Mini-Project Report On

QGEN (An Application for generating question paper from an input PDF)

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science & Engineering

By

Nevin Aju (U2003151)
Rohan Jose Paul (U2003171)
Ronit John Daniel (U2003174)
Sebin Bejoy (U2003190)

Under the guidance of
Ms. Sangeetha Jamal

Department of Computer Science & Engineering
Rajagiri School of Engineering and Technology (Autonomous)
Rajagiri Valley, Kakkanad, Kochi, 682039

July 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RAJAGIRI SCHOOL OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
RAJAGIRI VALLEY, KAKKANAD, KOCHI, 682039

CERTIFICATE

This is to certify that the mini-project report entitled ***"QGEN (An Application for generating question paper from an input PDF)"*** *is a bonafide work done by* ***Nevin Aju (U2003151), Rohan Jose Paul (U2003171), Ronit John Daniel (U2003174), Sebin Bejoy (U2003190)****, submitted to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2022-2023.*

Dr. Preetha K. G.	**Ms. Anita John**	**Ms. Sangeetha Jamal**
Head of Department	Mini-Project Coordinator	Mini-Project Guide
Dept. of CSE	Asst. Professor	Asst. Professor
RSET	Dept. of CSE	Dept. of CSE
	RSET	RSET

ACKNOWLEDGEMENTS

We wish to express our sincere gratitude towards **Dr. P. S. Sreejith**, Principal of RSET, and **Dr. Preetha K. G.**, Head of Department of Computer Science and Engineering for providing us with the opportunity to undertake our mini-project, "QGEN Application".

We are highly indebted to our mini-project coordinators, **Ms. Anita John**, Assistant Professor, Department of Computer Science and Engineering and **Mr. Sajanraj T D**, Assistant Professor, Department of Computer Science and Engineering for their valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our mini-project guide **Ms. Sangeetha Jamal**, Assistant Professor, Department of Computer Science and Engineering , for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Nevin Aju

Rohan Jose Paul

Ronit John Daniel

Sebin Bejoy

ABSTRACT

Traditional methods of creating question papers manually by educators can be time-consuming, repetitive, and often prone to human errors. To overcome these challenges, the concept of a Question Paper Generator has emerged as a valuable tool for automating the process of generating assessment materials.

The goal of this AI enabled Question Paper Generator is to develop an automated system that can efficiently set a question paper in seconds. It significantly reduces effort required to create question papers, allowing educators to focus on other critical aspects of teaching.

The project will be implemented as a web application which utilizes the client-server architecture. A login interface is used for the project that displays the dashboard for each user when you login. The dashboard will show users their previously generated question papers with an option to download them.

It provides an easy user experience that is guaranteed to make question paper generation an easier, more natural process. This would ease the burden on educators when setting a question paper.

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 General Background

In today's age, education is the most important way of achieving success. When we discuss education, it is imperative to mention tests and examinations. Examinations prepare students in their quest for knowledge. So, having a proper examination paper and format is quite necessary. Now the traditional method of generating question papers has been manual. But this method can be ineffective at times owing to time consuming, repetition, etc. Thus having an Automated process of Question Paper Generation will be very useful.

Paper Based Systems-

- Human process.
- Slow as human labor is involved.

Paperless Based Systems-

- Automated process.
- Faster due to computer based automation.

1.2 Objective

The objective of a Question Paper Generation application is to automate and streamline the process of creating question papers, in the field of education. It will optimize the assessment process in education, making it more efficient, and tailored to support student learning and achievement. By providing an easy user interface it makes question paper generation easier thus the application will meet the user's needs. The primary goals are:

- **Utility**-Question paper generators serve as a valuable tool in schools, colleges, and universities.
- **Time Saving**-Generating question papers manually can be a time-consuming task. Thus it eliminates the need for manual creation and selection of questions..
- **Scalability**- A question paper generator can handle the scalability effortlessly, generating papers for any number of students or exams.

1.3 Motivation

The motivation for creating a question paper generation application comes from the need to embrace technological advancements and leverage the power of automation in education. By developing such an application, we aim to harness the capabilities of artificial intelligence and machine learning to revolutionize the assessment process.

This technology-driven solution seeks to enhance the overall educational experience by providing educators with a tool that can generate dynamic and engaging question papers. It helps in enabling educators to effectively measure and support student learning thus enhancing their learning experience.

1.4 Summary Of Report

The primary goal of this report explains the processes and efforts that went into developing a Question Paper Generation Application that helps in easing the burden on educators for setting question papers.

The first chapter deals with the general background of the project, its objective and the motivation behind it.

The second chapter includes the literature survey that was done to get an idea of the existing methods used.

The third chapter contains the hardware and software requirements.

The fourth chapter describes the problem definition,the scope of the project and the module division.

The fifth chapter contains architecture diagram and use case diagram.

The sixth chapter contains testings associated to the project.

The seventh chapter contains the results associated with the project.

The last two chapters outlines the risks, challenges, conclusion of the entire project and its future scope.

The report also includes the various references used as well as appendixes of sample code and specifying the project code and project objectives.

Chapter 2

LITERATURE SURVEY

2.1 Transformer-based End-to-End Question Generation

-Luis Enrico Lopez, Diane Kathryn Cruz, Jan Christian Blaise Cruz, Charibeth Cheng

This research paper focuses on the task of generating questions from a given context paragraph. The authors present a simple yet effective approach to this task using transformer-based finetuning techniques. The model is trained on a single pretrained language model and does not require the use of additional mechanisms, answer metadata, or extensive features.

The authors compare their approach to previous techniques for question generation, such as RNN-based Seq2Seq models and models that employ answer-awareness and other special mechanisms. They show that their model outperforms previous more complex RNN-based Seq2Seq models and performs on par with Seq2Seq models that employ answer-awareness and other special mechanisms.

The paper also discusses factors that affect the performance of the model, such as input data formatting and the length of context paragraphs. The authors provide examples of the model's failure modes and possible reasons why it fails. They also discuss possible future work, such as incorporating additional features and mechanisms to improve the model's performance.

Overall, this research paper presents a promising approach to the task of generating questions from a given context paragraph using transformer-based finetuning techniques. The model is simple yet effective and outperforms previous more complex models. The paper provides valuable insights into the factors that affect the performance of the model and possible future directions for research in this area.

2.2 Scalable Educational Question Generation with Pre-trained Language Models

-Sahan Bulathwela, Hamze Muse and Emine Yilmaz

This research paper focuses on the development of a novel model for generating educational questions called EduQG. The authors argue that generating scalable educational questions is crucial for democratizing education and enabling self-assessment at scale. While existing language models are used for question generation, their utility in education has only been explored recently. This work demonstrates how a large language model can be adapted for educational question generation.

The paper begins by discussing related work in the field of AI systems capable of generating educational questions for technology-enhanced learning. The authors explain that this involves two main sub-tasks: Question Generation (QG), where a model generates a question based on given information, and Question Answering (QA), where a model generates a response to a question. QG is essential for QA, and both tasks are part of reading comprehension tasks. This paper focuses on QG specifically.

The authors then introduce their proposed model, EduQG, which is based on the T5 language model. EduQG is pre-trained on a large corpus of educational text and fine-tuned on a science question dataset. The paper describes the experiments conducted to validate the effectiveness of EduQG, including pre-training with educational text, investigating the impact of pre-training data size on question generation, and enhancing educational questions through fine-tuning with a science question dataset. The experimental results show that pre-training and fine-tuning with domain-specific scientific text can outperform a state-of-the-art baseline, providing significant evidence for building an effective educational question-generation model.

The paper also discusses potential applications of EduQG in online education, such as enabling self-assessment and personalized learning. The authors suggest that EduQG can be adapted to generate questions for subjects other than science, and that future research could explore the use of EduQG in other domains.

Overall, this research paper presents a novel approach to generating educational questions that can help democratize education and enable self-assessment at scale. The au-

thors demonstrate the effectiveness of their proposed model, EduQG, through a series of experiments and suggest potential applications for online education.

Chapter 3

SYSTEM ANALYSIS

3.1 Hardware Requirements

The following are the hardware requirements to develop the QGEN Application.

- Hardware: Keyboard, Mouse, Monitor, CPU for a standard i3 processor
- Processor: Intel Core i3
- RAM: Minimum 8GB

3.2 Software Requirements

The following are the softwares used in the development of the application.

3.2.1 Operating System: Windows 8 or above

Windows 8 is a personal computer operating system that is part of the Windows NT family. Windows 8 introduced significant changes to the Windows operating system and its user interface (UI), targeting both desktop computers and tablets.

It is a touch-optimized platform based on the modern Metro design architecture, which specifies how applications are delivered and rendered in the UI. Along with having a much different look and feel from its predecessor Windows 7, Windows 8 also boasted faster startup times and better performance.

3.2.2 Python 3.10

Python is a versatile and widely used programming language known for its simplicity and readability. It offers an extensive standard library and a vibrant ecosystem of third-party packages, making it suitable for various domains, including web development, data

analysis, artificial intelligence, and automation.It supports RegEX which is needed for this project.

It also supports multiple programming paradigms, including procedural, object-oriented, and functional programming, allowing developers to choose the approach that best suits their needs. Python 3.10 introduces improved error messages, making debugging and troubleshooting easier.

3.2.3 MongoDB 6.0

MongoDB is a popular NoSQL database management system that provides a flexible and scalable solution for storing and retrieving data. As a document-oriented database, MongoDB stores data in a JSON-like format called BSON, making it easy to work with and highly compatible with modern web applications.

It offers features such as high availability, horizontal scaling, and automatic sharding, allowing for seamless handling of large datasets and high traffic loads. With its flexible schema and dynamic document model, MongoDB is an excellent choice for agile development, rapid prototyping, and applications requiring frequent schema changes.

3.2.4 Flask(Python Web Framework)

Flask is a lightweight and versatile micro web framework for Python, widely acclaimed for its simplicity and efficiency in building web applications. With its minimalistic design and intuitive syntax, Flask offers developers a straightforward and elegant approach to web development.

It provides a wide range of features, including routing, templating, and request handling, making it suitable for projects of any size. Flask's modular structure allows developers to choose and integrate various extensions based on their specific needs, enhancing its flexibility and scalability.

Chapter 4

PROPOSED METHOD

4.1 Problem Definition

Faculties often face difficulties in creating questions for exams. This project aims to develop an application that can extract text from study material and generate a questions paper from it .

4.2 Scope Of The Work

The application will help you to generate question papers from an uploaded document. However the uploaded document is restricted to a pdf document. There is no option of selection of questions. All the questions will be used to generate the question paper.

4.3 Module Division

The project will be divided into the following modules:

- **Module 1: Flask App Development**- Web application that has a login and register option. After logging in it is redirected to a user dashboard page where you can view the previously generated question papers and option to upload a pdf file for generating a new question paper.
- **Module 2: Document Text Extraction and Text Pre-processing**-Extract text from the uploaded study material using PyPDF2 and preprocess the text into suitable format for the transformer using RegEx.
- **Module 3: Question Generation using T5 Transformer**-Fine tune the T5 base model for question generation. Utilize this trained T5 model to generate questions from the extracted text.

- **Module 4: Question Paper Generation**-Convert the generated questions into a question paper in PDF format and save it in the database.

