AI Generated Feedback for Students' Assignment Submissions

A case study in generating feedback for students' submissions using ChatGPT

Lars Nysom

Assistant Professor

University College of Northern Denmark

lany@ucn.dk

August 2023

Abstract

In the evolving landscape of education, the transformative potential of AI is gaining momentum. While AI-powered systems contribute to teaching, a void persists in generating personalized feedback for assignments. This study bridges this gap by integrating AI-generated feedback within the UCN didactical framework, exploring a resource-efficient, student-centric approach.

The study delves into prerequisites to empower a chatbot, utilizing a generative language model like ChatGPT, to craft tailored feedback for assignments. Educational context, assignment specifics, evaluation criteria, and an indicative solution serve as input prompts, generating valuable feedback aligned with lecturer perspectives. The analysis of AI-generated feedback for the assignments included in the case study reveals consistent quality.

Developing a feedback system extends beyond ChatGPT, as reliability demands lecturer validation pre-student receipt, ensuring transparency. A pipeline leveraging the OpenAI API is proposed for operationalization. This efficient approach resonates with personalized feedback objectives, complementing pedagogical methods and prompting subsequent dialogues.

The study's conclusion synthesizes findings and contemplates AI integration implications. Insights emerge on benefits and challenges, guiding future research. Establishing AI's role in feedback augments educational needs, fostering enriched student engagement.

Table of Content

1. Introduction

In recent years, the increasing availability and advancements in AI technologies have opened new possibilities for enhancing various aspects of education. One such area is the generation of feedback on student hand-ins, which plays a crucial role in fostering student growth and improving their understanding of subject matter. This case study explores the integration of ChatGPT, an AI language model, into the feedback generation process in the context of applied computer science education.

At the Computer Science programme at University College of Northern Denmark (UCN) in Aalborg, the subject of programming as well as technology and system development are taught during the first three semesters. In most of these subjects, the students work with different assignments where they hand in a synopsis, an extended abstract, or a piece of software code they have produced.

We encourage the students to reflect over problem solutions rather than learn practices by heart, so we much prefer assignments there has no fixed solution, that forces the students to search for knowledge and reflect over the content of their answer more than remember a phrase or a sentence from a book, since we believe that it will prepare them better for a job in the industry. Also, one objective for the programme is that the students become able to assess practice-related problems in relation to computer systems and select solution options [1]. A characteristic feature in an open assignment is that it does not have a single solution, even though the question or task is well defined, the answers can be formulated in different ways and an assignment that results in a piece of software, can also be programmed differently and still work and contain all the requested properties.

In these cases, feedback is a very important part of the learning experience, and it is essential that it is valid and continuous. The fundamental difference between effective teaching and merely giving out content is, from our point of view, that effective teaching includes giving constructive feedback. The students must therefore receive feedback that is adapted to their individual assignment submissions, which can be an enormous and very time-consuming task for the lecturer.

Due to lack of resources, individual feedback is rarely given and, even though it is the best option for the individual student, it is only given after major project work, and in some cases, only on a group basis and orally. Another form of feedback is an indicative solution to the assignment, and it is then up to the students to self-assess their own solution or do a peer review. The experience is, however, that the students gain little from this because of the informal and unstructured approach, and they prefer personalized and specific written feedback, so they can review it and reflect on their solution.

1.1. Problem Statement

At UCN we consider these limitations in giving good feedback a problem which prevents us from delivering good and up-to-date education and we are considering ways to provide the students with better and more consistent feedback within the financial framework we are subject to, or in other words, we want to provide lecturers with a tool that can help them in creating a less generic and more specific feedback to individual students, considering that time and resources are limited.

Artificial intelligence (AI) has recently been the subject of much debate, including also in connection with education and study work, but the prospect of students cheating for an exam or use AI to write reports takes up most of it. However, it cannot be denied that it is a tool that can be utilized in many ways, and it is therefore worth examining for its usefulness in aiding lecturers giving feedback to student's assignment submissions.

One popular tool is ChatGPT developed by OpenAI[1] which is a natural language processing tool driven by AI technology that allows you to have human-like conversation with it. It is trained to follow an instruction in a prompt and provide a detailed response [2] [3]. In this study we will investigate how ChatGPT can be used to generate feedback for individual assignment submissions from our students.

The hypothesis for this project is that an AI assisted feedback system will be beneficial for the lecturers because they do not have to spend a lot of time testing or thoroughly understand a student's solution to a given assignment and subsequently write feedback. They only need to read the feedback generated by the AI and validate the quality of it, make corrections, if necessary, and give it to the student. Also, the lecturer has the possibility to refer to previous given feedback for similar assignments to track individual students progress and to adapt the didactic and pedagogical approach to the subject. This can be of benefit to the students because they get concrete written suggestions for improvements they can use in upcoming assignments. Other gains are that the AI provides fair and open-minded feedback to all students, regardless of factors unrelated to the assignment (e.g., absence from lectures, political standpoints, etc.).

1.2. Problem Definition

Even though there are already systems that can support teaching using AI (e.g., Coding Rooms[2], MyViewBoard Sens[3]), we have not found any product that is specifically able to generate personalized feedback for general assignments which is more suitable for our didactical method at UCN which is based on three concepts that students must perform – experience, reflect, and act [4].

Based on the assumption that utilizing AI in the process of providing students with constructive feedback on their assignments is helpful for the lecturers, save resources, and is beneficial for both lecturers and students, this report aims to provide an answer to the following three questions:

1. What is needed to let a chatbot that adopts a generative large pretrained language model such as ChatGPT generate feedback for a student's suggested solution to an assignment?
2. Are any specific kind of assignments better suited for generating feedback?

If we find that an AI can generate personalized feedback to a student's assignment, the next step is to describe a formalized system where submissions can be uploaded by the students after which they receive the feedback. This can be expressed in this final question, that this study will try to answer:

[1] https://openai.com/chatgpt
[2] https://www.codingrooms.com
[3] https://myviewboard.com/products/sens/

3. How can the process of giving feedback to students be operationalized?

2. Crafting Feedback to Students

Feedback is an essential component of the learning process in higher education. As educators, providing constructive and meaningful feedback to university college students is crucial for their academic and personal growth. In UCN's basic didactical principles, feedback is described as an important part of the continuous dialog with the students about their work and learning process, where the lecturer, students and fellow students are in a targeted partnership about the work's qualities and development potential [4].

Feedback is therefore defined as a part of the dialog with each individual student about their learning process, and as such encourages using oral feedback more than written. There are good reasons for this as the general understanding of feedback has changed over the years from a linear approach to a more interactive one based on dialog between lecturer and student [5].

The scope of this project is, however, not to determine if written feedback is better than oral, but only to accommodate a request for written feedback from the students. Also, the definition for good feedback in UCN's didactical approach is rather vague since it is based on dialog and in that context will adapt according to the specific scenarios it is given in. The purpose is to encourage the students to reflect and think innovative by asking them "why"-questions rather that telling them "how" or "what" in relation to their work [5]. To force the students to reflect we prefer to give feedback as part of a dialog with the students and with an intuitive approach where we for instance ask a question like "why have you chosen this technology" or "why does your implementation of the singleton design pattern not work". Feedback is an essential tool when it comes to innovative thinking, where the student becomes able to push to already existing solutions they have previously developed, without telling them how.

We have observed that the students seldom take elaborate notes in the feedback situation, even if they are encouraged to do so, and the information is, to a certain degree, often forgotten a few days later, especially when it comes to smaller and less important details. It is also not possible for them to go back to earlier submissions for a recap of the information that was given.

In the light is this ad hoc approach for giving feedback, providing the students with written feedback is unsolicited in the context of UCN's didactical approach, and there are not any guidelines on how to structure this. To make it fit into the lessons it must be perceived as a tool for enhancement of the dialog between student and lecturer with the purpose of contributing to students' overall learning journey.

To do that, feedback should not only highlight areas for improvement but also acknowledge students' strengths and successes. A balance between positive reinforcement and constructive criticism fosters a supportive learning environment and encourages students to build upon their existing skills and knowledge.

Vague or general feedback can be confusing and discouraging to students. Instead, we should offer specific comments on their work, pinpointing areas that require attention and offering actionable steps for improvement. This helps students understand the precise changes they need to make and boosts their confidence in implementing those changes effectively.

To summarize good feedback should incorporate the following properties to create a supportive and productive learning environment:

Acknowledgement: Feedback that starts with acknowledging the student's efforts and achievements. This creates a positive tone and demonstrates that their work has been recognized, helping to build a rapport between the student and the instructor.

Guidance: Feedback that provides direction and advice on how to improve. It goes beyond simply pointing out mistakes, offering suggestions on how to enhance their understanding and skills. This could involve recommending resources, suggesting alternative approaches, or providing references for further study.

Constructive: Feedback that is designed to be helpful and encouraging. It highlights both strengths and areas for improvement, aiming to foster a sense of progress and growth. It avoids being overly negative or discouraging, focusing on the student's development.

Specific: Feedback that is detailed and specific to the student's work. It avoids vague statements and generalizations, pinpointing exactly what aspects of their work are strong and what needs improvement. Specific feedback allows the student to understand the context and apply changes effectively.

Actionable: Feedback that suggests actionable steps for improvement. It goes beyond just identifying issues by providing concrete advice on how to address those issues. This empowers the student to make meaningful changes and see immediate progress.

Highlights areas for improvement: Feedback that doesn't just focus on strengths but also points out specific areas where the student can enhance their skills or understanding. It helps the student prioritize their efforts and focus on aspects that will lead to significant growth.

To be able to assess the quality of feedback, we will do a small comparison between what is considered good and bad feedback. The assignment below is imaginary, and the feedback examples are made to highlight the qualitative differences.

Assignment:

Design a simple login system using Python that takes a username and password, checks them against a predefined list of credentials, and grants access if the credentials are valid. Provide comments in your code to explain each step.

Bad feedback:

Your explanation is confusing, and I don't get what you're trying to say. Your code doesn't make sense. It's not what I expected, and you need to improve your code and explanations.

Good feedback:

Your code successfully checks credentials but consider using functions to improve modularity. Your comments provide a clear outline of the steps in your code. To take it further, you can explain the reasoning behind your choice of the list data structure and explore how this might affect performance with larger datasets.

Analysis:

The bad feedback is unhelpful, vague, and overly negative. It lacks specificity, guidance, and constructive elements that can guide the recipient toward improvement.

The good feedback on the other hand is constructive, specific, actionable, and delivered in a supportive and respectful manner. It helps the recipient understand both their strengths and areas for improvement, guiding them toward enhancing their skills and performance.

By understanding the role of the written feedback and how it fits into the didactics of UCN, we can create a feedback culture that empowers students to thrive academically and personally. As a result, students become more self-aware, motivated, and confident learners, better equipped to achieve their academic and professional goals.

3. Applying AI in Teaching

In the realm of education, the integration of AI has emerged as a transformative force, offering new possibilities to enhance the learning experience for students. In this case study we will explore the application of AI in the generation of personalized feedback for students' assignment submissions.

AI-powered generative language models have demonstrated remarkable capabilities in understanding and producing human-like text. Leveraging their proficiency in natural language processing, these models hold the potential to generate coherent and contextually relevant feedback on students' work. By analyzing the intricacies of the submitted assignments and referencing vast repositories of knowledge, these language models can produce detailed and targeted feedback that addresses individual students' strengths and weaknesses. This personalized approach to feedback not only encourages continuous improvement but also fosters a more engaging and dynamic learning environment.

Central to this research is the exploration of large pre-trained language models, which have showcased unparalleled linguistic prowess across a myriad of language tasks. The pre-training process on vast datasets equips these models with a deep understanding of grammar, semantics, and subject-specific knowledge, making them ideal candidates for providing insightful and domain-specific feedback. By fine-tuning these models on educational data and assignment samples, their ability to offer precise and accurate feedback can be honed, significantly enhancing the overall learning experience for students [6] [7].

Throughout this case study, we will delve into the practical implications of applying AI in generating feedback for students' assignment submissions. By experimenting with generative language models and large pre-trained language models, we seek to uncover the potential benefits and challenges associated with AI-driven feedback in education.

The goal of this research is to push the boundaries of educational technology and redefine the feedback process, paving the way for more effective and personalized learning experiences. By marrying the capabilities of AI with the expertise of educators, we envision a future where students receive tailored feedback that nurtures their academic growth and fosters a deeper understanding of the subject matter.

3.1. Neural Networks

A neural network is a computational model inspired by the structure and function of the human brain's neural networks. It is a type of machine learning algorithm that is used for various tasks, including pattern recognition, classification, regression, and more. The basic building block of a neural network is a neuron, which is a mathematical function that takes input, processes it using weights and biases, and produces an output. Neurons are organized into layers, with each layer connecting to the next, forming a network-like structure. The first layer is the input layer where the network receives data to be processed and the last the output layer, with the layer in between, which can be multiple, is called the hidden layer. [8] [9].

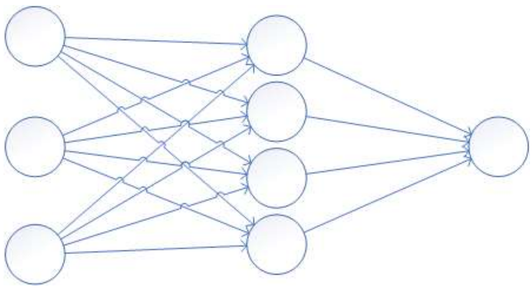

Figure 1: Layers in a neural network

Each individual neuron in the network is composed of input, weight, bias, and output. The weight represents the importance of that input in determining the output. Higher weights mean that the input has a larger influence on the outcome. Bias in a neural network is like an inherent inclination or prejudice for or against a particular outcome. It serves as an additional parameter that aids the network in making decisions. In the context of neural networks, bias helps shift the output of the neuron either up or down, ensuring flexibility in the decision-making process.

In a neural network, a neuron receives various inputs. Each of these inputs has an associated weight, indicating its importance. The neuron multiplies each input by its weight and then combines these products into a single sum. This sum is then processed by an activation function, which gives an output value. If this output value is above a certain set limit or threshold, the neuron is considered *activated* and sends its output to the following layer. This means that the output from one neuron can serve as an input for a neuron in the next layer.

Consider a scenario where a student, named Alice, is faced with a multiple-choice question on her test. To determine if she will answer it, she considers several factors:

Inputs: These represent the different pieces of information Alice remembers or pieces of knowledge that might help:

- Did I study this topic? (Yes: 1, No: 0)
- Did my friend discuss this with me? (Yes: 1, No: 0)
- Did my teacher emphasize this during a lecture? (Yes: 1, No: 0)

Weights: Each piece of remembered information does not have equal importance in helping Alice decide. Weights are like confidence levels:

- Studying is the most reliable source of knowledge for Alice (Weight = 5).
- Discussions with friends are sometimes helpful but not always accurate (Weight = 2).
- Alice trusts her teacher's emphasis because important topics are usually highlighted (Weight = 4).

Multiplication and Summation: Just like in our earlier example, Alice multiplies each piece of knowledge by its weight (importance) and then sums them up. Suppose Alice studied the topic (Value = 1), did not discuss it with a friend (Value = 0), but the teacher emphasized it (Value = 1). The total becomes:

$$(1 * 5) + (0 * 2) + (1 * 4) = 9$$

Threshold & Activation Function: Alice has an internal confidence threshold. If the total from the above step is greater than, say, a value of 6, Alice feels confident enough to answer the question. If not, Alice might leave it or guess.

Since 9 > 6, Alice answers the question.

The example above does not describe the concept of bias, which in a neural network is like an inherent inclination or prejudice for or against a particular outcome. It serves as an additional parameter that aids the network in making decisions. In the context of neural networks, bias helps shift the output of the neuron either up or down, ensuring flexibility in the decision-making process. This could be integrated in the example as if Alice has a personal bias towards always attempting questions, even if they are not entirely confident. This inherent bias can stem from an experience, such as receiving partial marks for partially correct answers or being advised to never leave questions unanswered.

Bias Value: We introduce a bias value. This value represents Alice's inclination to answer questions even if the other factors are not strongly pointing towards an answer. Let us give this a value of -2.

Incorporating Bias into the Decision: Using the previous scenario, where the sum of Alice's factors was 9, we add the bias:

$$Total\ confidence = 9 + (-2) = 7$$

The threshold remains 6. So, even after accounting for the bias, Alice's total confidence of 7 is still above the threshold, leading her to answer the question.

Despite being negative, which might seem counterintuitive since Alice's bias is to attempt questions, the bias did not change her decision. However, biases in neural networks can often significantly affect the outcome. For instance, if the summation of the weights and inputs was closer to the threshold, say 7, and Alice's inherent bias was -2, then the bias would push the result below the threshold, influencing the decision to not answer the question.

