**Problem Statement:**

The goal of this project is to develop an educational math game using Pygame that not only engages users but also adapts to their performance by generating math problems based on data analysis. Additionally, a Flask web app will be created to display educational content related to math topics. The primary objectives are to enhance math skills and provide statistical insights to help users identify their strengths and weaknesses.

**Expected Functionality:**

1. **Math Game (Pygame):**

.

* Dynamic math problem generation.
* Scoring system and immediate feedback.

1. **Flask Web App:**

* Display educational content (lessons, tutorials) related to math topics.
* Performance statistics.
* Progress tracking and visualization.

1. **Enhanced Features (for creativity):**

* Multiplayer mode for competitive math challenges.
* Customizable avatars for user profiles.
* Topic selection for specific math areas.

1. Additional Considerations:

* Create a user-friendly interface.
* Provide high-quality educational content.
* Plan for future scalability.
* Thoroughly test and debug the application.
* Document code and usage for developers and users.

**Project Structure:**

**Flask Web App**:

1. app.py: The main Flask application script, includes the problem generation and statistics functions
2. templates/: Folder containing HTML templates for web pages.
3. static/: Folder for static assets (CSS, images).:

static/

|-- css/

| |-- style.css # Your CSS stylesheet(s)

1. templates/: Folder for .html files

templates/

|-- index.html -> Home page

|-- result.html -> Result page

|-- stats.html -> Statistics page

**How to run the app ?**

To run your Flask web app, follow these steps:

* Install Flask:

If you haven't already installed Flask, you can do so using pip. Open your command prompt or terminal and run:

* Create a Flask Project Directory:

Create a folder for your Flask project if you haven't already. Place your Python scripts, HTML templates, and the "static" folder (containing your CSS, images, and JavaScript) inside this directory.

* Set Up Your Flask App:

In your project directory, create a Python script (e.g., app.py) for your Flask application. Define your app, routes, and views within this script.

Here's an example of a simple app.py script:

* Create HTML Templates:

In the "templates" folder of your project directory, create HTML templates for your web pages. For example, you can create a home.html template.

* Run Your Flask App:

Open your command prompt or terminal, navigate to your project directory, and run the following command:

This will start your Flask app, and you should see output indicating that the app is running locally.

* Access Your App:

Open a web browser and enter http://localhost:5000/ in the address bar. You should see your Flask app's home page. If you have other routes defined, you can access them by entering their URLs.

That's it! You've successfully set up and run your Flask web app. Remember that this is a basic example, and you can expand upon it by adding more routes, views, and functionality as needed for your project.