

*A project report submitted to ICT Academy of Kerala  
in partial fulfillment of the requirements  
for the certification of*

**CERTIFIED SPECIALIST  
IN  
DATA SCIENCE & ANALYTICS INTERNSHIP**

submitted by

**Sreya Thangam Mathew**



**ICT ACADEMY OF KERALA  
THIRUVANANTHAPURAM, KERALA, INDIA  
Sep 2024**

## List of Figures

<b>Fig. No:</b>	<b>Figure Name</b>	<b>Page No:</b>
5.4.1	Manual Test Output	10
6.1.1	Home Page Pre-login	14
6.1.2	Home Page Post-login	14
6.2.1	MCQ Exam Page	15
6.2.2	Subjective Exam Page	15
6.3.1	View Result Page	16
7.1.1	Training and Validation Loss	17

## List of Tables

<b>Table No:</b>	<b>Table Name</b>	<b>Page No:</b>
4.1	Tools used for Development	10
4.2	Libraries used for Development	10
7.1.1	Test Set Metrics	17

# **List of Abbreviations**

1. FYUGP : Four-Year Undergraduate Program
2. AI : Artificial Intelligence
3. MCQ : Multiple Choice Question
4. LLM : Large Language Model
5. NLP : Natural Language Processing

## Table of Contents

<b>Sl. No.</b>	<b>Content</b>	<b>Page No:</b>
<b>i</b>	Abstract	6
<b>1.</b>	Problem Definition	7
1.1	Objective	7
1.2	Problem Statement	7
<b>2.</b>	Introduction	8
<b>3.</b>	Dataset Description	9
3.1	Student Dataset	9
3.2	Grading Dataset	9
<b>4.</b>	Tools & Libraries Used	10
<b>5.</b>	Methodology	11
5.1	Database Creation	11
5.2	Application Overview	11
5.3	Objective Question Exam	12
5.4	Subjective Question Exam	12
<b>6.</b>	Implementation	14
6.1	Home Page	14
6.2	Exam Page	15
6.3	View Result Page	16
<b>7.</b>	Result	17
7.1	Model Performance	17
<b>8.</b>	Conclusion	18
<b>9.</b>	References	19

# Abstract

In the world of Artificial Intelligence, most tasks are being automated as we progress. In the field of education too, AI has a large scope. Specially, in helping students learn with the right feedback. The manual process of grading exams and giving feedback takes weeks, if not months if some external issues affect educational organizations. With the newly introduced frameworks like Four-Year Undergraduate Program (FYUGP) framework which helps students get relevant knowledge outside of the main curriculum, and chances to expedite their graduation by a year, it has become crucial for educational institutions to offer faster results for exams and assessments.

In this project, the goal was to build a web application that would allow students in FYUGP to attend exams and receive automated grading and feedback services. The exam consists of MCQ as well as subjective questions. The application is built using the Flask framework, HTML, CSS and Bootstrap. It also utilises LLM model T5 small and MySQL databases for storing student details and scores. The project follows a multi-task learning approach, where the model predicts both a score and constructive feedback for each student answer. The preprocessing pipeline includes data cleaning, tokenization, augmentation, and structured labeling of responses to improve model generalization. Fine-tuning is performed in Google Colab using free and open-source tools, ensuring accessibility and cost efficiency. VSCode and pythonanywhere were used to deploy the app locally and online, respectively. Future extensions may explore other modes of exam evaluation, larger transformer models, cross-domain grading, and further dataset expansion to improve accuracy.

# 1. Problem Definition

## 1.1 Objective

The objective is as follows:

- To automate the exam grading process for objective (MCQ) and subjective questions
- To provide constructive feedback to students

## 1.2 Problem Statement

The problem statement chosen for this project is a **Smart Grading and Feedback System for FYUGP** that leverages technologies like Artificial Intelligence and data science.

## 2. Introduction

With the increasing adoption of digital learning platforms, universities and colleges need efficient and scalable solutions to grade student assignments, quizzes, and exams. Traditional manual grading is time-consuming, subjective, and prone to human error. An AI-powered system can help streamline the grading process while ensuring fairness and consistency.