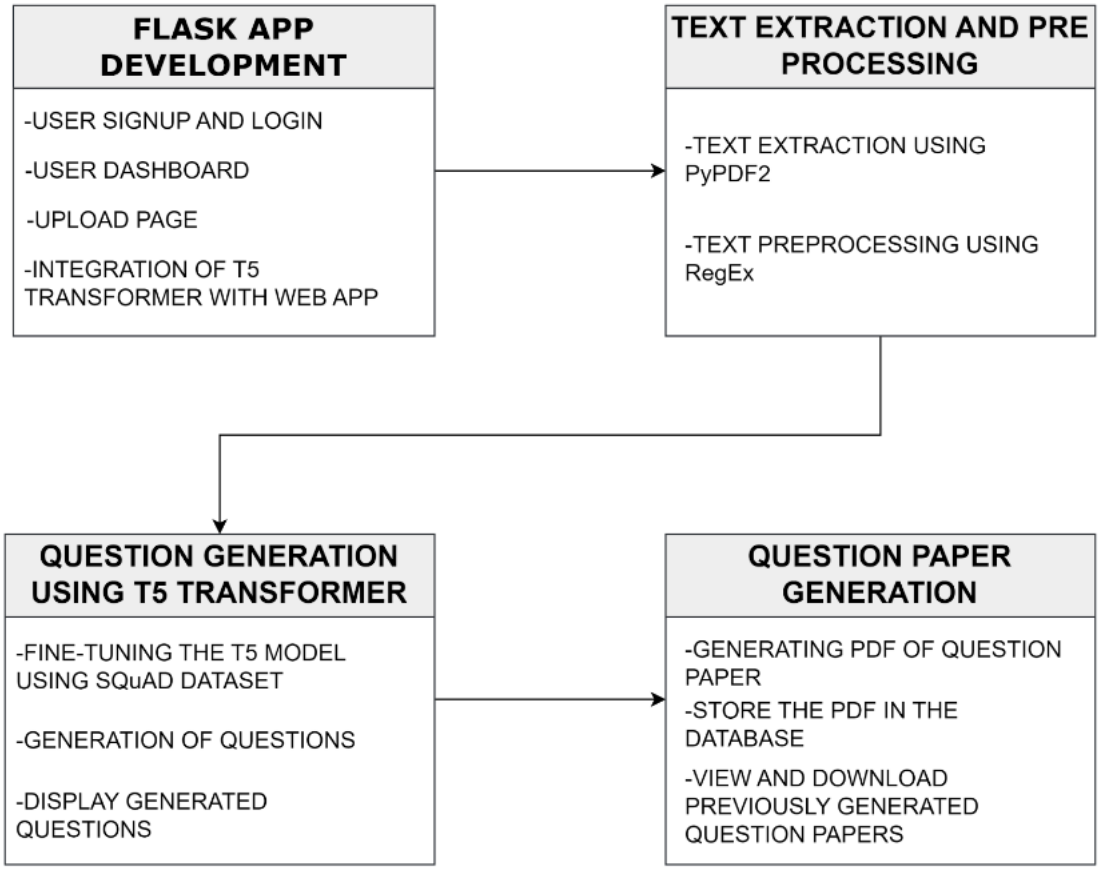


Figure 4.1: Module Wise Diagram

Chapter 5

SYSTEM DESIGN

5.1 Architecture Diagram of the System

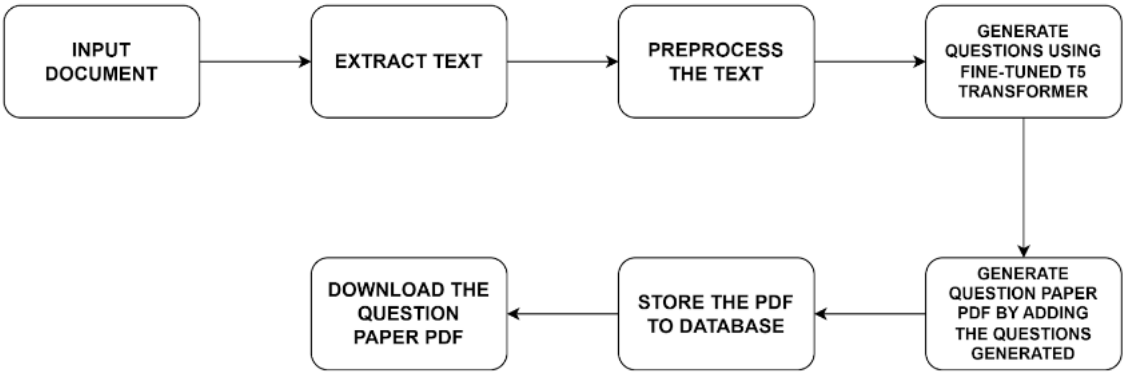


Figure 5.1: Architecture diagram

5.2 Use Case diagram

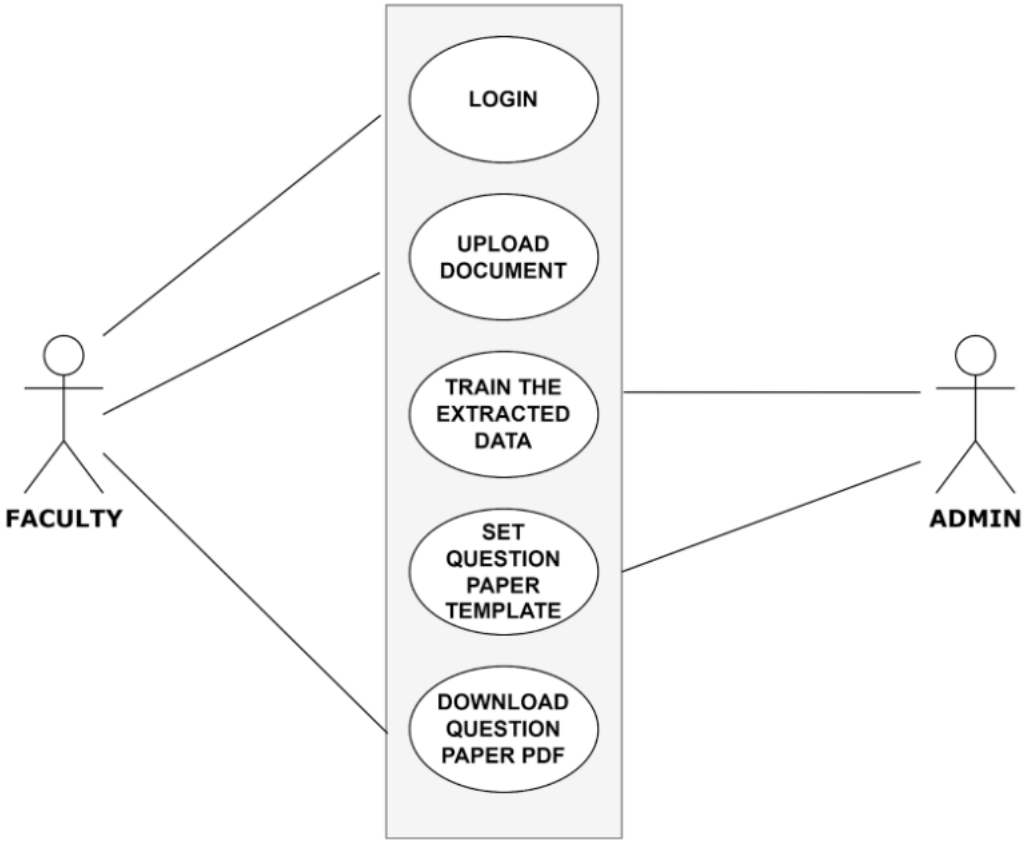


Figure 5.2: Use Case Diagram

Chapter 6

TESTING

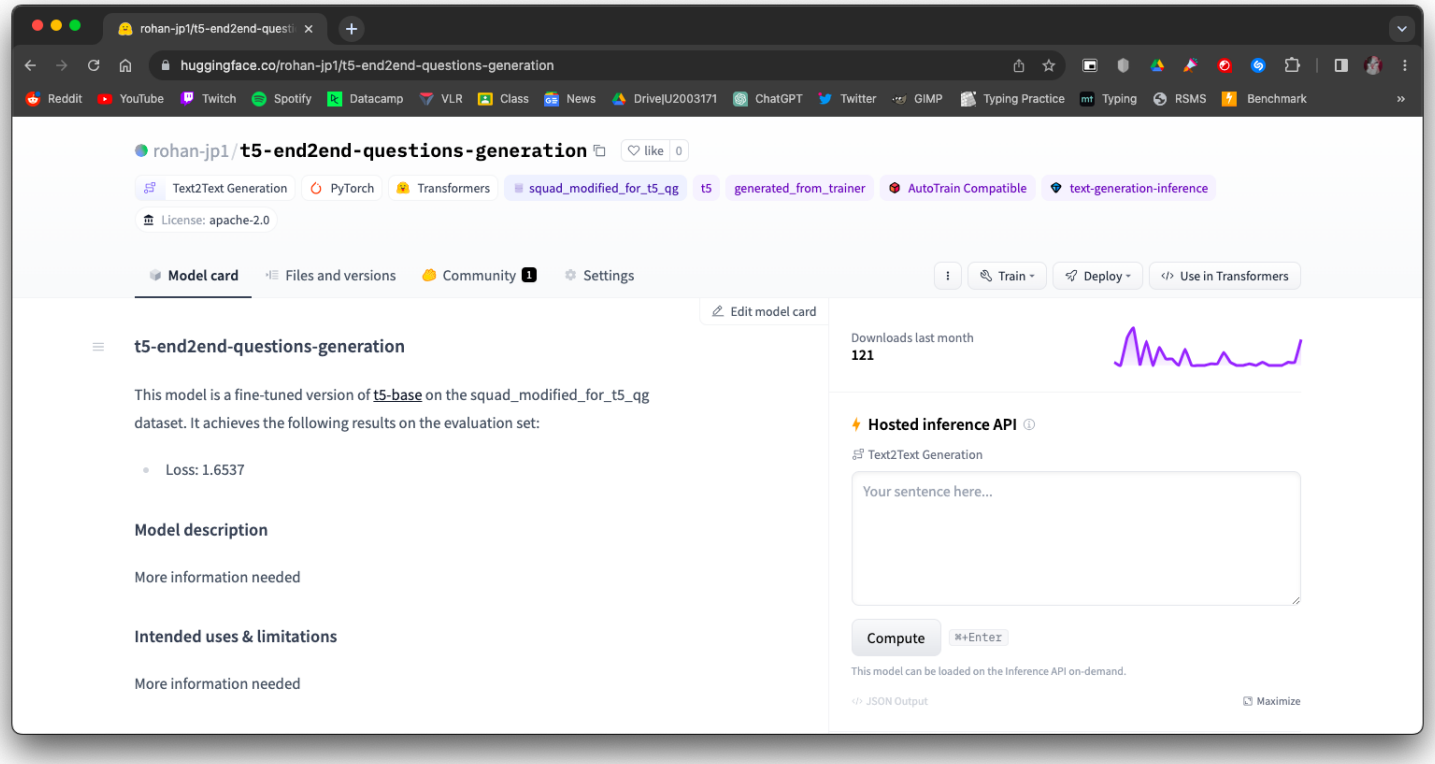


Figure 6.1: Fine tuned model pushed to hub

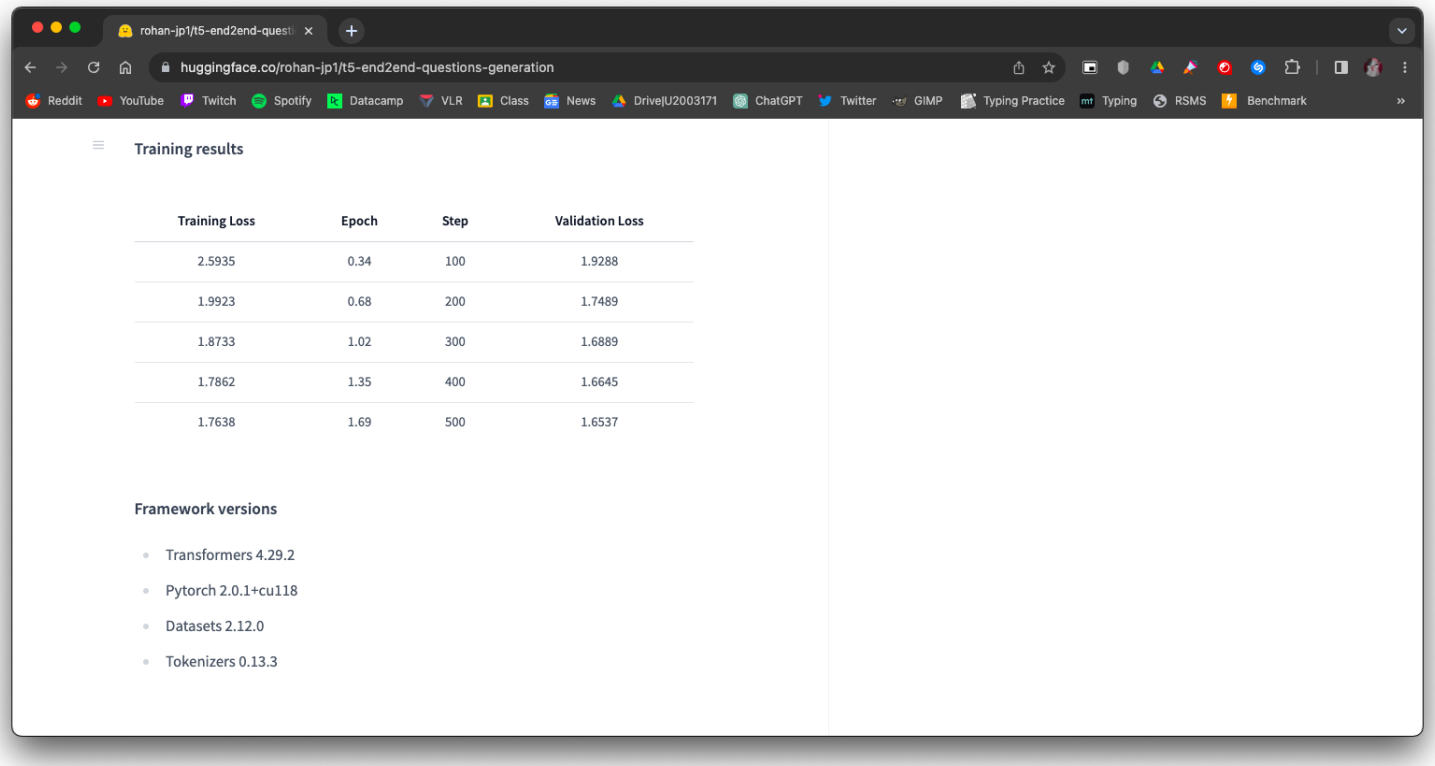


Figure 6.2: Training Summary

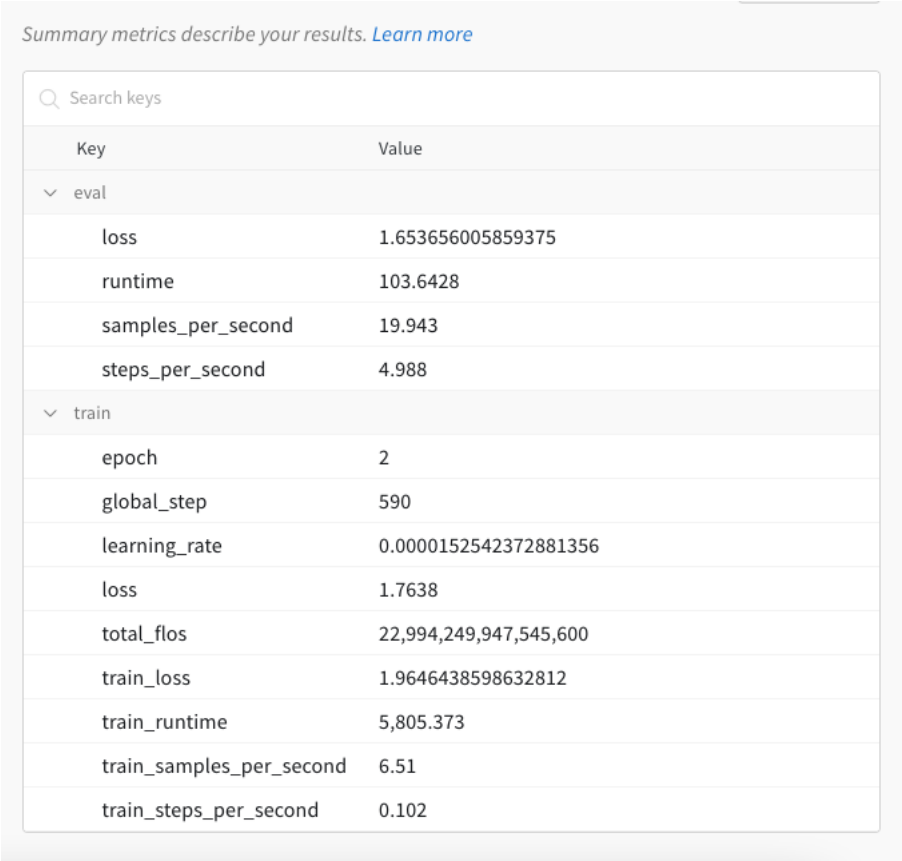


Figure 6.3: Overall Summary

Figure 6.4: Training Report

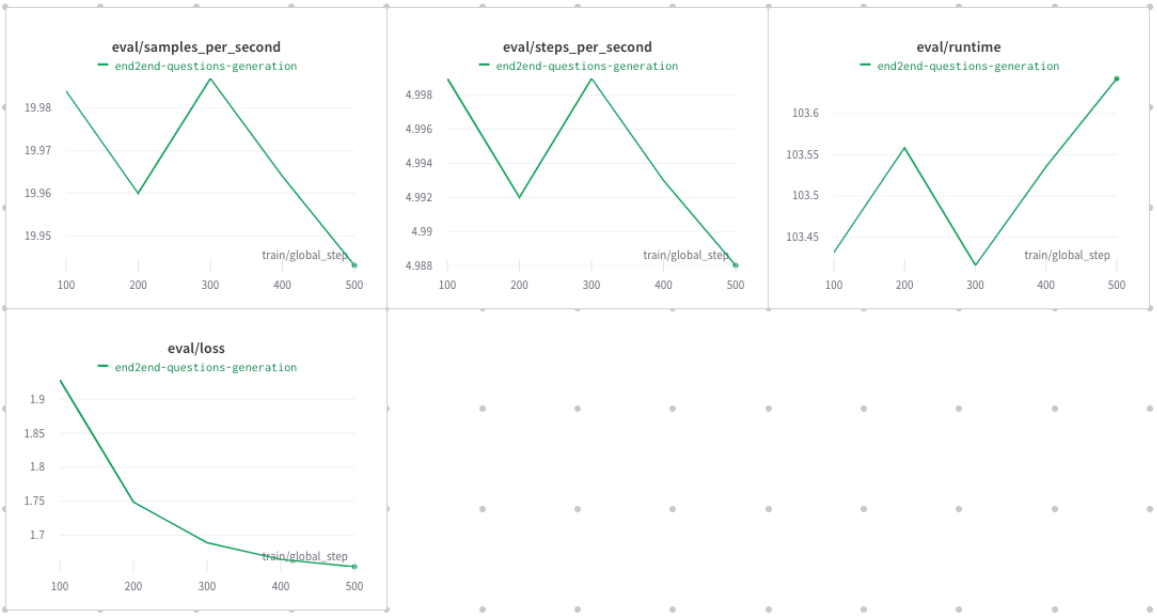


Figure 6.5: Testing Report

Chapter 7

RESULTS

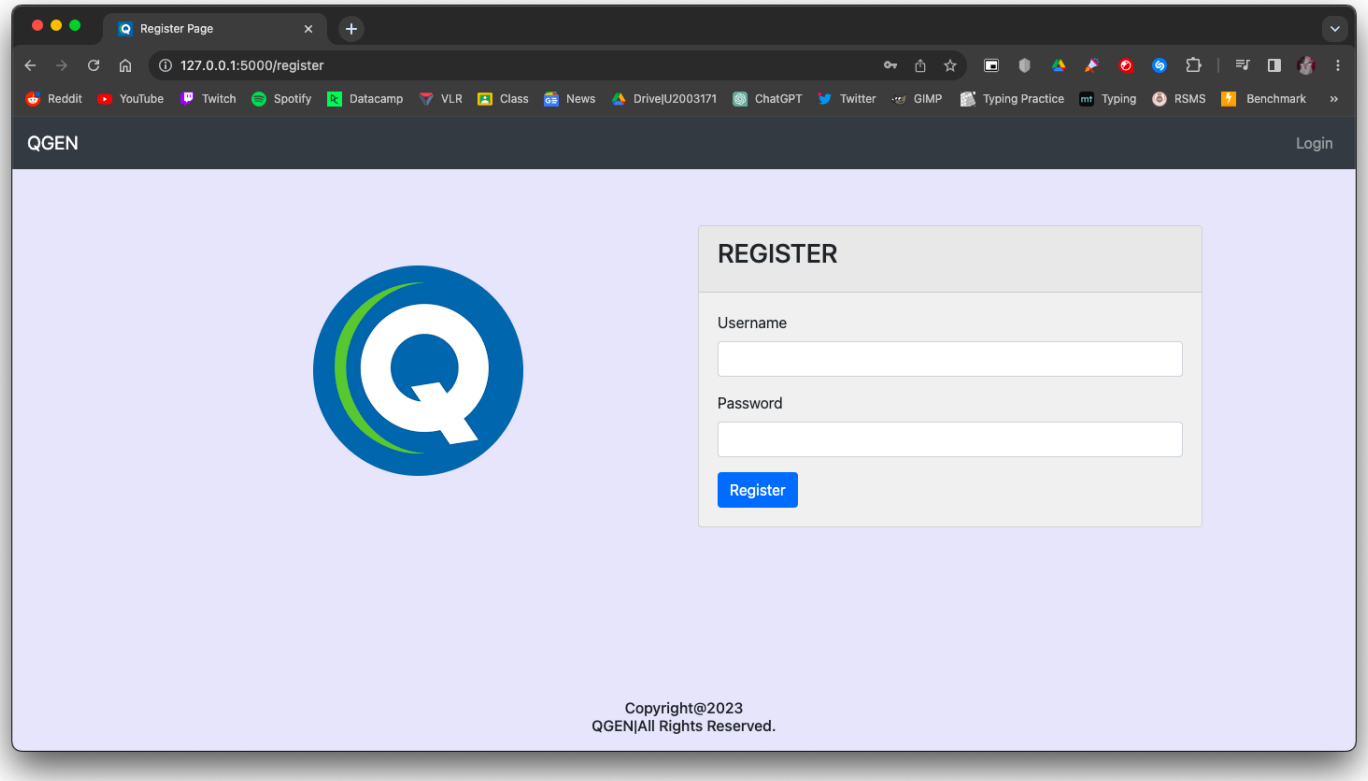


Figure 7.1: Register page

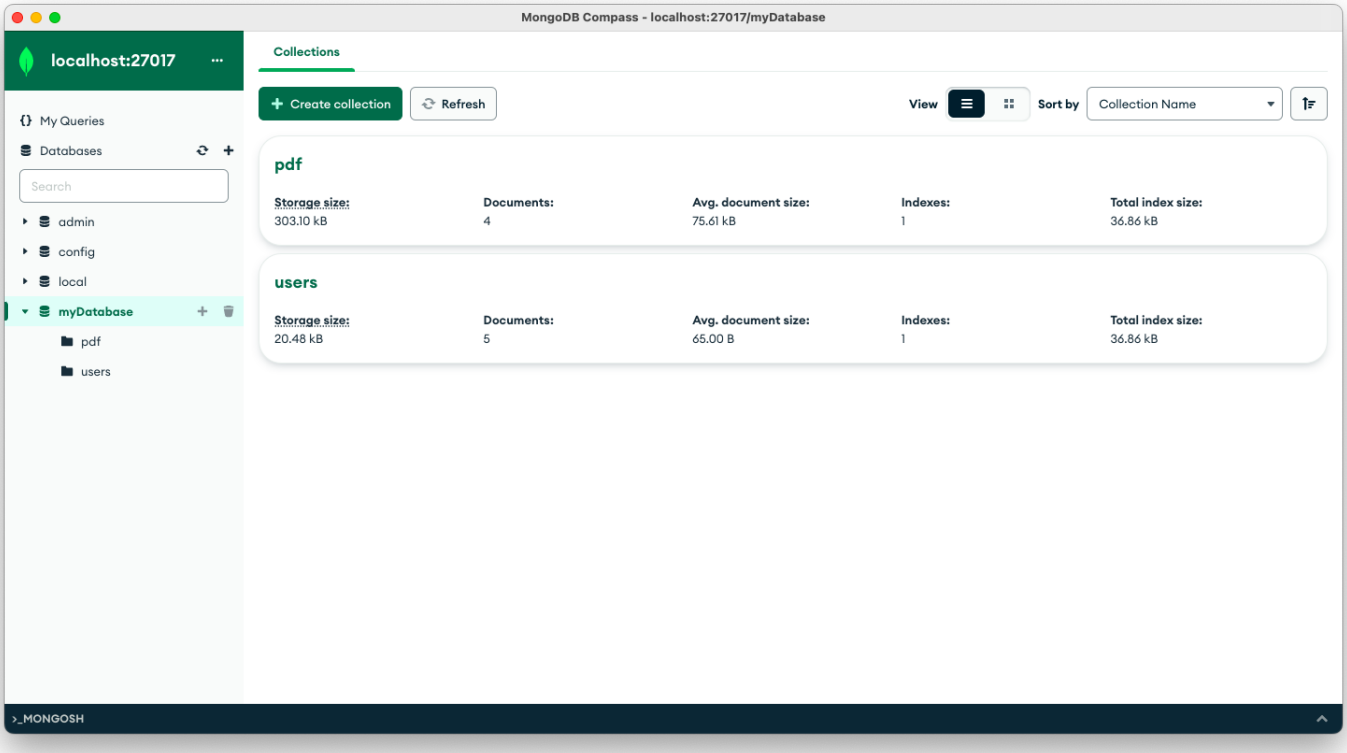


Figure 7.2: Database

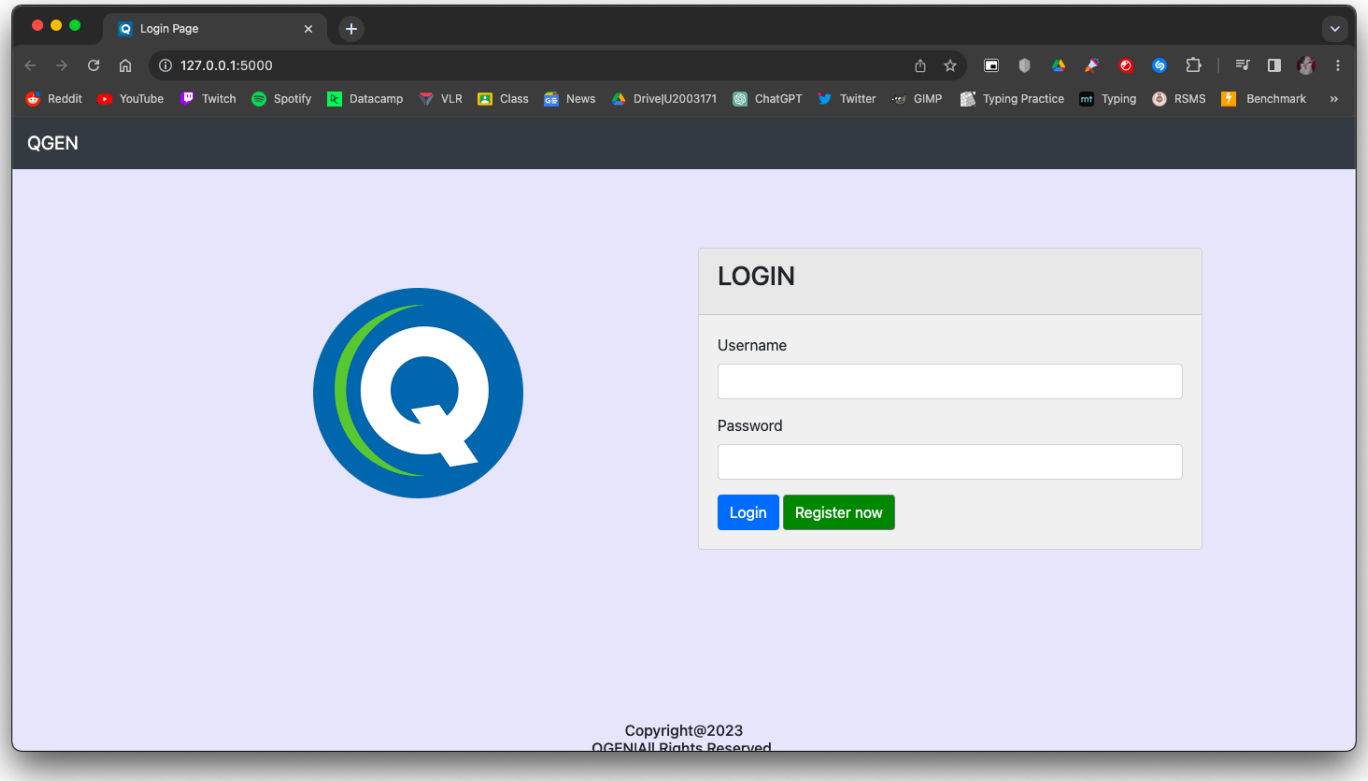


Figure 7.3: Login Page

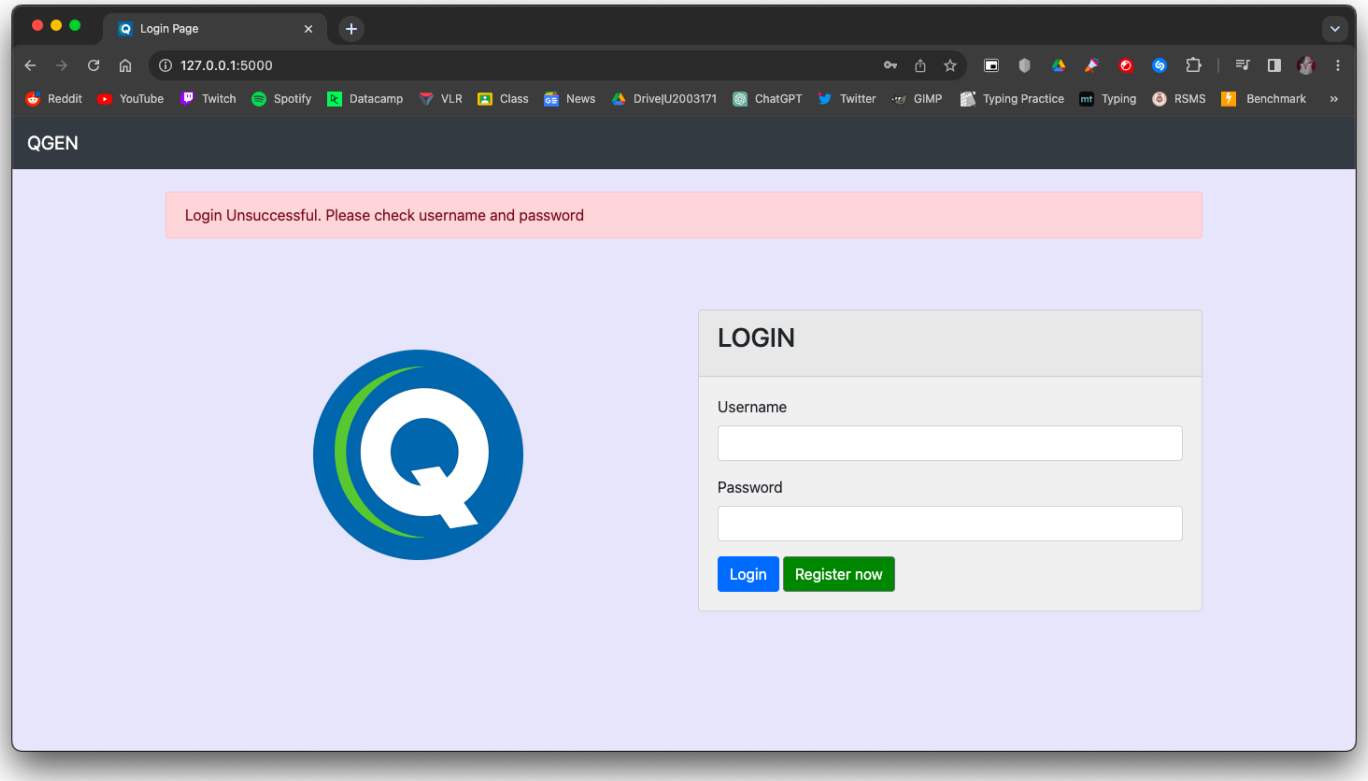


Figure 7.4: Login Unsuccessful

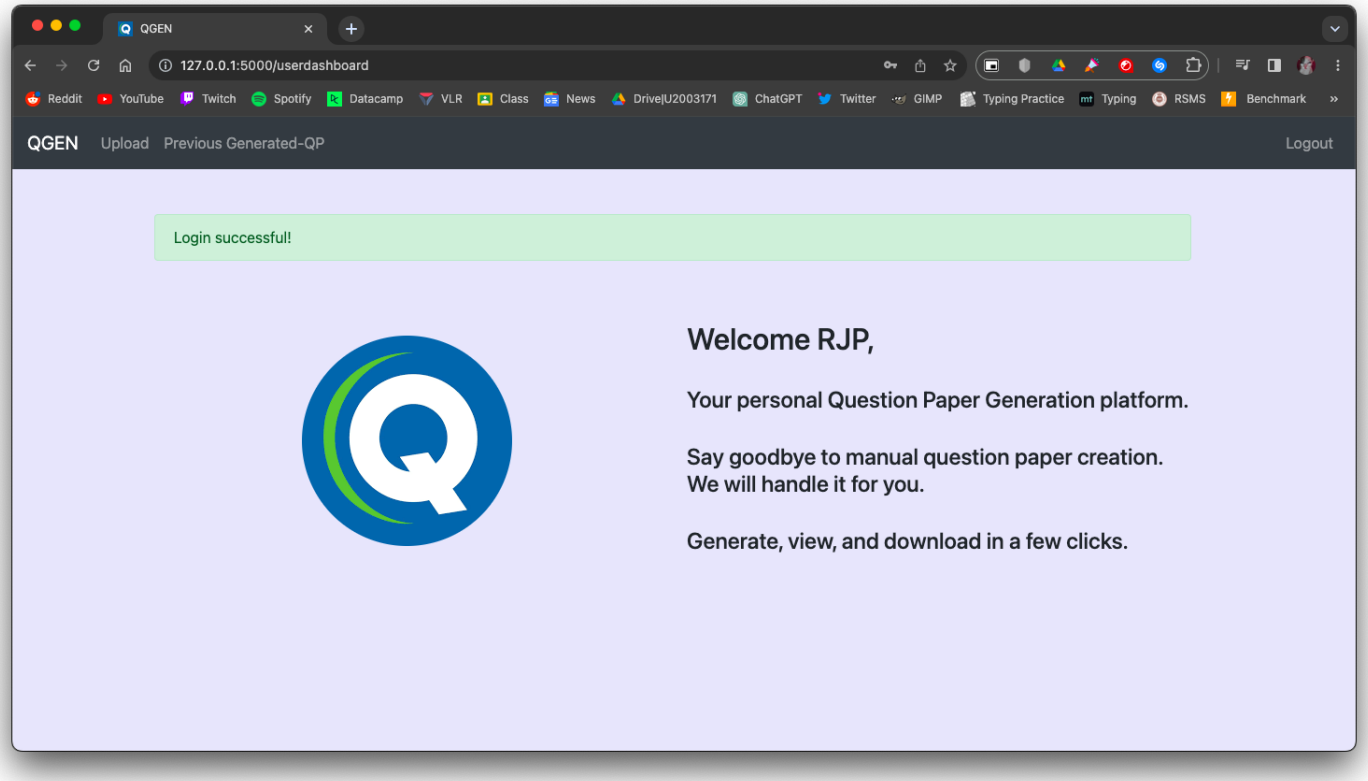


Figure 7.5: Dashboard

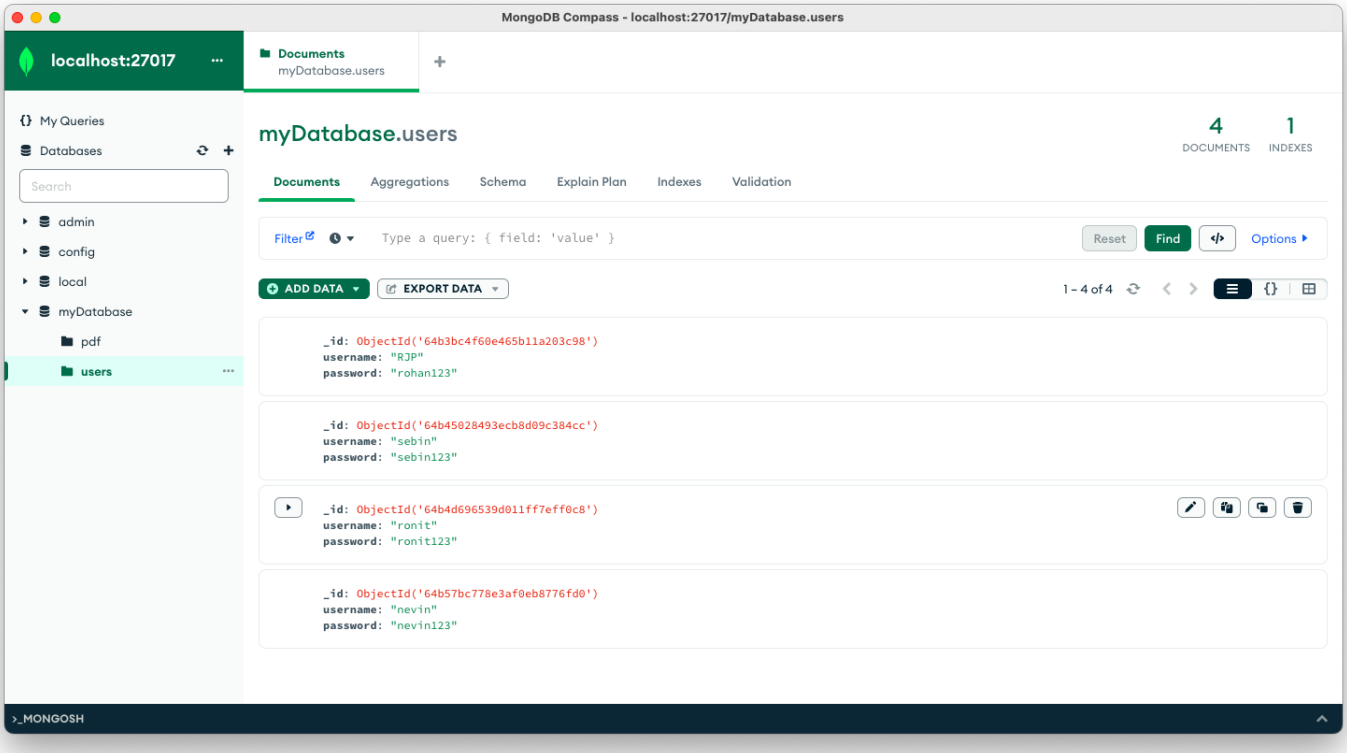


Figure 7.6: Users Collection

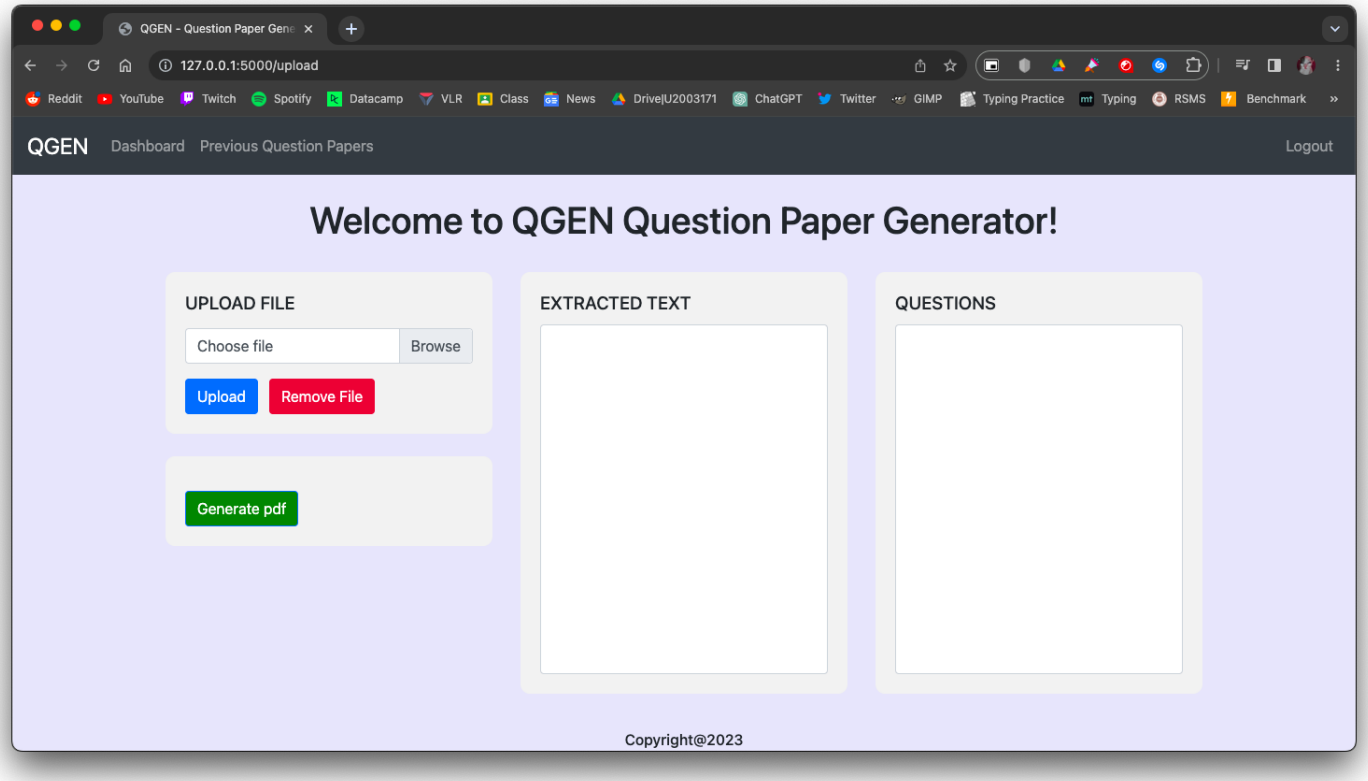


Figure 7.7: Upload Page

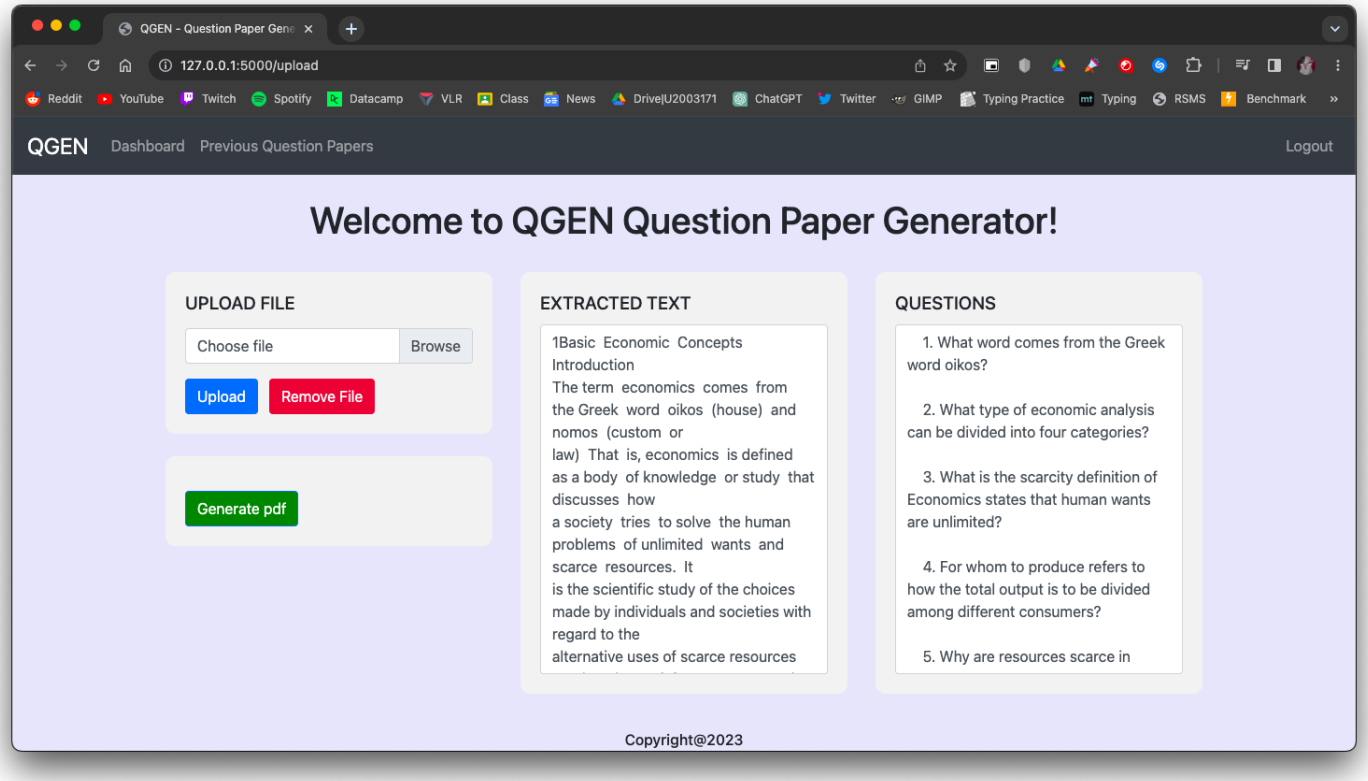


Figure 7.8: Upload Page after question generation

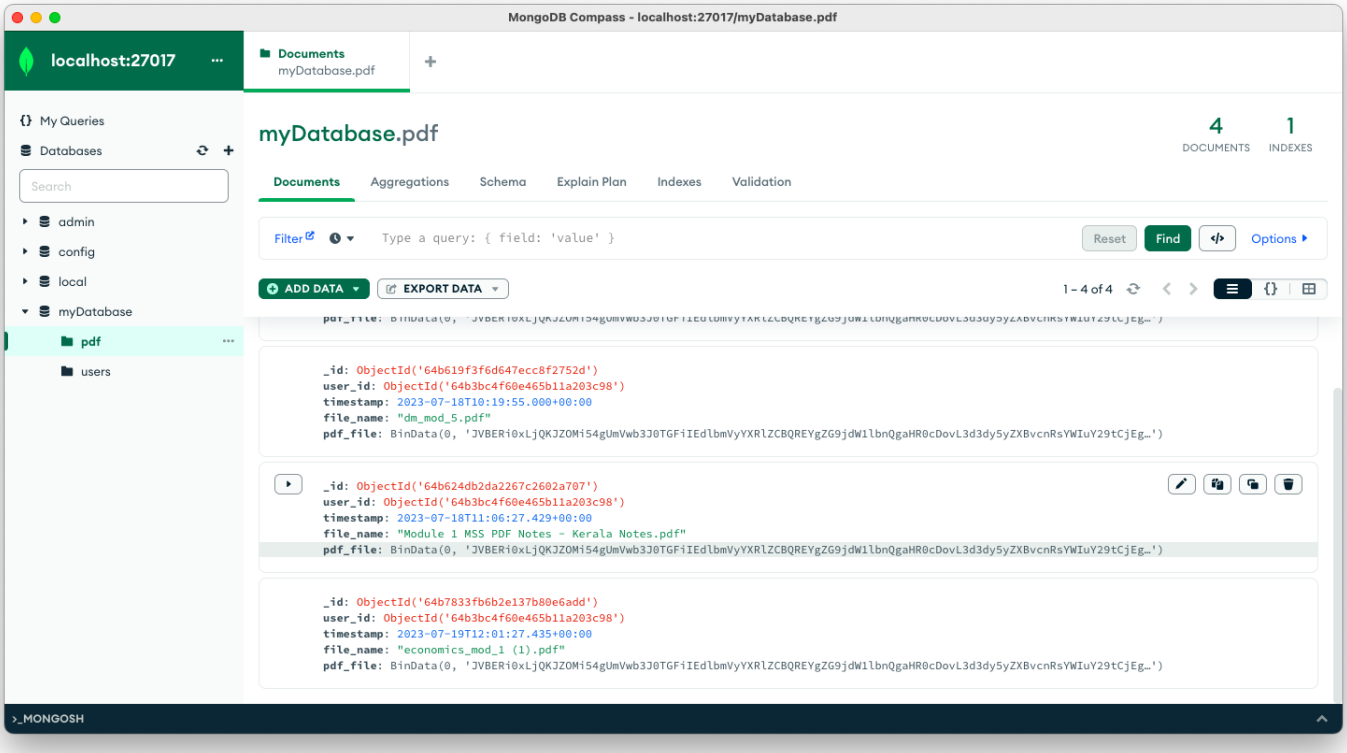


Figure 7.9: PDF Collection

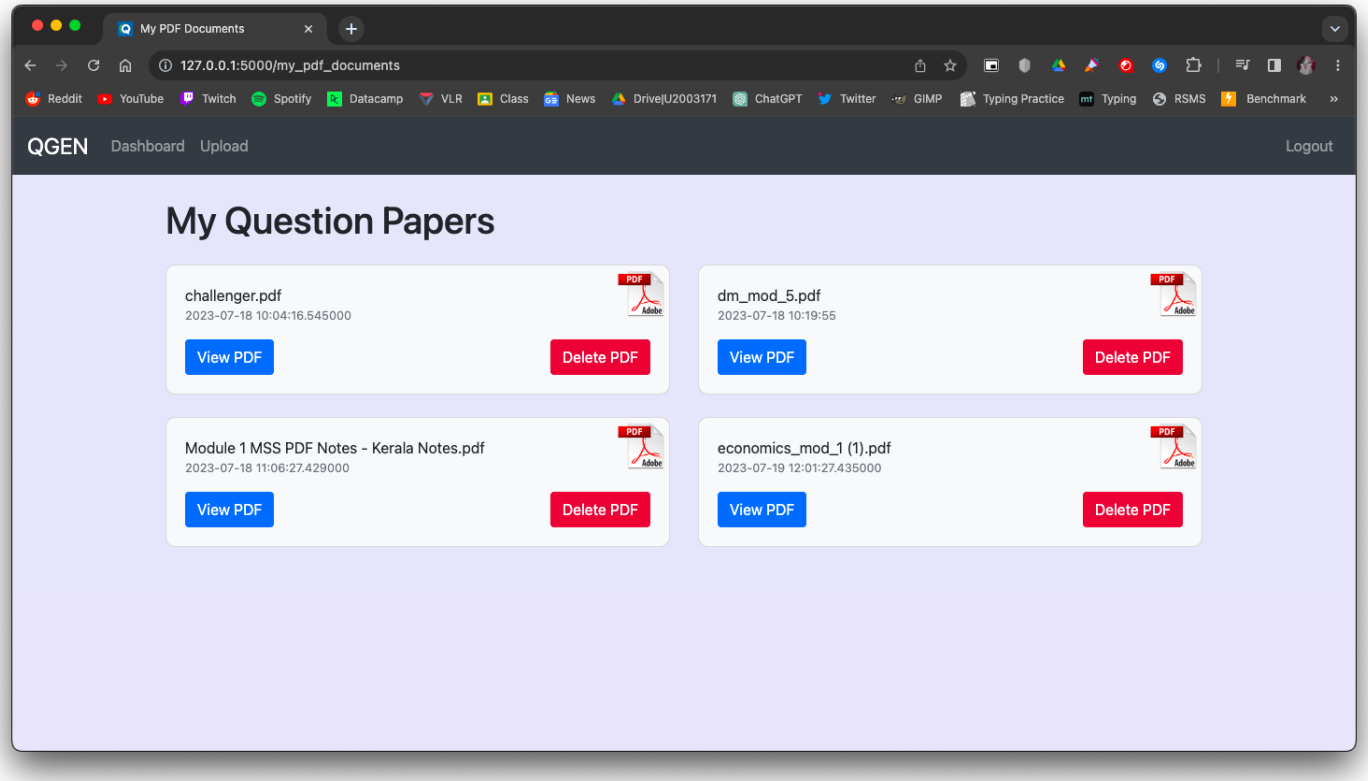


Figure 7.10: Previously Generated Question Papers

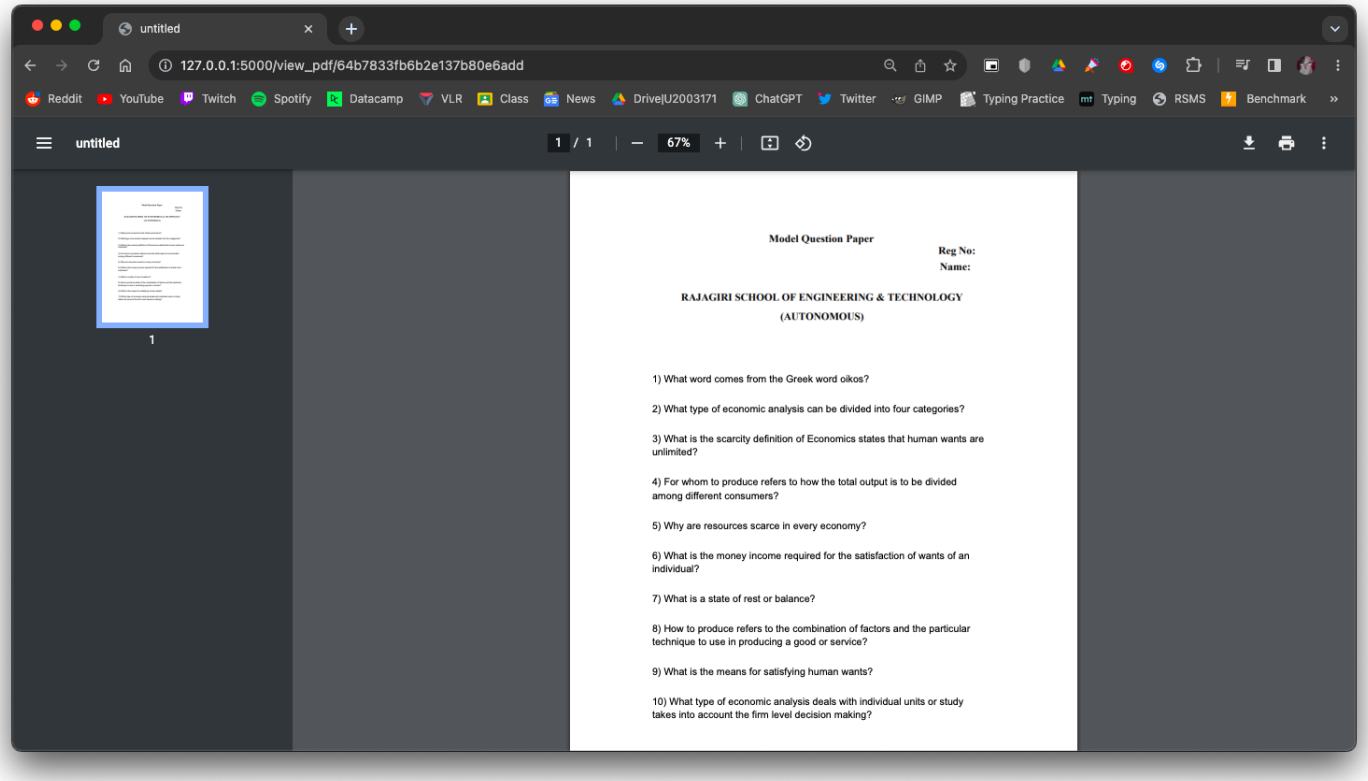


Figure 7.11: Generated Question Paper

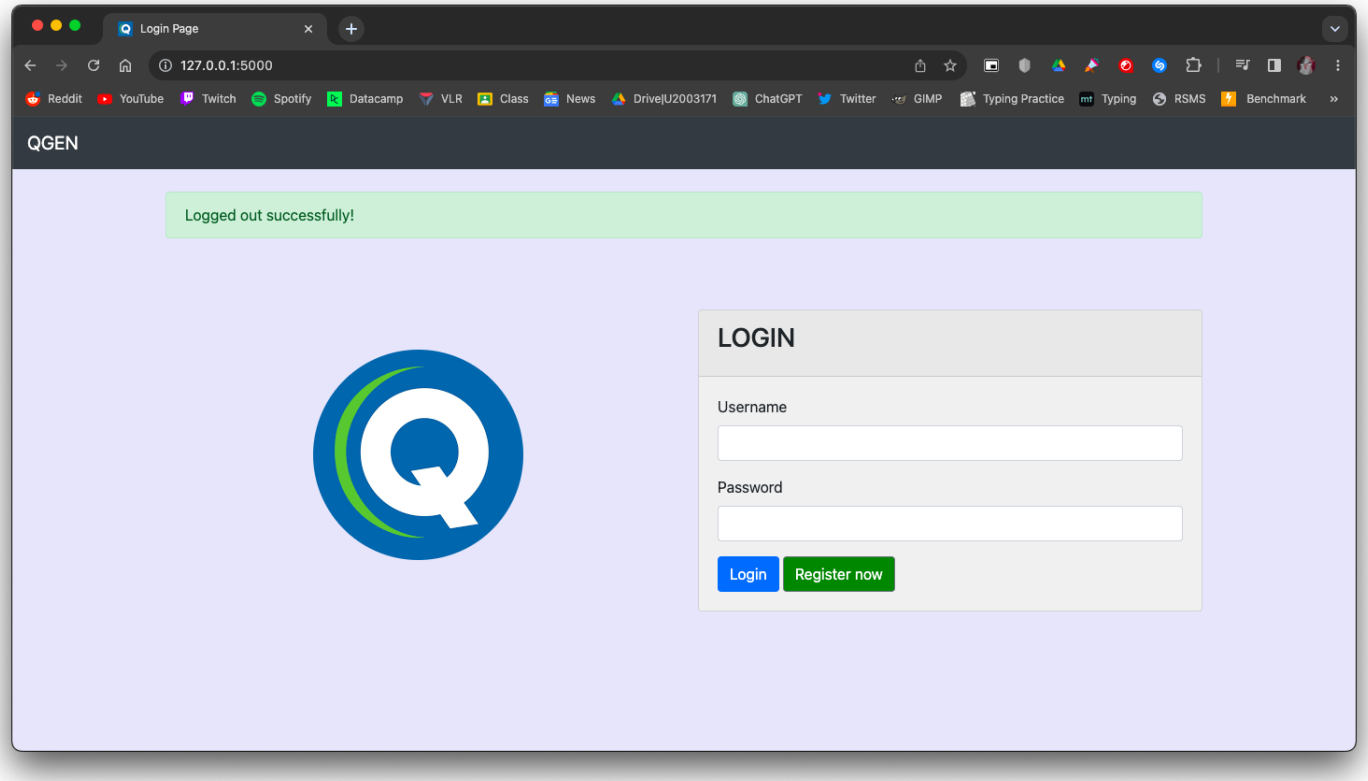


Figure 7.12: Screen after logging out

Chapter 8

RISKS AND CHALLENGES

We faced several realistic challenges for the perfect implementation of QGEN in the given time frame. This includes:

1. Ensuring the accuracy and reliability of the generated questions is crucial.
2. Preventing security breaches, such as unauthorized access to the question paper, is critical to maintain the integrity of the application.
3. The input file must be in a pdf format.
4. The equations present in the input file may not be relevant to the questions generated.
5. The fine tuned T5 model may produce irrelevant questions.
6. Questions may not be diverse.

Chapter 9

CONCLUSION AND FUTURE SCOPE

The project describes an automated system that progresses from the traditional method of paper generation to an automated process by uploading the study material.By utilizing T5 transformers and training them with the SQuAD dataset, relevant questions can be generated. A question paper is generated using these questions and is downloaded in pdf format.