The process through which the model adjusts its weights and biases, referred to as *internal parameters*, is called *learning*, and improves its performance on a specific task. These weights are numerical values that determine how the model transforms its inputs into outputs. Learning occurs through a process called *training*, where the model is provided with input data along with corresponding target outputs, and it adjusts its weights to minimize the difference between its predictions and the true targets [9].

In the context of a predefined vocabulary of tokens, such as in natural language processing, the model's inputs are tokenized text sequences, and the weights learn to represent the relationships between these tokens. Each token is often represented as a high-dimensional vector (embedding), and the model adjusts these embeddings to capture semantic meanings and relationships between words.

So, training a neural network involves feeding it with input data and adjusting the weights and biases iteratively through a process called *backpropagation* where the model calculates the gradient of the loss with respect to its weights. Backpropagation uses optimization techniques like gradient descent to minimize the difference between the predicted output and the actual target output. This process helps the neural network learn to make more accurate predictions over time. During the training process, the model adjusts the weights and biases based on the input data to learn patterns and relationships within the data [9].

3.2. Generative Language Models

A generative language model is a neural network architecture, specifically based on the Transformer model, designed for natural language processing tasks [7]. It relies on the concept of self-attention mechanisms, which allows the model to capture dependencies between different words in a sentence effectively. The model is "generative" because it can generate coherent and contextually relevant text by predicting the likelihood of the next word or token in a sequence given the preceding tokens. Through extensive training on large corpora of text data, the generative language model learns to model the statistical patterns and relationships within the language, enabling it to generate human-like text for various applications, such as machine translation, text completion, and creative writing [8].

This human-like text is generated in response to prompts. In the context of natural language processing (NLP), generative language models focus on generating coherent and contextually relevant text based on the patterns and structures they learn from vast amounts of training data comprising diverse examples of human language, allowing them to learn the statistical patterns and relationships between words, phrases, and sentences.

However, generative language models can sometimes generate outputs that may seem plausible but are factually incorrect or misleading, especially if the training data contains biased or unreliable information. Researchers and developers work continuously to improve these models and make them more reliable and useful for various real-world scenarios [9] [10].

In other words, a generative language model can take what it has learned from the examples it's been shown and create something entirely new based on that information.

3.3. Large Pre-Trained Language Models

A Large Pre-Trained Language Model (LPLM) is a type of artificial intelligence model designed to understand and generate human language. It is "pre-trained" in the sense that it has been exposed to vast amounts of text data from various sources before being fine-tuned for specific tasks. These models are built using deep learning techniques, particularly using neural networks [7].

The main idea behind LPLMs is to leverage the vast amount of textual information available on the internet and use it to train a general language understanding capability. This pre-training phase involves predicting missing words or next words in a sentence, given the context of the surrounding words. The model learns to understand the syntax, grammar, semantics, and other linguistic patterns during this phase.

The most well-known and widely used examples of Large Pre-trained Language Models include:

GPT (Generative Pre-trained Transformer): Developed by OpenAI, it was one of the first LPLMs to gain widespread attention. GPT is based on a transformer architecture, which allows it to handle long-range dependencies in text efficiently [7].

BERT (Bidirectional Encoder Representations from Transformers): Developed by Google, BERT is another influential LPLM. It introduces a "masked language model" pre-training task, where it predicts randomly masked words in a sentence, considering the context [11].

These LPLMs have revolutionized natural language processing (NLP) tasks due to their ability to generate coherent and contextually relevant text. After pre-training, these models can be fine-tuned on specific tasks, such as text classification, sentiment analysis, question-answering, language translation, and more. Fine-tuning involves taking the pre-trained model and training it further on a smaller, task-specific dataset, thereby adapting its language understanding to a particular application.

Large Pre-trained Language Models have been widely adopted because they offer numerous benefits, including:

Transfer Learning: Pre-training allows models to learn general language representations that can be transferred to various downstream tasks without extensive training on task-specific data.

Reduced Data Requirements: Fine-tuning on smaller datasets can yield impressive results, even with limited labeled examples.

Versatility: These models can be applied to a wide range of NLP tasks with only slight adjustments and achieve state-of-the-art performance in many cases.

3.4. ChatGPT

Chatbot GPT (Generative Pre-trained Transformer) is a language model developed by OpenAI. It is part of the GPT series of models, which are designed to generate human-like text based on the input they receive. It is a highly advanced language model that is trained on a vast amount of data from the internet. It uses a transformer architecture, which enables it to process and generate text more effectively, making it capable of understanding context and generating coherent responses.[4]

[4] https://openai.com/chatgpt

It can interact with users through text-based conversations, answering questions, providing information, and engaging in natural language dialogues. With over 175 billion learnable elements, it is one of the most powerful language models available. Its extensive training allows it to perform a wide range of tasks, including language translation, question-answering, text completion, text generation, and more.

This makes it interesting in relation to this project because it should be able to understand the same assignments that the students get as well as context about the assignment, and therefore validate the quality of the student's submission.

3.5. Prompt Engineering

Prompt engineering refers to the practice of crafting effective prompts or instructions to interact with a language model like ChatGPT. It involves formulating clear and specific prompts that guide the model's responses towards the desired output [12] [13].

In the context of using a language model, prompt engineering is important because it helps achieve more accurate and relevant results. By carefully designing the prompt, users can influence the model's behavior and steer it towards providing helpful and informative responses.

Some key aspects of prompt engineering are:

Clarity: A well-crafted prompt should be clear and unambiguous, providing precise instructions to the model. Vague or ambiguous prompts may lead to unpredictable or nonsensical outputs.

Context: Including relevant context in the prompt can help the model understand the desired direction of the conversation. By providing background information or specifying the desired format, users can guide the model's response more effectively.

Examples: Providing specific examples related to the desired output can help the model understand the expected format, style, or content. This can improve the relevance and accuracy of the responses.

System Messages: System messages are instructions or guidance provided to the model within the conversation. They can be used to set the behavior or role of the AI assistant in the conversation. System messages are often used to establish a consistent persona for the model or to remind it of specific guidelines.

Experimentation: Prompt engineering often involves an iterative process of trial and error. Users may need to experiment with different prompts, parameters, or techniques to find the most effective approach for their specific use case.

It's worth noting that prompt engineering is not about "hacking" or tricking the model but rather about providing explicit instructions and guidance to achieve the desired outcomes. Responsible and thoughtful prompt engineering is essential to ensure that the model provides accurate, helpful, and ethical responses.

It I also worth noticing that using the ChatGPT web interface has its issues related to the prompt. There are certain limitations in place to ensure that the use of AI technology remains safe, respectful, and appropriate for users[5].

4. Method

The Computer Science programme at UCN covers topics from the process of developing software, technology, and programming and the main goal for the students is to become software developers. Every assignment the students work with is designed to help them achieve knowledge, skills, and competencies towards that.

The types of assignments range widely, from programming exercises to multiple-choice questionnaires to small essays with reflections over a relevant topic or a simple short review question, it will be relevant for this experiment to select assignments of different types to be able to find out if some types are better suited or more relevant than others. A multiple-choice questionnaire typically holds the correct answers when given to the students, meaning that it is only possible for the student to select a correct or incorrect answer and feedback can be instant. Also, a solution to a programming exercise seems at first rather straightforward since it can be evaluated as "if it works, it is correct", but there are still parameters that can be looked at and evaluated, for instance how well does the proposed solution comply with the SOLID principles[6] in relation to object-oriented programming, or could the solution be done in a simpler way with fewer lines of code? To generate feedback for the student regarding those features would need a thorough analysis of the code.

4.1. Case study

A case study is a qualitative research method that involves in-depth investigation and analysis of a specific subject, event, group, organization, or situation. It is commonly used in various fields, including business, psychology, medicine, social sciences, and more [13] [14].

This method allows us to delve deeply into a specific AI system or application, in this case it will be ChatGPT, understanding its unique characteristics and context. This depth of exploration can uncover nuances and complexities that might be missed in broader quantitative studies. By focusing on a real AI system, a case study provides a more authentic and realistic context for understanding how feedback generation works in practice. This can help bridge the gap between theory and application.

Using the case study method to investigate how feedback could be generated by an AI can provide several benefits and insights about the subject, and can, because of the explorative and evolving format, easily be adapted based on our findings along the way.

A case study will be conducted using ChatGPT, based on the three different types of assignments described in the following section. The assignments were given to students on the 2nd semester of the Computer Science programme, and they have handed in their submissions as text files. These submissions will be used individually as input to ChatGPT along with the description, indicative solution, and evaluation criteria, asking

[5] https://openai.com/policies/usage-policies
[6] The SOLID abbreviation describes five different principles that must be observed in object-oriented programming (https://en.wikipedia.org/wiki/SOLID).

the chatbot to generate feedback. The output from ChatGPT will afterwards be examined by experienced lecturers to find whether the output is useful or not.

4.2. Assignment Cases

The three assignments that have been chosen for this case study, are from subjects that are taught in the 2nd semester of the Computer Science programme. They are all assignments with open solutions, where students would benefit from individual feedback to highlight their professional strengths and weaknesses and they are also representing three of the most common assignment types given to students: the review question, the programming exercise, and the formal essay.

Case 1: A short review question related to the subject of operating systems

Subject	Operating Systems
Assignment	Briefly explain each state in the Five-State Process model.
Description	This is a review question that the student answer by handing in a short description (max. 1200 characters including spaces).
Indicative solution	Running: The process that is currently being executed. Ready: A process that is prepared to execute when given the opportunity. Blocked: A process that cannot execute until some event occurs, such as the completion of an I/O operation. New: A process that has just been created but has not yet been admitted to the pool of executable processes by the operating system. Exit: A process that has been released from the pool of executable processes by the operating system, either because it halted or because it aborted for some reason.
Evaluation Criteria	All five states described in the book should be included and described correctly in the students answer.

Case 2: A small programming assignment

<table>
<tr><td>Subject</td><td>Programming with Java</td></tr>
<tr><td>Assignment</td><td>Explain the singleton design pattern and give an example implementation.</td></tr>
<tr><td>Description</td><td>This is a programming exercise where the students must demonstrate that they are able to implement a specific programming pattern and explain how it works. The solution must be handed in as a code-snippet[7] where the explanation can be embedded in the code as a comment.</td></tr>
<tr><td>Indicative answer</td><td>The Singleton pattern is a creational software design pattern that restricts the instantiation of a class to a singular instance. This can be done by making the constructor private and create a public static method in the class that creates and returns an instance once and subsequently returns that instance.

```
public class Singleton {
    private static Singleton instance;

    private Singleton() {
        System.out.println("Creating new instance");
    }

    public static Singleton getInstance() {
        if (instance == null) {
            instance = new Singleton();
        }
        return instance;
    }
}

public class Main {
    public static void main(String[] args) {
        // Test 1: Create an object
        ProductContainer obj1 = ProductContainer.getInstance();

        // Test 2: Try to create a second object, the first one
should be returned
        ProductContainer obj2 = ProductContainer.getInstance();

        // Test 3: Ensure both references point to the same object
        System.out.println(obj1 == obj2); // This should print
"true"

        // Test 4: Attempt to call the class constructor directly
(should not be possible)
        // ProductContainer obj3 = new ProductContainer(); //
Uncommenting this line should result in a compile-time error
    }
}
```
</td></tr>
<tr><td>Evaluation Criteria</td><td>The purpose and the functionality of the singleton pattern must be described in a few sentences.
The sample implementation should ensure that only one instance of an object can be created, by running tests that:
• Creates an object.
• When trying to create a second object, the first one is returned.
• It must not be possible to call the class constructor directly.</td></tr>
</table>

[7] A code-snippet in this context is a piece of code that is syntactically correct, but not executable. It is possible to test it, but not run it without integrating it in an executable program.

Case 3: A formal essay about a specific topic.

Subject	Asynchronous Programming
Assignment	Explain the properties of mutual exclusion in concurrent operations using the Dining Philosophers Problem. Be sure to emphasize on some of the side effects such as deadlock and starvation and give examples on how they can be solved.
Description	In this type of assignment, the students must demonstrate a deeper understanding of a specific topic. In this case it is the property of mutual exclusion that relates to concurrency control in asynchronous, or parallel, execution of code where a resource is shared between concurrent operations. The students will hand in a short essay with their answer (max. 7200 characters including spaces)
Indicative answer	The Dining Philosophers Problem is a well-known example on the challenges in concurrent programming related to synchronization issues when accessing shared resources. Five philosophers dine together at the same table eating from one bowl of noodles placed at the center of the table. There are five chopsticks placed on the table, one between every philosopher, and any philosopher needs to pick up both the chopstick to the left and to the right of him, thus making every chopstick a shared resource. The philosophers alternate between thinking and eating, and when in thinking mode both chopsticks are placed at the table making them available for the philosophers next to the one thinking. The challenge is to create an algorithm such that no philosopher will starve, which will happen if all philosophers want to eat at the same time and picks up one chopstick and waits for the other to be released, which will never happen since they all are waiting. To solve this, we look at Stallings lists of requirements for mutual exclusion: 1. Mutual exclusion must be enforced: only one process at a time is allowed into its critical section, among all processes that have critical sections for the same resource or shared object. When a philosopher has picked up a chopstick, it will be flagged as in use so no other philosopher can pick it up. 2. A process that halts in its non-critical section must do so without interfering with other processes. When the philosophers are thinking they are in their non-critical section of the code where they do not access any shared resource. If the thinking process halts, it will therefore not affect any other process. 3. It must not be possible for a process requiring access to a critical section to be delayed indefinitely: no deadlock or starvation. A simple solution is found in the asymmetric approach where one philosopher picks up the chopsticks in reverse order, which will remove the problem of all picking up the left chopstick at the same time. One philosopher will try to pick up the chopstick to the right, and if he succeeds this will prevent the philosopher next to him to pick up his left chopstick, and forces him to wait, or the philosopher himself must wait, until the philosopher next to him is finished. If all philosopher wants to eat at the same time, there will always be one that cannot eat before the one next to him is finished. 4. When no process is in a critical section, any process that requests entry to its critical section must be permitted to enter without delay. When a philosopher is done thinking and wants to pick up a chopstick, this is allowed only if the chopstick is placed on the table. This can be done using the synchronized keyword in Java. 5. No assumptions are made about relative process speeds or number of processors. Depending on the programming language, decisions about which processor to use is up to the operating system, which is the case with Java. Otherwise, this must be handled in another process running independently of the program running the Dining Philosophers. 6. A process remains inside its critical section for a finite time only. A philosopher is only allowed eating for a finite time, which must be implemented in the code.

Evaluation Criteria	All six requirements from Stallings book must be referenced and taken into consideration in relation to solving the Dining Philosophers Problem. There are several known solutions to the Dining Philosophers Problem (e.g., Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, Asymmetric) and the students should name and describe at least one.

The cases described above are real assignments given to and answered by students (see Appendix AAppendix A) who submit a text containing their solution. The submission should comprehend the descriptions in the assignments and can be their reflections on the matter, a piece of code or a short concise text. The assignment and the indicative solution are formulated in natural language, or code, but there will of course be some variations in them depending on the assignment. For example, in a programming assignment, the syntactic limitations of the chosen coding language could determine the formulation of the question. In the same way, the answers to the tasks will be formulated within such a delimitation, but in addition there will be variations based on the students' personality, professional level, knowledge, etc. The answers must be evaluated based on the criteria that defines the correctness of the proposed assignment solution.

To answer the first question from the problem definition (see section 1.2), it is necessary to work with the structure of the prompt, to make ChatGPT generate promising feedback for the student's assignment submission.

1) The educational context must be provided to the prompt so the chatbot can generate the feedback in the correct academic language.
2) The exact wording of the assignment must also be added to the prompt to make the model understand what problem the student is trying to solve.
3) A description of what is formally expected of the students' submissions (e.g., type of answer, length, code language, etc.)
4) To be able to identify which criteria the submission must meet, they must also be provided.
5) The indicative solution to the assignment should also be added to the prompt since we cannot trust the model to be able to answer the assignment correctly.
6) The student's submission.

Whether to provide the prompt with the indicative solution or not, has been decided on the background that ChatGPT is not necessarily able to generate a correct answer itself. This became apparent in the preliminary work for this case study, where some conversations with the chatbot indicated that this was indeed the case (see Appendix C).

Another question that was raised in this process was whether to include a reference to the literature and sources used in the subject. This was deselected as the data for the model comes from the internet, and we cannot be sure that the books are available in the public domain.

To summarize, the form on the input to the chatbot will as a starting point contain the following:

Context: A short description of the context in which the feedback should be given. Since all assignments are from the same education, this will be the same for all the cases and will contain the level of education, the purpose of the request, what information is provided in the prompt, and what type of assignment is in question.

Assignment, **Indicative solution**, **Evaluation Criteria**: These three will be sent to the chatbot as they are described in the cases above.

Students Submission: The text that constitutes the student's assignment submission, cleared of personal information such as name or class.

