OnBoardIQ -AI-Powered Automated Onboarding System

Name :- Girish Anil Amrutkar

Contact :-9503007889

Email :- girishamrutkar977@gmail.com

Documentation

Table Of Content

1. Introduction

1.1 Problem Statement

The manual onboarding procedure entails completing printed forms, keeping hard copies of them, and manually entering data into the database, which can result in errors and inefficiencies. An automated system that can upload scanned forms, use AI to extract details, and create database records with ease is required to streamline this workflow.

1.2 Objectives

The primary objective of this project is to replace current manual onboarding procedure with a more efficient and automated system. The current workflow is inefficient because it requires a lot of manual labor and is prone to human error. The project intends to improve accuracy, speed, and overall efficiency by utilizing automation and artificial intelligence. In order to successfully meet the requirements and tackle the challenges outlined in the problem statement, the project aims to accomplish the following objectives**.**

- **Automate the Uploading Process :-** To create and implement a system that enables HR staff to upload scanned forms (in PDF or image format), facilitating a seamless shift from paper-based to digital onboarding.

- **AI-Powered Data Extraction:-** To incorporate an artificial intelligence (AI) model that can efficiently retrieve candidate data from scanned forms, minimizing the mistakes that come with manual data entry.
- **Database Integration and Structuring:-** To ensure effective data management and speedy retrieval, a normalized database should be created to hold the extracted data.
- **Streamline HR Workflow:-** To increase the overall efficiency of onboarding by lowering the amount of time and manual labor required for data entry, freeing up HR staff to concentrate on more important duties.
- **Enhance Operational Accuracy:-** To increase the accuracy of the data stored in the system by reducing the possibility of mistakes and inconsistencies during the onboarding process.

2. Tools And Technologies

The development of this project required a combination of tools and frameworks to ensure seamless integration of machine learning, web development, and database management. Each tool was carefully selected to address specific aspects of the workflow, from data extraction to user interaction and storage.

2.1. List Of Tools And Framework

- **Machine Learning Tools :-**
  a. Python
  b. OCR
  c. Pytesseract
- Web Development Tools:-
  a. Flask
  b. HTML
  c. CSS
- Database Management System
  a. MySQL
- Other Tools
  a. Google Colab
  b. VS Code

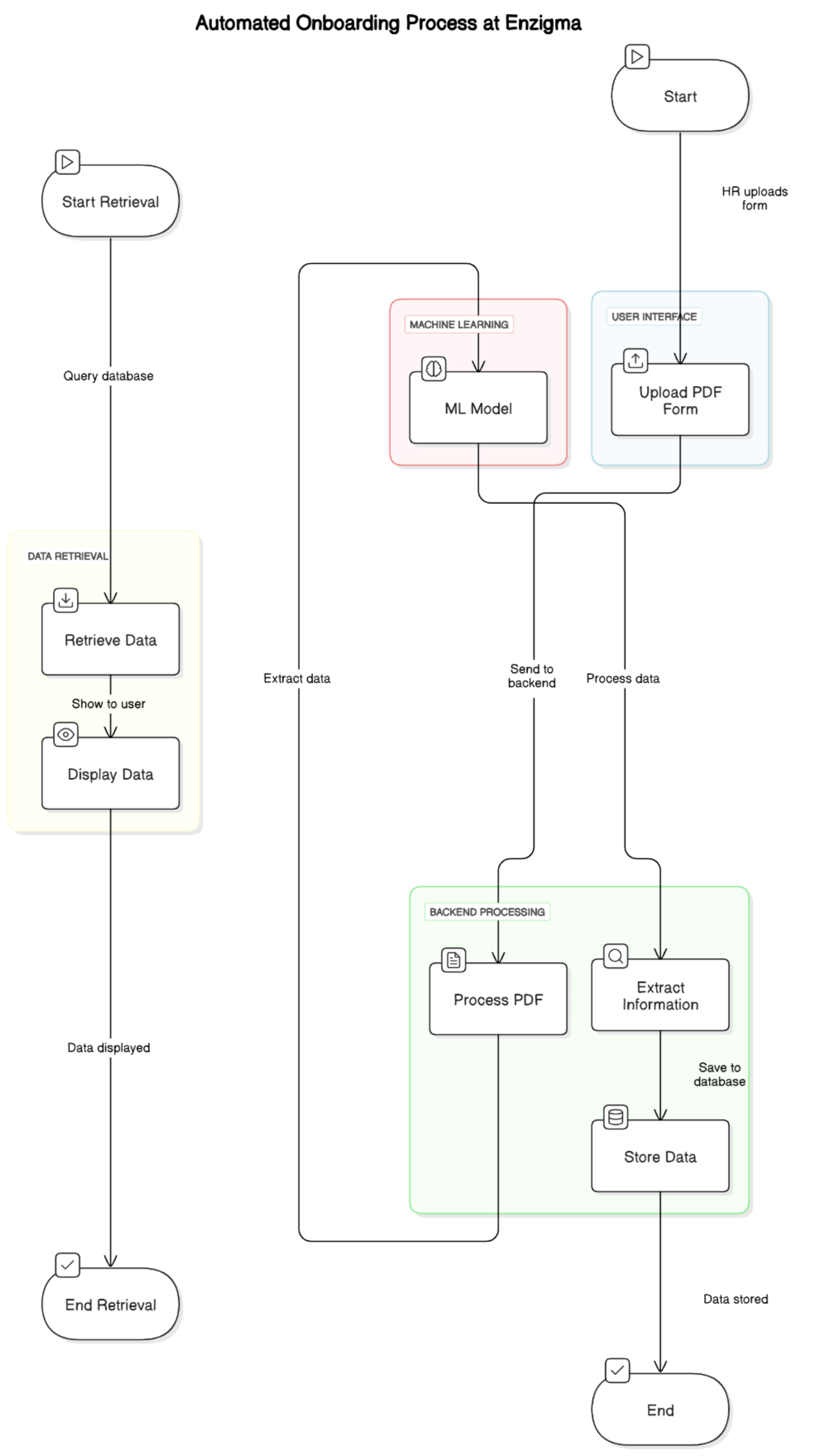
2.2. Justification Of Technology Choice

- **Python :-** Python's extensive ecosystem of libraries, including Scikit-learn, Pandas, and NumPy, which make data manipulation and machine learning easier, makes it the perfect choice for this project. The ML model is seamlessly integrated into the web application thanks to its compatibility with Flask.

- **Flask :-** Flask is ideal for small-scale web applications like this one because of its lightweight and modular design. It makes it simple to integrate Python-based machine learning models into the backend server and enables rapid prototyping.
- **HTML & CSS :-** These technologies were used to create a user-friendly and visually appealing interface. The simplicity of HTML and CSS ensured that the webpage could efficiently handle file uploads while maintaining a professional look.
- **MySQL :-** MySQL was chosen as the database because of its user-friendliness and resilience. Its effective handling of structured data fits in nicely with the requirement that candidate details extracted by the ML model be stored and retrieved.
- **Pytessract:-** For machine learning tasks, these libraries are considered industry standards. They guarantee the accuracy and effectiveness of the AI solution by offering prebuilt tools for data preprocessing and model training.
- **Google Colab:-** Used to test and visualize the ML model's output during the development stage, which facilitates debugging and fine-tuning.

3.System Architecture

3.1. Block Diagram



3.2. Component Interaction Overview

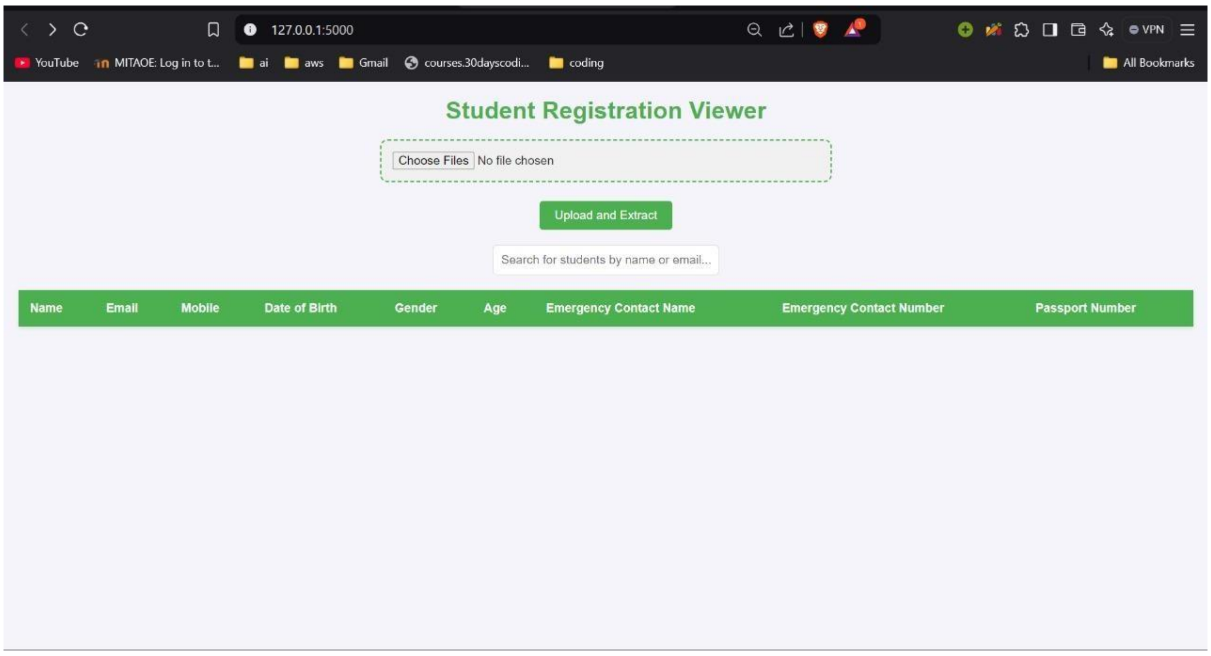
- **User Interface:-** Onboarding forms in PDF format can be uploaded by users through the user interface (UI). The webpage, which was created with HTML and CSS, offers users an easy-to-use interface for interacting with the system. A form submission mechanism ensures a smooth integration with the backend workflow by sending the uploaded form to the backend for additional
- **Flask (Backend) :-** The application's central component is the Flask backend, which effectively handles requests, communicates with the database, and works with the machine learning model. The user interface uploads files to it, which are then sent to the ML model for data extraction. The backend saves the pertinent data in the MySQL database after it has been extracted. It also makes it easier for the database and user interface to communicate, allowing stored data to be retrieved and shown when needed.
- **Machine Learning Models**:- The machine learning model plays a crucial role in extracting relevant data fields, such as names and contact details, from the uploaded PDF forms. It receives the file from the backend for processing and utilizes techniques like Optical Character Recognition (OCR) to extract structured data accurately. Once the extraction is complete, the model sends the processed data back to the backend for storage in the database.