We hope to extend the application by:

1. Input file can be of any file format.
2. Equations in the input file may be used to generate questions.
3. Setting the difficulty level of questions.
4. Allowing user to input number of questions he want and select which questions he want.
5. Generating MCQ questions.
6. Extend other aspects of Bloom's Taxonomy to generate more diverse questions.

REFERENCES

[1] Luis Enrico Lopez, Diane Kathryn Cruz, Jan Christian Blaise Cruz, Charibeth Cheng,"Transformer-based End-to-End Question Generation.[Accessed:May18,2023].[Online].Available:https://arxiv.org/abs/2005.01107

[2] Sahan Bulathwela, Hamze Muse and Emine Yilma,"Scalable Educational Question Generation with Pre-trained Language Model".[Accessed:May18,2023].[Online].Available:https://arxiv.org/abs/2305.07871

APPENDIX A: Sample Code

main.py

```
from questionpapergenerator import app

if __name__ == '__main__':
    app.run(debug=True)
```

……

models.py

```
from flask_pymongo import ObjectId
from questionpapergenerator import mongo

class User:
    def __init__(self, username, password):
        self.username = username
        self.password = password

    @staticmethod
    def from_dict(user_dict):
        username = user_dict.get('username')
        password = user_dict.get('password')
        return User(username, password)

    def to_dict(self):
        return {'username': self.username, 'password': self.password}

    @staticmethod
    def find_by_username(username):
        user_dict = mongo.db.users.find_one({'username': username})
        if user_dict:
            return User.from_dict(user_dict)
        return None

    def save(self):
        user_dict = self.to_dict()
        mongo.db.users.insert_one(user_dict)
```

……

__init__.py

```
from flask import Flask
from flask_pymongo import PyMongo

app = Flask(__name__)
app.config["MONGO_URI"] = "mongodb://localhost:27017/myDatabase"
app.config['SECRET_KEY'] = '5791628bb0b13ce0c676dfde280ba245'
db = PyMongo(app).db
mongo = PyMongo(app)
pdf_collection= mongo.db.pdf
```

users_collection = mongo.db.users

from questionpapergenerator import routes

……

login.html

```
<!DOCTYPE html>
<html>

<head>
  <title>Login Page</title>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <link rel="icon"

href="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6M"
    type="image/gif" sizes="16x16">
  <style>
    body {

      background-color: #e6e6fa;

    }

    .card {
