

# AI model report

In the realm of modern education, the need for personalized, adaptive, and engaging learning experiences is more critical than ever. Leveraging advanced AI technologies, our newly developed AI tutor, powered by the LLaMA (Large Language Model Architecture) model, aims to revolutionize the way students learn, manage their schedules, and assess their knowledge. This report delineates the key features, functionalities, and benefits of this AI tutor, designed to cater to the unique needs and talents of each student.

---

## Features and Functionalities

### 1. Personalized Tutoring

- Adaptive Learning Pathways: The AI tutor tailors educational content and pathways based on the student's performance, learning style, and pace. It continuously adjusts the difficulty and type of content to ensure optimal learning outcomes.
- Interactive Sessions: Through natural language processing, the AI engages in interactive dialogues with the student, providing explanations, answering questions, and facilitating deep understanding of complex concepts.

### 2. Schedule Management

- Customizable Timetables The AI creates personalized study schedules that balance academic, extracurricular, and leisure activities, ensuring a holistic development approach.

### 3. Creative Assessments

- Dynamic Quizzes and Exams The AI generates quizzes and exams tailored to the student's current knowledge level, focusing on areas that need improvement while also challenging their strengths.
- Skill-based Evaluation: Assessments are designed to evaluate not just rote memorization, but critical thinking, problem-solving skills, and creativity.

---

## Benefits

### 1. Individualized Attention

- Each student receives personalized attention and support, akin to having a dedicated

tutor, ensuring that their unique learning needs are met.

## **2. Enhanced Learning Outcomes**

- By adapting to the student's learning style and pace, the AI tutor enhances understanding, retention, and application of knowledge, leading to improved academic performance.

## **3. Time Management**

- Students learn effective time management skills through the AI's scheduling and reminder features, balancing academic responsibilities with personal growth and well-being.

## **4. Comprehensive Skill Development**

- The AI tutor's creative assessments and interactive learning materials promote the development of a wide range of skills, including critical thinking, creativity, and problem-solving.

## **5. Continuous Improvement**

- With real-time progress tracking and feedback, students are continually aware of their progress and areas needing improvement, fostering a proactive approach to learning.

---

Here are some screenshots of the raw code outputs (Before implementing the model in the website and adding good looking UI/UX)

---

**Asking the AI tutor to summarize a sample pdf of Object Oriented Programming**

```
(shahzan@kali) - [~/Desktop/Work/Eduthon]
$ /bin/python "/home/shahzan/Desktop/Work/Eduthon/AI Tutor/main.py"
What would you like to run 1) AI tutor or 2) AI Evaluator: 1
Hello! How can I help you today? (Type 'attach notes' to attach notes, 'remove notes' to remove notes, and press Enter to exit)

You: attach notes
-----
Please attach your notes. I can read pdf, ppt, pptx, jpeg, png, txt and everything that is text based.
(Enter the file path): /home/shahzan/Desktop/Work/Eduthon/Samples/sample00PNotes.pdf
Notes attached successfully! Ask me anything.

You: could you please summarize the notes for me?
-----
AI tutor:
Sure! Here's a summary of the Object-Oriented Programming (OOP) concepts in Python:

1. Class and Object:
   - Class: A blueprint for creating objects, defining attributes (data) and methods (functions).
   - Object: An instance of a class, representing a unique entity with access to the class's attributes and methods.

2. Encapsulation: Bundling data and methods that operate on the data into a single unit (class), hiding internal state and exposing methods to interact with the data.

3. Abstraction: Hiding complex implementation details of a class and showing only essential features, focusing on what an object does rather than how it does it.

4. Inheritance: A mechanism where one class acquires properties and behaviors of another class, promoting code reusability and hierarchical classification.

5. Polymorphism: Objects taking on multiple forms, allowing objects of different classes to be treated as objects of a common superclass, enabling design flexibility and extensibility through method overriding and overloading.
```

```
Key Terms:
- Instance: An object created from a class.
- Method: A function defined inside a class.
- Attribute: A variable associated with a class or instance.
- Constructor ('__init__'): A special method for initializing objects.
- Instance Variable: Variables defined inside methods and tied to specific instances.
- Class Variable: Variables defined inside the class and shared among all instances.
- Instance Method: Methods that operate on instances and can access instance variables.
- Class Method: Methods that operate on the class itself rather than instances.
- Static Method: Methods that do not access instance or class variables, typically used for utility functions.

OOP Benefits:
- Modularity, Reusability, Maintainability, and Flexibility.

Best Practices (SOLID principles):
- Single Responsibility Principle (SRP)
- Open/Closed Principle (OCP)
- Liskov Substitution Principle (LSP)
- Interface Segregation Principle (ISP)
- Dependency Inversion Principle (DIP)

Summary:
OOP is a powerful paradigm for structuring and organizing code, promoting encapsulation, inheritance, polymorphism, and abstraction to create modular and reusable software components, leading to maintainable, scalable, and flexible codebases.
```

## Asking AI tutor some more questions related to the attached notes

```
You: what do you mean by instance?
-----
AI tutor:
In the context of Object-Oriented Programming (OOP), an instance refers to an object that is created from a class. It represents a unique entity and can access the class's attributes (data) and methods (functions). Instances are also known as objects or class instances.

In the provided example, 'car1' and 'car2' are instances created from the 'Car' class. They have their own unique attribute values (brand and model) and can use the class's methods ('display_info').

Here's the relevant part of the example demonstrating the creation and use of instances:
```python
# Create objects (instances)
car1 = Car("Toyota", "Camry")
car2 = Car("Honda", "Accord")

# Access instance method
print(car1.display_info()) # Output: Toyota Camry
```

So, in this context, an instance is a specific object with its own unique attribute values based on the class blueprint.

You: █
```

You: how can i make an instance?

