Documentation for MCQ-Based Online Exam Application

# Project Overview:

This project is a web-based multiple-choice question (MCQ) exam application built using Python Flask. The application allows participants to take an exam, view their scores, and check the leaderboard. The questions are stored in a CSV file (questions.csv), and participant scores are saved in a CSV file (leaderboard.csv). It provides both an admin interface (for managing questions) and a user interface (for taking the exam and viewing results).

# Table of Contents:

1. Folder Structure  
2. Setup and Installation  
3. Files Description  
 - app.py  
 - question\_master.py  
 - questions.csv  
 - leaderboard.csv  
 - HTML templates  
4. Application Features  
 - Home Page  
 - Exam Page  
 - Result Page  
 - Leaderboard Page  
5. How to Use  
 - Running the App  
 - Interacting with the App  
6. Future Enhancements

## Folder Structure:

/mcq\_exam\_app  
 /static # For static files (CSS, JavaScript, etc.)  
 /templates # For HTML templates  
 app.py # Main Flask application  
 question\_master.py # For managing questions  
 questions.csv # CSV file containing questions  
 leaderboard.csv # CSV file containing leaderboard  
 requirements.txt # Dependencies (Flask, etc.)

## Setup and Installation:

1. Clone or download this repository.  
2. Install the required dependencies using pip. Use the following command to install Flask and other necessary packages:  
 pip install flask  
3. Ensure that you have the following files in your project directory:  
 - questions.csv  
 - leaderboard.csv  
4. Run the application using:  
 python app.py  
5. Visit http://127.0.0.1:5000/ in your browser to use the application.

## Files Description:

### 1. app.py:

This is the main Flask application file. It handles the following:  
- Loading questions from the CSV file.  
- Serving web pages (home, exam, result, and leaderboard).  
- Handling form submissions for the exam and saving the participant's score.  
- Saving leaderboard data to leaderboard.csv.

### 2. question\_master.py:

This is an admin tool for managing the questions. It allows the admin to:  
- Add a new question to questions.csv.  
- Display all questions in the system.

### 3. questions.csv:

This file stores all the questions for the exam. The format is as follows:  
num,question,option1,option2,option3,option4,correctoption  
1,10+20 is,20,30,40,10,op2  
2,5/2 is,2.5,3,6,7,op1  
Each question has 4 options, and the correct option is indicated as op1, op2, etc.

### 4. leaderboard.csv:

This file stores the leaderboard data, including the name of the participant, the university, the score, and the timestamp. It follows this format:  
name,university,score,timestamp  
Francis,VTU,10,2024-09-21 12:45:00

### 5. HTML Templates:

- index.html: The home page where users can start the exam or view the leaderboard.  
- exam.html: The page where users answer the exam questions.  
- result.html: The result page that shows the user's score after submitting the exam.  
- leaderboard.html: The page that displays the list of participants and their scores.

## Application Features:

### 1. Home Page:

- Displays a welcome message.  
- Allows users to start the exam by clicking the 'Start Exam' link.  
- Provides a link to view the leaderboard.

### 2. Exam Page:

- Participants are required to enter their name and university.  
- Presents the multiple-choice questions (loaded from questions.csv).  
- Users can select their answers for each question and submit the form.

### 3. Result Page:

- Displays the participant's score after completing the exam.  
- Shows how many questions were answered correctly.

### 4. Leaderboard Page:

- Displays the leaderboard with participant names, universities, scores, and timestamps.  
- The leaderboard data is loaded from leaderboard.csv.

## How to Use:

### 1. Running the App:

- Start the Flask application by running the following command:  
 python app.py  
- Open the browser and navigate to http://127.0.0.1:5000/.

### 2. Interacting with the App:

- Home Page: Click 'Start Exam' to begin the test or 'View Leaderboard' to see the leaderboard.  
- Exam Page: Enter your name and university, answer the questions, and submit.  
- Result Page: After submitting the exam, you will be shown your score.  
- Leaderboard: See how your score compares to others on the leaderboard.

## Future Enhancements:

- Admin Interface: Provide a web-based admin interface for adding, modifying, or deleting questions rather than using question\_master.py.  
- Authentication: Add authentication for users to allow participants to register and log in.  
- Question Randomization: Add randomization of the order of questions and answer options.  
- Pagination: Break the questions into pages so participants can navigate through them rather than showing all questions on one page.  
- Timer: Add a timer to limit the time for completing the exam.  
- Email Notifications: Send an email to participants with their scores after completing the exam.