      margin-top: 60px;
    }
  </style>
</head>

<body>
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
    <a class="navbar-brand" href="{{url_for('login')}}">QGEN</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
      aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <!-- Remove the <ul> for the "Logout" option -->
    </div>
  </nav>
  <div class="container">
    <div class="row align-items-center">
      <div class="col-md-12">
        <br>
        {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
        {%for category, message in messages %}
```

```
        <div class="alert alert-{{category}}">
          {{message}}
        </div>
        {% endfor %}
        {% endif %}
        {% endwith %}
      </div>
      <div class="col-md-6 d-flex justify-content-center pt-5">
        <img

src="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6MA
"   class="img-fluid rounded-circle">
      </div>
      <div class="col-md-6 text-black">
        <div class="card" style="background-color: #f0f0f0;"> <!-- Adding a bit more grey to the card -->
          <div class="card-header">
            <h3 class="card-title">LOGIN</h3>
          </div>
          <div class="card-body">
            <form action="/login" method="post">
              <div class="form-group">
                <label for="username">Username</label>
                <input type="text" class="form-control" id="username" name="username" required>
              </div>
              <div class="form-group">
                <label for="password">Password</label>
                <input type="password" class="form-control" id="password" name="password" required>
              </div>
              <button type="submit" class="btn btn-primary">Login</button>
              <a href="/register" class="btn btn-secondary" style="background-color: green;">Register
now</a>
            </form>
          </div>
        </div>
      </div>
    </div>
  </div>
  <footer class="page-footer font-small blue">
    <br>
    <br>
    <br>
    <br>
    <br>
    <br>
    <br>
    <div class="footer-copyright text-center py-3">
      <h6>Copyright@2023<br>QGEN|All Rights Reserved.</h6>
    </div>
```

```
    <!-- Copyright -->

  </footer>
  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</body>
</html>
```

……

register.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Register Page</title>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
   <link rel="icon"
href="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6MA