The FYUGP framework (Four-Year Undergraduate Program) emphasizes continuous assessment, skill-based evaluation, and personalized feedback. This means the system should be capable of handling various assessment formats. This project aims to develop an **automated grading** for **objective questions** using a **rule-based system**. And to provide an **automated grading and feedback system** for **subjective programming theory answers** using a **fine-tuned T5-small model**. The model is trained to generate **both scores and qualitative feedback**, ensuring that students not only receive a numerical evaluation but also constructive insights into their responses.



## 3. Dataset Description

### 3.1 Student Dataset

The dataset of student details was created after domain research regarding Four Year Undergraduate students, The dataset consists of 150 entries of students enrolled in a Data Science course in an institution. The fields include:

- **Admission no.**
- **Name**
- **Gender**
- **Department**
- **Address**

### 3.2 Grading Dataset

This dataset consists of 186 subjective questions with 5 answers each of varying understanding with scores ranging from 0-5 with the associated feedbacks.

## 4. Tools & Libraries Used

The tools used for this projects is given in the table below:

Python 3.10	The language used.
VS Code	For developing the flask app
Google Colab	For fine-tuning LLM
MySQL Workbench	Database
PythonAnywhere	Hosting the web application
HTML, CSS, Bootstrap	Frontend

Table 4.1 :Tools used for Development

The python libraries used are given below:

LLM	<ol style="list-style-type: none"><li>1. torch</li><li>2. transformers</li><li>3. datasets</li><li>4. scikit-learn</li><li>5. json</li></ol>
Web Application	<ol style="list-style-type: none"><li>1. flask</li><li>2. mysql-connector-python</li><li>3. transformers</li><li>4. torch</li></ol>

Table 4.1 :Libraries used for Development

## 5. Methodology

### 5.1 Database Creation

Database student\_db was created using MySQL. The csv files were imported to the tables in the database as tables student\_details and users. The tables to store scores were kept empty after creation, only to be updated once the students attempted the exam. It contains 3 tables:

1. **student\_details** : 150 students in total enrolled in the course
2. **users** : 125 students who have an account on the web application
3. **scores** : For storing the scores of the exam for the registered students
4. **subjective\_scores** : For storing the subjective question, student's answer, score and feedback from model

### 5.2 Application Overview

The Student can register as a user as long as they are in the student table. Then log in, take exams, submit exams, and view scores. HTML, CSS and Bootstrap were used to create the frontend. The backend was developed using python, flask, and MySQL.

- **Register**
  - Student provides the required details for registration.
  - System validates and stores the student information in the database.
  - If the student admission\_no matches the one in student details, the username and password given are added as an entry along with admission\_no in users table.
- **Login**
  - Student enters the user credentials.
  - System verifies credentials and grants access.
- **Home page**
  - Displays the student details from the database.
- **Take Exam**
  - Student selects an available exam: Currently MCQ and subjective exam.
  - The selected exam page is shown.
- **Submit Exam**
  - Student submits their answers.
  - System evaluates answers and calculates scores.

- **View Scores**
  - Student can view their scores by clicking the result button on the nav bar or will be redirected to it on submit.
  - Admin can access and manage all student scores.
- **Store Scores**
  - System stores the calculated scores into the database.

### 5.3 Objective Question Exam

This part of the flask app is developed to allow students to participate in a MCQ exam and receive their grades instantaneously. This follows a rule based method. Five questions were chosen from the course and displayed as a quiz form using HTML and CSS. It was ensured that a student can only attempt the exam once, in the flask code through checking entries in the database table **scores**.

Once the student submitted the quiz, the answers were checked against the correct ones in the python code. If it matches with the correct option, the score is incremented. After submission, the user will be redirected to the result page which shows the score out of 5, which is also stored in the database.

### 5.4 Subjective Question Exam

This part of the app uses means of an LLM for grading and giving feedback to students on their subjective or short answer questions. Students are allowed to type in their answers through form which then passes it on as input to T5-small model which was fine-tuned on the grading dataset.

#### Data Preprocessing

The dataset was created manually with questions from the course and then answers ranging from scores 0-5.

- **Feature Engineering**
  - Another feature, feedback, was added to provide appropriate feedback to student answers.
- **Data Augmentation**
  - The original entries in the dataset were paraphrased to increase the size of data.


The Json file was converted to text-to-text format. Then split into training and testing and validation sets. The split chosen was a 80-20 split with the latter split in half for validation set as well. The input was tokenized using **T5Tokenizer** from transformer library

## Model Fine-tuning

T5-small was chosen since it was a free model and less resource intensive compared to the T5 model. It also requires less data preprocessing normally required for NLP tasks. The augmented grading dataset.