The second question in the problem statement requires a comparison of the generated feedback for the different types of assignments, which can be difficult since they are different and shared properties used for comparison can be hard to identify. However, the generated feedback needs to be evaluated and validated by both experts and students to make sure it is valid and useful for the students, so based on such an assessment, it can be established if some assignment types are better suited than others. In other words, a grading of the quality of the generated feedback must be done between the different assignment types.

4.3. Evaluation

The findings from the evaluation and student perception study are analyzed and discussed. The strengths and limitations of using ChatGPT for feedback generation in applied computer science education are critically examined. Factors such as accuracy, efficiency, scalability, and adaptability to different assignments and educational settings are considered.

If the evaluation of a generated feedback fails, it simply cannot be considered useful, and we must reconsider if the prompt could be changed and how. As mentioned in the previous section, both experts and students should evaluate the generated feedback to be able to make a qualified assumption of its usefulness. It could however be questioned; does it make sense to have students evaluate? From their perspective there will be no difference if they were provided with human-generated feedback or AI-generated, and before this they were only used to self-assess against the indicative solution. It would make more sense to involve the students in relation to the third and last question from the problem definition since they would constitute most of the user group and should be considered as such in relation to operationalizing the feedback process. That said, the students can still contribute information about the content of the generated feedback in relation to the feedback, or lack thereof, they receive today so this will be the focus.

4.3.1. Expert Feedback Evaluation

Two experienced professors that are familiar with the specific domain of applied computer science will first evaluate the students' assignments and then evaluate the quality and effectiveness of the feedback generated by ChatGPT compared to their own. This is to compares the AI-generated feedback against human-generated feedback for a set of hand-ins and metrics such as accuracy, clarity, and relevance are considered.

4.3.2. Student Feedback Perception

To gauge the perception and acceptance of feedback by students, a survey or interviews will be conducted later. The students will be asked to compare the AI-generated feedback with the feedback forms they normally receive which is an indicative solution or general lecturer-provided feedback in plenum and provide their opinions on its usefulness, comprehensibility, and potential for improvement. This will take place when the first prototype is ready for testing in a realistic environment.

5. Generating Feedback for Cases

In the following section, the students' submissions to the three cases will be reviewed and the lecturers will summarize their observations and comments. It is important to emphasize that this is not normal practice, and the students will normally only be given the indicative solutions for the purpose of self-evaluation. Nevertheless, we have found it useful letting the lecturers make comments to compare with the AI-generated feedback, so that the quality of it can be assessed. The lecturers have been told to comment the submission in the same way they would if they were to give feedback to the students, but since some only provides feedback orally, the result shown here is a summary of several lecturers' contribution. An important point here is, that the lecturers are not asked to maintain a specific form or template when generating their feedback, but they are encouraged to do as they usually do. The feedback texts have been approved by all before it was added to the report.

The three cases will be reviewed by comparing the comments from the lecturers with the generated feedback from ChatGPT.

The feedback generated by ChatGPT is quite comprehensive, so it is assembled in Appendix B and the case reviews will not reference it explicitly.

5.1. Case 1: Review Question

Student 1

The lecturer has evaluated the student's submission and has the following comments:

> The student has answered the question quite thoroughly and includes five stages in the submitted text. Compared to the indicative solution, it is not a brief description though, and it exceeds the 1200-character limit. There are two states that is named slightly wrong compared to the indicative solution and the subject book, but the student's description of them demonstrates some understanding of their purpose. In general, it is a rather vague answer that shows an overall understanding of the subject without concretizing the five states.

The lecturers point out in their feedback that the student answered the question thoroughly and included all five stages in the response. Also, that the response is not brief and exceeds the character limit. Some state names are slightly wrong, and the overall answer is somewhat vague and lacking concretization.

The AI-generated feedback highlights the strengths of the student's submission, showing understanding of the model and some effort to explain each state in their own words. There are also specific and constructive suggestions to improve the response, addressing issues related to length, accuracy, terminology, completeness, and focus.

The AI-generated feedback is comprehensive and constructive, offering praise for what the student did well and providing clear guidance on how to enhance their submission. By pointing out strengths, such as understanding the model and trying to explain the states, the student gains confidence and motivation. The feedback's areas for improvement are specific and actionable, giving the student a roadmap to enhance their response effectively.

In contrast, the lecturers feedback focuses on general observations of the submission without providing actionable steps for improvement. While it acknowledges the thoroughness of the answer, it lacks specific guidance on how to address the identified issues, such as exceeding the character limit, using inaccurate state names, and offering vague descriptions. As a result, the feedback may not be as beneficial for the student's growth and improvement.

The AI-generated feedback encourages the student to be more concise and accurate while emphasizing the importance of using the correct terminology. It also directs the student to explicitly describe all the required states and stay focused on the model itself. Finally, the feedback offers a revised suggestion that serves as a model for the student to follow when revising their response.

Student 2

The lecturer has evaluated the student's submission and has the following comments.

> This answer covers all five states in the model and shows an overall understanding of their individual purpose.

Comparing the two feedbacks, the AI-generated feedback is more detailed, specific, and helpful for the student's learning and improvement, where the lecturers' feedback is too vague and general. While it acknowledges that the answer covers all five states and shows an overall understanding, it lacks the crucial details and specific pointers needed for improvement.

The AI-generated feedback provides details on each state in the Five-State Process model, pointing out the specific inaccuracies and inconsistencies in the student's descriptions. This specificity helps the student understand precisely where they went wrong and what needs improvement. Not only identifies this feedback the errors, but it also offers constructive criticism on how to correct them, which guides the student towards the right direction and helps them rectify their mistakes. It also emphasizes the importance of clarity and precision in the descriptions and provide clear and concise definitions for each state, ensuring the student understands the correct way to explain each state in the model.

In conclusion, the AI-generated feedback is the most useful for the student. It addresses the inaccuracies, provides constructive criticism, offers a revised version, and evaluates the submission comprehensively. This kind of feedback not only helps the student improve their assignment but also enhances their understanding of the subject matter, making it a more valuable learning experience.

Student 3

The lecturer has evaluated the student's submission and has the following comments.

> The student covers all five states in the model. However, it is not all states that are described correct. A process is not blocked by the CPU, but for example by an I/O request where the process is forced to wait until the request is answered. The explanation of the Ready state is not precise enough.

The lecturers' feedback acknowledges that the student covers all five states in the model and points out and clarifies that the student got it wrong according to a process that is blocked and why. It also points out that not all states are described correct but does not specify which states are described incorrectly. Neither does it provide guidance on how to improve the explanation of the "Ready" state.

The AI-generated feedback provides specific remarks on each state, highlighting what the student got right and what needs improvement. It suggests clarifications and elaborations for each state, helping the student understand the concepts better. It is a little fussy though and calls out minor inaccuracies, but it is not clear which states are inaccurate without a comparison to the student's original submission. Also, it says the students' explanation for the "exit" state is acceptable, but the student has not mentioned this state in the submission.

Example 4: Excerpt from AI-generated feedback

Exit: The student's explanation is acceptable as it correctly mentions that a process in the "Exit" state has been released from the pool of executable processes by the operating system, either because it halted or aborted.

A quick comparison with the indicative solution reveals that the AI might have wrongly perceived that as a part of the students' submission. However, this does not apply to the rest of the submission.

Example 5: Excerpt from indicative solution

Exit: A process that has been released from the pool of executable processes by the operating system, either because it halted or because it aborted for some reason.

Even so, the AI-generated feedback is more useful and effective for the student's learning and improvement. It provides constructive criticism, specific areas for improvement, and clear suggestions on how to enhance the submission. It acknowledges the student's effort and points out the areas where the student demonstrates a basic understanding. Moreover, the AI-generated feedback offers a revised and improved submission, making it easier for the student to comprehend the correct explanations for each state. It is necessary to make changes before the student get it back.

In contrast, the lecturers' feedback is less specific and fails to provide the necessary guidance for the student to correct their errors. It only points out the presence of inaccuracies without specifying which states need improvement or how to enhance the explanations.

Yes
No
Neutral

Table 1: Submission meet evaluation criteria

Evaluation Criteria	Student 1	Student 2	Student 3
All five states described in the book should be included and described correctly in the students answer.	Yes	Yes	No

Table 2: Feedback quality comparison summary

	Student 1		Student 2		Student 3	
Criteria for good feedback	**Human**	**AI**	**Human**	**AI**	**Human**	**AI**
Acknowledgement						
Guidance						
Constructive						
Specific						
Actionable						
Highlights areas for improvement						

5.2. Case 2: Programming Assignment

Student 1

The lecturer has evaluated the student's submission and has the following comments.

> The explanation demonstrates an understanding of how the singleton pattern works. However, the implementation lacks a private constructor, which means the class can be instantiated more than once and is therefore not a singleton. If the code included a test to prove that it was in fact an implementation of a singleton, this could have been discovered.

The lecturers feedback recognizes the student's understanding of how the Singleton pattern works, which is a positive aspect. It also points out a major flaw in the implementation – the absence of a private constructor, which renders the class incapable of being a true singleton.

It might be debatable if the feedback provides specific guidance on how to fix the issue with the missing constructor. The sentence: "*the implementation lacks a private constructor*" can be interpreted as the correct solution will be to add a private constructor, however it could be made crystal clear with an example implementation of the pattern and an explanation of the impact of adding private constructors to a class. It does state though, that a test would have been helpful to identify this specific issue, but again without providing guidance how. It does not mention the general importance of including a test to verify the implementation's correctness.

The AI-generated feedback is organized into distinct sections, addressing both the purpose of the Singleton pattern and the implementation evaluation. It also provides constructive suggestions for improvement, including specific tests that need to be included.

While this is more comprehensive than the first, it doesn't clearly praise the student for understanding the Singleton pattern and providing a reasonable implementation. It could also have emphasized the positive aspects more explicitly before diving into the areas for improvement.

Overall, the AI-generated feedback is more detailed and helpful. It takes a more constructive approach by not only pointing out the areas that need improvement but also providing specific suggestions on how to improve the assignment. It also provides a clear evaluation criterion, such as purpose, functionality, and sample implementation, which helps the student understand what aspects to focus on. The feedback emphasizes the importance of including tests to validate the correctness of the implementation, which is crucial for assessing whether the code follows the Singleton pattern correctly.

Student 2

The lecturer has evaluated the student's submission and has the following comments.

This submission lacks a sample implementation, and all the explanation illuminates, is a basic understanding of the purpose of the pattern and an example on its use.

The lecturers feedback points out that the student has a basic understanding of the purpose of the Singleton pattern, which indicates some level of comprehension. It also highlights the presence of an example on the use of the pattern in the submission.

However, the feedback is mostly negative, mentioning the lack of a sample implementation without providing specific guidance on how to improve it. It doesn't acknowledge any positive aspect of the student's submission, which might be demotivating. The feedback is too brief and doesn't provide constructive feedback or evaluation criteria to guide the student's improvement.

The AI-generated feedback provides specific points of improvement, highlighting the need for more clarity and precision in the explanation of the Singleton pattern's purpose and functionality, as well as acknowledging the students' correct understanding of the Singleton pattern's core concept. It points out that the student has implemented the pattern correctly and appreciates the relevance and effectiveness of the tests provided by the student, which is wrong since the student has not submitted any implementation. As for case 1, it seems like the AI takes the indicative solution and thinks it is part of the students' submission, since it mentions specifically that a private static instance variable, a private constructor, and a public static method to retrieve the instance are defined.

It could have included more positive reinforcement to motivate the student and build confidence in their understanding of the Singleton pattern. The feedback could also have given more guidance on how to improve clarity and precision in the explanation as well as an example implementation of the pattern (e.g., the indicative solution).

Despite its shortcomings, the AI-generated feedback is still more comprehensive and constructive than the human-generated. It highlights both the strengths and areas for improvement in the student's submission and it acknowledges the correct understanding of the Singleton pattern and providing specific guidance on how to enhance the explanation.

On the other hand, the human-generated feedback is rather discouraging, focusing only on the absence of a sample implementation, and giving little constructive input. It does not give credit for the student's basic understanding or acknowledge any positive aspects of the submission.

The error in the generated feedback points out the necessity of human validation before it is given back to the students.

Student 3

The lecturer has evaluated the student's submission and has the following comments.

> The student explains the overall purpose of the singleton design pattern but is wrong in relation to the implementation. The initializer method cannot be private, as in that case no instances of the class can be created at all. the implementation lacks a constructor even though it is mentioned in the explanation that a private constructor is needed to implement the pattern.

The lecturers feedback clearly highlights the incorrect points in the student's explanation and implementation. It provides specific details on what is wrong with the implementation, such as the incorrect private modifier for the initializer method and the lack of a constructor. There is a lack of constructive guidance on how to improve the assignment and it does not mention any positive aspects of the submission.

The AI-generated feedback acknowledges the student's basic understanding of the Singleton pattern and provides detailed constructive feedback on the purpose, functionality, and implementation of the pattern. It points out the error in the implementation but downplays the severity of it by calling it a "small issue", which it is not.

The modified description and implementation examples provided are clear and accurate and it includes suggestions for adding test cases, which were part of the evaluation criteria and is valuable for the student as it helps the student to demonstrate a deeper understanding of the pattern and its practical application. However, it could have been more explicit in pointing out the specific errors made by the student, especially because the error in the code means that the implementation does not work at all.

Both feedbacks point out areas for improvement in the student's assignment submission, but the AI-generated offers constructive criticism and provides specific directions for improvement. It helps the student to enhance their knowledge and comprehension of the Singleton design pattern and its implementation.

Yes
No
Neutral

Table 3: Submission meet evaluation criteria

Evaluation Criteria	Student 1	Student 2	Student 3
The purpose and the functionality of the singleton pattern must be described in a few sentences.			
The sample implementation should ensure that only one instance of an object can be created			
Contains minimum number of tests (3)			

Table 4: Feedback quality comparison summary

	Student 1		Student 2		Student 3	
Criteria for good feedback	Human	AI	Human	AI	Human	AI
Acknowledgement						

Guidance						
Constructive						
Specific						
Actionable						
Highlights areas for improvement						

5.3. Case 3: The Formal Essay

The three students' submissions to this assignment reflect that all have either misunderstood the assignment or have not been able to solve it.

Student 1

The lecturer has evaluated the student's submission and has the following comments.

> This submission is not sufficient, and the student must do a new attempt. Only four requirements are described, and these are not linked to the dining philosopher's problem. The 5th requirement "No assumptions are made about relative process speeds or number of processors" is described correct but very briefly. None of the known solutions are mentioned.

The lecturers feedback points out one requirement that is described correctly, albeit briefly. Other than that, the feedback is mostly negative without specific guidance on how to improve. If does not provide any direction on linking the content to the Dining Philosophers Problem, nor does not mention any solutions to the problem or side effects like deadlock and starvation.

Even though the submission might be insufficient, as the lecturers feedback claims, the AI-generated feedback starts with a neutral remark on the submission's overall content as being concise and it makes it clear that there are important errors and omissions in the submitted work. It continues giving specific guidance on what aspects of the assignment the student needs to address and lists clear criteria for improvement, such as linking to the Dining Philosophers Problem, emphasizing side effects, providing examples, and referencing requirements mentioned in the subject book.

The AI-generated feedback is the most useful because it not only points out the areas that need improvement but also provides specific guidance on how to enhance the submission. The student is given clear directions on what is missing and what to include to meet the assignment requirements fully. Moreover, the feedback encourages the student to research and provide concrete examples, which can significantly enrich the submission. In summary, even though the student's submission does not meet the requirements for the assignment, this feedback keeps the constructive tone by providing some concrete actionable points the student can work with.

In contrast, the feedback from the lecturers is more discouraging and lacks specificity on how to improve. It does not provide any valuable direction to the student, making it less useful for their learning and progress of the subject matter.

Student 2

The lecturer has evaluated the student's submission and has the following comments.

> The student only mentions four requirements, and although they are mostly correct, the description of them is very brief and superficial. Also, they are not linked to the dining philosopher's problem. None of the known solutions are mentioned.

The lecturers feedback acknowledges that the student has mentioned four requirements, which are mostly correct. The feedback is too brief though and does not provide specific areas of improvement or guidance on how to address the shortcomings in the submission. So, while acknowledging the correctness of some parts, the feedback lacks depth and direction for the student to improve their work effectively.

In the AI-generated feedback, details and specific points for improvement are provided, that identifies areas that needs clarification and elaboration, as well as additional content that should be included in the submission. The need to include all six requirements from the subject book is also pointed out, as is the importance of understanding the Dining Philosopher's Problem and its relationship to mutual exclusion. The feedback encourages the student to research and cite external sources properly, which will enhance the credibility and validity of the essay.

The AI-generated feedback is well-structured and covers multiple aspects of the assignment, giving the student clear directions on how to improve their work. It helps the student understand the areas where they need to focus and the specific actions, they can take to enhance their submission.

