Project Name

M-Tech Admissions

Project Supervisor
Dr. Satyanath Bhat

Team Members
Kastala john Agusteen-21031
D Arun Kumar-2103112

1. Introduction..... 4
Objective:..... 4
To design and implement a web-based portal for IIT Goa to streamline the COAP (Common Offer Acceptance Portal) process, enabling applicant login, admin functionalities, and applicant management..... 4
Scope of the Project:..... 4
● Develop a user-friendly interface for applicants and admins..... 4
● Secure user authentication using backend APIs and session management..... 4
● Store and retrieve applicant data using PostgreSQL..... 4
Technology Stack:..... 4
● Frontend: HTML, CSS, JavaScript..... 4
● Backend: Flask (Python)..... 4
● Database: PostgreSQL..... 4
2. Frontend Development..... 5
2.1 Login Page Design:..... 5
2.2 Frontend Field and Component Table..... 6
2.3 JavaScript Logic Notes..... 7
3.1 Registration Page:..... 7
Objective..... 7
Components of the Portal..... 8
1. HTML Structure..... 8
2. CSS Styling..... 8
3. JavaScript Functionality..... 8
Code Features..... 9
Validation Rules..... 9
3.2 Frontend Field and Component Table..... 9
4.1 Admin Login Page:..... 11
Structure and Design..... 11
JavaScript (Dynamic Functionality)..... 11
4.2 Frontend Field and Component Table..... 12
5.1 Password Change Page..... 13
1. Header Section..... 13
2. Input Fields..... 13
3. Password Change Button..... 13
4. Password Validation..... 14
5. Backend Request..... 14
6. Response Handling..... 14
7. Error Handling..... 14
6 Applicant and Dashboard Page..... 16
6.1 Project Overview:..... 16
Features Implemented..... 16
1. Dashboard Interface..... 16
2. Personal Details Form..... 16

3. GATE Examination Details..... 18
4. Academic Details.....19
5. Certificate Upload Section.....19
6. Actions and Functionality..... 20
7. Backend Communication..... 20
Technologies Used.....20
1. Frontend:.....20
2. Backend:..... 21
3. Database:.....21
Progress Since Last Milestone.....21
Challenges Faced..... 21
6.2 Frontend Field and Component Table.....22
5. Admin Dashboard for Applicant Profile Management and Eligibility Criteria Filtering. 26
Introduction.....26
Objectives of the Project..... 26
Current Progress..... 26
Challenges and Solutions.....27
3. Report on Flask-Based Backend for COAP Registration and Management System... 28
1. Introduction.....28
2. System Features..... 28
2.1. User Registration.....28
2.2. User Login..... 28
2.3. Change Password..... 29
2.4. Store Applicant Details..... 29
2.5. Shortlist Calculation.....29
2.6. Retrieve Applicant Details..... 29
2.7. Admin Dashboard.....30
2.8. Admin Login.....30
3. Database Interaction..... 30
3.1. PostgreSQL Database.....30
3.2. Database Tables.....30
4. Security Measures.....31
5. Conclusion.....31

Frontend and Backend Development of COAP Registration Portal

1. Introduction

Objective:

To design and implement a web-based portal for IIT Goa to streamline the COAP (Common Offer Acceptance Portal) process, enabling applicant login, admin functionalities, and applicant management.

Scope of the Project:

- Develop a user-friendly interface for applicants and admins.
- Secure user authentication using backend APIs and session management.
- Store and retrieve applicant data using PostgreSQL.