The model was trained on the dataset for 5 epochs with a batch size of 4. The saved model was also tested manually on student answers as shown below.

Test 1:

 Student Answer: A list is mutable, while a tuple is immutable. Lists are slower than tuples.


◆ Model Output: 3 | Partially correct, but missing some key aspects.

Test 2:

 Student Answer: Lists are changeable, but tuples are not. Lists use square brackets, tuples use parentheses.


◆ Model Output: 3 | Partially correct, but missing some key aspects.

Test 3:

 Student Answer: Lists and tuples are the same and can be modified.


◆ Model Output: 3 | Partially correct, but missing some key aspects.

Test 4:

 Student Answer: Lists and tuples store data. Lists are mutable, meaning they can be changed, while tuples are immutable. Lists use square brackets, whereas tuples use parentheses. Tuples are slightly faster.

◆ Model Output: 5 | Excellent! Well-explained with clarity.

Test 5:

 Student Answer: Lists can change, tuples can't.

◆ Model Output: 3 | Partially correct, but missing some key aspects.

Fig. 5.4.1 : Manual Test Output

## Model Deployment

Locally, the model was deployed using flask on VSCode with the exam having two subjective type questions. Due to size restriction on free hosting service, it could not be deployed online. Only the objective question part was deployed online on PythonAnywhere.

## 6. Implementation

### 6.1 Home Page

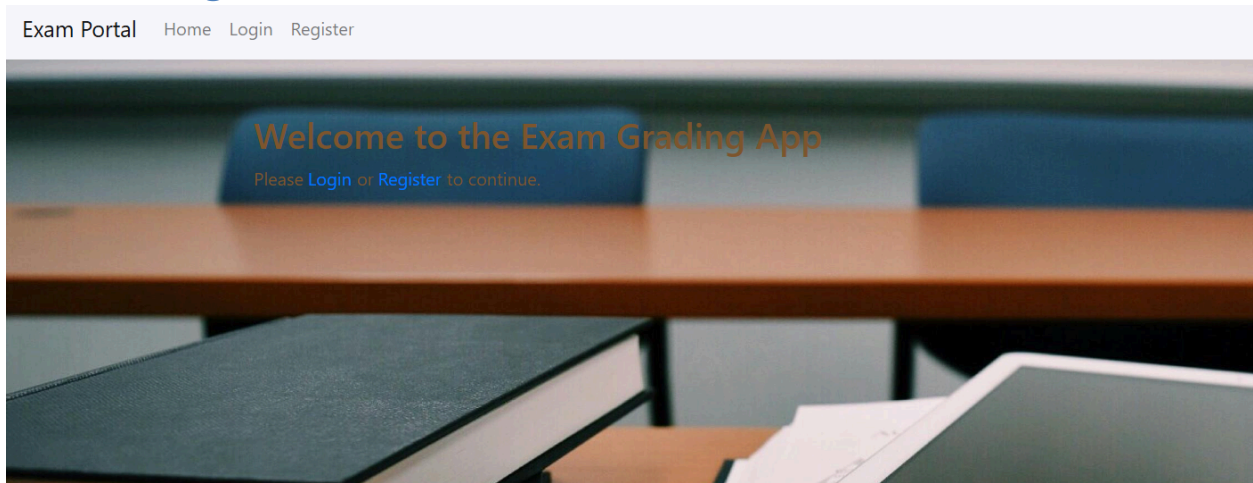


Fig. 6.1.1 : Home Page Pre-login

The user may login or register as a user here.