In conclusion, the AI-generated feedback is the best and most useful for the student. It provides a comprehensive assessment of the student's submission and offers constructive guidance on how to address the identified weaknesses. The feedback emphasizes the importance of research, accuracy, and proper citation while encouraging the student to delve deeper into the topic and provide a more comprehensive explanation.

Student 3

The lecturer has evaluated the student's submission and has the following comments.

> This submission is not sufficient, and the student must hand in a new attempt. The list in the submission contains only four points which, by the way, have nothing to do with the requirements for mutual exclusion, but can only be seen as results of correctly implemented mutual exclusion. We suspect that the student has misunderstood the assignment.

The lecturer's feedback is mostly negative and does not provide any specific guidance on how to improve the submission. They suspect that the student misunderstood the assignment but do not clarify what exactly is wrong with the submission.

The AI-generated feedback acknowledges the students attempt and appreciates their effort to solve the problem. It also provides clear and specific points for improvement, covering aspects such as comprehensiveness, requirement references, known solutions, clear explanations, examples, academic references, and code implementation. It offers actionable advice on how the student can enhance their work. For instance, they advise the student to elaborate on each property, reference the requirements from the textbook, provide examples, and include code implementations.

By suggesting the inclusion of academic references and exploring well-known solutions to the Dining Philosophers Problem, ChatGPT motivates the student to conduct more research and broaden their understanding of the topic. It also highlights the importance of explaining how the properties of mutual exclusion are essential in the context of the Dining Philosophers Problem. This helps the student understand the practical relevance of the topic.

Instead of naming the suspicion that the student has misunderstood the assignment, ChatGPT clarifies the requirements of the assignment, emphasizing the importance of relating the properties of mutual exclusion to the Dining Philosophers Problem explicitly. This ensures that the student understands what is expected.

By comparing the feedbacks, it is evident that the AI-generated is more constructive and helpful for the student. It offers constructive criticism, actionable steps for improvement, and highlights the aspects that need enhancement to meet the assignments requirements. It encourages the student to dive deeper into the topic, provide examples, use academic references, and better explain their points.

Yes
No
Neutral

Table 5: Submission meet evaluation criteria

Evaluation Criteria	Student 1	Student 2	Student 3
All six requirements from Stallings book must be referenced.			
The requirements must be taken into consideration in relation to solving the Dining Philosophers Problem.			
The student should name and describe at least one of the following solutions: Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, Asymmetric			

Table 6: Feedback quality comparison summary

	Student 1		Student 2		Student 3	
Criteria for good feedback	**Human**	**AI**	**Human**	**AI**	**Human**	**AI**
Acknowledgement						
Guidance						
Constructive						
Specific						
Actionable						
Highlights areas for improvement						

5.4. Summary of Case Studies

After collecting the data used in this case study, it was known that the quality of the human-generated feedback was exceptionally bad and did not comply with the guidelines given by UCN. Considering this, it has been easy to ascertain that the quality of the generated feedback has been significantly better, so the question must be: is it good enough?

In all the cases above, the chatbot has generated feedback for the students that is mostly positive and constructive to foster a supportive learning environment. Also, it has offered specific comments on the students work, pointing out areas that needed their attention and given actionable steps for improvement.

It is evident that the AI-generated feedback is comprehensive and complies with the principles for feedback as stated in section 2. Even in cases where the students submit work, that is unacceptable from an academic point of view, which lecturers across the board reject, the chatbot zealously handles the analysis of their effort and generates useful and constructive feedback. However, in some cases the AI-generated feedback is overly positive, and confuses the indicative solution with the students' work.

6. Operationalized Feedback

Since we cannot trust the model to generate correct feedback, it is necessary to involve the lecturer in the process to validate the generated output. That way, we can be sure it complies to the curriculum as well as the indicative solution and the evaluation criteria for the assignment.

This also means that it is not possible to create a fully automated system where the students upload their submissions and receive the feedback instantly. There must be at least one step in between where the generated feedback will be validated and approved by an expert or a lecturer. However, this does not make it impossible to operationalize the feedback process or make a system designed to streamline the feedback loop by automating certain stages while maintaining expert validation for accuracy and educational quality.

Figure 2: Feedback workflow

Input Stage: The first step in the proposed system is the input stage, where students submit their assignments through an online platform. This platform should be user-friendly and accessible, allowing students to upload various file formats, such as documents, code files or multimedia content. The platform will be a web application where students can upload their documents or enter their responses for the assignments.

Generate Feedback: Upon submission, the system leverages AI technology to generate initial feedback for each student's assignment. This feedback will cover the areas mentioned in the assignment description as well as the evaluation criteria. To do this, the system will call the OpenAI API[8] with the assignment text as input and receive the AI-generated feedback as the output. The API will use the pre-trained language model to analyze the content and provide relevant feedback.

Evaluate Feedback: To maintain the integrity and accuracy of the feedback, the system incorporates a crucial step of expert validation. After the AI generates the feedback, it is sent to a qualified lecturer or subject matter expert who reviews and validates the comments. The expert's role is to ensure that the feedback aligns with the learning objectives, evaluation criteria, and provides actionable insights to help students improve their work.

Correcting Flawed Feedback: Recognizing that AI-generated feedback may not always be flawless, the system should include a mechanism for students and educators to flag and correct any inaccuracies or errors in the feedback. If a student believes that the feedback provided is incorrect or inappropriate, they can submit a request for review. Additionally, the lecturer or subject matter expert who validated the feedback in the previous step may also identify areas where the AI-generated feedback can be improved.

Making Feedback Available for Students: Once the feedback has been duly validated, it is made available to the students via the same online platform used for assignment submission. Each student receives their personalized feedback securely and privately. The feedback should be presented in a clear and understandable format, highlighting both strengths and areas for improvement, thereby fostering a growth-oriented mindset among the students. In this case, nothing is mentioned about the use of AI, and it is also not important from the user's perspective.

6.1. Pipeline

The steps described in the previous section are interconnected stages through which the students' submissions flow in a specific order to achieve feedback for it. Such a construction can be called a *pipeline* which is derived from its real-world counterpart, where fluids, materials, or products flow through a series of connected pipes in a manufacturing or distribution process.

In software development, the pipeline pattern is used to automate and streamline some process of tasks or operations setup in a series of stages. An example is the Azure Pipeline that automates software deployment in stages of integrating the code in the code repository, running unit tests, and deploying the code to various environments such as test, demo, or production[9]. Each stage in the pipeline represents a specific task or operation that takes the output of the previous stage as input and produces its output. This facilitates the efficient and reliable delivery of the outcome.

A pipeline can be implemented in several ways. In this section, we present the design and implementation of a service-oriented architecture for a web-based feedback pipeline, catering to the dynamic needs of generating, evaluating, and delivering assignment feedback to students. The primary objective of this system is to streamline

[8] https://platform.openai.com/
[9] https://learn.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops

the process of generating constructive feedback for assignment submissions while ensuring efficient collaboration between lecturers and students in an academic environment.

To achieve the desired flexibility, scalability, and maintainability for our feedback pipeline, we have chosen a service-oriented architecture (SOA) approach. The SOA paradigm allows us to decompose the system into a set of loosely coupled, independent services, each responsible for specific functionalities. By doing so, we promote modularity and reusability, enabling seamless integration with existing systems and accommodating future expansions with ease.

The web-based feedback pipeline consists of three main steps derived from the steps described above:

Text processing and feedback generation: The pipeline begins with the ingestion of a student's assignment submission as text. The first step in the pipeline is responsible for the text processing and feedback generation. The upload of the submission will be handled through a web-based portal, where the students have a dedicated area for all their submissions that gives them a status overview. The OpenAI API are employed by a web-service to analyze the content and generate constructive feedback. The generated feedback is temporarily stored in a data repository by that service.

Lecturer evaluation and approval: After the feedback generation step, the participation of lecturers is needed. They are provided with a user-interface to review the generated feedback. Lecturers have the option to approve the feedback as is or make necessary edits and improvements. Upon evaluation, the revised feedback is stored in the repository for further processing.

Feedback delivery to students: In the final step of the pipeline, the approved or edited feedback is delivered to the respective students. Students can access the feedback through a secure web-based portal, allowing them to understand their performance better and make necessary improvements.

The entire pipeline will be handled by another service to lower the coupling between the individual steps. That way it is relatively easy to add more steps (e.g., automated unit testing) to the pipeline or replace existing.

6.2. System Description

The system servicing the pipeline is designed as a web-based system, accessible through standard web browsers. This approach ensures that users, including students and lecturers, can conveniently access and interact with the system from any internet-enabled device, eliminating the need for specialized software installations.

The adoption of a service-oriented architecture for the feedback pipeline offers numerous benefits. Firstly, it facilitates enhanced modularity, enabling the individual services to evolve independently, thereby simplifying updates and maintenance. Secondly, it promotes reusability, allowing us to leverage existing services for other future applications within the academic domain. Lastly, the web-based implementation ensures widespread accessibility and ease of use, fostering greater engagement and participation among stakeholders.

In conclusion, the utilization of a service-oriented architecture for our web-based feedback pipeline presents a robust and adaptable solution to provide timely, personalized, and constructive feedback to students, thus fostering a collaborative and supportive learning environment. This approach aligns with the dynamic requirements of modern educational systems and positions our feedback pipeline to effectively cater to the evolving needs of students and educators alike.

6.2.1. Applications and Services

The system is composed of the following applications and services that are illustrated in Figure 3: System Architecture:

1) **Authentication Service**
   The users of the system are lecturers and students and to be able to secure their data and limit their options for action, it is necessary to have a mechanism to authenticate and authorize them, when they log into the system. It is not decisive to implement the service specifically for this system since UCN already offers an authentication service that can be integrated.
2) **Feedback Service**
   The whole purpose of this system is to provide feedback for students' assignment submissions. To generate that we have in this report used OpenAI's ChatGPT to prove whether it can be done or not, and to integrate that into the system, we can call the service equivalent called OpenAI Chat Completions API. Instead of implementing this function directly into the Pipeline Service, we have chosen to implement a separate service to remove the dependency.
   Doing this we can easily change the AI platform in the Feedback Service and if the interface to the Pipeline Service is complied with, this will not be affected.
3) **Pipeline Service**
   This service is running the core functionality of the system by keeping track of the students' assignments, initiate the generation of feedback, make data available to lecturers, handle changes made by lecturers, and make data available to students. We might benefit from splitting this up in more services, each handling a single step in the pipeline so they can be developed independently, but this may be subject to a later project.
4) **Web Application**
   The systems entry point is the web application students and lecturers use for uploading assignment submissions, evaluating the generated feedback, and reviewing said feedback. It updates data through the Pipeline Service and must not be aware of, nor dependent on, how the pipeline works. The exact behaviors of the interfaces are described in the sections below.

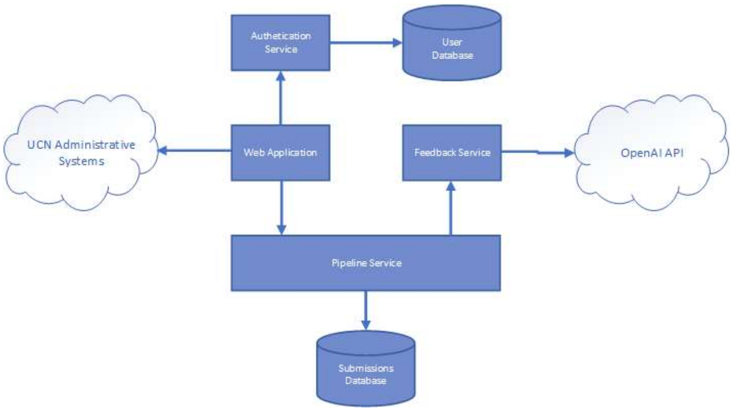

Figure 3: System Architecture

6.2.2. Calling OpenAI API

The OpenAI API allows developers to integrate and interact with OpenAI's language models, like the GPT-3.5 model, in their own applications, products, or services. So, instead of using ChatGPT, which is an application that makes the model available via a web-based user interface, the API enables you to send prompts or text inputs to the model using HTTP and receive generated text outputs as responses.

The API has various use cases, such as generating human-like text, answering questions, creating conversational agents, providing language translation, summarizing content, and more. In short, it opens opportunities for developers to leverage the power of advanced AI language models without having to build the models from scratch.

As mentioned earlier, we wish to create a separate service for calling the OpenAI API with the purpose of generating feedback for a students' assignment. This service will be designed as an API itself and function as an adapter between our pipeline application and the OpenAI API. In this way, our system becomes independent of the AI platform, and we can, in principle, change it later without it affecting the system's functionality.

Architecture and implementation of our API will be trivial, and will not be reviewed here, but calling the OpenAI API is a central concept, no matter which platform for implementation is chosen and a minimally structured HTTP request can look like the one shown in the sample below.

Code Sample 1: The HTTP request to OpenAI API made for student 1s submission to the first case. The prompt has been removed to save space. Also, the API key has been removed for obvious reasons.

```
POST /v1/chat/completions HTTP/1.1
Host: api.openai.com
Content-Type: application/json
Authorization: Bearer <YOUR_API_KEY>
Content-Length: 2791

{
     "model": "gpt-3.5-turbo",
     "messages": [{"role": "user", "content": <INSERT_PROMPT_HERE>}],
     "temperature": 1
}
```

Headers

This is a common HTTP header where information related to the http request can be added. The *Authorization* header is necessary though since it describes the authentication method provided with the request. In the example above a bearer token is used, which means the server is using OAuth 2.0 to protect the endpoint. This method requires a token that here is an API key obtained from OpenAI here:

https://platform.openai.com/account/api-keys

Body

The body of the HTTP POST request contains the payload of the request with various properties that allow us to customize and control the behavior of the language model.

The OpenAI API documentation[10] explains the properties in the data parameter like this:

model: Specifies the language model you want to use. In this case, it's set to "gpt-3.5-turbo", which is the GPT-3.5 Turbo model.

messages: A list of message objects that simulate a conversation between a user and the AI model. Since HTTP is stateless, so is the conversation between the user and the AI which means that the service cannot remember the conversation between requests. So, to simulate a continuous conversation, we need to send the whole list of messages sent and received in every request to the service. This list will thus over time accumulate all messages that have been exchanged.

Each message contains two properties:

role: Represents the role of the message sender. It can be "user" or "assistant" representing the input and responses respectively.

content: The content of the message sent by the sender. For the initial message, we need to replace `<INSERT_PROMPT_HERE>` with the actual text prompt we want to provide to the model. If the lecturer wants to continue the conversation with the model when validating the generated feedback, it is necessary to include the initial prompt as well.

temperature: The *temperature* property controls the randomness of the model's output. Higher values (e.g., 0.8) make the output more diverse and creative, while lower values (e.g., 0.2) make it more focused and deterministic. In the case study this property has not been explicitly set, so the default value of 1 has been used which means the output is very creative and can vary a lot if we send the same initial message more than once.

The choice of temperature depends on the use case and the desired tone of the generated text. If we are aiming for more controlled and coherent responses, we should use a lower temperature. If we are looking for more creativity and diversity, we could consider using a higher temperature. In Appendix D we have conducted a small experiment with different temperature values generating feedback, which shows a lower value for temperature might be preferrable for the system.

The properties listed above are not exhaustive, and there are other advanced options and parameters available in the API to further customize the behavior of the language model.

Code Sample 2: Response from OpanAI API

```
{
    "id": "chatcmpl-7kUo43Ff7M4Y2gB6EOq9lqBeW8cE8",
    "object": "chat.completion",
    "created": 1691315864,
    "model": "gpt-3.5-turbo-0613",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": <RESPONSE_TEXT>
            },
            "finish_reason": "stop"
```

[10] https://platform.openai.com/docs/api-reference

```
        }
    ],
    "usage": {
        "prompt_tokens": 515,
        "completion_tokens": 416,
        "total_tokens": 931
    }
}
```


6.2.3. Student User Interface

Student users of the system have three actions they can perform on the system when they are logged in. They are:

1) Upload assignment submission
2) Read status of feedback progress
3) Review feedback for assignments

Upon logging in, students will be greeted with a dashboard or homepage that provides an overview of their assignment submissions and feedback progress. The dashboard should prominently display the key actions they can take, such as uploading assignments, accessing feedback, and checking progress.

There should be a dedicated section or button on the dashboard to allow students to upload their assignment submissions. When students click on this option, they can browse their computer or device to select the relevant file and submit it.

The dashboard should include a section where students can track the status of their assignment feedback. This section might include a list of submitted assignments with details on their current feedback status. For example, it could show whether the feedback is pending, in progress, or complete.

Another section on the dashboard should provide students with access to their received feedback. Students can click on a specific assignment to view the feedback provided by the instructor. The feedback will be presented as text.

Other features could be provided as well such as: A notification system to inform students about updates regarding their submissions, historic information about all previous feedback received, or support for mobile platforms. However, the first step is to create a barebone prototype that can be used for testing the concept, so these features can be added in later development.