AI tutor:

To create an instance (object) from a class in Python, you use the class name and call it like a function, passing any required arguments in parentheses. Here's an example of creating an instance from the `Car` class used in the notes you provided:

```
```python
# Define the class
class Car:
    # Class variable
    car_count = 0

    # Constructor
    def __init__(self, brand, model):
        # Instance variables
        self.brand = brand
        self.model = model
        Car.car_count += 1

    # Instance method
    def display_info(self):
        return f"{self.brand} {self.model}"

# Create an instance (object) of the Car class
my_car = Car("Tesla", "Model 3")

# Access the instance method
print(my_car.display_info()) # Output: Tesla Model 3

# Access the class variable
print(Car.car_count) # Output: 1
```
```

In this example, I created an instance named `my\_car` from the `Car` class by calling `Car("Tesla", "Model 3")`. Make sure the class name is capitalized and follows PascalCase naming convention, while the instance name should be in lowercase or camelCase, following Python naming conventions.

## Asking the AI model to generate an easy schedule based on the subjects provided by the university

What would you like to run 1) AI tutor or 2) AI Evaluator: 2

Welcome! How can I help you today?

1. Make a schedule

2. Evaluate

Enter 1 or 2: 1

Please attach a document that contains subjects, time period of the semester and course material. I can read pdf, ppt, pptx, jpeg, png, txt and everything that is text based.

(Enter the file path): /home/shahzan/Desktop/Work/Eduthon/Samples/Schedule.pdf

Sure thing! I'll be happy to help you schedule your courses for the next 5 months. To make sure you don't feel overwhelmed and have enough time to study, I'll spread out your quizzes and exams as evenly as possible.

Here's a proposed schedule for you:

**\*\*September\*\***

Week 1:

\* Computer organization (3 credits)

Week 2:

\* Electrical circuits (3 credits)

Week 3:

\* Semiconductors (1 credits)

Week 4:

\* No classes, study group session for all subjects

**\*\*October\*\***

Week 1:

\* Object Oriented Programming (4 credits)

```
Week 2:
* Linear Algebra (2 credits)

Week 3:
* Signals and systems (3 credits)

Week 4:
* Digital logic design (3 credits)

**November**
Week 1:
* Computer organization (3 credits)

Week 2:
* Electrical circuits (3 credits)

Week 3:
* Semiconductors (1 credits)

Week 4:
* No classes, study group session for all subjects

**December**
Week 1:
* Object Oriented Programming (4 credits)
```

```
Week 2:
* Linear Algebra (2 credits)

Week 3:
* Signals and systems (3 credits)

Week 4:
* Digital logic design (3 credits)

**January**
Week 1:
* Review week for all subjects

Week 2:
* Final exams for all subjects

Let me know if this schedule works for you, and if you have any questions or suggestions for modifications. I'm here to help you!
```

---

## Asking the AI to suggest creative and innovative ideas for quiz and exams based on course material

```
What would you like to run a quiz or exam on?
Welcome! How can I help you today?
1. Make a schedule
2. Evaluate
Enter 1 or 2: 2
Please attach all the notes on which the student should be evaluated on. I can read pdf, ppt, pptx, jpeg, png, txt and everything that is text based.
(Enter the file paths separated by '--'): /home/shahzan/Desktop/Work/Eduthon/Samples/sampleOOPNotes.pdf
Thank you for providing the OOP notes. Here are some unique ideas to evaluate students based on their understanding of these concepts:

1. **Code Implementation:**
   - Provide students with a set of requirements or a problem statement and ask them to implement the solution using OOP principles. Assess their understanding of classes, objects, inheritance, encapsulation, and polymorphism.

2. **Code Review:**
   - Give students a piece of code that exhibits good or bad OOP practices. Ask them to review the code and identify areas for improvement, such as following best practices, encapsulation, or abstraction.

3. **Multiple Choice Questions (MCQs):**
   - Create questions that test students' understanding of key terms, definitions, and concepts. Make sure to include questions covering inheritance, polymorphism, encapsulation, and abstraction.

4. **True or False Questions:**
   - Present statements related to OOP and ask students to determine whether they are true or false. This can help assess their understanding of key concepts and details.

5. **Fill in the Blanks:**
   - Provide incomplete definitions or statements related to OOP and have students fill in the missing parts. This can test their recall and understanding of important terms.

6. **Short Answer Questions:**
   - Formulate questions that require students to write short answers, such as explaining a concept in their own words, comparing and contrasting different OOP principles, or giving examples of real-world applications of inheritance and polymorphism.
```

```
7. **Pair Programming:**
  - Divide students into pairs and have them work together on a coding project or exercise. Monitor their interactions and assess their ability to collaborate, explain concepts, and provide constructive feedback.

8. **Design Patterns:**
  - Present students with a design pattern, such as Singleton, Factory, or Decorator, and ask them to identify the OOP principles being used. This can test their ability to recognize and apply advanced OOP concepts and patterns.

9. **Refactoring Challenge:**
  - Provide students with a codebase that is poorly designed or implemented using OOP principles. Ask them to refactor the code to improve encapsulation, abstraction, inheritance, or polymorphism.

10. **Case Studies:**
  - Present real-world case studies or examples where OOP principles have been applied. Ask students to analyze the case and identify the principles and best practices used, as well as areas for improvement.

These creative evaluation methods can help ensure that students have a deep understanding of OOP concepts and principles.
```

---

Note: the sample files like OOP notes or schedule.pdf is uploaded on [github.com](https://github.com)

**The above codes clearly demonstrate working of our model and how we can use its output to develop a full fledged application that can become the best personal tutor for a student.**