- **DataBase(MySQL) :-** The extracted data is stored in the MySQL database, which is designed to handle and retrieve information effectively. Because it guarantees data consistency, dependable storage and retrieval procedures are made possible. HR staff can more easily search and efficiently manage candidate records thanks to the database's querying capabilities.
- **System Output:-** The system output, designed as an HR view, displays the stored information to HR personnel for review and verification. The backend retrieves the requested data from the database and presents it on the user interface, ensuring easy access to the required records. This feature allows HR staff to efficiently search, view, and verify candidate details, streamlining the overall onboarding process.

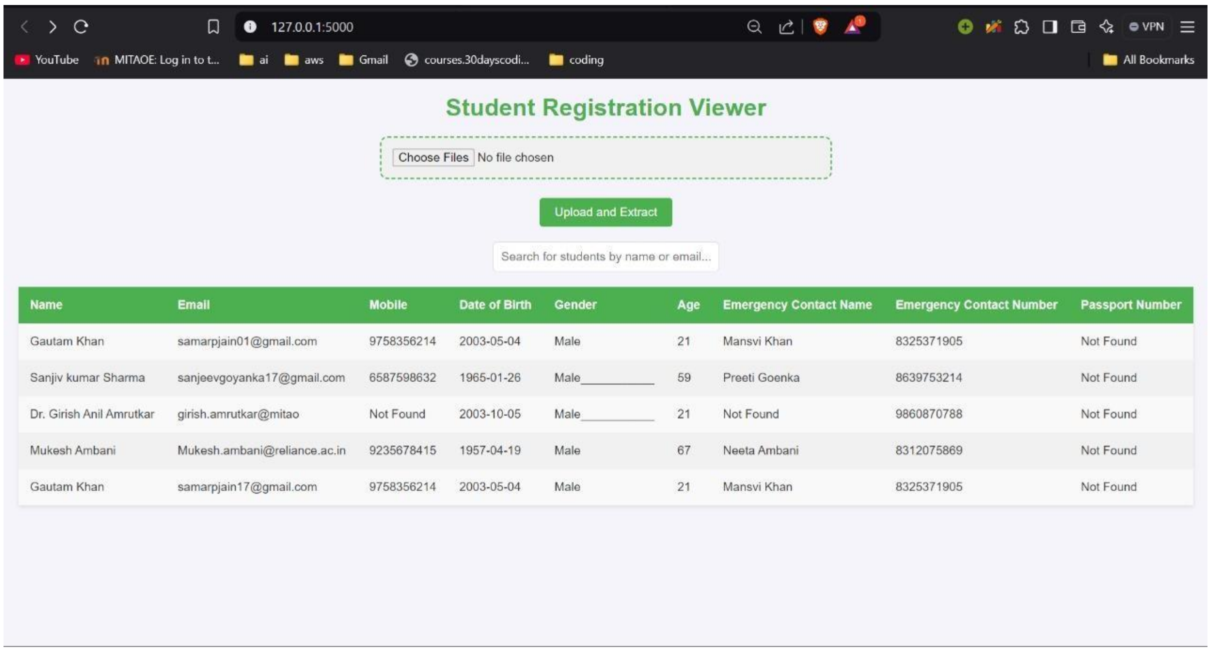
4. Implementation Details

4.1. User Interface Development

To guarantee a straightforward, responsive, and user-friendly design, HTML and CSS were used in the development of the onboarding system's user interface (UI). HR staff can upload and process multiple PDF forms with it. Additionally, the user interface has a search bar that allows you to look up records by candidate name or email. The results are shown in an easy-to-read tabular format. Accessibility across platforms is guaranteed by the interface, which is optimized for desktop and tablet devices. It offers a smooth and effective user experience by enabling real-time data processing and retrieval through integration with the Flask backend.



Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number

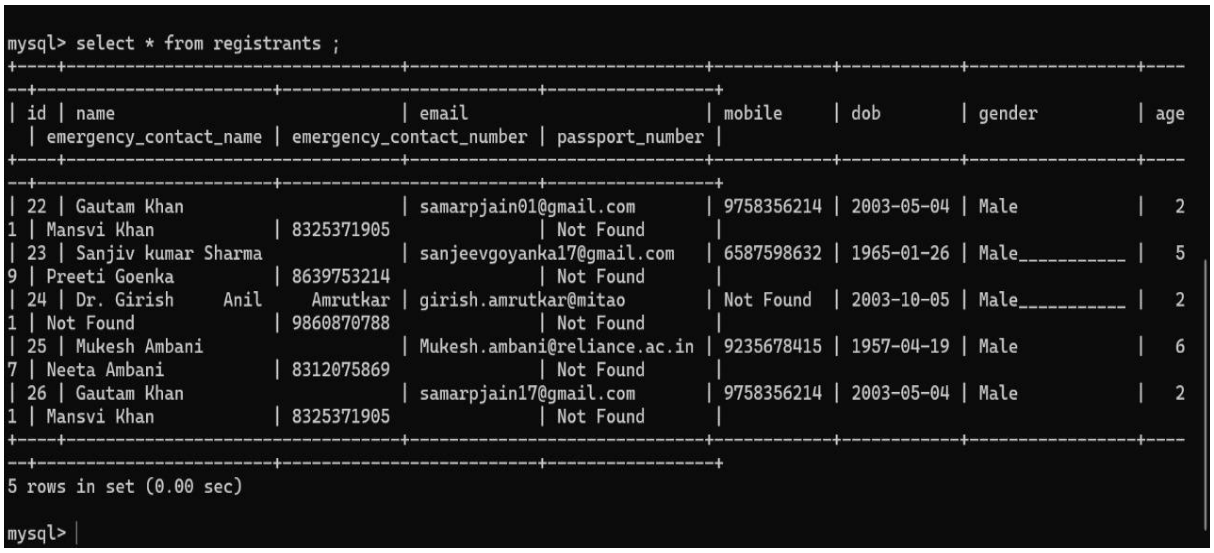


Student Registration Viewer

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number
Gautam Khan	samarpjain01@gmail.com	9758356214	2003-05-04	Male	21	Mansvi Khan	8325371905	Not Found
Sanjiv kumar Sharma	sanjeevgoyanka17@gmail.com	6587598632	1965-01-26	Male__________	59	Preeti Goenka	8639753214	Not Found
Dr. Girish Anil Amrutkar	girish.amrutkar@mitao	Not Found	2003-10-05	Male__________	21	Not Found	9860870788	Not Found
Mukesh Ambani	Mukesh.ambani@reliance.ac.in	9235678415	1957-04-19	Male	67	Neeta Ambani	8312075869	Not Found
Gautam Khan	samarpjain17@gmail.com	9758356214	2003-05-04	Male	21	Mansvi Khan	8325371905	Not Found

4.2. Database Design

Additionally, the user interface has a search bar that allows you to look up records by candidate name or email. The results are shown in an easy-to-read tabular format. Accessibility across platforms is guaranteed by the interface, which is optimized for desktop and tablet devices. It offers a smooth and effective user experience by enabling real-time data processing and retrieval through integration with the Flask backend. While security measures like encryption and access restrictions guarantee the safe storage of sensitive data, constraints like not null and unique are used to maintain data accuracy. This design ensures scalability and consistency as the system expands by facilitating effective candidate record retrieval and querying.