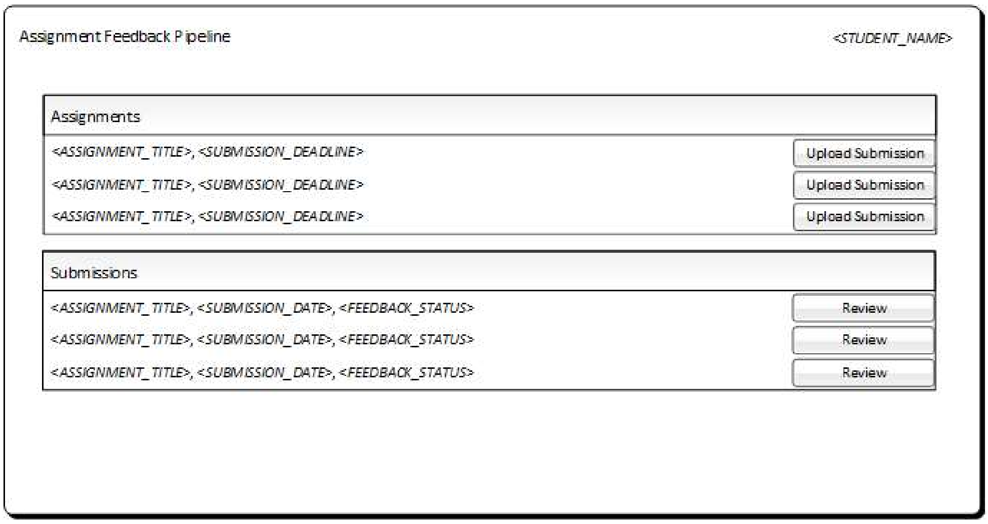

Figure 4: Students' user interface

FEEDBACK_STATUS indicates if the feedback for the submission is: *pending*, *in progress*, or *ready*.

6.2.4. Lecturers User Interface

The lecturers' role is to evaluate and validate the generated feedback in the system before the students can see it. They need to see a list of the assignment submissions they relate to and their status in the system. The submission will pop-up when they are uploaded to the system and as soon as the feedback is generated, it will change state, so it is easy to identity the feedback that needs attention. The lecturer can then open the feedback, make any corrections, and mark the feedback as approved for release back to the student. When the student has opened the feedback for review, that information will also be available.

Before this is possible, the interface must have an administrative screen where assignments can be created, and students attached to them. Information about students and classes will be extracted from UCNs administrative system and therefore be available for the lecturers.

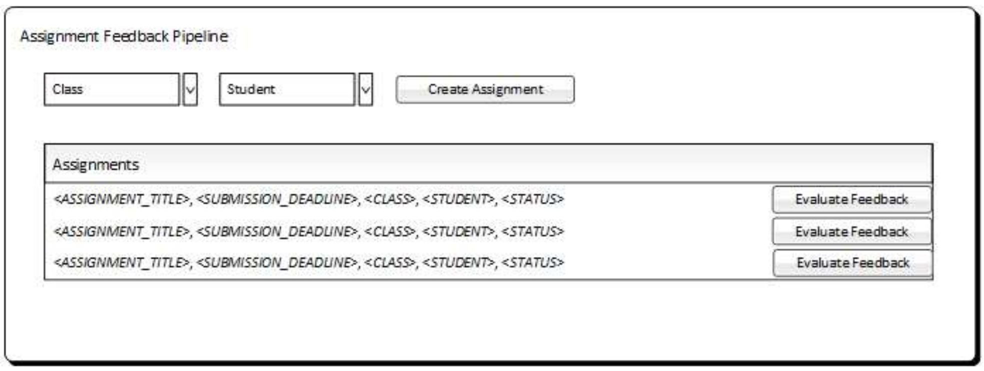

Figure 5: Lecturers' administrative user interface

In the wireframe shown in Figure 5, the lecturers can create and add assignments for classes or individual students. In the list it is possible to see current assignments waiting for students' submissions and their status in the pipeline. When they are submitted and feedback is generated, the lecturer presses the **Evaluate Feedback** button to open the screen shown in Figure 6. Here the lecturer evaluates and approves the generated feedback compared to the assignment and the student's submission. It is also possible to correct the feedback if necessary.

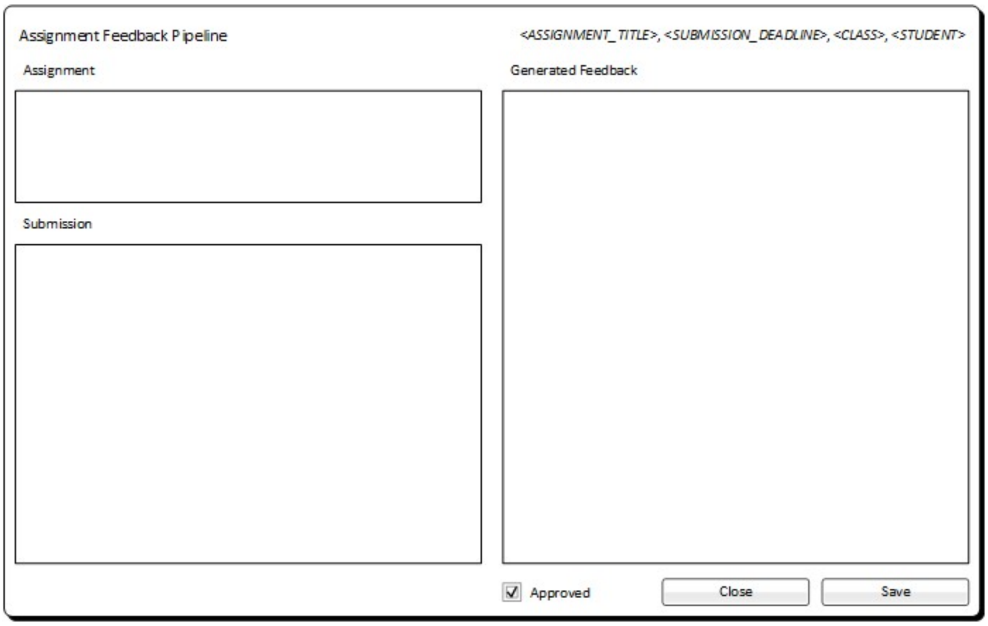

Figure 6: Lecturers' user interface for correcting and approving the generated feedback.

7. Conclusion

The conclusion summarizes the key findings and discussing the implications of using AI, specifically ChatGPT, as a tool for generating feedback on student hand-ins in applied computer science education. The case study provides insights into the potential benefits and challenges associated with integrating AI into the feedback generation process, and it highlights directions for future research and improvement.

To answer the problem definition:

> *What is needed to let a chatbot that adopts a generative large pretrained language model such as ChatGPT generate feedback for a student's suggested solution to an assignment?*

In the preparation of the case study, it was decided to include educational context, the assignment wording, description of formal expectations, evaluation criteria, and the indicative solution as input to the prompt. The indicative solution was particularly a subject of consideration but was added because we could not trust ChatGPT to be able to solve the assignment correctly on its own. Furthermore, the indicative solution was already available in the source material, so it did not require additional work. With these parameters it was possible to generate feedback for the individual submissions. The following discussion of this clarified that the generated feedback will be useful from a lecturer's point of view. Whether the students think the same is yet to be investigated.

> *Are any specific kind of assignment better suited for generating feedback?*

There are no indications of difference in the quality of the generated feedback between the selected assignment types in this case study. The feedback generated for the formal essay case was all for submissions that had failed of various reasons, so it might be necessary to run more cases with better submissions, to obtain a more realistic picture of this. However, in the other cases, where the quality of the submissions where more diverse, ChatGPT generally generated good quality feedback, that complies with the principles stated in section 2. In some cases, though, the generated feedback failed in addressing important errors and generated feedback for parts of the assignment that was not answered in the submission, which supports an assumption that human validation is necessary.

How can the process of giving feedback to students be operationalized?

Creating a system with the purpose of operationalize the process of giving feedback to students is not bound by using ChatGPT or any other tool for generating it. Because of the unreliability of the models, we know that it is necessary to have a lecturer evaluating the generated feedback before the students receive it, so the process can in all circumstances not be entirely automated, and the involvement of an AI is therefore transparent for the students.

To operationalize the feedback generation process, a pipeline that leverages the language model with the OpenAI API can be developed. The pipeline must take an assignment submission as input and produces a feedback report that is available for the student when the generated feedback is validated and approved by the lecturer. This will most likely save much time and give better more personalized feedback for each student, which is the main goal of this project and is as such an acceptable solution.

Finally, to comply with the didactical and pedagogical methods we are subject to at UCN, written feedback for students' submission cannot stand alone and must be followed up in the dialogue with the student.

References

[1] *Curriculum for AP in Computer Science,* Aalborg: UCN, 2019.

[2] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller og M. Simens, »Training language models to follow instructions with human feedback,« 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2203.02155. [Senest hentet eller vist den July 2023].

[3] OpenAI, »Custom instructions for ChatGPT,« 20 July 2023. [Online]. Available: https://openai.com/blog/custom-instructions-for-chatgpt. [Senest hentet eller vist den July 2023].

[4] L. H. Horn, C. G. Jensen, T. Kjærgaard, N. B. Lukassen, I. M. Sørensen, C. Valbak-Andersen og S. B. Bundgaard, »Hvidbog om Refleksiv Praksislæring,« 2020. [Online]. Available: https://blad.ucn.dk/hvidbog-om-rpl/. [Senest hentet eller vist den July 2023].

[5] N. B. Lukassen, »Feedback og innovativ tænkning,« i *Teori og praksis i professionsdidaktik*, Aalborg Universitetsforlag, 2021, pp. 127-136.

[6] D. Baidoo-Anu og L. Owusu Ansah, »Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning,« *Available at SSRN 4337484,* 2023.

[7] E. Kasneci, K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann og E. Hüllermeier, »ChatGPT for good? On opportunities and challenges of large language models for education,« *Learning and Individual Differences,* årg. 103, p. 102274, 2023.

[8] A. Krogh, »What are Artificial Neural Networks,« *Nature Biotechnology,* årg. 26, nr. 2, pp. 195-197, 2008.

[9] M. Nielsen, »Neural Networks and Deep Learning,« 2019. [Online]. Available: http://neuralnetworksanddeeplearning.com/. [Senest hentet eller vist den July 2023].

[10] OpenAI, »GPT-4 Technical Report,« 2023. [Online]. Available: https://arxiv.org/abs/2303.08774. [Senest hentet eller vist den July 2023].

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser og I. Polosukhin, »Attention is all you need,« 2017. [Online]. Available: https://arxiv.org/abs/1706.03762v5. [Senest hentet eller vist den 07 2023].

[12] OpenAI, »Pioneering research on the path to AGI,« OpenAI, 2023. [Online]. Available: https://openai.com/research. [Senest hentet eller vist den July 2023].

[13] P. Schramowski, C. Turan, N. Andersen, C. A. Rothkopf og K. Kersting, »Large pre-trained language models contain human-like biases of what is right and wrong to do,« *Nature Machine Intelligence,* årg. 4, nr. 3, pp. 258-268, 2022.

[14] J. Devlin, M.-W. Chang, K. Lee og K. Toutanova, »BERT: Pre-training of deep bidirectional transformers for language understanding,« *arXiv preprint arXiv:1810.04805,* 2018.

[15] P. L. a. W. Y. Neubig, J. Fu, Z. Jiang, H. Hayashi og G. Neubig, »Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing,« 28 July 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2107.13586.

[16] j. white, q. fu, g. s. hays, m. sandborn, c. olea og h. gilbert, »A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT,« *arXiv preprint arXiv:2302.11382,* 2023.

[17] L. Rienecker og P. S. Jørgensen, Den Gode Opgave, Samfundslitteratur, 2022.

[18] T. A. Schwandt og E. F. Gates, »Case Study Methodology,« i *The SAGE Handbook of Qualitative Research*, 5th red., Los Angeles, SAGE, 2018, p. Chapter 14.

Appendix A: Students Answers

Case 1

Student 1	New: During this phase, an operation is initiated. Resources such as memory and input/output devices are assigned, and the operation is registered in the operation control structure (OCS) and given an operation ID (OID). However, it has not yet been transferred to the primary memory. Ready: After initialization, the operation is transferred to the primary memory and is set for execution. It resides in the prepared line until the organizing system grants it access to the CPu for execution. In the prepared state, the operation possesses all necessary resources and is merely awaiting its opportunity to execute. Running: The CPU is actively executin the operation at this stage. It is the active operation, and its commands are being performed by the processing unit. To be in the executing state, an operation must have been allocated to a CPU. Blocked/waiting: In this phase, the operation is on hold for an external occurrence or resource to become accessible, such as user interaction or the conclusion of an I/O task. While paused, the operation is unable to proceed with its execution, even if the CPU is available. The operation returns to the prepared state once the necessary resource or event becomes available. Exit/Terminate: The operation's execution is finished, and it is no longer required. In this phase, the computer's operating system deallocates resources and memory assigned to the operation, and the operation is removed from the operation control structure. The operation ceases to exist following this stage.
Student 2	The 5 states in the five-state process model is: New - refers to a new process having been created Ready - refers to the process having been loaded into the main memory and is ready to run Running - refers to the process being executed Blocked - refers to the process quitting the CPU, entering a waiting state, until the CPU is ready to run the process again, in that case the process is moved from the blocked stage, to the ready stage and then the running stage . Exit - refers to the process having been terminated and removed from main memory
Student 3	New: when a process is first accepted into a query. Ready: ready to run Running: when a process is actively executing Blocked: a process is blocked when the process is blocked by the CPU. From here the process can move back to the ready state when either the CPu becomes available again or the ressource the process has been waiting for is available.

Case 2

<table>
<tr><td>Student 1</td><td>The singleton pattern is a creational design pattern, that makes sure that a class can only have 1 instance in the lifetime of the application. It provides a global point off acces to the class instance.

<pre>
public class ProductContainer {

   private static ProductContainer instance;

   public   static   ProductContainer
getInstance() {
      if (instance == null) {
         instance = new ProductContainer();
      }
      return instance;
   }
</pre>

This example is from a 1st semester project, and the singleton pattern is used for the container classes. That way the corresponding controller class and call the getInstance() method and make a single instance of the container.</td></tr>
<tr><td>Student 2</td><td>When implementing the singleton pattern, you ensure that only one instance of the singleton class will exist at a time and is accessed in a static way. Is often used in container classes when building a smaller system where a database isn't needed.</td></tr>
<tr><td>Student 3</td><td>The singleton pattern is used to ensure that only one instance of a given class can be initiated. This is accomplished by having private constructors and a private static method that returns a reference to the object or initiates a new object if an instance has not yet been initiated.