" type="image/gif" sizes="16x16">
  <style>
    body {

        background-color: #e6e6fa;

    }
    .card{
        margin-top: 60px;

    }
  </style>
</head>
<body>
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <a class="navbar-brand" href="{{url_for('login')}}">QGEN</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav ml-auto">
                <li class="nav-item">
                    <a class="nav-link" href="{{url_for('login')}}">Login</a>
                </li>
            </ul>
        </div>
    </nav>
    <div class="container">
        <div class="row align-items-center">
```

```
          <div class="col-md-6 d-flex justify-content-center pt-5">
             <img
src="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6M"
              alt="" class="img-fluid rounded-circle">
          </div>
          <div class="col-md-6 text-black">
             <div class="card" style="background-color: #f0f0f0;">
                <div class="card-header">
                  <h3 class="card-title">REGISTER</h3>
                </div>
                <div class="card-body">
                  <form action="/register" method="post">
                    <div class="form-group">
                      <label for="username">Username</label>
                      <input type="text" class="form-control" id="username" name="username" required>
                    </div>
                    <div class="form-group">
                      <label for="password">Password</label>
                      <input type="password" class="form-control" id="password" name="password" required>
                    </div>
                    <button type="submit" class="btn btn-primary">Register</button>
                  </form>
                </div>
              </div>
          </div>
       </div>
    </div>
    <footer class="page-footer font-small blue">
       <br>
       <br>
       <br>
       <br>
       <br>
       <br>
       <br>
      <div class="footer-copyright text-center py-3">
       <h6>Copyright@2023<br>QGEN|All Rights Reserved.</h6>
      </div>
      <!-- Copyright -->

    </footer>
  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</body>
</html>
,,.....
```

dashboard.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>QGEN</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
    <link rel="icon"
href="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6MA
" type="image/gif" sizes="16x16">

    <style>
        body {
            background-color: #e6e6fa;
        }