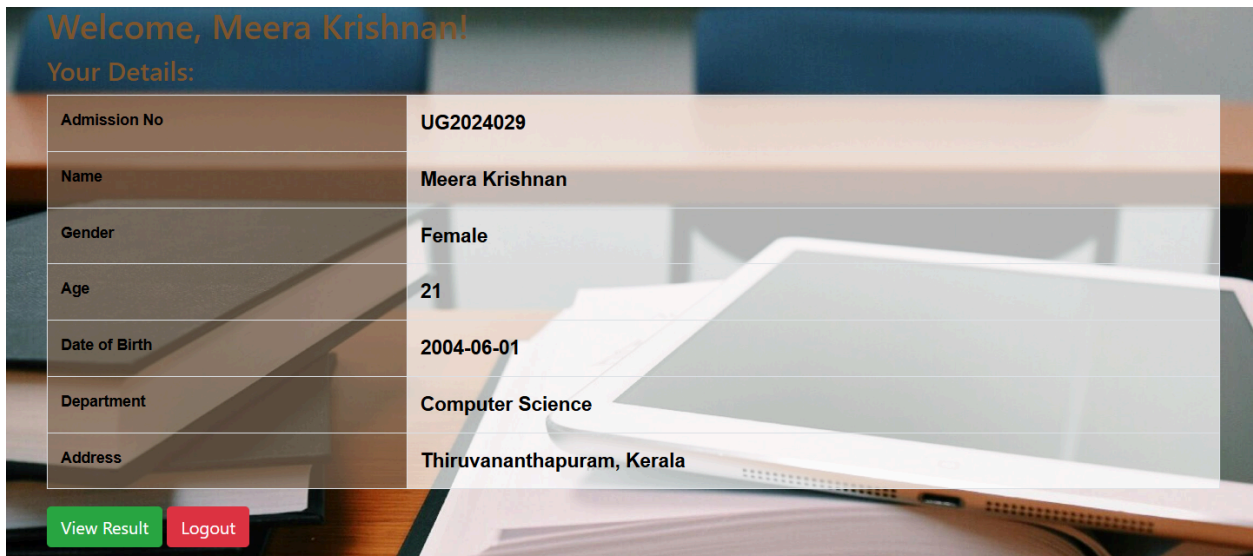


Fig. 6.1.2 : Home Page Post-login

The student user's details are displayed on successful login.

## 6.2 Exam Pages

The **objective question** exam is displayed on clicking the Take Exam button.

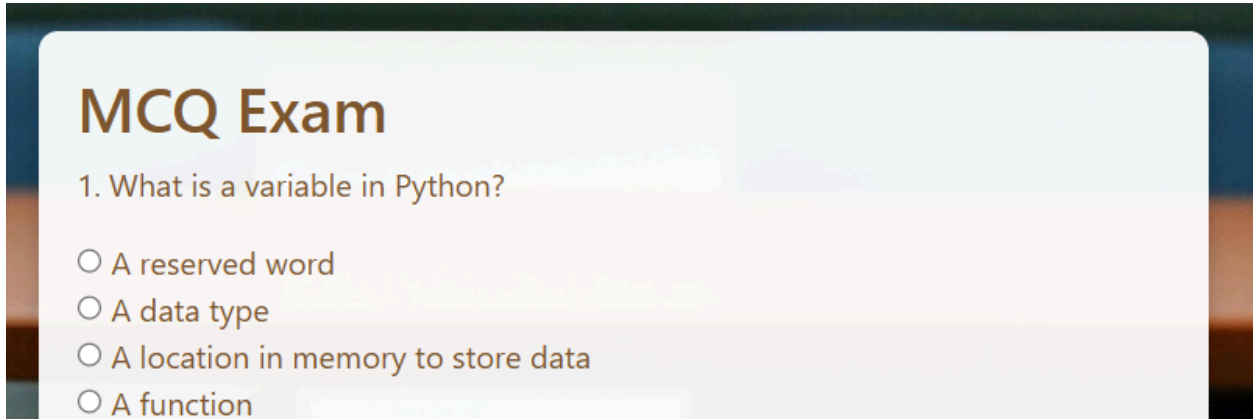


Fig. 6.2.1 : MCQ Exam Page

The **subjective question** exam is displayed on clicking the Take S Exam button.