Example:

<pre>
public class Singleton{

    private static Singleton instance;

    private static Singleton getInstance(){
        if(instance == null){
            instance = new Singleton();
        }
        return instance;
    }
}
</pre></td></tr>
</table>

Case 3

Student 1	Mutal exclusion is an important concept in concurrent programming, where multiple processes or threads need to access shared resources without causing conflicts or race conditions. Key requirements are: 1. Mutual Exclusion: Only one process accesses a shared resource at a time. 2. Progress: The next process to enter the critical section isn't indefinitely postponed. 3. Bounded Waiting: No process is perpetually denied access, preventing starvation. 4. No Speed Assumptions: The solution works regardless of execution speed and process count.
Student 2	There is four requirement for mutual exclusion: 1. No process more than one process can be in their respective critical section at any time 2. There should be no assumptions to the relative speed of conflicting processes. 3. Processes outside their critical section should not interrupt another process access to the critical section. 4. When more than one process wants to enter its critical section, it must be granted entry in a finite time, so that it will never be kept waiting in a infinite loop.
Student 3	No deadlocks No starvation Fairness Fault tolerance

Appendix B: Generated Feedback from ChatGPT

Case 1

Student 1	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions. This is a review question that the student answer by handing in a short description (max. 1200 characters including spaces). The review question is: Briefly explain each state in the Five-State Process model. Here is the indicative solution: Running: The process that is currently being executed. Ready: A process that is prepared to execute when given the opportunity. Blocked: A process that cannot execute until some event occurs, such as the completion of an I/O operation. New: A process that has just been created but has not yet been admitted to the pool of executable processes by the operating system. Exit: A process that has been released from the pool of executable processes by the operating system, either because it halted or because it aborted for some reason. The evaluation criteria are: All five states described in the book should be included and described correctly in the students answer. Here is the students submission: New: During this phase, an operation is initiated. Resources such as memory and input/output devices are assigned, and the operation is registered in the operation control structure (OCS) and given an operation ID (OID). However, it has not yet been transferred to the primary memory. Ready: After initialization, the operation is transferred to the primary memory and is set for execution. It resides in the prepared line until the organizing system grants it access to the CPu for execution. In the prepared state, the operation possesses all necessary resources and is merely awaiting its opportunity to execute. Running: The CPU is actively executin the operation at this stage. It is the active operation, and its commands are being performed by the processing unit. To be in the executing state, an operation must have been allocated to a CPU. Blocked/waiting: In this phase, the operation is on hold for an external occurrence or resource to become accessible, such as user interaction or the conclusion of an I/O task. While paused, the operation is unable to proceed with its execution, even if the CPU is available. The operation returns to the prepared state once the necessary resource or event becomes available. Exit/Terminate: The operation's execution is finished, and it is no longer required. In this phase, the computer's operating system deallocates resources and memory assigned to the operation, and the operation is removed from the operation control structure. The operation ceases to exist following this stage.
Generated Feedback	Overall, the student's submission covers the required states of the Five-State Process model, but there are some areas for improvement. The feedback should highlight the strengths and provide constructive suggestions to improve the response. Here's a detailed feedback: Strengths: 1. The student's submission demonstrates an understanding of the Five-State Process model by describing the different states that a process can be in during its lifecycle. 2. The descriptions provided by the student show an effort to explain each state in their own words, which indicates some level of comprehension of the topic. Areas for Improvement:

	1. Length: The student's submission exceeds the maximum character limit of 1200 characters. Encourage the student to be more concise and focus on the essential aspects of each state to meet the assignment's requirements. 2. Accuracy: While the student attempted to explain the states, some descriptions lack accuracy and precision. Suggest that they review the model's definitions and provide more accurate descriptions. 3. Terminology: The student uses some technical terms that are not part of the Five-State Process model. Advise them to stick to the terminology used in the indicative solution to ensure consistency and accuracy. 4. Completeness: The student's submission lacks a clear mention of all the required states (New, Ready, Running, Blocked, and Exit). Suggest that they revise their answer and ensure that each state is explicitly described. 5. Focus: Some parts of the student's response focus on additional details not directly related to the Five-State Process model, such as memory allocation and operation control structure. Encourage them to stay focused on the states and their meanings in the context of the model. Revised Suggestion: New: The process is newly created and awaiting admission to the pool of executable processes by the operating system. It has been assigned necessary resources, but it is not yet in the primary memory. Ready: The process is in the prepared queue and can be executed if granted access to the CPU by the operating system. It has all the required resources and is waiting for its turn to run. Running: The process is actively executing on the CPU. It is the currently active process, and its commands are being processed by the CPU. Blocked: The process is on hold, waiting for an external event or resource to become available, such as user input or the completion of an I/O operation. While blocked, the process cannot proceed with its execution. Exit: The process has completed its execution and is released from the pool of executable processes. The operating system deallocates resources and memory assigned to the process, and it is removed from the system. By providing this constructive feedback, you can help the student improve their understanding and explanation of the Five-State Process model. Encourage them to revise their response and focus on being clear, accurate, and concise while incorporating all the required states.
Student 2	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions. This is a review question that the student answer by handing in a short description (max. 1200 characters including spaces). The review question is: Briefly explain each state in the Five-State Process model. Here is the indicative solution: Running: The process that is currently being executed. Ready: A process that is prepared to execute when given the opportunity. Blocked: A process that cannot execute until some event occurs, such as the completion of an I/O operation. New: A process that has just been created but has not yet been admitted to the pool of executable processes by the operating system. Exit: A process that has been released from the pool of executable processes by the operating system, either because it halted or because it aborted for some reason. The evaluation criteria are: All five states described in the book should be included and described correctly in the students answer. Here is the students submission: The 5 states in the five-state process model is: New - refers to a new process having been created

	Ready - refers to the process having been loaded into the main memory and is ready to run Running - refers to the process being executed Blocked - refers to the process quitting the CPU, entering a waiting state, until the CPU is ready to run the process again, in that case the process is moved from the blocked stage, to the ready stage and then the running stage . Exit - refers to the process having been terminated and removed from main memory
Generated Feedback	Overall, the student's submission covers all five states of the Five-State Process model, but there are some inaccuracies and inconsistencies in their descriptions. Here's the constructive feedback for the student: 1. **New**: The student correctly states that it refers to a new process that has been created. However, they should also mention that it has not yet been admitted to the pool of executable processes by the operating system. 2. **Ready**: The student mentions that the process is loaded into main memory and ready to run. It's mostly correct, but they should clarify that the process is prepared to execute when given the CPU time. 3. **Running**: The student's definition is accurate, as it refers to the process currently being executed. 4. **Blocked**: The student describes the blocking situation reasonably well but incorrectly mentions that the process is moved from blocked to ready and then running stages. In the Five-State Process model, a blocked process transitions back to the ready state when the event it was waiting for (e.g., completion of I/O) occurs, not directly to the running state. 5. **Exit**: The student correctly defines the exit state as the process being terminated and removed from main memory. Based on these points, the student's submission is mostly correct but lacks some clarity and precision in the descriptions. Here's a revised and concise version of their submission: "The Five-State Process model consists of the following states: 1. New: A newly created process waiting for admission to the pool of executable processes by the OS. 2. Ready: Process loaded into main memory and prepared to execute when given CPU time. 3. Running: The process currently being executed. 4. Blocked: A process waiting for an event (e.g., I/O completion) before transitioning back to the ready state. 5. Exit: A process that has been terminated and removed from main memory." With this feedback, the student's submission should be improved and meet the evaluation criteria for accuracy and completeness.
Student 3	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions. This is a review question that the student answer by handing in a short description (max. 1200 characters including spaces). The review question is: Briefly explain each state in the Five-State Process model. Here is the indicative solution: Running: The process that is currently being executed. Ready: A process that is prepared to execute when given the opportunity. Blocked: A process that cannot execute until some event occurs, such as the completion of an I/O operation. New: A process that has just been created but has not yet been admitted to the pool of executable processes by the operating system. Exit: A process that has been released from the pool of executable processes by the operating system, either because it halted or because it aborted for some reason. The evaluation criteria are:

	All five states described in the book should be included and described correctly in the students answer. Here is the students submission: New: when a process is first accepted into a query. Ready: ready to run Running: when a process is actively executing Blocked: a process is blocked when the process is blocked by the CPU. From here the process can move back to the ready state when either the CPu becomes available again or the ressource the process has been waiting for is available.
Generated Feedback	Overall, the student's submission shows a basic understanding of the Five-State Process model, but there are some inaccuracies and areas for improvement. Let's provide constructive feedback on their submission: 1. New: The student correctly mentions that it is when a process is first accepted into a queue. However, it's important to clarify that the queue they are referring to is the pool of executable processes managed by the operating system. 2. Ready: The student's explanation is brief but acceptable. However, to provide a more complete understanding, they should mention that the "Ready" state represents a process that is waiting to be scheduled by the operating system to run on the CPU. 3. Running: The student's explanation is quite concise but accurate. However, it can be improved by mentioning that a process is actively executing on the CPU during this state. 4. Blocked: The student's explanation contains some inaccuracies. They mention that a process is blocked when it is blocked by the CPU, which is not entirely accurate. Instead, the "Blocked" state occurs when a process cannot proceed because it is waiting for an external event, such as the completion of an I/O operation or the availability of a resource. 5. Exit: The student's explanation is acceptable as it correctly mentions that a process in the "Exit" state has been released from the pool of executable processes by the operating system, either because it halted or aborted. Based on the evaluation criteria, the student's submission meets the requirement of including all five states of the Five-State Process model, but there are some minor inaccuracies and lack of elaboration in some states. Suggested improved submission: - New: A process enters the "New" state when it is first created but has not yet been admitted to the pool of executable processes by the operating system. It waits in a queue until the OS allocates resources and prepares it for execution. - Ready: The "Ready" state represents a process that is waiting to be scheduled by the operating system to run on the CPU. It has all the required resources and is prepared for execution. - Running: During the "Running" state, a process is actively executing on the CPU. It is the current state of a process when it is being processed by the CPU. - Blocked: A process enters the "Blocked" state when it cannot proceed because it is waiting for an external event, such as the completion of an I/O operation or the availability of a resource. Once the event occurs, the process may move back to the "Ready" state. - Exit: The "Exit" state is reached when a process has completed its execution or has been terminated for some reason. The operating system releases the process from the pool of executable processes. By providing this feedback, the student can gain a clearer understanding of the Five-State Process model and improve their knowledge of the subject matter.

Case 2

Student 1	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions.

	This is a programming exercise where the students must demonstrate that they are able to implement a specific programming pattern and explain how it works. The solution must be handed in as a code-snippet where the explanation can be embedded in the code as a comment.
	
	The assignment is:
	Explain the singleton design pattern and give an example implementation.
	
	Here is the indicative solution:
	Description: The Singleton pattern is a creational software design pattern that restricts the instantiation of a class to a singular instance. This can be done by making the constructor private and create a public static method in the class that creates and returns an instance once and subsequently returns that instance.
	
	Implementation:
	public class Singleton {
	    private static Singleton instance;
	
	    private Singleton() {
	        System.out.println("Creating new instance");
	    }
	
	    public static Singleton getInstance() {
	        if (instance == null) {
	            instance = new Singleton();
	        }
	        return instance;
	    }
	}
	
	public class Main {
	    public static void main(String[] args) {
	        // Test 1: Create an object
	        Singleton obj1 = Singleton.getInstance();
	
	        // Test 2: Try to create a second object, the first one should be returned
	        Singleton obj2 = Singleton .getInstance();
	
	        // Test 3: Ensure both references point to the same object
	        System.out.println(obj1 == obj2); // This should print "true"
	
	        // Test 4: Attempt to call the class constructor directly (should not be possible)
	        // Singleton obj3 = new Singleton (); // Uncommenting this line should result in a compile-time error
	    }
	}
	
	The evaluation criteria are:
	The purpose and the functionality of the singleton pattern must be described in a few sentences.
	The sample implementation should ensure that only one instance of an object can be created, by running tests that:

	1. Creates an object.
	2. When trying to create a second object, the first one is returned.
	3. It must not be possible to call the class constructor directly.
	
	Here is the students submission:
	The singleton pattern is a creational design pattern, that makes sure that a class can only have 1 instance in the lifetime of the application. It provides a global point off acces to the class instance.
	
	public class ProductContainer {
	
	    private static ProductContainer instance;
	
	    public static ProductContainer getInstance() {
	        if (instance == null) {
	            instance = new ProductContainer();
	        }
	        return instance;
	    }
	
	This example is from a 1st semester project, and the singleton pattern is used for the container classes. That way the corresponding controller class and call the getInstance() method and make a single instance of the container.

Generated Feedback	Overall, the student's submission provides a reasonable understanding of the Singleton design pattern and demonstrates a basic implementation. However, there are some areas where the submission can be improved. Let's provide constructive feedback based on the evaluation criteria:
	
	1. Purpose and functionality of the Singleton pattern:
	The student's description is mostly accurate, but it lacks a few essential details. Here's how it can be improved:
	
	Feedback:
	"The student correctly describes the Singleton pattern as a creational design pattern that ensures a class has only one instance throughout the application's lifetime. However, the description could be enhanced by mentioning the importance of making the class constructor private to prevent direct instantiation and relying on a static method (like getInstance()) to control access to the single instance. Additionally, it would be helpful to mention that the Singleton pattern is commonly used to provide a global point of access to this instance."
	
	2. Sample Implementation Evaluation:
	The student's implementation seems to fulfill the criteria, but there are a couple of minor improvements needed:
	
	Feedback:
	- Test 1 and 2: Creating an object and ensuring only one instance is returned.
	  The student's implementation appears to be correct as it follows the Singleton pattern by using a private constructor and a static method (getInstance) to return the instance. However, it is essential to ensure that the object is instantiated lazily (only when needed) to optimize resource utilization. The current implementation uses lazy initialization, which is good.
	
	- Test 3: Ensuring both references point to the same object.
	  The student did not include a test to validate that both obj1 and obj2 point to the same object. It's essential to verify this aspect of the Singleton pattern.

<table>
<tr><td></td><td>
- Test 4: Ensuring the class constructor cannot be called directly.
The student's implementation is correct in this aspect as it is not possible to call the constructor directly due to its private access modifier. The commented-out code attempting to create a new instance should remain commented to avoid compilation errors.

3. General Feedback:
The student's submission is concise and provides a relevant example for the Singleton pattern, using a ProductContainer class. However, the comment in the code is missing, where the explanation of the pattern should be embedded. The comment should briefly describe the purpose and functionality of the Singleton pattern as discussed earlier.

To summarize the feedback:

1. Enhance the purpose and functionality description of the Singleton pattern by adding details about private constructor and global point of access.
2. Include a test to ensure that both obj1 and obj2 point to the same object.
3. Keep the commented-out code to demonstrate that direct instantiation is not allowed.
4. Add a comment to embed the explanation of the Singleton pattern within the code snippet.

After addressing these points, the student's submission will be well-rounded and demonstrate a better understanding of the Singleton pattern and its implementation.
</td></tr>
<tr><td colspan="2">Student 2</td></tr>
<tr><td>Prompt</td><td>
I am a lecturer at a university college, teaching computer science.
I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions.
This is a programming exercise where the students must demonstrate that they are able to implement a specific programming pattern and explain how it works. The solution must be handed in as a code-snippet where the explanation can be embedded in the code as a comment.

The assignment is:
Explain the singleton design pattern and give an example implementation.

Here is the indicative solution:
Description: The Singleton pattern is a creational software design pattern that restricts the instantiation of a class to a singular instance. This can be done by making the constructor private and create a public static method in the class that creates and returns an instance once and subsequently returns that instance.

Implementation:
<pre>
public class Singleton {
    private static Singleton instance;

    private Singleton() {
        System.out.println("Creating new instance");
    }

    public static Singleton getInstance() {
        if (instance == null) {
            instance = new Singleton();
        }
</pre>
</td></tr>
</table>

<table>
<tr><td></td><td>

```
        return instance;
    }
}

public class Main {
    public static void main(String[] args) {
        // Test 1: Create an object
        Singleton obj1 = Singleton.getInstance();

        // Test 2: Try to create a second object, the first one should be returned
        Singleton obj2 = Singleton .getInstance();

        // Test 3: Ensure both references point to the same object
        System.out.println(obj1 == obj2); // This should print "true"

        // Test 4: Attempt to call the class constructor directly (should not be possible)
        // Singleton obj3 = new Singleton (); // Uncommenting this line should result in a
compile-time error
    }
}
```

The evaluation criteria are:
The purpose and the functionality of the singleton pattern must be described in a few sentences.
The sample implementation should ensure that only one instance of an object can be created, by running tests that:
1. Creates an object.
2. When trying to create a second object, the first one is returned.
3. It must not be possible to call the class constructor directly.

Here is the students submission:
When implementing the singleton pattern, you ensure that only one instance of the singleton class will exist at a time and is accessed in a static way. Is often used in container classes when building a smaller system where a database isn't needed.
</td></tr>
<tr><td>Generated Feedback</td><td>
Overall, the student's submission touches upon the main points of the Singleton pattern but lacks clarity and some specific details. Let's provide constructive feedback on the student's submission and evaluate it based on the evaluation criteria.

1. **Purpose and Functionality of Singleton Pattern**:
The student's explanation is correct in stating that the Singleton pattern ensures only one instance of the class exists at a time and is accessed statically. However, it can be improved by adding a bit more clarity and precision. The Singleton pattern is typically used when you want to control the instantiation of a class to a single object, ensuring that all references to the class refer to the same instance. This is useful in scenarios where multiple instances of the class may lead to conflicts or unnecessary resource consumption. Additionally, the student mentioned that it is used in container classes, but this is not a universal use case, so it's better to avoid such specificity.

Feedback:
- You've correctly stated that the Singleton pattern ensures only one instance of the class exists and is accessed statically. However, it would be better to mention that this pattern is used when you want to control the instantiation of a class to a single object, ensuring that all references to the class refer to the same instance. Avoid being too specific about its use in container classes as it has broader applications.
</td></tr>
</table>

	2. **Sample Implementation and Tests**: The student's implementation is mostly correct and achieves the desired functionality. It correctly defines a private static instance variable, a private constructor, and a public static method to retrieve the instance. The tests provided are relevant and validate that: - Creating an object using `getInstance()` works. - When trying to create a second object, the first one is returned, demonstrating the Singleton behavior. - Attempting to call the class constructor directly should not be possible, which is correctly enforced by the private constructor. Feedback: - Your implementation of the Singleton pattern is on the right track. You have declared a private static instance variable and a private constructor, ensuring that only one instance can exist. Your `getInstance()` method correctly handles the instantiation of the class and ensures that only one instance is created and returned. - Your tests are relevant and demonstrate the Singleton behavior effectively. You create an object using `getInstance()` in the first test and show that when attempting to create a second object, the same instance is returned. The third test checks if both references point to the same object, which is a crucial property of the Singleton pattern. The commented-out code attempting to call the constructor directly shows that the private constructor prevents direct instantiation, which is good. In summary, your submission is on the right track, and your implementation of the Singleton pattern is mostly correct. However, for future assignments, try to provide a more detailed and precise explanation of the pattern's purpose and functionality. Also, remember to avoid being overly specific about use cases unless explicitly asked for. Keep up the good work!
Student 3	

Prompt	I am a lecturer at a university college, teaching computer science.
	I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions.
	This is a programming exercise where the students must demonstrate that they are able to implement a specific programming pattern and explain how it works. The solution must be handed in as a code-snippet where the explanation can be embedded in the code as a comment.
	
