**AI Assisted Online Learning Platform**

**Team A3**

Anirudh Vijayaraghavan, 002995983

Sunil Srinivaas, 002711958

Devki Patel, 002912471

Kaushik Elangovan, 002793494

Arvind Ram Mahesh, 002819495

Rohit Varma, 002688431

Github repository :

<https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/tree/main>

Website URL :

<https://assistedinstructai.anirudhvijayaraghavan.me/>

**Motivation and User Benefits**

In today’s digital age, the education industry (colloquially known as EdTech) is rapidly evolving, necessitating innovative solutions to cater to the diverse learning needs of individuals. Our project, the AssistedInstructAI, is what we feel, a cutting-edge application/concept designed to personalize the educational experience, offering a unique blend of technology and user-centric design to enhance learning outcomes.

Why is it a necessity?

Traditional educational systems often follow a one-size-fits-all approach, which may not effectively address the unique learning styles, preferences, and goals of everyone. With an increasing number of learners seeking personalized educational experiences, there is a pressing need for a platform that can dynamically adapt to each user's requirements, facilitating a more engaging and effective learning journey.

We believe our application is an answer to that, and combines the traditional style of watching videos, with interactive quizzes and custom course generation, as well as access to other courses/ideas generated by other users. These elements naturally complement each other.

Our key features, as will be shown in our demo, include :

* Personalized Course Recommendations: Utilizing AI to analyze user preferences and objectives, the platform offers highly relevant and engaging course suggestions.
* Streamlined Content Access: A vast library of courses and materials is easily accessible, simplifying knowledge acquisition.
* Interactive Assessments and Quizzes: These tools engage users in active learning, aiding in knowledge retention and skill development.

**Live Demo**

Demo URL :

**Architecture and Design Choices**

Architecture Description:

1. N-Tier Architecture:
   * Presentation Tier (Frontend): Includes user interface components like the navigation/landing page and theme toggle, facilitating user interaction and course creation.
   * Logic Tier (Application Server): Contains the core functionality, such as the Course Generation API, User Authentication and Management, Stripe Integration, and potentially AI models.
   * Data Tier (Database): Utilizes Prisma for data storage and management, handling user and course information.
2. Layered Architecture within Components:
   * Presentation Layer: Manages user interface aspects.
   * Application Layer: Handles business logic, user authentication, and interfaces with Stripe.
   * Persistence Layer: Deals with data access and includes mechanisms for interfacing with the database.
3. Model-View-Controller (MVC) for UI Components:
   * Model: Contains core business logic and data.
   * View: Manages visual elements and user interactions.
   * Controller: Acts as an intermediary between Model and View, handling user requests and updating the View.

Design Choices:

1. Class Diagram and Design Patterns:
   * The class diagram showcases the static structure of the system, including classes, attributes, methods, and their relationships.
   * Design patterns like MVC ensure modularity, easier debugging, and scalability. It allows the application to evolve with user demand and ensures a structured development approach.
2. Functional & Non-Functional Requirements Fulfillment:
   * The chosen architectures and design patterns fulfill several key requirements:
     + Scalability: Both the N-Tier and Layered architectures allow for independent scaling of different components.
     + Separation of Concerns: Each layer/component has a distinct responsibility, enhancing maintainability and reducing complexity.
     + Security: Robust security measures can be implemented at each tier/layer.
     + User-Friendly Interface: The MVC pattern facilitates a clear separation between the user interface and business logic, ensuring a smooth user experience.
     + Flexibility and Modularity: The architectures support easy updates and changes to individual components without affecting the whole system.

N-Tier Application Landscape Pattern relevant to our app.:

Once again, as per what we have learnt in class, we feel this is the architecture that fits our use cases and concept, based on our understanding.

A diagram of a computer server

Description automatically generated

**Implementation, Testing, CI/CD**

Implementation:

* Frontend: Developed using Next.js 13, with Shad CN and Tailwind CSS for UI.
* Backend: Integration of Prisma DB Toolkit for database management, PlanetScale DB (Scalable cloud based application / platform, using MySQL ) and Stripe for payment processing.
* APIs: Utilization of **Unsplash API** for images, **YouTube API** for video content, **and OpenAI** for content generation and analysis.