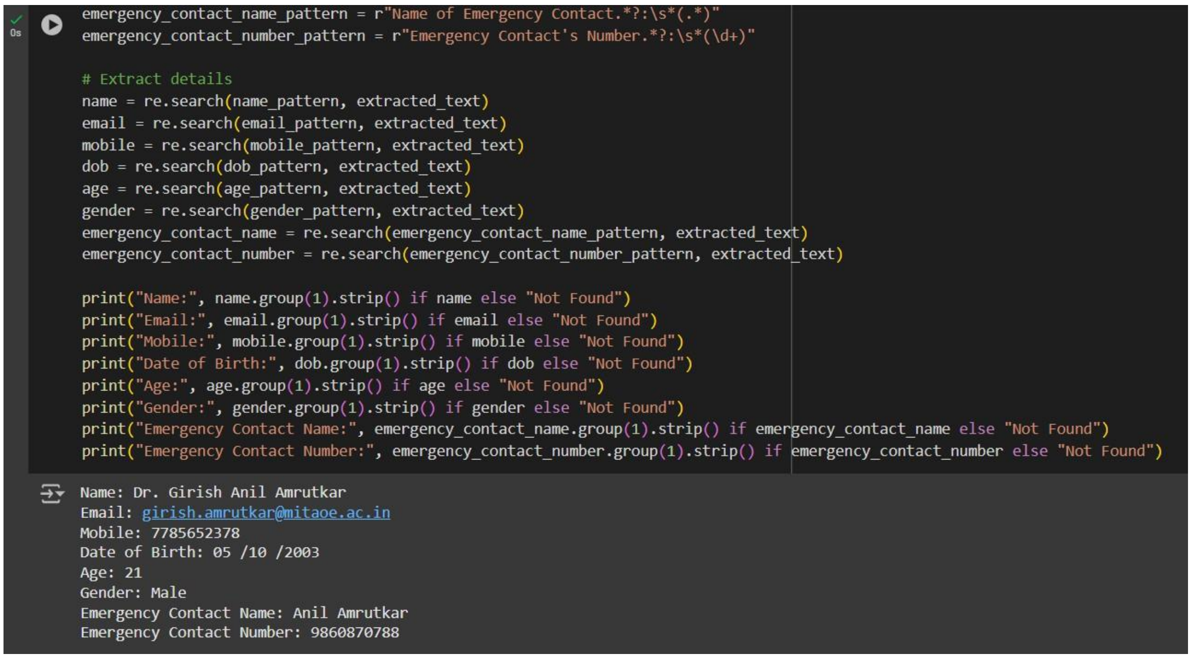
```
mysql> select * from registrants ;
+----+----------------------------------+------------------------------+------------+------------+-----------------+----
--+------------------------+--------------------------+-----------------+
| id | name                             | email                        | mobile     | dob        | gender          | age
  | emergency_contact_name | emergency_contact_number | passport_number |
+----+----------------------------------+------------------------------+------------+------------+-----------------+----
--+------------------------+--------------------------+-----------------+
| 22 | Gautam Khan                      | samarpjain01@gmail.com       | 9758356214 | 2003-05-04 | Male            |   2
1 | Mansvi Khan            | 8325371905               | Not Found       |
| 23 | Sanjiv kumar Sharma              | sanjeevgoyanka17@gmail.com   | 6587598632 | 1965-01-26 | Male___________ |   5
9 | Preeti Goenka          | 8639753214               | Not Found       |
| 24 | Dr. Girish     Anil     Amrutkar | girish.amrutkar@mitao        | Not Found  | 2003-10-05 | Male___________ |   2
1 | Not Found              | 9860870788               | Not Found       |
| 25 | Mukesh Ambani                    | Mukesh.ambani@reliance.ac.in | 9235678415 | 1957-04-19 | Male            |   6
7 | Neeta Ambani           | 8312075869               | Not Found       |
| 26 | Gautam Khan                      | samarpjain17@gmail.com       | 9758356214 | 2003-05-04 | Male            |   2
1 | Mansvi Khan            | 8325371905               | Not Found       |
+----+----------------------------------+------------------------------+------------+------------+-----------------+----
--+------------------------+--------------------------+-----------------+
5 rows in set (0.00 sec)

mysql>
```

4.3. AI Model Design & Functionality

The AI model used in this project is intended to automate the data entry process by extracting pertinent information from the uploaded PDF forms. The model, which was constructed with Python and machine learning libraries such as TensorFlow or Scikit-learn, processes the scanned or text-based forms using Optical Character Recognition (OCR) methods. By examining the format and content of the forms, the model is trained to recognize and extract important fields like candidate names, contact information, and other private data. The model is refined to identify different fonts during the training phase, guaranteeing high data extraction accuracy. The structured data is returned by the model after it has processed the uploaded form, and it is subsequently sent to the backend for database storage. As part of its functionality, the AI model can handle various form layouts and input variations, which makes it reliable and flexible for use in real-world scenarios. With more training data, the model gets better over time, increasing its efficiency and accuracy in extracting the relevant information.

The pertinent data is saved in a JSON file at the conclusion of the data extraction procedure for the machine learning model. A lightweight, human-readable data format called JSON (JavaScript Object Notation) is used to store and send data between a client and a server. The JSON file contains key-value pairs that structure the extracted data, which includes candidate details like name, contact information, and other fields. The data can be readily parsed and integrated with the backend system using this format, after which it is processed further and saved in the MySQL database. Easy data manipulation, compatibility with multiple programming languages, and the ability to preserve the data in a flexible structure that is simple to expand or alter are just a few benefits of using JSON. The data is prepared for additional processing after it has been saved in the JSON file, guaranteeing smooth system integration with other parts.

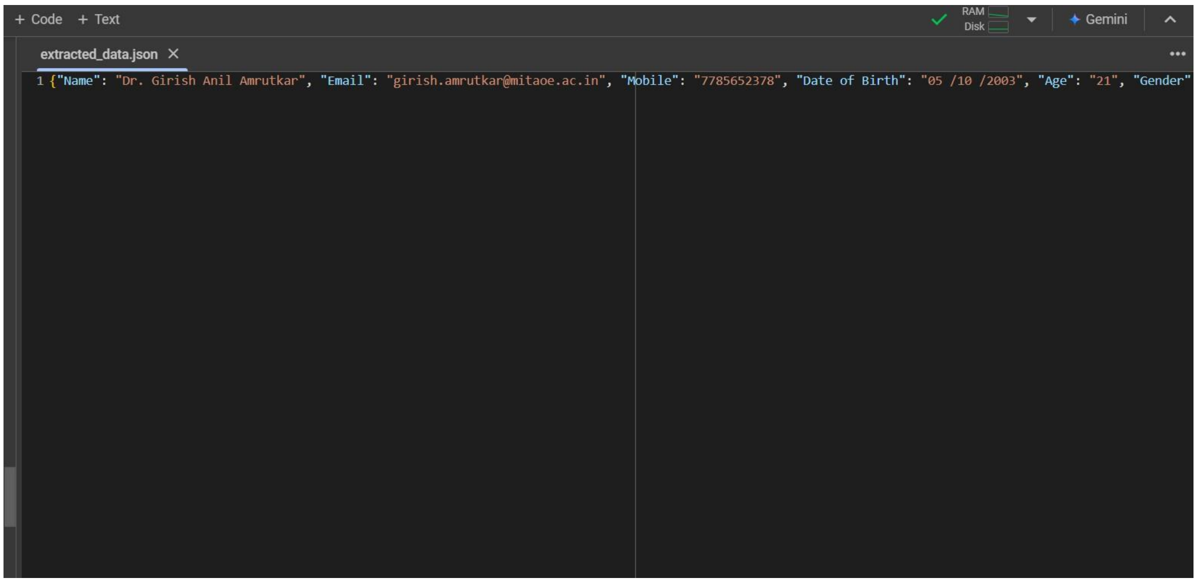


```python
emergency_contact_name_pattern = r"Name of Emergency Contact.*?:\s*(.*)"
emergency_contact_number_pattern = r"Emergency Contact's Number.*?:\s*(\d+)"

# Extract details
name = re.search(name_pattern, extracted_text)
email = re.search(email_pattern, extracted_text)
mobile = re.search(mobile_pattern, extracted_text)
dob = re.search(dob_pattern, extracted_text)
age = re.search(age_pattern, extracted_text)
gender = re.search(gender_pattern, extracted_text)
emergency_contact_name = re.search(emergency_contact_name_pattern, extracted_text)
emergency_contact_number = re.search(emergency_contact_number_pattern, extracted_text)

print("Name:", name.group(1).strip() if name else "Not Found")
print("Email:", email.group(1).strip() if email else "Not Found")
print("Mobile:", mobile.group(1).strip() if mobile else "Not Found")
print("Date of Birth:", dob.group(1).strip() if dob else "Not Found")
print("Age:", age.group(1).strip() if age else "Not Found")
print("Gender:", gender.group(1).strip() if gender else "Not Found")
print("Emergency Contact Name:", emergency_contact_name.group(1).strip() if emergency_contact_name else "Not Found")
print("Emergency Contact Number:", emergency_contact_number.group(1).strip() if emergency_contact_number else "Not Found")
```

```
Name: Dr. Girish Anil Amrutkar
Email: girish.amrutkar@mitaoe.ac.in
Mobile: 7785652378
Date of Birth: 05 /10 /2003
Age: 21
Gender: Male
Emergency Contact Name: Anil Amrutkar
Emergency Contact Number: 9860870788
```

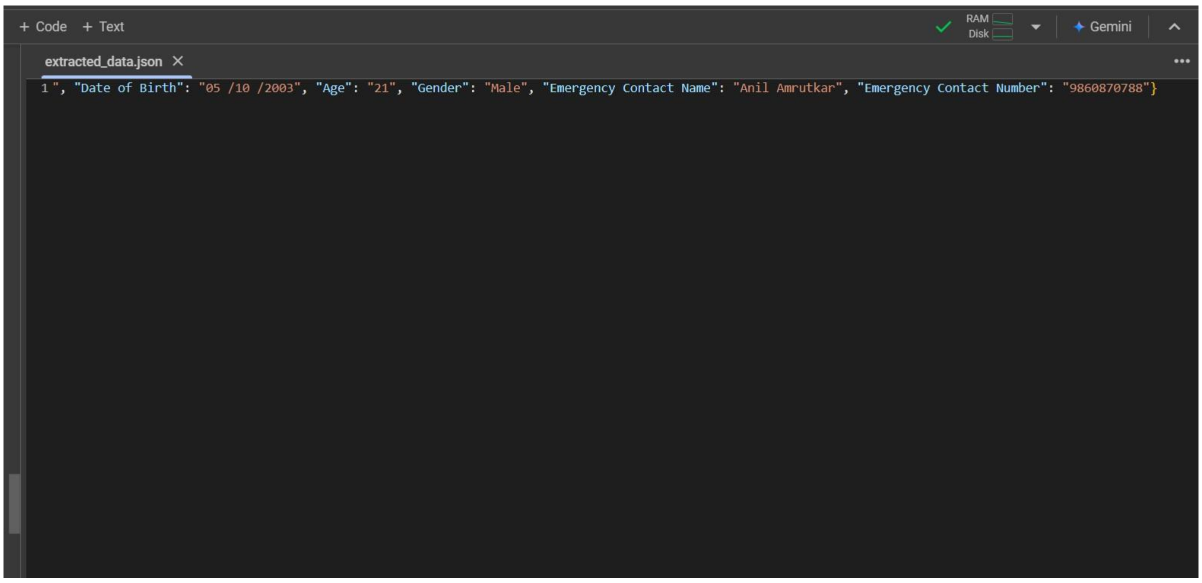


+ Code + Text

Gemini

extracted_data.json

```
{"Name": "Dr. Girish Anil Amrutkar", "Email": "girish.amrutkar@mitaoe.ac.in", "Mobile": "7785652378", "Date of Birth": "05 /10 /2003", "Age": "21", "Gender"
```