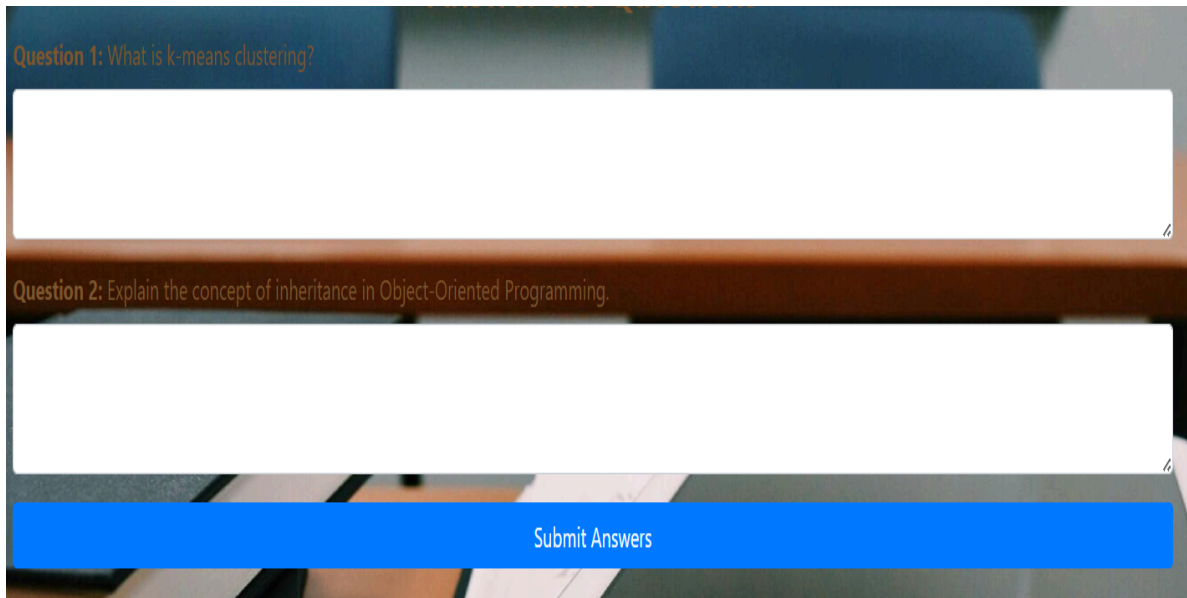


Fig. 6.2.2 : Subjective Exam Page

### 6.3 View Result Page

The score is displayed once the exam is attempted. Only one attempt is allowed.

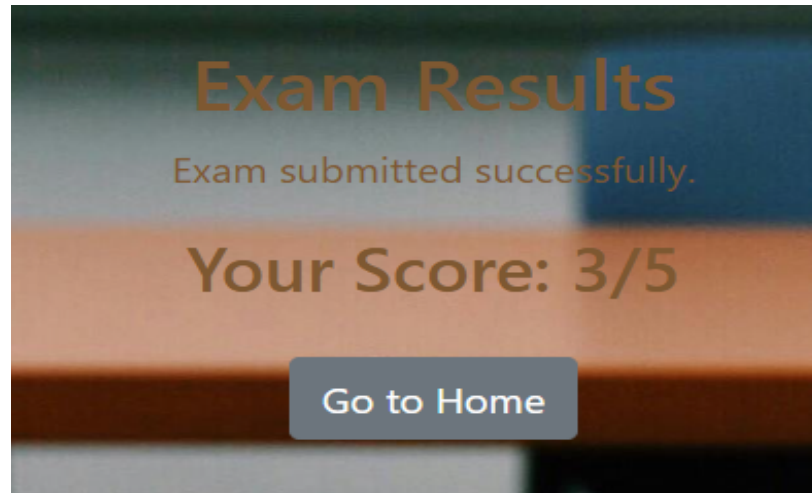


Fig. 6.3.1 : MCQ Exam Page



## 7. Result

- The system was able to grade the MCQ questions and subjective ones well.
- The database was updated accordingly.

### 7.1 Model Performance

Epoch	Training Loss	Validation Loss
1	0.201900	0.213620
2	0.192700	0.190988
3	0.136700	0.187734
4	0.188900	0.182611
5	0.125800	0.180845

Fig.7.1.1 : Training and Validation Loss

Model stabilised at the fifth and final epoch.

Metric	Test Set
Evaluation Loss	0.181

Table 7.1.1 : Test Set Metrics

## 8. Conclusion

The web application performs well. It effectively automated the process of grading exams and gave appropriate feedback to the student. Thus, it can reduce the workload on teachers and institutions who offer FYUGP frameworks. Further improvements may be added to enhance the system. Such as:

- A coding evaluation feature.
- Larger, more diverse dataset for LLM.
- More resources for complete online deployment.

## 9. References

1. google-t5/t5-small · Hugging Face
2. Noam Shazeer, et.al., Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Colin Raffel, Google, Mountain View, CA 94043, USA
3. Flask Tutorial - GeeksforGeeks