Testing:

1. Frontend Testing:
   * Manual testing is the primary method for the frontend. This includes checking UI elements, responsiveness, and general functionality.
2. Backend Testing:
   * Combination of manual and automated testing, using Mocha for TypeScript/JavaScript.
   * Focus on healthcheck endpoint and database connectivity, with plans for more comprehensive automated unit tests.

Continuous Integration / Continuous Deployment:

Link to our CI script : <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/blob/main/.github/workflows/nextjsCI.yml>

For CI, we’ve used GitHub Actions. Inside the .github/workflows directory, we can see a nextCIjs.yml file, that consists of our CI script. It contains specifications on triggers, steps / jobs, and actions. They are triggered on Pull requests, as mentioned below.

**Trigger**: Any pull request that targets the main branch will also trigger the pipeline. This ensures that proposed changes are tested before they are merged into the main codebase.

**Jobs and Steps**:

* Runs-on: The job runs on the latest Ubuntu runner provided by GitHub Actions.
* Node.js Versions: The job tests the application against multiple Node.js versions (14.x, 16.x, and 18.x) to ensure compatibility.
* Checkout: The first step checks out the code from the repository so that it can be built and tested.
* Node.js Setup: This step sets up the Node.js environment based on the versions specified in the matrix.
* Install Dependencies: Executes npm install to install all necessary dependencies for the project.

**Actions**:

* actions/checkout@v2: Used for checking out the repository.
* actions/setup-node@v2: For setting up the Node.js environment.

**Outputs**:

* The primary outputs are the results of the build process, test execution, and linting. These results are visible directly on the GitHub Actions page.
* The secondary outputs, also equally important, are that it allows the pull request to be merged with our main branch of the repository.

Link to our CD script : <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/blob/main/.github/workflows/deploy.yml>

**Trigger**: Any push that targets the main branch will trigger this pipeline. This ensures that proposed changes are deployed to production seamlessly.

**Jobs and Steps**:

* Runs-on: The job runs on the latest Ubuntu runner provided by GitHub Actions.
* Environment: Production – This is important as it takes the secret keys and variables present in our Github Actions production environment.
* Steps: Essentially, we are :
  1. Connecting to the DigitalOcean Droplet through the SSH keys.
  2. Going into the .git folder where the project is present.
  3. Running the git pulls, .env variables copy, and docker down/up commands.

**Actions**:

* actions/checkout@v3: Used for checking out the repository.

**Outputs**:

* The primary outputs are the production website being updated, live with the relevant changes.

**Reflections**

Scrum Process:

The scrum process was managed by Sunil and Anirudh. We used Github to create a Kanban board, and put all relevant Epics and Stories, pertaining to our application.

We used the Kanban board along with an AGILE sprint model to develop, build, test and deploy our application. Our sprint review and meeting happened every 2 weeks.

For further information and more details, the link to our dashboard is here :

[https://github.com/users/AnirudhVijayaraghavan/projects/2](https://github.com/users/AnirudhVijayaraghavan/projects/2%20)

Contributions:

Backend (Major API routes such as course generation, generating chapter information and syllabus/quiz): Sunil, and Devki.

ALL Front end components: Kaushik, Arvind, with some assistance from Rohit.

DevOps (Testing, Continuous Integration, Continuous Deployment, Github Structure management, Scrum Process & Kanban, Project Fine Tuning and Bug fixes): Anirudh and Sunil.

Sprint Statistics:

We had 4 major sprints, with each sprint ending with approximately 2 epics, and 7 user stories completed.

Lessons learned:

Throughout the entire project, our team enjoyed working around and developing our web application as it forced all of us to get out of our comfort zones. The scrum process was new to all of us, and it was intensive, and an eye-opening glimpse into what goes on in software development.

All of us learnt a little bit about each others roles. Anirudh managed to incorporate what he learned from Network Structures and Cloud computing graduate course, into the DevOps pipeline, as well as teach his teammates about common Github, and DevOps strategies. Sunil and Devki were able to work with Kaushik and Arvind and teach what they knew from UI/UX, and vice versa about backend development.

The Scrum Process, forced us to adhere to a timeline, and other aspects of this course, such as the architecture selection, etc, taught us the theory behind Software Development in a hands on way.

Future work and potential improvements:

This app, can use some UI improvements. It is also limited in the sense, we have to be on the constant monitor and lookout for API changes to OpenAI, and Youtube. We also have to monitor for API\_KEY rate limits. In a way, we are limited to the services we have selected.

To improve :

* 1. We can probably automate some of the monitoring, using Splunk, or other tools to check the health of the application, and daily reports which scrape through the youtube and OpenAI websites using our key to check if they are nearing their limits.
  2. We can also probably include other features, such as an interactive communities section. Right now, we can only view what others have created. An additional blog site would be a fantastic user experience.

You can refer to our wiki link given below, on how to get started with and use our AI assisted platform. Using a combination of description text and screenshots, users can expect how to generate the course list as well as check out the videos and quiz sections.

Github wiki link : [Wiki Link.](https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/wiki)

**Scrum Summary**

We have designed a user-friendly navbar that simplifies navigation and access to various parts of the application. This navbar ensures easy traversal through different sections of the platform, making it effortless for users to explore and interact with our services.

Next, we created the User Interaction Form, a vital component for user-generated content. This form plays a crucial role in ensuring the accuracy and reliability of user input data. By parsing and validating the data provided by users, we guarantee that the information entered is error-free and meets our quality standards.

In addition to the user-facing components, we have developed a powerful Syllabus Fetching API Endpoint. This endpoint provides users with a convenient way to retrieve syllabus-related information. The data is returned in a structured format, typically JSON, making it straightforward for us to process and display this valuable content to our users.

To enhance security and maintain control over syllabus creation, we have implemented authentication checks. These checks ensure that only authorized users can create and modify the syllabus. This adds an extra layer of security to our platform, protecting sensitive educational content.

Currently, we are working on integrating Stripe payment services into our application. This integration will enable us to handle payments efficiently and securely, giving our users the option to upgrade to a pro account. This enhancement will further expand our platform's capabilities and provide valuable premium features to our users.

Epics and User Stories with sprints:

Sprint 1 - User Authentication And User Input Fetching

EPICS:

* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/5>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/11>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/7>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/12>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/22>

User Stories:

* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/20>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/21>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/14>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/6>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/13>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/4>

Sprint 2 - Use user data - to create syllabus and extract information needed to generate videos

Epics:

* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/17>

User Stories:

* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/15>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/16>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/25>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/24>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/27>

Sprint 3 - Fetch Summary, Transcript , Video ID and generate questions for the user with proper Front End display

Epics:

* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/33>
* [https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project\_A3/issues/37 (In progress)](https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/37%20(In%20progress))

User Stories:

* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/30>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/31>
* <https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/32>
* [https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project\_A3/issues/34](https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/34%20%20)  (In progress)
* [https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project\_A3/issues/35](https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/35%20%20)  (In progress)
* [https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project\_A3/issues/36](https://github.com/AnirudhVijayaraghavan/CSYE7230-Software-Engineering-Project_A3/issues/36%20%20)  (In progress)

In the front end, Kaushik has created the navbar component that provides options for user account actions, such as signing in and accessing account details. Users can also switch between light and dark modes using the theme toggle.

Arvind implemented the User interaction form which allows users to create courses by providing a title and specifying subtopics (units). The form enables users to dynamically add or remove units.

Anirudh and Devki generated chapter end point in backend which checks the user's authentication status and verifies if they are authorized. The end point facilitates the dynamic creation of course content and management of user interaction.

Sunil and Rohit worked on displaying the course information and generated chapters, fetching the data related to the course and its chapters, handles user authentication and presents the course details in a user-friendly interface.

We are actively working on the integration of Stripe into our application. Stripe integration will enable us to handle payments and subscriptions seamlessly, providing our users with a secure and efficient payment experience.