+ Code + Text

Gemini

extracted_data.json

```
", "Date of Birth": "05 /10 /2003", "Age": "21", "Gender": "Male", "Emergency Contact Name": "Anil Amrutkar", "Emergency Contact Number": "9860870788"}
```

5. Testing And Result

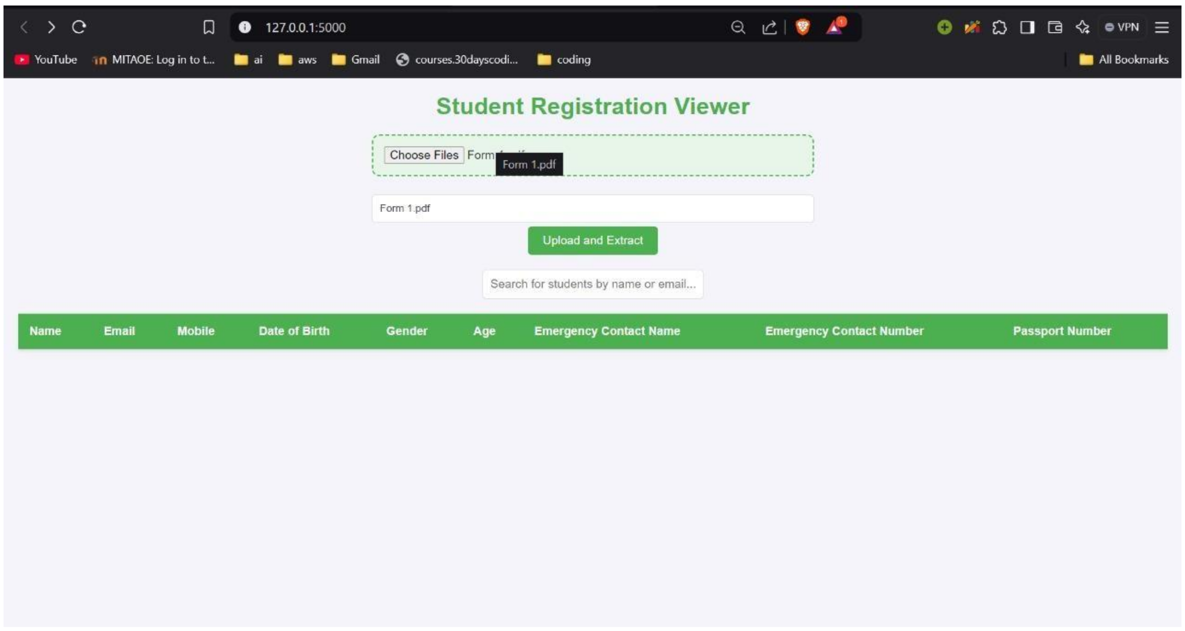
5.1. Testing Scenarios

1. Test Case 1:- Upload the valid PDF
2. Test Case 2:- Upload an invalid or unsupported file type
3. Test Case 3:- Upload multiple PDF files simultaneously
4. Test Case 4:- Upload a PDF with clear, machine-readable text.
5. Test case 5:- Ensure data is correctly inserted into the MySQL database after extraction.
6. Test case 6:- Search for a candidate using their name or email.
7. Test Case 7:- Search for a non-existent candidate.
8. Test Case 8:- Upload the same form which has been added already in the system
9. Test Case 9:- Retrieve multiple records if there are matching search criteria.

5.2. System Output

The output of the above testcases are as follows :-

Test case 1:- Upload the valid PDF

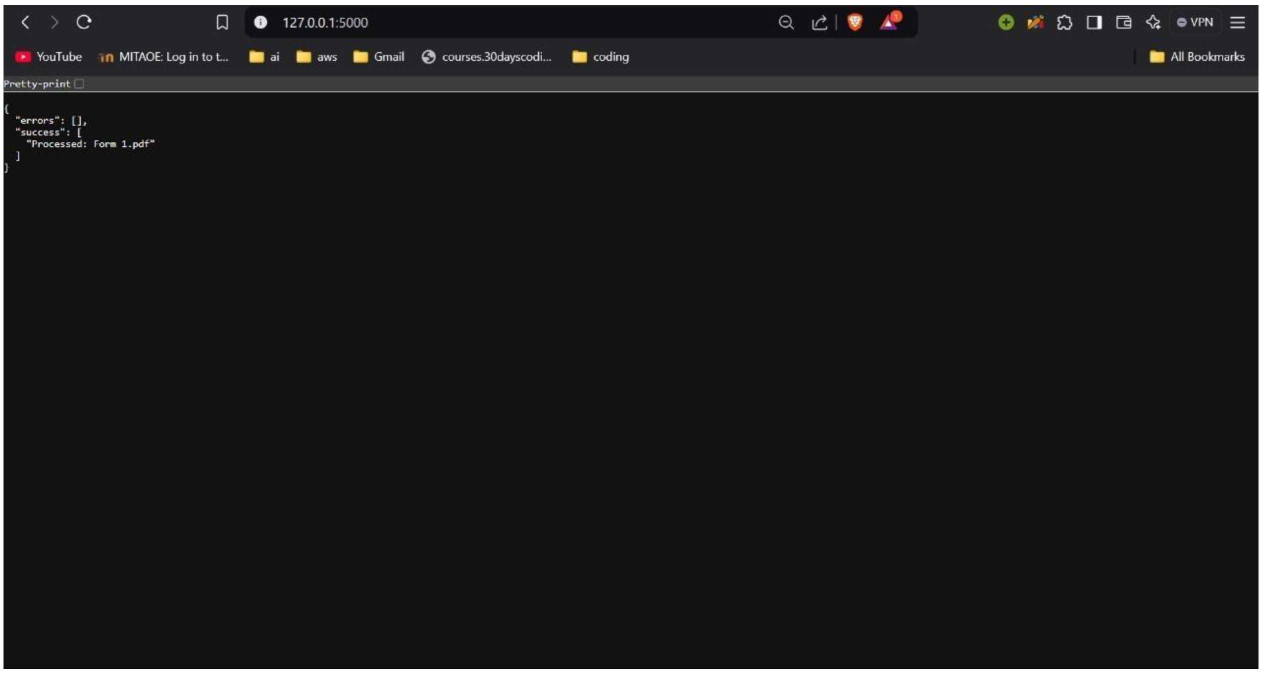


Student Registration Viewer

Form 1.pdf

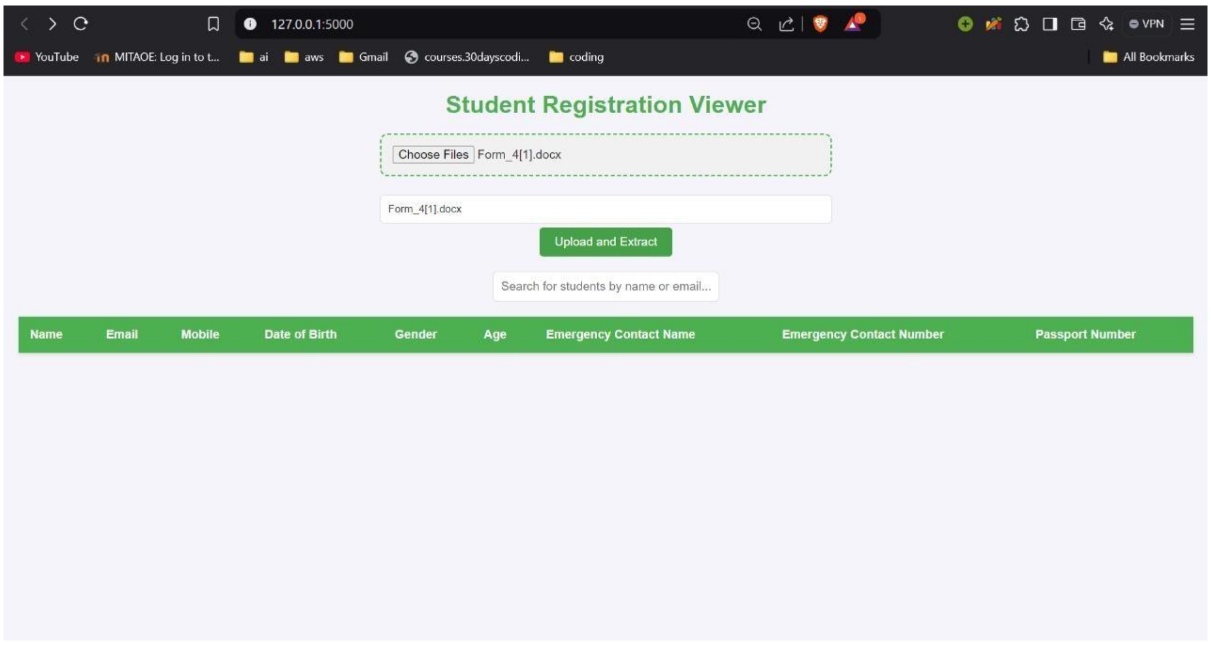
Search for students by name or email...