	The assignment is:
	Explain the singleton design pattern and give an example implementation.
	
	Here is the indicative solution:
	Description: The Singleton pattern is a creational software design pattern that restricts the instantiation of a class to a singular instance. This can be done by making the constructor private and create a public static method in the class that creates and returns an instance once and subsequently returns that instance.
	
	Implementation:
	public class Singleton {
	    private static Singleton instance;
	
	    private Singleton() {
	        System.out.println("Creating new instance");
	    }
	
	    public static Singleton getInstance() {

	``
	        if (instance == null) {
	            instance = new Singleton();
	        }
	        return instance;
	    }
	}
	
	public class Main {
	    public static void main(String[] args) {
	        // Test 1: Create an object
	        Singleton obj1 = Singleton.getInstance();
	
	        // Test 2: Try to create a second object, the first one should be returned
	        Singleton obj2 = Singleton .getInstance();
	
	        // Test 3: Ensure both references point to the same object
	        System.out.println(obj1 == obj2); // This should print "true"
	
	        // Test 4: Attempt to call the class constructor directly (should not be possible)
	        // Singleton obj3 = new Singleton (); // Uncommenting this line should result in a compile-time error
	    }
	}
	``
	
	The evaluation criteria are:
	The purpose and the functionality of the singleton pattern must be described in a few sentences.
	The sample implementation should ensure that only one instance of an object can be created, by running tests that:
	1. Creates an object.
	2. When trying to create a second object, the first one is returned.
	3. It must not be possible to call the class constructor directly.
	
	Here is the students submission:
	The singleton pattern is used to ensure that only one instance of a given class can be initiated. This is accomplished by having private constructors and a private static method that returns a reference to the object or initiates a new object if an instance has not yet been initiated.
	Example:
	``
	public class Singleton{
	
	    private static Singleton instance;
	
	    private static Singleton getInstance(){
	        if(instance == null){
	            instance = new Singleton();
	        }
	        return instance;
	    }
	}
	``

Generated Feedback	Overall, the student's submission demonstrates a basic understanding of the Singleton design pattern. However, there are some areas that could be improved, and some important evaluation criteria are not fully met. Below is the constructive feedback for the student:
	
	1. Purpose and functionality of the Singleton pattern:
	The student's description is on the right track but lacks a few important points. They should describe that the pattern ensures that a class has only one instance and provides a global access point to that instance. Additionally, it should mention that the constructor is made private to prevent direct instantiation and that the pattern is often used to manage shared resources and configurations.
	
	Modified Description:
	"The Singleton pattern is a creational design pattern that guarantees a class has only one instance and provides a global access point to that instance. This is achieved by making the constructor private and creating a static method that returns the single instance if it exists or creates a new instance if none has been created yet. It is commonly used to manage shared resources, such as configuration settings and database connections."
	
	2. Implementation:
	The student's implementation is almost correct, but there is a small issue. The constructor should be private, not the \`getInstance()\` method. Also, the student should add comments to explain each step of the implementation, including the purpose of the \`getInstance()\` method.
	
	Modified Implementation:
	\`\`\`java
	public class Singleton {
	    private static Singleton instance;
	
	    private Singleton() {
	        System.out.println("Creating new instance");
	    }
	
	    public static Singleton getInstance() {
	        if (instance == null) {
	            instance = new Singleton();
	        }
	        return instance;
	    }
	}
	\`\`\`
	
	3. Test Cases:
	The student has not provided test cases as part of the submission, which were specified in the evaluation criteria. They should add the test cases to the Main class as demonstrated in the indicative solution. The test cases should cover the following scenarios:
	
	a. Test 1: Create an object using \`Singleton.getInstance()\`.
	b. Test 2: Try to create a second object using \`Singleton.getInstance()\`, and check if it returns the same instance as Test 1.
	c. Test 3: Ensure that both references (obj1 and obj2) point to the same object.
	d. Test 4: Attempt to call the class constructor directly (should result in a compile-time error).
	
	Modified Main class with Test Cases:

	``java
	public class Main {
	public static void main(String[] args) {
	// Test 1: Create an object
	Singleton obj1 = Singleton.getInstance();
	
	// Test 2: Try to create a second object, the first one should be returned
	Singleton obj2 = Singleton.getInstance();
	
	// Test 3: Ensure both references point to the same object
	System.out.println(obj1 == obj2); // This should print "true"
	
	// Test 4: Attempt to call the class constructor directly (should not be possible)
	// Singleton obj3 = new Singleton(); // Uncommenting this line should result in a compile-time error
	}
	}
	``
	
	After addressing these points, the student's submission will provide a more complete and accurate understanding of the Singleton pattern and its implementation.

Case 3

Student 1	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions. In this type of assignment, the students must demonstrate a deeper understanding of a specific topic. In this case it is the property of mutual exclusion that relates to concurrency control in asynchronous, or parallel, execution of code where a resource is shared between concurrent operations. The students will hand in a short essay with their answer (max. 7200 characters including spaces) The assignment is: Explain the properties of mutual exclusion in concurrent operations using the Dining Philosophers Problem. Be sure to emphasize on some of the side effects such as deadlock and starvation and give examples on how they can be solved. Here is the indicative solution: The Dining Philosophers Problem is a well-known example on the challenges in concurrent programming related to synchronization issues when accessing shared resources. Five philosophers dine together at the same table eating from one bowl of noodles placed at the center of the table. There are five chopsticks placed on the table, one between every philosopher, and any philosopher needs to pick up both the chopstick to the left and to the right of him, thus making every chopstick a shared resource. The philosophers alternate between thinking and eating, and when in thinking mode both chopsticks are placed at the table making them available for the philosophers next to the one thinking. The challenge is to create an algorithm such that no philosopher will starve, which will happen if all philosophers want to eat at the same time and picks up one chopstick and waits for the other to be released, which will never happen since they all are waiting. To solve this, we look at Stallings lists of requirements for mutual exclusion:

	1. Mutual exclusion must be enforced: only one process at a time is allowed into its critical section, among all processes that have critical sections for the same resource or shared object. When a philosopher has picked up a chopstick, it will be flagged as in use so no other philosopher can pick it up. 2. A process that halts in its non-critical section must do so without interfering with other processes. When the philosophers are thinking they are in their non-critical section of the code where they do not access any shared resource. If the thinking process halts, it will therefore not affect any other process. 3. It must not be possible for a process requiring access to a critical section to be delayed indefinitely: no deadlock or starvation. A simple solution is found in the asymmetric approach where one philosopher picks up the chopsticks in reverse order, which will remove the problem of all picking up the left chopstick at the same time. One philosopher will try to pick up the chopstick to the right, and if he succeeds this will prevent the philosopher next to him to pick up his left chopstick, and forces him to wait, or the philosopher himself must wait, until the philosopher next to him is finished. If all philosopher wants to eat at the same time, there will always be one that cannot eat before the one next to him is finished. 4. When no process is in a critical section, any process that requests entry to its critical section must be permitted to enter without delay. When a philosopher is done thinking and wants to pick up a chopstick, this is allowed only if the chopstick is placed on the table. This can be done using the synchronized keyword in Java. 5. No assumptions are made about relative process speeds or number of processors. Depending on the programming language, decisions about which processor to use is up to the operating system, which is the case with Java. Otherwise, this must be handled in another process running independently of the program running the Dining Philosophers. 6. A process remains inside its critical section for a finite time only. A philosopher is only allowed eating for a finite time, which must be implemented in the code. The evaluation criteria are: All six requirements from Stallings book must be referenced and taken into consideration in relation to solving the Dining Philosophers Problem. There are several known solutions to the Dining Philosophers Problem (e.g., Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, Asymmetric) and the students should name and describe at least one. Here is the students submission: Mutal exclusion is an important concept in concurrent programming, where multiple processes or threads need to access shared resources without causing conflicts or race conditions. Key requirements are: 1. Mutual Exclusion: Only one process accesses a shared resource at a time. 2. Progress: The next process to enter the critical section isn't indefinitely postponed. 3. Bounded Waiting: No process is perpetually denied access, preventing starvation. 4. No Speed Assumptions: The solution works regardless of execution speed and process count.
Generated Feedback	Overall, the student's submission provides a concise overview of mutual exclusion and its key requirements in concurrent programming. However, there are some important aspects of the assignment that the student has not addressed yet. Let's provide constructive feedback to help the student improve their submission: 1. **Linking with the Dining Philosophers Problem**: The student needs to explicitly connect the concept of mutual exclusion with the Dining Philosophers Problem, as stated in the assignment. They should explain how mutual exclusion is relevant in the context of philosophers trying to access shared chopsticks while avoiding deadlock and starvation.

	2. **Emphasize on Side Effects**: The assignment explicitly asks the students to emphasize side effects such as deadlock and starvation. The student should provide a more detailed explanation of these side effects and how they can occur in the Dining Philosophers scenario. 3. **Describe at Least One Solution**: The student should name and describe at least one solution to the Dining Philosophers Problem. The indicative solution mentioned "asymmetric approach" as a solution. The student should research and describe this solution or any other known solution to the problem. 4. **Detailed Examples**: The student should provide concrete examples to illustrate the concepts they discuss. For instance, they could explain a scenario in the Dining Philosophers context where deadlock or starvation occurs and how it can be resolved using the mentioned solution. 5. **Reference Stallings' Requirements**: The student should explicitly reference and discuss how each of Stallings' requirements for mutual exclusion applies to the Dining Philosophers Problem. They should show how the proposed solution satisfies these requirements. 6. **Addressing All Evaluation Criteria**: The student should ensure they address all the evaluation criteria listed in the assignment prompt, including discussing all six requirements from Stallings' book and naming/describing at least one solution to the Dining Philosophers Problem. 7. **Utilize the Word Limit Effectively**: The student is provided with a maximum word limit of 7200 characters. While being concise is essential, they should also make sure to use this word limit effectively to provide a comprehensive and detailed explanation. Given this feedback, the student can revise their essay to better meet the assignment requirements and provide a more comprehensive and accurate response to the prompt.
Student 2	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions. In this type of assignment, the students must demonstrate a deeper understanding of a specific topic. In this case it is the property of mutual exclusion that relates to concurrency control in asynchronous, or parallel, execution of code where a resource is shared between concurrent operations. The students will hand in a short essay with their answer (max. 7200 characters including spaces) The assignment is: Explain the properties of mutual exclusion in concurrent operations using the Dining Philosophers Problem. Be sure to emphasize on some of the side effects such as deadlock and starvation and give examples on how they can be solved. Here is the indicative solution: The Dining Philosophers Problem is a well-known example on the challenges in concurrent programming related to synchronization issues when accessing shared resources. Five philosophers dine together at the same table eating from one bowl of noodles placed at the center of the table. There are five chopsticks placed on the table, one between every philosopher, and any philosopher needs to pick up both the chopstick to the left and to the right of him, thus making every chopstick a shared resource. The philosophers alternate between thinking and eating, and when in thinking mode both chopsticks are placed at the table making them available for the philosophers next to the one thinking.

	The challenge is to create an algorithm such that no philosopher will starve, which will happen if all philosophers want to eat at the same time and picks up one chopstick and waits for the other to be released, which will never happen since they all are waiting. To solve this, we look at Stallings lists of requirements for mutual exclusion: 1. Mutual exclusion must be enforced: only one process at a time is allowed into its critical section, among all processes that have critical sections for the same resource or shared object. When a philosopher has picked up a chopstick, it will be flagged as in use so no other philosopher can pick it up. 2. A process that halts in its non-critical section must do so without interfering with other processes. When the philosophers are thinking they are in their non-critical section of the code where they do not access any shared resource. If the thinking process halts, it will therefore not affect any other process. 3. It must not be possible for a process requiring access to a critical section to be delayed indefinitely: no deadlock or starvation. A simple solution is found in the asymmetric approach where one philosopher picks up the chopsticks in reverse order, which will remove the problem of all picking up the left chopstick at the same time. One philosopher will try to pick up the chopstick to the right, and if he succeeds this will prevent the philosopher next to him to pick up his left chopstick, and forces him to wait, or the philosopher himself must wait, until the philosopher next to him is finished. If all philosopher wants to eat at the same time, there will always be one that cannot eat before the one next to him is finished. 4. When no process is in a critical section, any process that requests entry to its critical section must be permitted to enter without delay. When a philosopher is done thinking and wants to pick up a chopstick, this is allowed only if the chopstick is placed on the table. This can be done using the synchronized keyword in Java. 5. No assumptions are made about relative process speeds or number of processors. Depending on the programming language, decisions about which processor to use is up to the operating system, which is the case with Java. Otherwise, this must be handled in another process running independently of the program running the Dining Philosophers. 6. A process remains inside its critical section for a finite time only. A philosopher is only allowed eating for a finite time, which must be implemented in the code. The evaluation criteria are: All six requirements from Stallings book must be referenced and taken into consideration in relation to solving the Dining Philosophers Problem. There are several known solutions to the Dining Philosophers Problem (e.g., Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, Asymmetric) and the students should name and describe at least one. Here is the students submission: There is four requirement for mutual exclusion: 1. No process more than one process can be in their respective critical section at any time 2. There should be no assumptions to the relative speed of conflicting processes. 3. Processes outside their critical section should not interrupt another process access to the critical section. 4. When more than one process wants to enter its critical section, it must be granted entry in a finite time, so that it will never be kept waiting in a infinite loop.
Generated Feedback	Overall, the student's submission shows some understanding of mutual exclusion and the Dining Philosophers Problem, but there are areas that need further clarification and elaboration. Here's a constructive feedback for the student: 1. Depth of Explanation: The student's submission lacks sufficient depth in explaining the concepts of mutual exclusion and the Dining Philosophers Problem. They need to provide a more detailed and comprehensive explanation of each requirement.

	2. Completeness: The student's submission only covers four requirements for mutual exclusion, while the assignment explicitly asks for all six requirements from Stallings book. They need to refer to Stallings' book and include all six requirements in their answer. 3. Understanding of the Dining Philosophers Problem: The student briefly mentions that no process can be in its critical section at the same time, but they need to elaborate on how this relates to the Dining Philosophers Problem specifically. They should explain the challenge of managing shared resources (chopsticks) and how it leads to issues like deadlock and starvation. 4. Solutions to the Dining Philosophers Problem: The student is expected to name and describe at least one known solution to the Dining Philosophers Problem. They should research and include information about the different solutions, such as Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, and Asymmetric. They should explain the core idea behind at least one of these solutions and how it addresses the problem of deadlock or starvation. 5. Inaccuracies: The student mentions four requirements for mutual exclusion, but some of them are not entirely accurate. For example, the second requirement about "relative speed of conflicting processes" is not a standard requirement for mutual exclusion. The student needs to correct this and provide accurate information. 6. Structure and Clarity: The student's submission lacks a clear structure and organization. They should divide their essay into clear sections, each addressing a specific requirement or aspect of the problem. This will make their answer more readable and coherent. 7. Citation and References: The student should provide proper citations and references, especially when mentioning Stallings' book or any other external sources they might use in their essay. 8. Grammar and Language: The student's submission has some grammar issues and sentence structure problems. They should revise their essay to improve its overall readability and clarity. To summarize, the student has made a good start, but they need to expand on their explanation, address all six requirements from Stallings' book, include information about at least one solution to the Dining Philosophers Problem, and improve the overall structure and clarity of their essay. Encourage them to research more, provide accurate information, and cite their sources properly.
Student 3	
Prompt	I am a lecturer at a university college, teaching computer science. I need to evaluate an assignment of programming for students and provide constructive feedback for the students submissions. In this type of assignment, the students must demonstrate a deeper understanding of a specific topic. In this case it is the property of mutual exclusion that relates to concurrency control in asynchronous, or parallel, execution of code where a resource is shared between concurrent operations. The students will hand in a short essay with their answer (max. 7200 characters including spaces) The assignment is: Explain the properties of mutual exclusion in concurrent operations using the Dining Philosophers Problem. Be sure to emphasize on some of the side effects such as deadlock and starvation and give examples on how they can be solved. Here is the indicative solution: The Dining Philosophers Problem is a well-known example on the challenges in concurrent programming related to synchronization issues when accessing shared resources. Five