        .q{

            padding-left: 17px;

        }
        .navhead {
            background-color: #343a40;
        }

    </style>
</head>


<body>
<header>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <a class="navbar-brand" href="{{url_for('user_dashboard')}}">QGEN</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav">
                <li class="nav-item ">
                    <a class="nav-link" href="{{url_for('upload')}}">Upload</a>
                </li>
                <li class="nav-item">
                     <a class="nav-link" href="{{url_for('my_pdf_documents')}}">Previous Generated-QP</a>
                </li>
```

```
          </ul>
          <ul class="navbar-nav ml-auto">
            <li class="nav-item">
              <a class="nav-link" href="{{url_for('logout')}}">Logout</a>
            </li>
          </ul>
        </div>
    </nav>
</header>
<section class="p-4 pr-5 bg">
    <div class="container">
        <br>
        <div class="row align-items-center">
            <div class="col-md-12">
                {% with messages = get_flashed_messages(with_categories=true) %}
                {% if messages %}
                {%for category, message in messages %}
                <div class="alert alert-{{category}}">
                 {{message}}
                </div>
                {% endfor %}
                {% endif %}
                {% endwith %}
            </div>
            <div class="col-md-6 d-flex justify-content-center pt-5">
                <img
src="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6M"
                 alt="" class="img-fluid rounded-circle">
            </div>
            <div class="col-md-6 text-black">
                <br>
                <br>
                <h2>Welcome {{username}},</h2>
                <br>
                <h4>Your personal Question Paper Generation platform.</h4>
                <br>
                <h4>Say goodbye to manual question paper creation. We will handle it for you.</h4>
                <br>
                <h4>Generate, view, and download in a few clicks.</h4>

            </div>
        </div>
    </div>
</section>
<footer class="page-footer font-small blue">
    <br>
    <br>
    <br>
```

```
    <br>
    <br>
    <br>
    <br>
    <br>
  <div class="footer-copyright text-center py-3">
    <h6>Copyright@2023<br>QGEN|All Rights Reserved.</h6>
  </div>
  <!-- Copyright -->

</footer>
<!-- Footer -->

    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
</body>
</html>
```