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number



```
{
  "errors": [],
  "success": [
    "Processed: Form 1.pdf"
  ]
}
```

Test Case 2:- Upload an invalid or unsupported file type



Student Registration Viewer

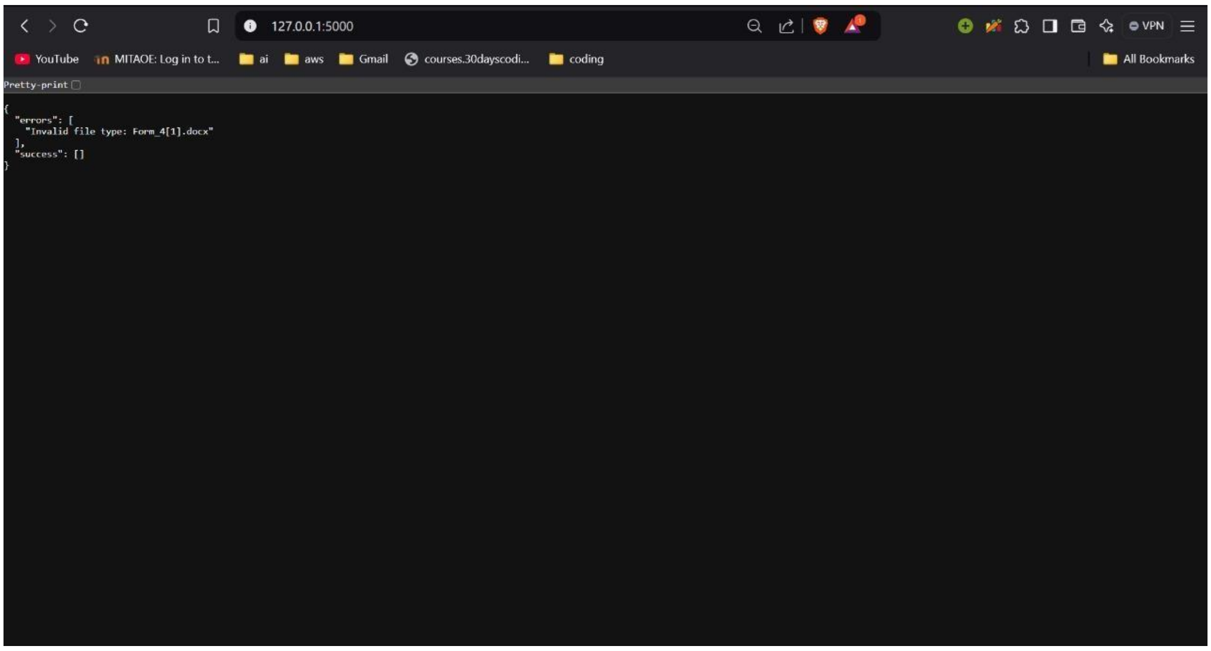
Choose Files Form_4[1].docx

Form_4[1].docx

Upload and Extract

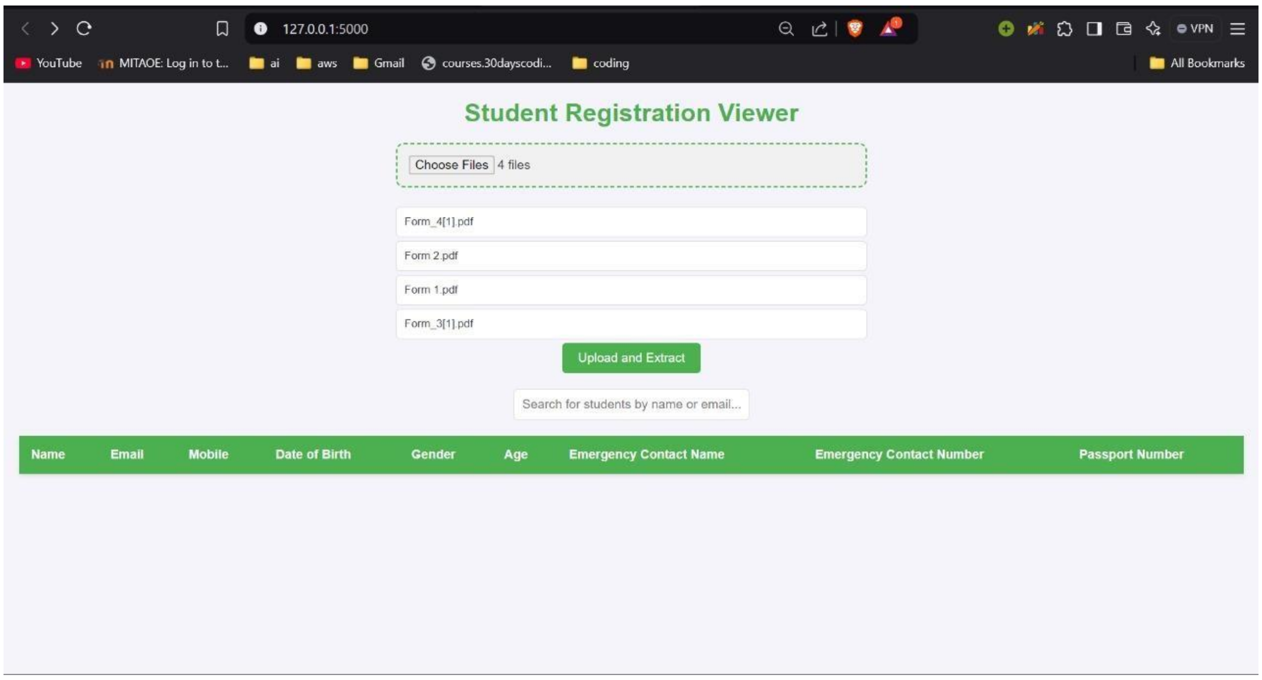
Search for students by name or email...

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number



```
{
  "errors": [
    "Invalid file type: Form_4[1].docx"
  ],
  "success": []
}
```