	philosophers dine together at the same table eating from one bowl of noodles placed at the center of the table. There are five chopsticks placed on the table, one between every philosopher, and any philosopher needs to pick up both the chopstick to the left and to the right of him, thus making every chopstick a shared resource. The philosophers alternate between thinking and eating, and when in thinking mode both chopsticks are placed at the table making them available for the philosophers next to the one thinking. The challenge is to create an algorithm such that no philosopher will starve, which will happen if all philosophers want to eat at the same time and picks up one chopstick and waits for the other to be released, which will never happen since they all are waiting. To solve this, we look at Stallings lists of requirements for mutual exclusion: 1. Mutual exclusion must be enforced: only one process at a time is allowed into its critical section, among all processes that have critical sections for the same resource or shared object. When a philosopher has picked up a chopstick, it will be flagged as in use so no other philosopher can pick it up. 2. A process that halts in its non-critical section must do so without interfering with other processes. When the philosophers are thinking they are in their non-critical section of the code where they do not access any shared resource. If the thinking process halts, it will therefore not affect any other process. 3. It must not be possible for a process requiring access to a critical section to be delayed indefinitely: no deadlock or starvation. A simple solution is found in the asymmetric approach where one philosopher picks up the chopsticks in reverse order, which will remove the problem of all picking up the left chopstick at the same time. One philosopher will try to pick up the chopstick to the right, and if he succeeds this will prevent the philosopher next to him to pick up his left chopstick, and forces him to wait, or the philosopher himself must wait, until the philosopher next to him is finished. If all philosopher wants to eat at the same time, there will always be one that cannot eat before the one next to him is finished. 4. When no process is in a critical section, any process that requests entry to its critical section must be permitted to enter without delay. When a philosopher is done thinking and wants to pick up a chopstick, this is allowed only if the chopstick is placed on the table. This can be done using the synchronized keyword in Java. 5. No assumptions are made about relative process speeds or number of processors. Depending on the programming language, decisions about which processor to use is up to the operating system, which is the case with Java. Otherwise, this must be handled in another process running independently of the program running the Dining Philosophers. 6. A process remains inside its critical section for a finite time only. A philosopher is only allowed eating for a finite time, which must be implemented in the code. The evaluation criteria are: All six requirements from Stallings book must be referenced and taken into consideration in relation to solving the Dining Philosophers Problem. There are several known solutions to the Dining Philosophers Problem (e.g., Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, Asymmetric) and the students should name and describe at least one. Here is the students submission: No deadlocks No starvation Fairness Fault tolerance
Generated Feedback	It's great that your students have attempted to solve the Dining Philosophers Problem and submitted their work. However, their solution seems to be quite brief and lacks the necessary depth and connection to the requirements outlined in the assignment. To provide constructive feedback, you can address the following points:

	1. **Comprehensiveness**: The students' submission is quite concise, and it does not fully explain how the properties of mutual exclusion relate to the Dining Philosophers Problem. Encourage them to elaborate on each property in the context of the problem, how it affects the philosophers' behavior, and the consequences of not ensuring these properties. 2. **Requirement References**: The assignment explicitly requires students to reference all six requirements from Stallings book in relation to solving the Dining Philosophers Problem. The students' submission does not directly refer to any of these requirements. Advise them to identify which requirements their solution addresses and explain how their solution meets each of these requirements. 3. **Known Solutions**: The students should name and describe at least one known solution to the Dining Philosophers Problem. Their current submission doesn't mention any specific solution, which is a significant gap in their response. They should research and describe at least one of the well-known solutions like Dijkstra's, Resource hierarchy, Arbitrator, Chandy/Misra, or Asymmetric and explain how it satisfies the mutual exclusion property. 4. **Clear Explanation**: The students' submission lacks clarity. Ask them to provide clear and detailed explanations for each point they mention, including "No deadlocks," "No starvation," "Fairness," and "Fault tolerance." How do they achieve these properties in the context of the Dining Philosophers Problem? 5. **Examples**: Encourage the students to provide concrete examples to illustrate their points. For instance, they can describe a scenario where deadlock or starvation can occur in the Dining Philosophers Problem and explain how their solution prevents such situations. 6. **Academic References**: Ensure that students back up their explanations with credible academic references or authoritative sources. This will add credibility to their work and demonstrate a deeper understanding of the topic. 7. **Code Implementation**: While the assignment does not explicitly mention code implementation, it can be beneficial to include code examples to demonstrate how the proposed solution addresses the problem's requirements. When providing feedback, it's essential to be encouraging and supportive while also highlighting the areas where improvement is needed. Encourage the students to revise their submission and provide more detailed explanations and examples, backed by academic references, to demonstrate their understanding of mutual exclusion in the context of the Dining Philosophers Problem.

Appendix C: Preliminary Conversations with ChatGPT

In this conversation the chat bot was asked to briefly explain each state in the Five-State Process model (related to Case 1: Review Question). The indicative answer to this is:

Running: The process that is currently being executed.

Ready: A process that is prepared to execute when given the opportunity.

Blocked: A process that cannot execute until some event occurs, such as the completion of an I/O operation.

New: A process that has just been created but has not yet been admitted to the pool of executable processes by the operating system.

Exit: A process that has been released from the pool of executable processes by the operating system, either because it halted or because it aborted for some reason.

The first attempt was the exact question that the students was asked in the assignment, and since they are using Stallings book, it is implicit that they should use his description of the model.

LN Briefly explain each state in the Five-State Process model.

The Five-State Process Model is a framework for understanding how individuals process information and make decisions. Here is a brief explanation of each state in the model:

1. Perceive: This state involves taking in information from the environment through our senses. We perceive the world around us through sight, sound, touch, taste, and smell.
2. Interpret: Once we have perceived the information, we interpret it based on our past experiences and knowledge. This step involves assigning meaning to the information we have received.
3. Evaluate: In this state, we assess the significance and relevance of the information we have interpreted. We evaluate whether the information is important, accurate, and reliable.
4. Decide: Based on our evaluation of the information, we make a decision about what action to take. This may involve choosing between multiple options or deciding whether to take a specific course of action.
5. Act: The final state involves taking action based on the decision we have made. We put our decision into practice and observe the outcome, which may influence our future perception and interpretation of similar information.

Overall, the Five-State Process Model helps to explain how our perceptions, interpretations, evaluations, decisions, and actions are all interconnected and shape our behavior.

The answer provided by ChatGPT is a little unexpected and a quick read-through suggests that it is talking about how individuals process information and make decisions, so it is completely out of context and not necessarily wrong. However, a google search on the term "Five-State Process model"[11] produce a list consisting exclusively of results related to operating systems processes, so it would be reasonable to assume that this is the context the question should be interpretated in, and so is its answer a little surprising.

Next, the question was expanded with a little contextual information.

LN Briefly explain each state in the Five-State Process model in relation to operating systems.

Sure, here's an explanation of each state in the Five-State Process Model in relation to operating systems:

1. Perceive: In the context of operating systems, this state involves the computer system perceiving inputs from the user or from external devices. This could include user input via a keyboard, mouse, or touch screen, or data coming in from a peripheral device like a printer or scanner.
2. Interpret: Once the input has been received, the operating system interprets it and determines what action needs to be taken. For example, if the user types a command into the command prompt, the operating system will interpret the command and determine which program to run.
3. Evaluate: The operating system then evaluates the input and any associated data to determine whether it is valid and can be acted upon. This step helps to prevent errors or system crashes due to invalid input.
4. Decide: Based on its evaluation of the input, the operating system makes a decision about what action to take. This could include running a program, opening a file, or sending data to a peripheral device.
5. Act: The final state involves the operating system carrying out the action it has decided upon. For example, if the user clicks on a file to open it, the operating system will retrieve the file from storage and open it in the appropriate program.

Overall, the Five-State Process Model provides a helpful framework for understanding how operating systems process inputs and carry out actions in response to user and system requests.

The question was then clarified further…

[11] https://www.google.com/search?q=Five-State+Process+model

LN

Briefly explain each state in the Five-State Process model in relation to processes in operating systems.

Sure, here's an explanation of each state in the Five-State Process Model in relation to processes in operating systems:

1. Perceive: In the context of processes in operating systems, this state involves the operating system perceiving the need for a new process. This may occur when a user initiates a new program or when the operating system itself requires a new process to perform a task.
2. Interpret: Once the need for a new process has been perceived, the operating system interprets the request and determines what resources will be required to create and run the process. This may include allocating memory, assigning processor time, and setting up any necessary data structures.
3. Evaluate: The operating system then evaluates the request and any associated data to ensure that the requested process can be created and run without interfering with other processes or system functions.
4. Decide: Based on its evaluation of the request, the operating system makes a decision about whether to create and run the new process. If the request is approved, the operating system determines the priority of the new process and assigns it the appropriate resources.
5. Act: The final state involves the operating system creating and running the new process. This includes setting up the necessary data structures, allocating memory, and assigning processor time. The operating system also monitors the new process to ensure that it is running correctly and does not interfere with other system functions.

Overall, the Five-State Process Model provides a helpful framework for understanding how operating systems manage and run processes. By perceiving, interpreting, evaluating, deciding, and acting on requests for new processes, operating systems can efficiently manage system resources and ensure that all processes run smoothly and without interference.

Only when the authors name was included in the question, did the bot provide a correct answer.

LN

Briefly explain each state in the Five-State Process model in relation to processes in operating systems as William Stallings explains them.

William Stallings provides a slightly different interpretation of the Five-State Process Model as it relates to processes in operating systems. Here is an explanation of each state according to his interpretation:

1. New: This state involves the operating system perceiving the need for a new process and creating a process control block (PCB) to manage the new process. The PCB contains information about the process, such as its priority level, memory requirements, and state.
2. Ready: Once the PCB has been created, the process enters the ready state, meaning it is ready to be assigned processor time and begin executing. In this state, the process is waiting for the operating system to allocate CPU time.
3. Running: When the operating system assigns CPU time to the process, it enters the running state, meaning it is actively executing instructions.
4. Blocked: If the process requires input/output (I/O) operations or encounters a wait state, it may enter the blocked state, meaning it cannot continue executing until the I/O operation is complete or the event that caused the wait state has occurred.
5. Exit: When the process has completed its task or is terminated by the operating system, it enters the exit state, meaning it is no longer executing and its resources are released back to the operating system.

Overall, Stallings' interpretation of the Five-State Process Model provides a helpful framework for understanding how operating systems manage and schedule processes. By moving processes through the new, ready, running, blocked, and exit states, the operating system can efficiently allocate CPU time, manage system resources, and ensure that all processes run smoothly and without interference.

This shows us that some context is needed to point ChatGPT in the correct direction, so that it generates useful feedback related to the syllabus. It makes good sense as people also would need that, to answer such a question. However, asking the chatbot a question such as the one above and providing the context for it, is no guarantee for the correctness of the answer. There is no way to know which sources the information the answer is based on comes from and we cannot be sure either, that the model knows the content of the book. In this case, some older editions of the book is available online, so it might know it, but if this particular topic has changed between editions, the chatbot might get it wrong.

Appendix D: Experimenting with the Temperature Property

Sending the same prompt to ChatGPT results in different outputs. The meaning of the message is the same, but the language changes. This is due to the *temperature* property of the request, which determines the randomness and creativity of the conversation.

Large language models try to predict the next best word given a prompt. One word at a time. They assign a probability to each word in their vocabulary, and then picks a word among those.

A temperature of 0 means the responses will be very straightforward, almost deterministic because the model will always select the highest probability word. This also means we almost always get the same response to a given prompt.

A higher temperature, with 1 being the highest, means that the model might select a word with slightly lower probability, leading to more variation, randomness, and creativity. Also, this raises the risk that the AI starts selecting words that will make no sense.

In the following experiments we have requested feedback for Student 1 submission in Case 1. As can be seen, there is only a very small difference between the output for the first and second attempt where temperature was set to 0.

The third and fourth attempt are, however, quite different from both each other but also from the first two. The fourth feedback generated even points out spelling errors which is outside the scope of the assignment.

The first attempt with temperature = 0

> The student's submission includes all five states of the Five-State Process model, but there are some inaccuracies and missing details in the descriptions.
>
> 1. New: The student accurately describes this state as the phase where an operation is initiated and resources are assigned. However, they do not mention that the process has not yet been admitted to the pool of executable processes by the operating system.
>
> 2. Ready: The student correctly mentions that the process is transferred to primary memory and is set for execution. However, they do not mention that it is in a queue, waiting for the operating system to grant it access to the CPU.
>
> 3. Running: The student accurately describes this state as the phase where the CPU is actively executing the process. However, they do not mention that the process must have been allocated to the CPU to be in this state.
>
> 4. Blocked: The student correctly mentions that the process is on hold for an external occurrence or resource to become accessible. However, they do not mention that it cannot proceed with execution even if the CPU is available. Additionally, they do not mention that the process returns to the ready state once the necessary resource or event becomes available.

5. Exit: The student accurately describes this state as the phase where the process's execution is finished and it is no longer required. However, they do not mention that the operating system deallocates resources and memory assigned to the process, and that it is removed from the operation control structure.

Overall, the student's submission includes the basic ideas of each state, but there are some missing details and inaccuracies. They should provide a more comprehensive and accurate description of each state.

The second attempt with temperature = 0

The student's submission includes all five states of the Five-State Process model, but there are some inaccuracies and missing details in the descriptions.

1. New: The student accurately describes this state as the phase where an operation is initiated and resources are assigned. However, they do not mention that the process has not yet been admitted to the pool of executable processes by the operating system.

2. Ready: The student correctly mentions that the process is transferred to primary memory and is set for execution. However, they do not mention that it is in a queue, waiting for the operating system to grant it access to the CPU.

3. Running: The student accurately describes this state as the phase where the CPU is actively executing the process. However, they do not mention that the process must have been allocated to the CPU to be in this state.

4. Blocked: The student correctly mentions that the process is on hold for an external occurrence or resource to become accessible. However, they do not mention that it cannot proceed with execution even if the CPU is available. Additionally, they do not mention that the process returns to the ready state once the necessary resource or event becomes available.

5. Exit: The student accurately describes this state as the phase where the process's execution is finished and it is no longer required. However, they do not mention that the operating system deallocates resources and memory assigned to the process, and that it is removed from the operation control structure.

Overall, the student's submission includes the basic understanding of each state, but there are some missing details and inaccuracies. They should provide a more comprehensive and accurate description of each state.

The third attempt with temperature = 1

The student's submission includes all five states of the Five-State Process model: New, Ready, Running, Blocked/Waiting, and Exit/Terminate.

However, there are some inaccuracies and missing details in the student's description.

1. New: The student correctly mentions that in this state, an operation is initiated and assigned resources. However, they missed mentioning the OCS and OID as specified in the indicative solution.

2. Ready: The student correctly states that the operation is transferred to primary memory and waits for the CPU. However, they should mention that the operation is also registered in the OCS and given an OID.

3. Running: The student accurately describes that the CPU actively executes the operation. However, they should include that the operation has been allocated to the CPU.

4. Blocked/Waiting: The student correctly states that the operation is on hold for an external occurrence or resource. However, they should mention that it returns to the ready state once the necessary resource or event becomes available.

5. Exit/Terminate: The student correctly mentions that the operation's execution is finished and resources are deallocated. However, they should include that the operation is removed from the OCS.

In order to provide constructive feedback, it would be helpful to point out these inaccuracies and missing details to the student. Additionally, it would be beneficial to suggest that the student refers to the book for a more comprehensive understanding of each state in the Five-State Process model.

The fourth attempt with temperature = 1

The student's submission includes all five states described in the book and provides a brief description of each state. However, there are some inaccuracies and inconsistencies in the description.

1. New: The student accurately describes this state as the phase when an operation is initiated and resources are assigned. However, they mention the operation control structure (OCS) and operation ID (OID), which are not mentioned in the given solution. These details are not necessary for the explanation and can be omitted.

2. Ready: The student correctly mentions that the operation is transferred to primary memory and is set for execution in this state. However, they refer to the "prepared line" and "organizing system," which are not mentioned in the given solution. The student should use the term "ready queue" and "operating system" respectively to maintain consistency.

3. Running: The student accurately describes this state as the phase when the operation is actively executed by the CPU. However, there is a typo in the word "executin" that should be corrected to "executing."

4. Blocked/waiting: The student correctly mentions that the operation is on hold for an external occurrence or resource to become accessible. However, they mention "user interaction" as an example, which is not mentioned in the given solution. The student should mention that a common reason for blocking is the completion of an I/O task, which was mentioned in the given solution.

5. Exit/Terminate: The student accurately describes this state as the phase when the operation's execution is finished and it is removed from the operation control structure. However, they mention "deallocated resources and memory assigned to the operation," which is not mentioned in the given solution. The student

should mention that the process is released from the pool of executable processes by the operating system, either because it halted or aborted.

Overall, the student's submission includes most of the necessary information, but there are some inaccuracies and inconsistencies. To improve the submission, the student should correct these issues and ensure consistency with the given solution.