……

upload.html

```
<!DOCTYPE html>
<html>
<head>
    <title>QGEN - Question Paper Generator</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <link rel="icon"
href="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6MA
" type="image/gif" sizes="16x16">
    <style>
        body {
            background-color: #e6e6fa;
        }

        .navbar {
            background-color: #343a40;
        }

        .navbar-brand {
            font-size: 1.5rem;
        }
```

```
        .card {
            background-color: #f8f9fa;
            border-radius: 10px;
        }

        .card-title {
            font-size: 1.2rem;
        }

        textarea {
            resize: none;
        }

        .card-custom {
            background-color: #f2f2f2;
            border-color: #f2f2f2;
        }
    </style>
    <script>
        function handleFileInput() {
            var fileInput = document.getElementById('fileInput');
            var fileNameLabel = document.getElementById('fileNameLabel');
            fileNameLabel.textContent = fileInput.files[0].name;
        }

        function removeFile() {
            var fileInput = document.getElementById('fileInput');
            fileInput.value = "";
            var fileNameLabel = document.getElementById('fileNameLabel');
            fileNameLabel.textContent = "Choose file";
        }

        function generatePdf() {
            fetch('/generate_pdf')
                .then(response => response.text())
                .then(data => {
                 alert(data);
                })
                .catch(error => {
                 console.error('Error:', error);
                });
        }

    </script>
</head>
<body>

    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
```

```
      <a class="navbar-brand" href="{{url_for('user_dashboard')}}">QGEN</a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav">
          <li class="nav-item">
            <a class="nav-link" href="{{url_for('user_dashboard')}}">Dashboard</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{{url_for('my_pdf_documents')}}">Previous Question Papers</a>
          </li>
        </ul>
        <ul class="navbar-nav ml-auto">
          <li class="nav-item">
            <a class="nav-link" href="{{url_for('logout')}}">Logout</a>
          </li>
        </ul>
      </div>
    </nav>

    <!-- Content -->
    <div class="container mt-4">
      <h1 class="text-center">Welcome to QGEN Question Paper Generator!</h1>
      <div class="row justify-content-center py-4">
        <!-- Column 1 -->
        <div class="col-md-4">
          <div class="card card-custom">
            <div class="card-body">
              <h5 class="card-title">UPLOAD FILE</h5>
              <div class="mt-3">
                <form method="POST" enctype="multipart/form-data">
                  <div class="custom-file">
                    <input type="file" class="custom-file-input" id="fileInput" name="file" accept=".pdf"
onchange="handleFileInput()">
                    <label class="custom-file-label" for="fileInput" id="fileNameLabel">Choose
file</label>
                  </div>
                  <br>
                  <div class="mt-3">
                    <button type="submit" class="btn btn-primary">Upload</button>
                    <button class="btn btn-danger ml-2" onclick="removeFile()">Remove File</button>
                  </div>
                </form>
              </div>
            </div>
          </div>
          <br>
```

```
            <div class="card card-custom">
               <div class="card-body">
                  <div class="mt-3">
                     <button class="btn btn-primary" style="background-color: green;"
onclick="window.location.href='{{ url_for('generate_pdf') }}'">Generate pdf</button>
                  </div>
               </div>
            </div>
         </div>

         <!-- Column 2 -->
         <div class="col-md-4">
            <div class="card card-custom">
               <div class="card-body">
                  <h5 class="card-title">EXTRACTED TEXT</h5>
                  <textarea class="form-control" rows="15">{{text1}}</textarea>
               </div>
            </div>
         </div>

         <!-- Column 3 -->
         <div class="col-md-4">
            <div class="card card-custom">
               <div class="card-body">
                  <h5 class="card-title">QUESTIONS</h5>
                  <textarea class="form-control" rows="15"id="questionTextarea">{% for question in text3
%}
   {{ loop.index }}. {{ question }}
   {% endfor %}
                  </textarea>
               </div>
            </div>
         </div>
      </div>
   </div>
   <footer class="page-footer font-small blue">
     <div class="footer-copyright text-center py-3">
       <h6>Copyright@2023<br>QGEN|All Rights Reserved.</h6>
     </div>
     <!-- Copyright -->

   </footer>
   <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</body>
</html>
```

……

my_pdf_documents.html

```
<!DOCTYPE html>
```

```
<html>
<head>
    <title>My PDF Documents</title>
    <!-- Add Bootstrap CSS -->
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">
    <link rel="icon"
href="
VEUgaqf///92wEUgaqn///3///x4v0Uiaan9//8ga6Z2wEb///p2wUIiaaYea6chaaoAX596xD8AY6UAYKUga6M"
type="image/gif" sizes="16x16">


   <style>
        body {
            background-color: #e6e6fa;
        }

        .navbar {
            background-color: #343a40;
        }

        .navbar-brand {
            font-size: 1.5rem;
        }

        .card {
            background-color: #f8f9fa;
            border-radius: 10px;
        }

        .card-title {
            font-size: 1.2rem;
        }

        textarea {
            resize: none;
        }

        .card-custom {
            background-color: #f2f2f2;
            border-color: #f2f2f2;
        }

        .file-name-left {
            font-size: 1rem;
            text-align: left;
            margin-bottom: 0;
        }

        .timestamp-left {
```

```
            font-size: 0.8rem;
            color: #6c757d;
            text-align: left;
            margin-top: 0;
        }

        .view-pdf-remove-file {
            display: flex;
            justify-content: space-between;
            align-items: center;
            margin-top: 10px;
        }

        .view-pdf-button {
            margin-right: 10px;
        }

        .card-body {
        position: relative;
        }

        .pdf-icon {
        position: absolute;
        top: 0;
        right: 0;
        width: 50px;
        height: 50px;
        margin: 5px;
        }
    </style>
</head>
<body>

    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <a class="navbar-brand" href="{{url_for('user_dashboard')}}">QGEN</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav">
                <li class="nav-item">
                    <a class="nav-link" href="{{url_for('user_dashboard')}}">Dashboard</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="{{url_for('upload')}}">Upload</a>
                </li>
            </ul>
```

```
        </div>
        <!-- Logout Button -->
        <ul class="navbar-nav ml-auto">
            <li class="nav-item">
                <a class="nav-link" href="{{url_for('logout')}}">Logout</a>
            </li>
        </ul>
    </nav>

    <div class="container">
        <h1 class="mt-4 mb-4">My Question Papers</h1>
        <div class="row">
            {% for pdf_doc in pdf_documents %}
                <div class="col-md-6">
                    <div class="card mb-4">
                        <div class="card-body">
                            <div class="file-timestamp">
                                <p class="file-name-left">{{ pdf_doc.file_name if pdf_doc.file_name else file_name
}}</p>
                                <p class="timestamp-left">{{ pdf_doc.timestamp }}</p>
                            </div>
                            <div class="view-pdf-remove-file">
                                <a href="/view_pdf/{{ pdf_doc._id }}" class="btn btn-primary view-pdf-button">View
PDF</a>
                                <form action="/remove_pdf/{{ pdf_doc._id }}" method="post">
                                    <button type="submit" class="btn btn-danger">Delete PDF</button>
                                </form>
                            </div>
                            <img
src="https://www.biochek.com/wp-content/uploads/2018/07/adobe-pdf-icon-logo-png-transparent.png"
alt="PDF Icon" class="pdf-icon">
                        </div>
                    </div>

                </div>
            {% endfor %}
        </div>
    </div>
    <footer class="page-footer font-small blue">
        <br>
        <br>
        <br>
        <br>
        <br>
        <br>
        <br>
        <br>
        <br>
        <br>
```

```
        <br>
        <br>
      <div class="footer-copyright text-center py-3">
        <h6>Copyright@2023<br>QGEN|All Rights Reserved.</h6>
      </div>
      <!-- Copyright -->

    </footer>
    <!-- Add Bootstrap JS (optional) -->
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"></script>
</body>
</html>
```

……

routes.py

```
from PyPDF2 import PdfReader
from flask import render_template, url_for, flash, redirect, request,session,send_file
from questionpapergenerator import app, users_collection , pdf_collection
import re
from transformers import AutoModelForSeq2SeqLM, AutoTokenizer, T5ForConditionalGeneration,
T5TokenizerFast
from reportlab.lib.pagesizes import letter
from reportlab.pdfgen import canvas
from reportlab.lib.units import inch
import torch,random
model_1=AutoModelForSeq2SeqLM.from_pretrained("rohan-jp1/t5-end2end-question-generation")

def extract_text_from_pdf(file_path):
    reader = PdfReader(file_path)
    text = ''
    for page in reader.pages:
        text += page.extract_text()
    return text

def preprocess_text(text, segment_length=1700):
    # Remove leading and trailing whitespace
    text = text.strip()

    # Replace bullet points with a space
    text = re.sub(r'\s*•\s*', ' ', text)

    # Replace newlines and multiple whitespaces with a single space
    text = ' '.join(text.split())

    # Split the text into segments of specified length
    segments = [text[i:i+segment_length] for i in range(0, len(text), segment_length)]

    return segments
```

```
checkpoint = "t5-base"
tokenizer = T5TokenizerFast.from_pretrained(checkpoint)

model = AutoModelForSeq2SeqLM.from_pretrained("rohan-jp1/t5-end2end-questions-generation")

import random

def hf_run_model(input_list, num_return_sequences=8, num_questions=2, max_sequence_length=512,
generator_args=None):
    if generator_args is None:
        generator_args = {
            "max_length": max_sequence_length,
            "num_beams": 10,
            "length_penalty": 1.5,
            "no_repeat_ngram_size": 6,
            "early_stopping": True,
            "temperature": 0.8,  # Adjust the temperature value (higher values for more randomness)
            "top_k": 50,  # Adjust the top_k value (higher values for more diverse output)
            "top_p": 0.95  # Adjust the top_p value (lower values for more focused output)
        }

    generated_questions = []
    unique_questions = set()

    #creating tensors of each input
    for input_string in input_list:
        input_string = "generate questions: " + input_string + " </s>"
        input_ids = tokenizer.encode(input_string, truncation=True, max_length=max_sequence_length,
return_tensors="pt")

        # Generate questions using the model
        res = model.generate(input_ids, **generator_args, num_return_sequences=num_return_sequences)
        output = tokenizer.batch_decode(res, skip_special_tokens=True,
clean_up_tokenization_spaces=True)

        segment_questions = []
        for sequence in output:
            sequence = sequence.split("<sep>")
            questions = [question.strip() + "?" for question in sequence[0].split("?") if question.strip()]
            segment_questions.extend(questions[:num_questions])  # Selecting the desired number of
questions from each segment

        # Filter out single-word questions for each segment
        segment_questions = [question for question in segment_questions if len(question.split()) > 1]
        generated_questions.extend(segment_questions)

    # Randomly sample questions until reaching the desired number of non-repeated questions
```

```
    while len(unique_questions) < num_questions * len(input_list):  # Generating questions from each
segment
        question = random.choice(generated_questions)
        generated_questions.remove(question)
        if question not in unique_questions:
            unique_questions.add(question)

    return list(unique_questions)


import datetime
import os
from reportlab.lib.pagesizes import letter
from reportlab.pdfgen import canvas
from reportlab.lib import colors
from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
from reportlab.platypus import Paragraph
from io import BytesIO

def convert_list_to_pdf_with_template(data_list, output_file):
    # Create the PDF canvas
    c = canvas.Canvas(output_file, pagesize=letter)

    # Set the font and size
    c.setFont("Helvetica", 12)

    # Add the template or background image
    template_path = 'template.png'
    c.drawImage(template_path, 0, 0, width=letter[0], height=letter[1])

    # Set up paragraph styles
    styles = getSampleStyleSheet()
    paragraph_style = ParagraphStyle(
        'normal',
        parent=styles['Normal'],
        textColor=colors.black,
        fontSize=12,
        leading=16  # Adjust the leading for more spacing between lines
    )

    # Write the list elements to the PDF
    y = 550  # Starting y position
    index = 1
    spacing = 20  # Fixed spacing between paragraphs

    for item in data_list:
        text = f"{index}) {item}"
        p = Paragraph(text, style=paragraph_style)
        p.wrapOn(c, 400, 0)
```

```
        # Check if there's enough space on the page for the paragraph
        if y - p.height < 50:
            c.showPage()  # Start a new page
            y = 750  # Reset the y position to the top of the new page

        p.drawOn(c, 100, y-p.height)
        y -= p.height + spacing  # Adjust the spacing between paragraphs
        index += 1

    # Save the canvas as the final PDF
    c.save()
    # Save the PDF file into MongoDB
    with open(output_file, 'rb') as pdf_file:
        pdf_data = pdf_file.read()
    username = session.get('username')  # Get the username from session or any relevant source

    user = users_collection.find_one({'username': username})  # Retrieve the user document from
MongoDB
    if user:
        user_id = user['_id']  # Assuming the user ID is stored in the '_id' field
        timestamp = datetime.datetime.now()  # Generate a timestamp
        file_name = session.get('file_name')
        pdf_document = {
            "user_id": user_id,
            "timestamp": timestamp,
            "file_name": file_name,
            "pdf_file": pdf_data
        }
        pdf_collection.insert_one(pdf_document)
        print("PDF saved to MongoDB successfully.")
    else:
        print("User not found. PDF not saved.")

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        existing_user = users_collection.find_one({'username': username})

        if existing_user:
            return "Username already exists!"

        user = {'username': username, 'password': password}
        users_collection.insert_one(user)
        session['username'] = username
        return redirect('/')
    else:
```

```
        return render_template('register.html')

@app.route('/')
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        existing_user = users_collection.find_one({'username': username, 'password': password})

        if existing_user:
            session['username'] = username
            flash('Login successful!', 'success')
            return redirect('/userdashboard')
        else:
            flash('Login Unsuccessful. Please check username and password', 'danger')
            return redirect('/')
    else:
        return render_template('login.html')

@app.route('/logout')
def logout():
    session.pop('username', None)
    flash('Logged out successfully!', 'success')
    return redirect('/')

@app.route("/userdashboard")
def user_dashboard():
    username = session.get('username')
    return render_template('dashboard.html',username=username)

import os
@app.route('/upload', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        file = request.files['file']
        if file:
            # Save the uploaded file to a temporary directory
            temp_dir = '/tmp'
            file_path = os.path.join(temp_dir, file.filename)
            file.save(file_path)
            session['file_name'] = file.filename
            # Perform text extraction
            extracted_text = extract_text_from_pdf(file_path)
            # Delete the temporary file
            os.remove(file_path)
            # Continue with text processing
            preprocessed_text = preprocess_text(extracted_text, segment_length=1700)
            print(len(preprocessed_text))
```

```
            questions = hf_run_model(preprocessed_text, num_return_sequences=8, num_questions=2)
            session['my_list'] = questions
            for count, ele in enumerate(questions):
                print(count + 1)
                print(ele)
            print(type(questions))
            return render_template('upload.html', file_name=file.filename, text1=extracted_text,
                                   text2=preprocessed_text, text3=questions)

    return render_template('upload.html')
@app.route('/generate_pdf', methods=['GET'])
def generate_pdf():
    items_1=session['my_list']
    output_path='output.pdf'
    convert_list_to_pdf_with_template(items_1,output_path)
    return redirect('/my_pdf_documents')

def fetch_pdf_documents_for_user(username):
    user = users_collection.find_one({'username': username})  # Retrieve the user document from
MongoDB
    if user:
        user_id = user['_id']  # Assuming the user ID is stored in the '_id' field
        pdf_documents = pdf_collection.find({'user_id': user_id})  # Fetch all PDF documents for the user
        return pdf_documents
    else:
        return None
import io
from bson import ObjectId

@app.route('/view_pdf/<pdf_id>')
def view_pdf(pdf_id):
    pdf_doc = pdf_collection.find_one({'_id': ObjectId(pdf_id)})
    if pdf_doc:
        pdf_file = pdf_doc['pdf_file']
        pdf_buffer = io.BytesIO(pdf_file)
        return send_file(pdf_buffer,mimetype='application/pdf')
    return "PDF not found"

@app.route('/my_pdf_documents')
def my_pdf_documents():
    username = session['username']  # Get the username from session or any relevant source
    if username:
        pdf_documents = fetch_pdf_documents_for_user(username)
        return render_template('my_pdf_documents.html', pdf_documents=pdf_documents)

    return "User not logged in"

@app.route('/remove_pdf/<pdf_id>', methods=['POST'])
def remove_pdf(pdf_id):
```

```
    # Remove the PDF file from the database
    pdf_collection.delete_one({'_id': ObjectId(pdf_id)})
    return redirect(url_for('my_pdf_documents'))
```