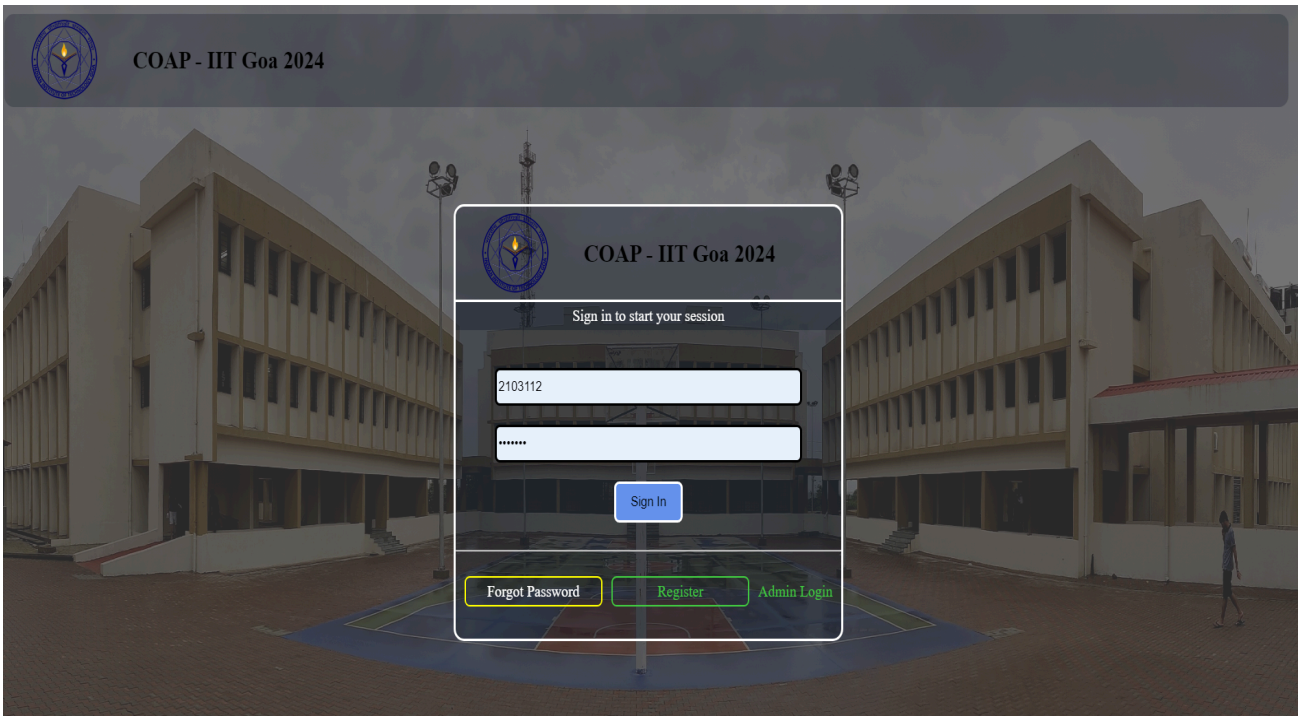
Technology Stack:

- **Frontend:** HTML, CSS, JavaScript
- **Backend:** Flask (Python)
- **Database:** PostgreSQL

2. Frontend Development

2.1 Login Page Design:

Objective: Create a responsive login page with inputs for COAP Registration ID and password.



2.2 Frontend Field and Component Table

Field Name	Field Type	Content	Required Field?	Min. Char.	Max Char	Notes
COAP Registration ID	Text Box	Alpha-Numeric	yes	8	16	User input is validated on form submission. Displays error if invalid.
Password	Text Box	Alpha-Numeric	yes	8	16	Password is validated; on error, prompts user to re-enter credentials.
Sign In	Button	N/A	N/A	N/A	N/A	Sends a POST request to the server to validate user credentials and redirects if successful.
Forgot Password	Hyper link	N/A	N/A	N/A	N/A	Redirects the user to changepassword.html for password recovery.
Register	Hyper link	N/A	N/A	N/A	N/A	Redirects the user to register.html for creating a new account.
Admin Login	Hyper link	N/A	N/A	N/A	N/A	N/A

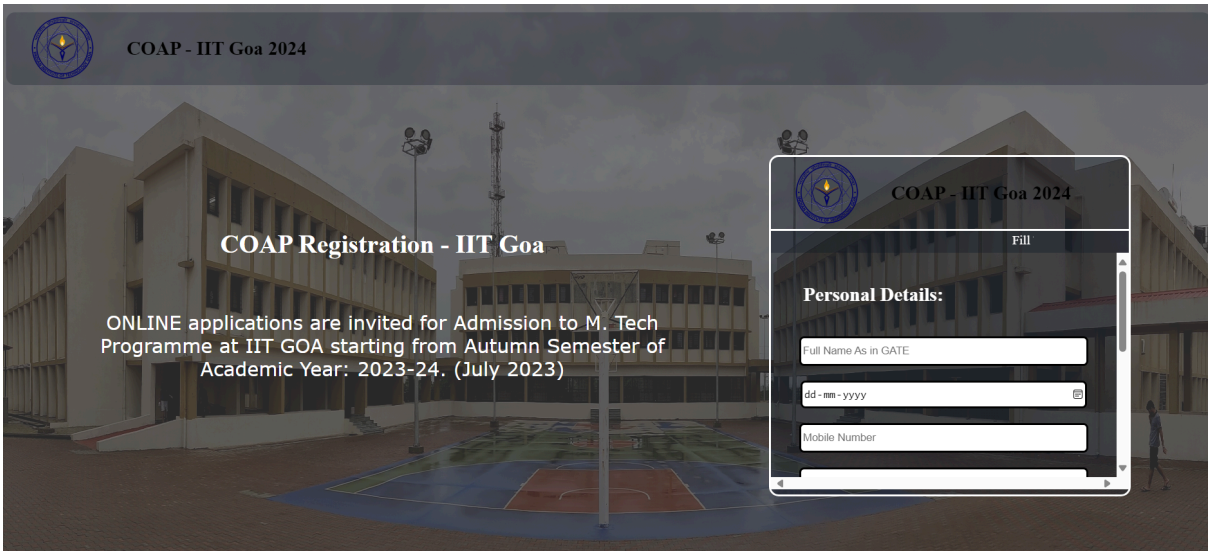
2.3 JavaScript Logic Notes

1. **Validation Logic**:
    - **COAP Registration ID** and **Password** fields must enforce minimum and maximum character limits (8-16).
    - If the user provides invalid credentials, display an error alert.
2. **Session Management**:
    - **Session Storage** is checked on page load to determine if a user is already logged in.
    - **Cookies** are used to persist login state for 1 hour. The page auto-redirects if the user is already authenticated.
3. **Sign In Button**:
    - Captures form data and sends a POST request to the backend at http://127.0.0.1:5000/login.
    - Validates the response to redirect users or display an error.
4. **Hyperlinks**:
    - All hyperlinks (Forgot Password, Register, Admin Login) navigate to their respective pages for additional functionality.

3.1 Registration Page:

Objective

The goal of this module is to implement the frontend for the COAP Registration Portal, enabling applicants to register by providing their personal and GATE details. The interface focuses on validation, user experience, and seamless integration with a backend API.



Components of the Portal

1. HTML Structure

The HTML defines the layout and structure of the registration page. Key elements include:

- **Form Inputs**: For capturing user details such as name, date of birth, mobile number, email, and GATE details.
- **Button**: A registration button triggers validation and communicates with the backend.
- **Headings and Paragraphs**: Provide guidance and information to the users.

2. CSS Styling

Although external styling (index.css) is referenced, inline styles are used to define text colour, alignment, and visual hierarchy for key elements.

3. JavaScript Functionality

The registration button is associated with a click event listener. Key features:

- **Field Validation**: Ensures required fields are filled and data formats are correct (e.g., email, mobile number).
- **API Integration**: Uses the fetch API to send registration data to the backend for processing.

Code Features

Validation Rules

1. **Name**: Cannot be empty; matches GATE record.
2. **Date of Birth**: Must be in the past.
3. **Mobile Number**: Exactly 10 digits.
4. **Email**: Valid format (e.g., example@domain.com).

3.2 Frontend Field and Component Table

Field Name	Field Type	Purpose	Validation Rules	Additional Notes
Full Name (name)	text	Captures the applicant's full name as in GATE.	Cannot be empty.	Placeholder: "Full Name As in GATE".
Date of Birth (dob)	date	Captures the applicant's date of birth.	- Cannot be empty. - Must be in the past.	No manual input; uses date picker.
Mobile Number (mobile	number	Captures the applicant's mobile number.	- Cannot be empty. - Must be exactly 10 digits. - Must contain only numeric characters.	Placeholder: "Mobile Number".
Email (email)	email	Captures the applicant's email address.	- Cannot be empty. - Must follow valid email format (e.g., example@domain.com).	Placeholder: "Enter Email".

COAP Registration ID (regid)	text	Captures the applicant's unique COAP registration ID.	- Cannot be empty.	Placeholder: "Enter COAP Registration ID".
Password (pass)	password	Captures the applicant's chosen password.	- Cannot be empty. - (Optional for future): Minimum length, alphanumeric validation.	Placeholder: "Enter Password".
Year (year)	text	Captures the applicant's GATE exam year.	- Cannot be empty. - Should be a valid year (e.g., 2023).	Placeholder: "Year".
"Complete Registration" Button (register)	button	Submits the registration data to the backend.		Triggers the onclick event to validate fields and send a POST request to the backend API.

4.1 Admin Login Page:

Structure and Design

1. **HTML**:
    - Contains a styled login form for admins to input their username and password.
    - Includes a responsive layout with clean styling for better user experience.
    - Uses input fields for Admin Username and Password.
2. **CSS (Inline)**:
    - Styling for buttons, form elements, and wrappers for better visual appearance.
    - Includes a box-shadow for a modern card-style form appearance.
    - Colors align with IIT branding (#00274D for dark blue background).
3. **User Experience**:
    - Input fields are required (required attribute), ensuring no empty fields on submission.
    - Inline error handling and visual confirmation via alerts for incorrect credentials.

JavaScript (Dynamic Functionality)

1. **Login Logic**:
    - Captures user inputs for adminid and adminpass.
    - Sends the inputs as JSON to the server endpoint /admin_login using a POST request.
2. **Response Handling**:
    - If the server response confirms the admin (Admin Matched), it:
        - **Sets a cookie** (adminLoggedIn=true) with a 1-hour expiration.
        - **Redirects** to the admin_dashboard.html page.
    - If credentials are incorrect:
        - Shows an alert (Incorrect credentials. Please try again.).
        - Clears the input fields.

3. **Session Persistence**:
    - On page load (window.onload), the script checks for the adminLoggedIn cookie.
    - If the cookie exists, it redirects the admin to the dashboard without requiring a new login.

4.2 Frontend Field and Component Table

Field Name	Field Type	Purpose	Validation Rules	Additional Notes
Admin Username	Text Input	Captures the applicant's full name as in GATE.	Required, non-empty	ID: adminid, Placeholder: "Enter Admin Username"
Password	Password Input	Captures the admin's password for login authentication.	Required, non-empty	ID: adminpass, Placeholder: "Enter Password"
Sign In Button	Button	Submits the login form to the backend for verification.	-	ID: signin, Triggers API /admin_login
Admin Login Banner	Static Text & Image	Displays branding and page title for admin login.	-	-

5.1 Password Change Page

This page allows users to change their passwords for their COAP IIT Goa account. It includes the following sections:

1. Header Section

- The top of the page contains the IIT Goa logo and a title stating "COAP - IIT Goa 2024". This ensures users know they are on the official IIT Goa COAP page.
- A small marquee section under the header displays a message stating: "Password Change In Progress" to inform the user about the ongoing process.

2. Input Fields

- **COAP Registration ID**: This is a required field where users must enter their COAP Registration ID. This helps to uniquely identify the user for the password change process.
- **New Password**: A password input field where users enter their new password. This is required and should be a strong, secure password.
- **Confirm Password**: This field allows the user to re-enter the new password for confirmation. It ensures that the user has not made a mistake when typing their new password.

3. Password Change Button

- After entering the required information, the user can click the "Change Password" button to initiate the password change process.

4. Password Validation

- When the user clicks the "Change Password" button, the system performs the following checks:
    - **Password Match**: It ensures that the new password and confirm password fields match. If they don't, an alert is shown to the user.
    - **COAP Registration ID**: If the COAP Registration ID field is empty, the system prompts the user to enter their registration ID before proceeding.

5. Backend Request

- After passing the validation, a request is sent to the backend server to update the password. This is done through an API call to the server at "http://127.0.0.1:5000/changepassword", which is expected to handle the logic of changing the password in the database.
- The request sends a JSON object containing the regid (COAP Registration ID) and the password the user has entered.

6. Response Handling

- Once the backend processes the password change request, it sends a response. The frontend (JavaScript) listens for this response:
    - If the password change is successful, a success message is displayed.
    - If there is any error or issue during the process, it is caught and displayed in the console for debugging.

7. Error Handling

- Any errors that occur while sending the request or processing it are caught by the catch block, and an error message is logged in the console for troubleshooting.

.

5.2 Frontend Field and Component Table

Field Name	Field Type	Purpose	Validation Rules	Additional Notes
COAP Registration ID	Text Input	Field for the user to enter their COAP Registration ID.	Required field; must not be empty.	Used to identify the user for the password change request.
New Password	Password Input	Field for the user to enter their new password.	Required field; must meet strength requirements (e.g., minimum length, mix of characters).	Ensures the new password is strong and secure.
Confirm Password	Password Input	Field for the user to re-enter the new password for confirmation.	Required field; must match the "New Password" field.	Ensures that the user has correctly typed the new password.
Change Password Button	Button	Button to submit the password change request.	None	Triggers the validation and backend request for password change.

6 Applicant and Dashboard Page

6.1 Project Overview:

The project aims to create an interactive and user-friendly dashboard to manage the **Common Offer Acceptance Portal (COAP)** at **IIT Goa**. The dashboard facilitates the collection, validation, and storage of applicant details, GATE scores, academic records, and relevant certificates. The application integrates a dynamic frontend and backend, ensuring seamless data handling and secure submission processes.

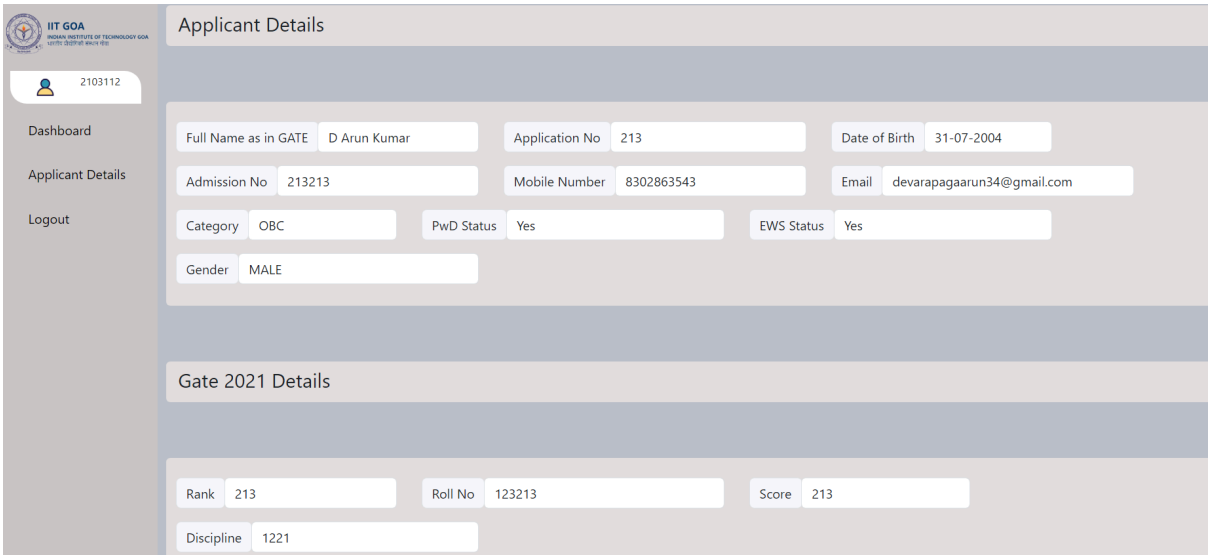
Features Implemented

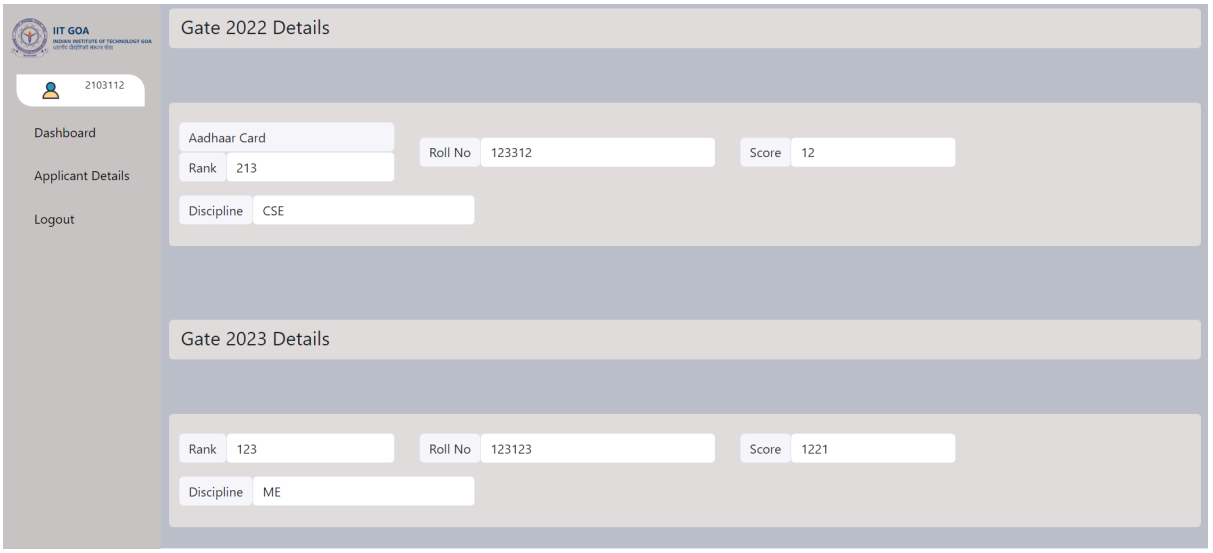
1. Dashboard Interface

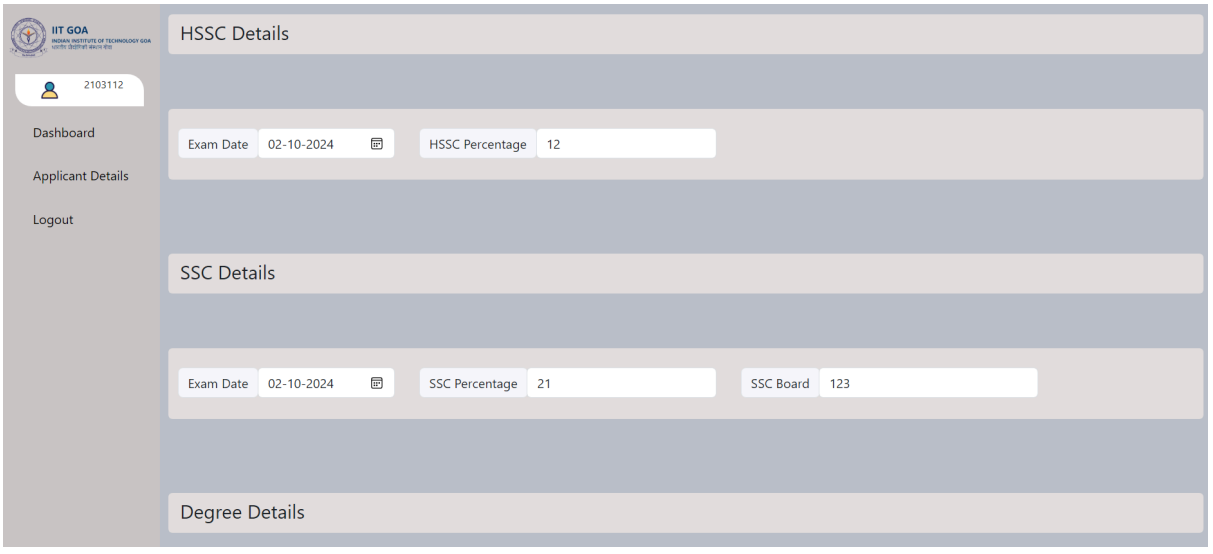
- **Sidebar Navigation**:
    - Provides access to:
        - **Dashboard**: Main landing page with user details.
        - **Applicant Details**: Page to input and review personal and academic information.
        - **Logout**: Ends user session and redirects to the login page.