Test Case 3:- Upload multiple PDF files simultaneously



Student Registration Viewer

Choose Files 4 files

Form_4[1].pdf

Form 2.pdf

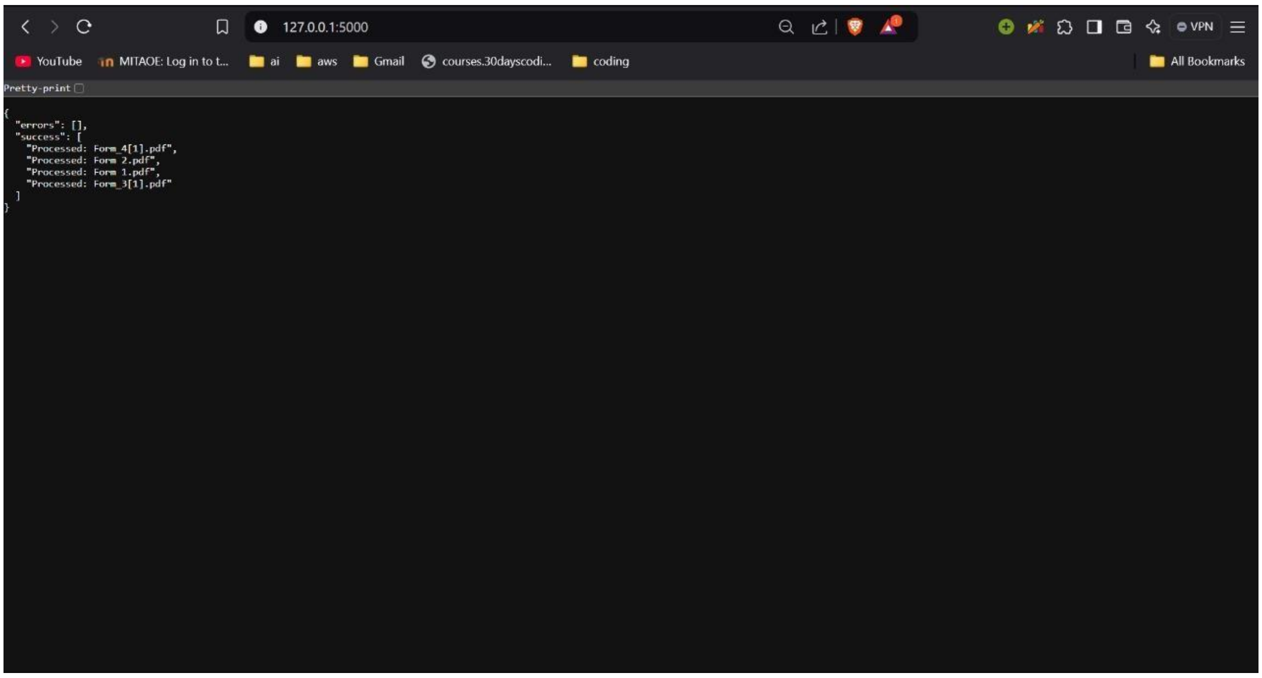
Form 1.pdf

Form_3[1].pdf

Upload and Extract

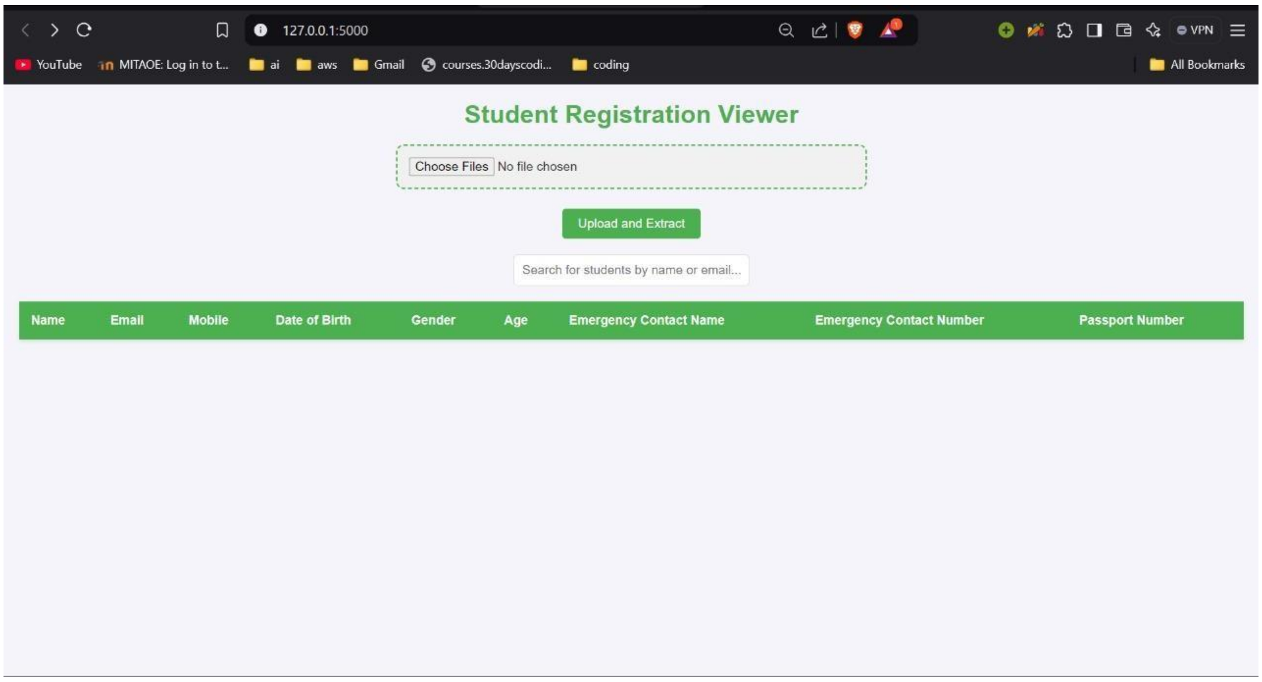
Search for students by name or email...

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number

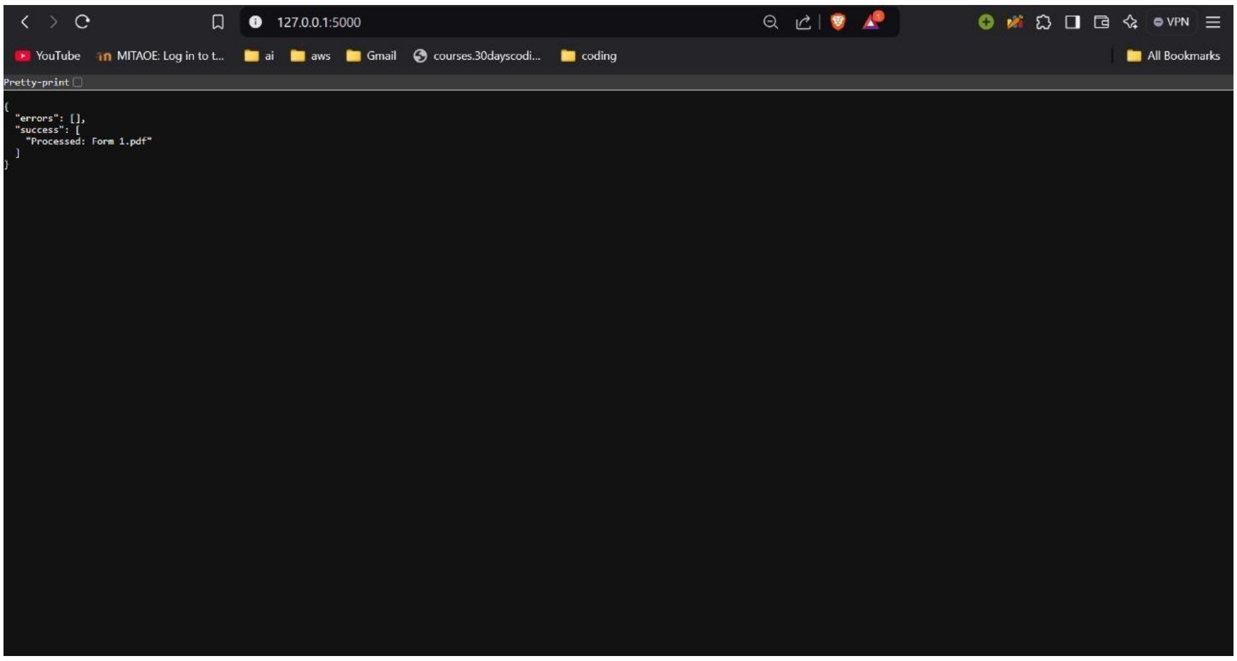


```
{
  "errors": [],
  "success": [
    "Processed: Form_4[1].pdf",
    "Processed: Form 2.pdf",
    "Processed: Form 1.pdf",
    "Processed: Form_3[1].pdf"
  ]
}
```

Test Case 4:- Upload a PDF with clear, machine-readable text.

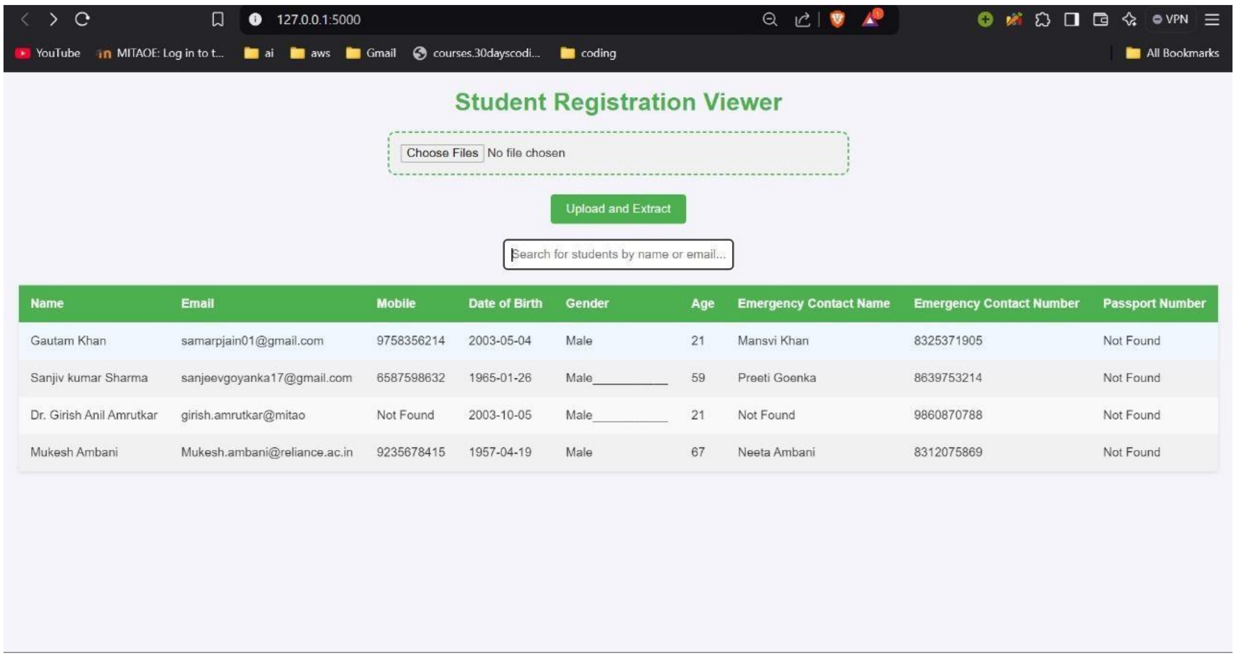


Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number

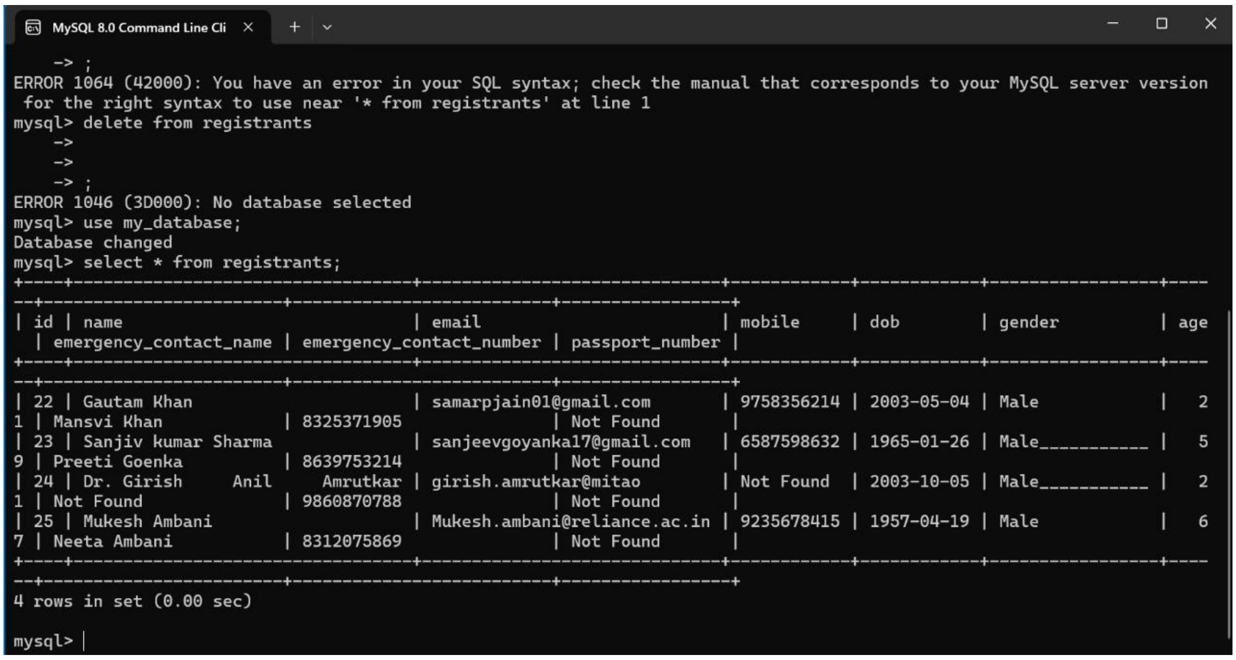


```
{
  "errors": [],
  "success": [
    "Processed: Form 1.pdf"
  ]
}
```

Test Case 5:- Ensure data is correctly inserted into the MySQL database after extraction.



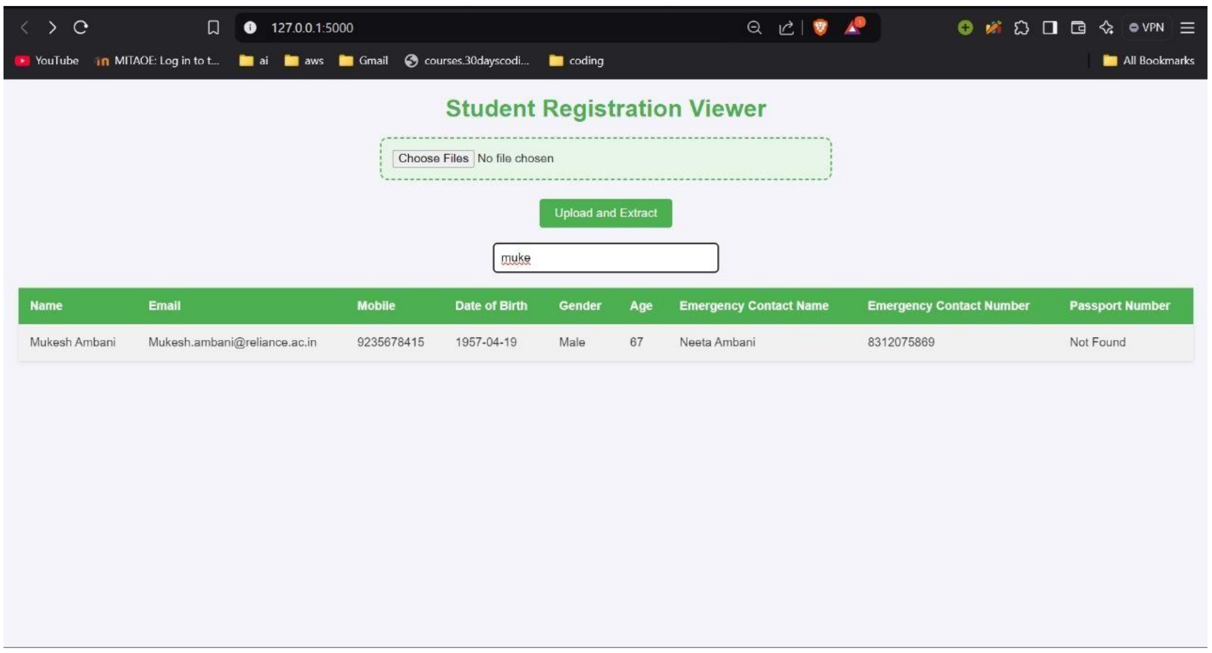
Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number
Gautam Khan	samarpjain01@gmail.com	9758356214	2003-05-04	Male	21	Mansvi Khan	8325371905	Not Found
Sanjiv kumar Sharma	sanjeevgoyanka17@gmail.com	6587598632	1965-01-26	Male__________	59	Preeti Goenka	8639753214	Not Found
Dr. Girish Anil Amrutkar	girish.amrutkar@mitao	Not Found	2003-10-05	Male__________	21	Not Found	9860870788	Not Found
Mukesh Ambani	Mukesh.ambani@reliance.ac.in	9235678415	1957-04-19	Male	67	Neeta Ambani	8312075869	Not Found



```
    -> ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
 for the right syntax to use near '* from registrants' at line 1
mysql> delete from registrants
    ->
    ->
    -> ;
ERROR 1046 (3D000): No database selected
mysql> use my_database;
Database changed
mysql> select * from registrants;
+----+-----------------------------------+-------------------------------+------------+------------+-------------------+----
--+------------------------+---------------------------+-----------------+
| id | name                              | email                         | mobile     | dob        | gender            | age
  | emergency_contact_name | emergency_contact_number | passport_number |
+----+-----------------------------------+-------------------------------+------------+------------+-------------------+----
--+------------------------+---------------------------+-----------------+
| 22 | Gautam Khan                       | samarpjain01@gmail.com        | 9758356214 | 2003-05-04 | Male              |   2
1 | Mansvi Khan            | 8325371905               | Not Found       |
| 23 | Sanjiv kumar Sharma               | sanjeevgoyanka17@gmail.com    | 6587598632 | 1965-01-26 | Male___________   |   5
9 | Preeti Goenka          | 8639753214               | Not Found       |
| 24 | Dr. Girish     Anil     Amrutkar | girish.amrutkar@mitao         | Not Found  | 2003-10-05 | Male___________   |   2
1 | Not Found              | 9860870788               | Not Found       |
| 25 | Mukesh Ambani                     | Mukesh.ambani@reliance.ac.in | 9235678415 | 1957-04-19 | Male              |   6
7 | Neeta Ambani           | 8312075869               | Not Found       |
+----+-----------------------------------+-------------------------------+------------+------------+-------------------+----
--+------------------------+---------------------------+-----------------+
4 rows in set (0.00 sec)

mysql> |
```

Test Case 6:- Search for a candidate using their name or email.



Student Registration Viewer

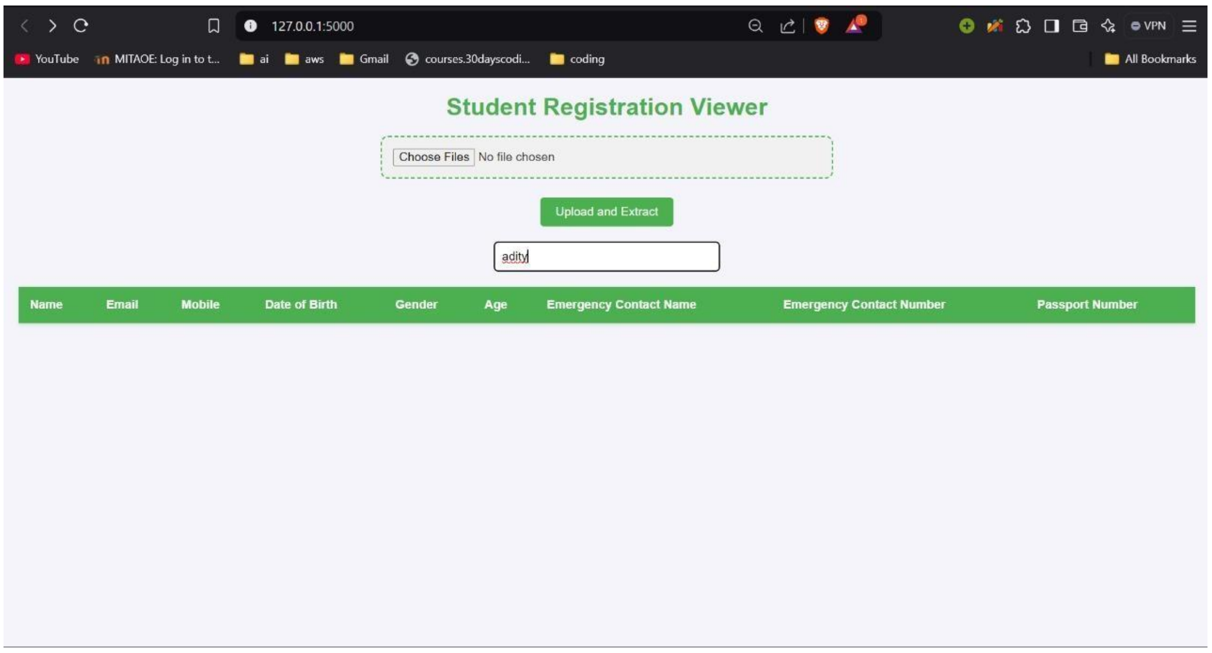
Choose Files No file chosen

Upload and Extract

muke

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number
Mukesh Ambani	Mukesh.ambani@reliance.ac.in	9235678415	1957-04-19	Male	67	Neeta Ambani	8312075869	Not Found

Test Case 7:- Search for a non-existent candidate.