……

Fine Tuning the T5 model for Question Generation

#Download and install the packages

```
!pip install transformers
!pip install datasets
!pip install sentencepiece
!pip install --upgrade accelerate
!pip install tqdm
!pip install huggingface-cli
!pip install wandb

!sudo apt-get install git-lfs


import torch

from datasets import load_dataset, load_metric, list_metrics
from transformers import AutoModelForSeq2SeqLM, AutoTokenizer, DataCollator,
T5ForConditionalGeneration, T5TokenizerFast

from tqdm import tqdm

from typing import Dict, List, Optional

import dataclasses
from dataclasses import dataclass, field

import logging
import os
import sys

import numpy as np
import torch

from huggingface_hub import notebook_login

from transformers import (
    T5ForConditionalGeneration,
    T5Tokenizer,
```

```
    EvalPrediction,
    DataCollator,
    Trainer,
    TrainingArguments)

from google.colab import files
```

#Connect to Weight and Biases

```
import wandb
wandb.login()

%env WANDB_PROJECT=t5-end-to-end-questions-generation
```

#Connect to Hugging Face

```
notebook_login()
```

#Loading the dataset

We use SQuAD v1.1, but a modified version where questions for a context are concatenated. You need to download the file here, unzip it and upload it in the next cell.

```
files.upload()

raw_dataset = load_dataset("squad_modified_for_t5_qg.py")
```

#Preprocessing the data

```
#loading t5-base model for fine tuning
checkpoint = "t5-base"
model = T5ForConditionalGeneration.from_pretrained(checkpoint)
tokenizer = T5TokenizerFast.from_pretrained(checkpoint)


tokenizer.sep_token = '<sep>'

tokenizer.add_tokens(['<sep>'])
model.resize_token_embeddings(len(tokenizer))
```

```
max_input_length =  2048
max_target_length = 64


# tokenize the examples
def convert_to_features(example_batch):

    input_encodings = tokenizer.batch_encode_plus(example_batch['context'],
                                        max_length=max_input_length,
                                        add_special_tokens=True,
                                        truncation=True,
                                        pad_to_max_length=True)

    target_encodings = tokenizer.batch_encode_plus(example_batch['questions'],
                                         max_length=max_target_length,
                                         add_special_tokens=True,
                                         truncation=True, pad_to_max_length=True)

    encodings = {
        'input_ids': input_encodings['input_ids'],
        'attention_mask': input_encodings['attention_mask'],
        'decoder_input_ids': target_encodings['input_ids']
        ,'decoder_attention_mask': target_encodings['attention_mask']
    }

    return encodings

def add_eos_examples(example):
  example['context'] = example['context'] + " </s>"
  example['questions'] = example['questions'] + " </s>"
  return example


def add_special_tokens(example):
  example['questions'] = example['questions'].replace("{sep_token}", '<sep>')
  return example

tokenized_dataset  = raw_dataset.map(add_eos_examples)
tokenized_dataset = tokenized_dataset.map(add_special_tokens)
tokenized_dataset  = tokenized_dataset.map(convert_to_features,  batched=True)

tokenized_dataset = tokenized_dataset.remove_columns( ["context", "questions"])

train_dataset = tokenized_dataset["train"]
```

```
valid_dataset = tokenized_dataset["validation"]

columns = ['input_ids', 'decoder_input_ids', 'attention_mask', 'decoder_attention_mask']
train_dataset.set_format(type='torch', columns=columns)
valid_dataset.set_format(type='torch', columns=columns)

torch.save(train_dataset, 'train_data.pt')
torch.save(valid_dataset, 'valid_data.pt')
```

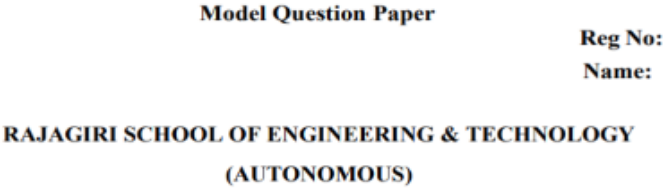
#Fine-Tuning the t5 model

```
@dataclass
class T2TDataCollator():
  def __call__(self, batch: List) -> Dict[str, torch.Tensor]:
    """
    Take a list of samples from a Dataset and collate them into a batch.
    Returns:
    A dictionary of tensors
    """
    input_ids = torch.stack([example['input_ids'] for example in batch])
    lm_labels = torch.stack([example['decoder_input_ids'] for example in batch])
    lm_labels[lm_labels[:, :] == 0] = -100
    attention_mask = torch.stack([example['attention_mask'] for example in batch])
    decoder_attention_mask = torch.stack([example['decoder_attention_mask'] for example in
batch])

    return {
        'input_ids': input_ids,
        'attention_mask': attention_mask,
        'labels': lm_labels,
        'decoder_attention_mask': decoder_attention_mask
    }
training_args = TrainingArguments(output_dir="./gdrive/My Drive/models",
                                  per_device_train_batch_size=4,
                                  per_device_eval_batch_size=4,
                                  gradient_accumulation_steps=16,
                                  learning_rate=1e-4,
                                  num_train_epochs=3,
                                  logging_steps=100,
                                  run_name="end2end-questions-generation",
                                  evaluation_strategy="steps",
                                  save_steps=500,
                                  report_to="wandb",
                                  push_to_hub=True,
```

```
                    push_to_hub_model_id="t5-end2end-questions-generation")

logger = logging.getLogger(__name__)

# Initialize our Trainer
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=valid_dataset,
    data_collator=T2TDataCollator()
)
# Training
trainer.train()

# When training is done, we push the fine-tuned model to the Hub
trainer.push_to_hub("t5-end2end-questions-generation")

wandb.finish()
```

……

template.png



Model Question Paper

Reg No:

Name:

RAJAGIRI SCHOOL OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

APPENDIX B: CO-PO And CO-PSO Mapping

Course Outcome

Sl No.	Description	Blooms Taxonomy Level
CSD334.1	Think innovatively on the development of components, products, processes or technologies in the engineering field.	Knowledge (Level 1) Analyse (Level 4)
CSD334.2	Apply knowledge gained in solving real life engineering problems.	Evaluate (Level 2) Understand (Level 5)

CO-PO Mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CSD334.1	-	3	-	2	-	-	1	-	3	2	1	3
CSD334.2	3	2	3	2	2	-	-	2	3	2	-	1

CO-PSO Mapping

	PSO1	PSO2	PSO3
CSD334.1	3	-	1
CSD334.2	3	3	2

Justifications for CO-PO/PSO Mapping

Mapping	Low/Medium/High	Justification
CSD334.1–PO4	M	Conduct investigations of complex problems : I used research based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
CSD334.1–PO7	L	Environment and sustainability : I understood the impact of the professional engineering solutions in societal and environmental contexts, and demonstrated the knowledge of- and the need for- sustainable developments.
CSD334.1–PO9	H	Individual: We were able to function effectively as an individual, in multi-disciplinary settings.
CSD334.1–PO10	M	Communication : We were able to communicate effectively on complex Engineering activities with the Engineering Community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
CSD334.1–PO11	L	Project Management and finance : Demonstrated knowledge and understanding of the Engineering and management principles and apply these to ones own work, to manage projects and in multi-disciplinary environments.
CSD334.1–PO12	H	Life-long learning : Recognized the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

CSD334.1–PSO1	H	Computer Science Specific Skills : Was able to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas by understanding the core principles and concepts of computer science.
CSD334.1–PSO3	L	Professional Skills : Was able to apply the fundamentals of computer science to formulate competitive research proposals and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur.
CSD334.2–PO1	H	Engineering Knowledge : Applied the knowledge of Mathematics, Science, Engineering fundamentals, and an Engineering discipline to the solution of complex engineering problems.
CSD334.2–PO2	M	Problem analysis : We were able to identify, formulate, review research literature, and analyze complex Engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and Engineering sciences.
CSD334.2–PO3	H	Design/Development of solutions : Designed solutions for complex Engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
CSD334.2–PO4	M	Conduct investigations of complex problems : Used research based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

CSD334.2–PO5	L	Modern Tool usage : Created, selected, and applied appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex Engineering activities with an understanding of the limitations.
CSD334.2–PO8	M	Ethics : Applied ethical principles and commit to professional ethics and responsibilities and norms of the Engineering practice.
CSD334.2–PO9	H	Individual: We were able to function effectively as an individual, and in multi-disciplinary settings.
CSD334.2–PO10	M	Communication : Communicated effectively on com- plex Engineering activities with the Engineering Community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
CSD334.2–PO12	L	Life-long learning : Recognized the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.
CSD334.2–PSO1	H	Computer Science Specific Skills : We were able to identify, analyze and design solutions for complex engineering problems in multi-disciplinary areas by understanding the core principles and concepts of computer science.
CSD334.2–PSO2	H	Programming and Software Development Skills : Acquired programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products.

CSD334.2–PSO3	M	Professional Skills : Applied the fundamentals of computer science to formulate competitive research proposals and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur.