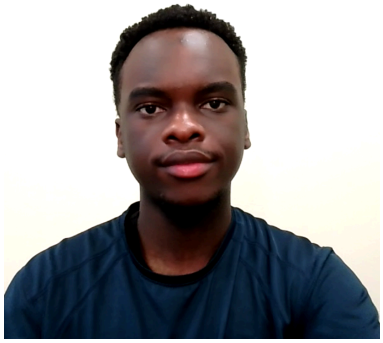
TZStudies



Overview

TZStudies is a ful -stack web application designed to support Tanzanian students and tutors. The primary purpose of the project is to provide students with easy, free access to past national exam PDFs and answer keys, while also serving as a platform for tutors to submit their applications. The application includes dynamic features such as an AI assistant for exam-related queries, email notifications for tutor applications, and a REST API endpoint for retrieving tutor data. The project demonstrates a practical implementation of modern web technologies and cloud integration.

Key Features

- **Exam PDF Access:**
  Students can browse and download past exam papers and corresponding answer keys from a dedicated folder on the server. The homepage displays the exams in a responsive built in search function that filters the displayed exams.

- **Tutor Application Form:**
  Tutors can apply by filling out a form on the “Become a Tutor” page. The form requires essential details such as name, location, school, hourly rate, experience, classes taught, and email (with an optional phone number). On submission, an email notification is sent to the admin and a confirmation email is sent to the applicant. The application data is stored in a PostgreSQL database.

- **AI Assistant (CURRENTLY IN DEVELOPMENT):**
  An AI assistant endpoint is implemented to answer exam-related queries. It uses the OpenAI API (specifically the gpt-3.5-turbo model) to generate detailed responses based on user input.

- **REST API Endpoint:**
  A REST API is exposed which returns all tutor application data in JSON format. This

endpoint can be useful for future integrations (such as administrative dashboard).

- **Admin Listing Page:**
  An admin page is provided to view all tutor submissions in a tabular format. This allows for quick review of applications directly from the web interface.

Tech Stack

- **Programming Language:**
  - **Python:** Used as the primary language to build the backend of the application.

- **Web Framework:**
  - **Flask:** Used to create the application's routes, handle HTTP requests, and render HTML templates.

- **Email Handling:**
  - **Flask-Mail:** An extension that integrates with Gmail's SMTP server to send out notification and confirmation emails. This module ensures that when a tutor submits an application, both an admin and the applicant receive email notifications.

- **Database and ORM:**
  - **PostgreSQL (Heroku Postgres):** A robust, cloud based relational database used to persist tutor application data.
  - **Flask-SQLAlchemy:** An ORM that simplifies database interactions by mapping Python classes (models) to database tables. The TutorApplication model represents the tutor applications stored in the database.

- **AI Integration (Work In Progess):**
  - **OpenAI API:** Integrated via the OpenAI Python library to provide AI powered responses to exam-related queries. The /ask route uses the gpt-3.5-turbo model to generate detailed answers.

- **Frontend Technologies:**
    - **HTML/CSS:** Used to structure and style the web pages. The layout includes a responsive grid for exam PDFs and a modern navbar with a logo.
    - **JavaScript:** Provides client-side interactivity such as filtering exam papers via a search bar.

- **Web Server:**
    - **Gunicorn:** A production-grade WSGI server used to serve the Flask application when deployed (for example, on Heroku).
- **Deployment and Version Control:**
    - **Heroku:** A cloud platform that hosts the application, handles scaling, and provides managed services such as Heroku Postgres.
    - **Git:** Used for version control and to push code to Heroku during deployment.

How It Works

1. **Exam PDFs Display:**
    - The Flask app reads PDF filenames from a local folder (the "exams" folder) and passes them to the index template.
    - The template uses a CSS grid layout to display these files as cards with download and answer key buttons.
    - A JavaScript search filter allows users to quickly narrow down the list of exam papers.
2. **Tutor Application Process:**
    - Tutors fill out the "Become a Tutor" form, which validates that all required fields (including email) are provided.
    - On submission, the app sends a notification email to the admin and a professional confirmation email to the tutor.
    - The tutor's information is stored in the PostgreSQL database using Flask-SQLAlchemy.
3. **AI Assistant:**
    - Currently In Development
4. **REST API Endpoint:**

- The /api/tutors endpoint returns all tutor application records in JSON format, enabling future integrations.

5. **Admin Data Review:**

    - An admin page (/admin_applicants) displays all tutor applications in a table.
    - This page allows the admin to review applications directly from the browser.

Why I Built This Project

I built TZStudies to address a real need in the Tanzanian education sector making past national exam resources easily accessible while streamlining the tutor recruitment process. The project demonstrates my ability to develop a full-stack application using modern technologies, from building a dynamic Flask backend and integrating third-party APIs (OpenAI and Gmail SMTP) to deploying a cloud-based solution with Heroku and managing data with PostgreSQL. I believe that this project shows my agile development practices, attention to user experience, and the practical application of cloud engineering, skills that are directly relevant to roles in software and full stack development.

- Kevin S. Shilla