Student Registration Viewer

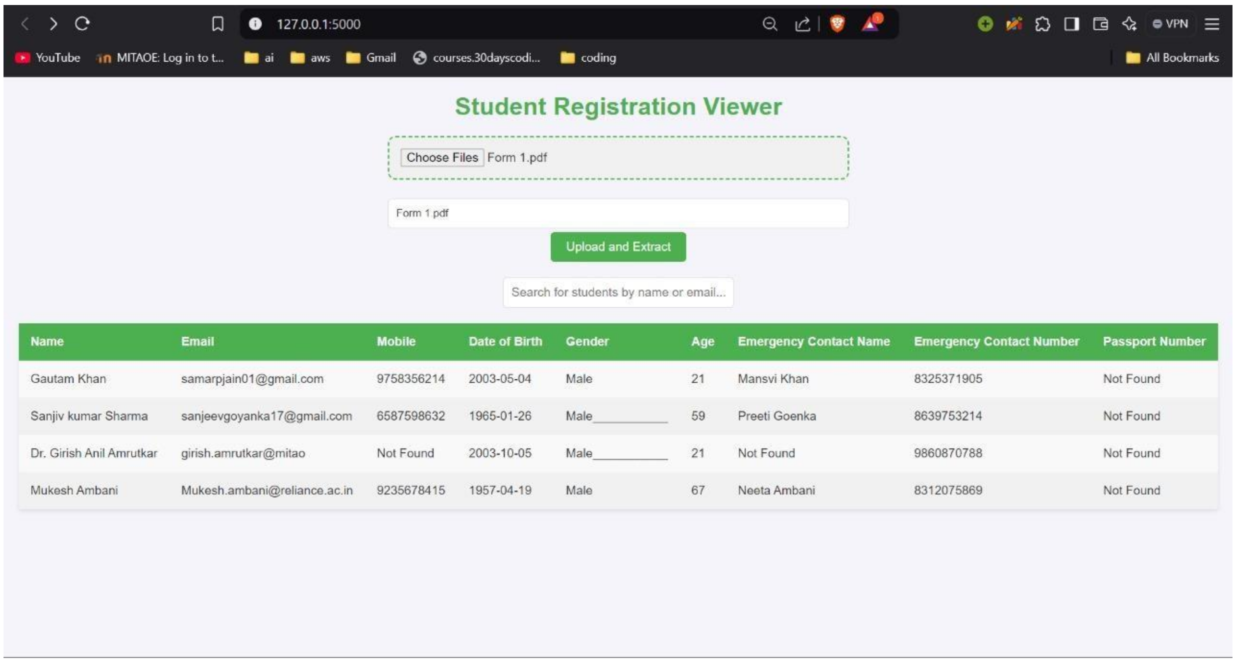
Choose Files No file chosen

Upload and Extract

adity

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number

Test Case 8:- Upload the same form which has been added already in the system



127.0.0.1:5000

Student Registration Viewer

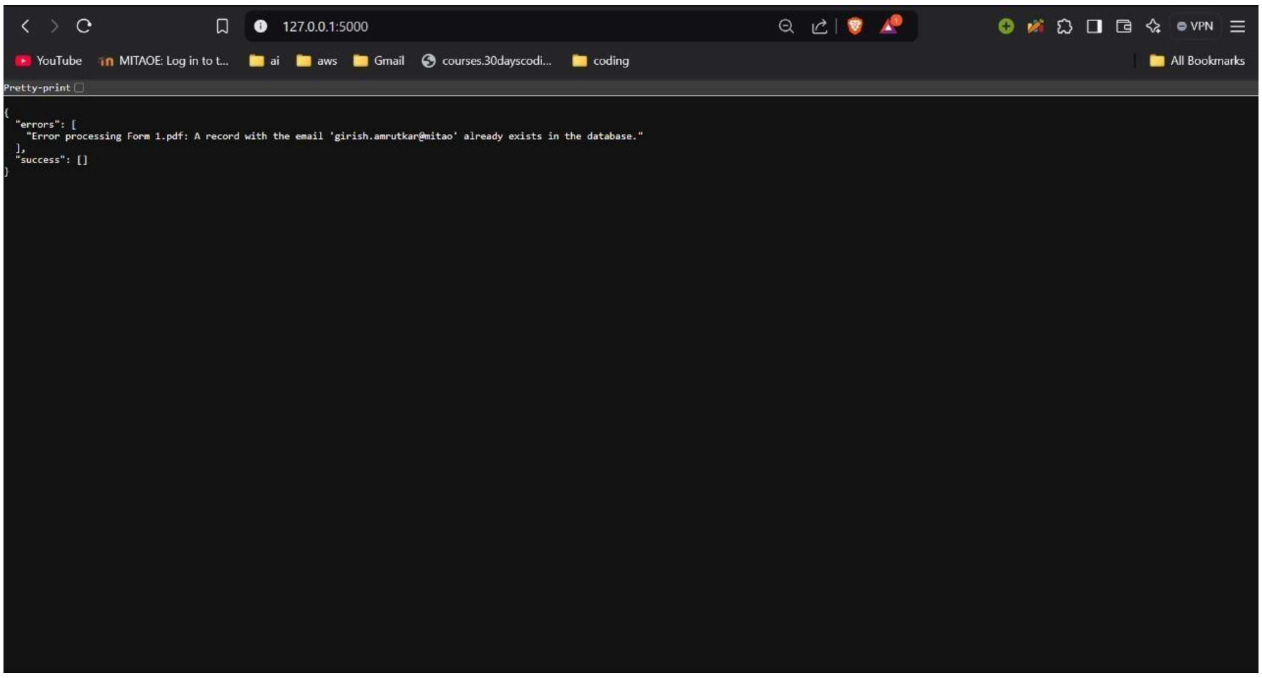
Choose Files Form 1.pdf

Form 1.pdf

Upload and Extract

Search for students by name or email...

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number
Gautam Khan	samarpjain01@gmail.com	9758356214	2003-05-04	Male	21	Mansvi Khan	8325371905	Not Found
Sanjiv kumar Sharma	sanjeevgoyanka17@gmail.com	6587598632	1965-01-26	Male__________	59	Preeti Goenka	8639753214	Not Found
Dr. Girish Anil Amrutkar	girish.amrutkar@mitao	Not Found	2003-10-05	Male__________	21	Not Found	9860870788	Not Found
Mukesh Ambani	Mukesh.ambani@reliance.ac.in	9235678415	1957-04-19	Male	67	Neeta Ambani	8312075869	Not Found

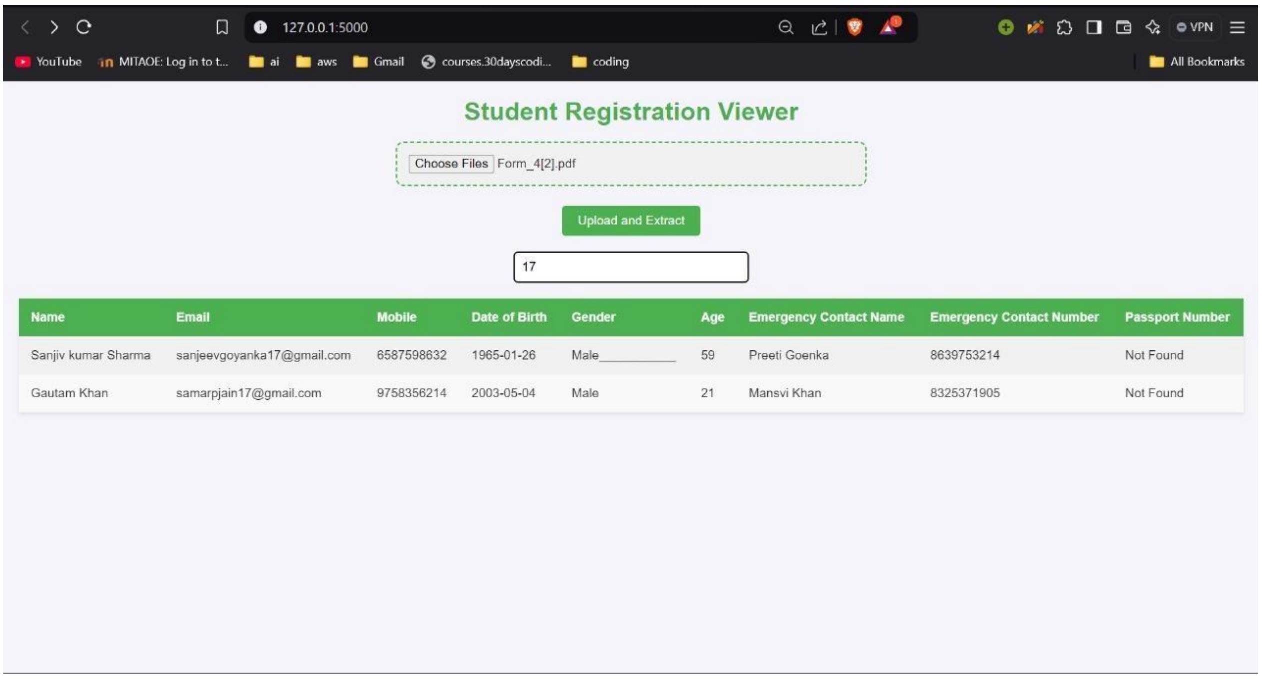


127.0.0.1:5000

Pretty-print

```
{
  "errors": [
    "Error processing Form 1.pdf: A record with the email 'girish.amrutkar@mitao' already exists in the database."
  ],
  "success": []
}
```

Test Case 9:- Retrieve multiple records if there are matching search criteria.



Student Registration Viewer

Name	Email	Mobile	Date of Birth	Gender	Age	Emergency Contact Name	Emergency Contact Number	Passport Number
Sanjiv kumar Sharma	sanjeevgoyanka17@gmail.com	6587598632	1965-01-26	Male__________	59	Preeti Goenka	8639753214	Not Found
Gautam Khan	samarpjain17@gmail.com	9758356214	2003-05-04	Male	21	Mansvi Khan	8325371905	Not Found

6. User Guide

6.1. Steps to use the System

To begin using the onboarding system, open a browser and navigate to the web interface using the application link or URL that was provided. Find the file upload section of the interface. From there, you can drag and drop PDF files into the appropriate spot or click the "Choose File" button. The system supports multiple file uploads, allowing you to process several forms at once. Click the "Upload" button to start the processing after choosing the files.

The uploaded files are processed by the Flask-powered backend system, and pertinent data, including names and contact information, is extracted by the integrated machine learning model. The extracted data is saved in a structured MySQL database after initially being saved in a JSON file. Use the interface's search bar to find and view records by typing in keywords like the candidate's name or email address. The system will retrieve pertinent records from the database and present them for review in an understandable tabular format. After that, HR staff can check the data and handle it as required, including updating or exporting data if supported. You can terminate the session by closing the application or logging out once all tasks have been finished.

You can terminate the session by closing the application or logging out once all tasks have been finished. For automating the onboarding process, this simple workflow guarantees a smooth, effective, and user-friendly experience.

7.Conclusion

The automated onboarding system successfully addresses the inefficiencies of the traditional manual process by incorporating advanced technologies such as an AI-powered data extraction model, a user-friendly web interface, and a structured MySQL database. HR staff can easily upload and process onboarding forms thanks to this integration, and the data that is extracted is safely stored and easily accessible when needed. The system dramatically lowers manual labor, minimizes errors, and boosts workflow efficiency by automating repetitive tasks. Its flexible and scalable architecture also guarantees that it can accommodate future needs, making it a strong option for efficiently handling onboarding duties.