    - The navigation sidebar dynamically fetches the logged-in user's registration ID and displays it alongside a placeholder profile image.
- **Responsive Design**:
    - Utilizes **Bootstrap 5.3** for styling and responsiveness.
    - Ensures proper alignment and usability across different devices.

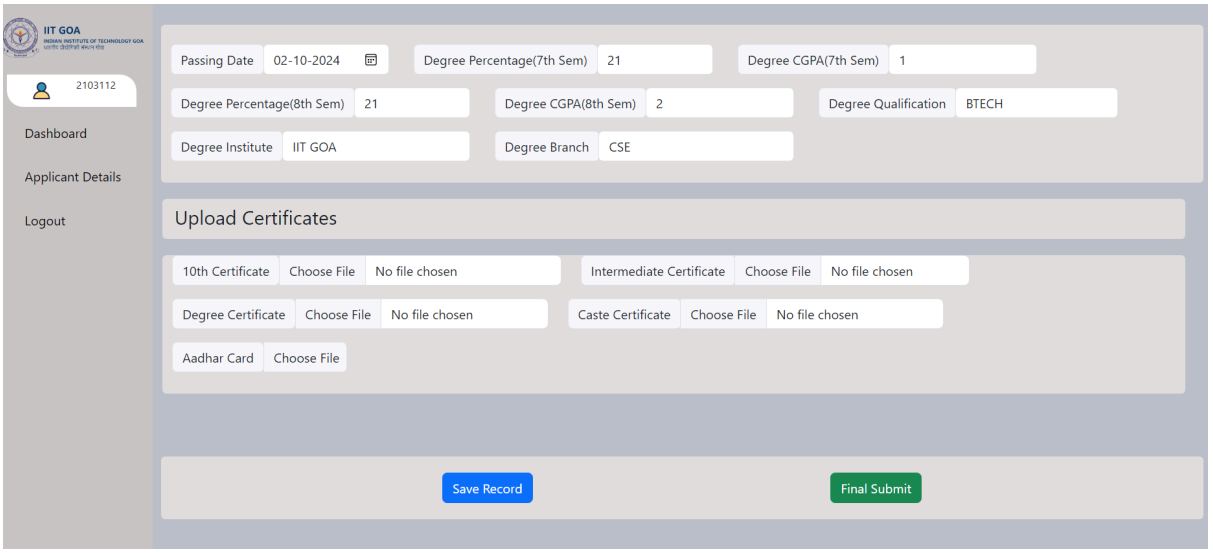
2. Personal Details Form

The form collects critical information about applicants:


IIT GOA
2103112
Dashboard
Applicant Details
Logout
Applicant Details
Full Name as in GATE
D Arun Kumar
Application No
213
Date of Birth
31-07-2004
Admission No
213213
Mobile Number
8302863543
Email
devarapagaarun34@gmail.com
Category
OBC
PwD Status
Yes
EWS Status
Yes
Gender
MALE
Gate 2021 Details
Rank
213
Roll No
123213
Score
213
Discipline
1221


IIT GOA
2103112
Dashboard
Applicant Details
Logout
Gate 2022 Details
Aadhaar Card
Rank
213
Roll No
123312
Score
12
Discipline
CSE
Gate 2023 Details
Rank
123
Roll No
123123
Score
1221
Discipline
ME


IIT GOA
2103112
Dashboard
Applicant Details
Logout
HSSC Details
Exam Date
02-10-2024
HSSC Percentage
12
SSC Details
Exam Date
02-10-2024
SSC Percentage
21
SSC Board
123
Degree Details



- **Fields**:
    - Full Name (as in GATE)
    - Application Number
    - Date of Birth
    - Admission Number
    - Mobile Number (validated for Indian formats)
    - Email Address (with format validation)
    - Category (General, OBC, SC, ST) via a dropdown menu
    - PwD (Persons with Disabilities) status
    - EWS (Economically Weaker Section) status
    - Gender
- **Validation**:
    - Uses JavaScript to check for required fields.
    - Implements regex for mobile numbers and email formatting.

3. GATE Examination Details

Separate sections for **GATE 2021**, **2022**, and **2023** scores:

- **Fields**:
    - Rank
    - Roll Number
    - Score
    - Discipline

- **Purpose**:
    - Ensures accurate tracking of applicant performance across multiple GATE attempts.
    - Supports eligibility verification for admission.

4. Academic Details

Captures critical academic data:

- **Higher Secondary School Certificate (HSSC)**:
    - Exam Date
    - Percentage
- **Secondary School Certificate (SSC)**:
    - Exam Date
    - Percentage
    - Board
- **Degree Details**:
    - Passing Date
    - Percentage (7th and 8th Semester)
    - CGPA (7th and 8th Semester)
    - Qualification
    - Institute Name
    - Branch

Validation Rules:

- Percentages must be between 0 and 100.
- CGPA must be between 0 and 10.

5. Certificate Upload Section

Allows applicants to upload required documents:

- **Certificates Supported**:
    - 10th Certificate
    - Intermediate Certificate
    - Degree Certificate
    - Caste Certificate
    - Aadhaar Card
- **Upload Mechanism**:

    - Uses **<form>** elements with POST requests to server endpoints.
    - Validates the presence of all required files before submission.

6. Actions and Functionality

- **Save Record**:
    - Temporarily stores applicant data for review before final submission.
- **Final Submit**:
    - Sends validated data to the backend for permanent storage.
- **Dynamic Fetching**:
    - Registration ID (regid) is dynamically retrieved from session cookies.
    - Fetches and pre-fills applicant details upon login, reducing redundancy.
- **Error Handling**:
    - Alerts users for invalid input (e.g., incorrect email format, missing data).
    - Provides feedback on successful or failed submissions.

7. Backend Communication

- **Endpoints**:
    - /storedetails: Accepts form data and stores it in the backend.
    - /getuserdetails: Retrieves user-specific information using the regid.
- **FormData API**:
    - Encodes file uploads and form fields for seamless backend transmission.

Technologies Used

1. Frontend:

- **HTML and CSS**:
    - Markup and styling for user interfaces.
- **Bootstrap 5.3**:
    - Ensures responsive design and consistent aesthetics.
- **JavaScript**:
    - Validates inputs.

    - Handles dynamic interactions and data pre-filling.

2. Backend:

- Server communication is facilitated via RESTful API endpoints.
- The backend likely uses **Flask** or a similar framework (based on endpoint structure).

3. Database:

- Although unspecified, the backend presumably integrates with a relational database (e.g., PostgreSQL or MySQL) to store applicant details and file metadata.

Progress Since Last Milestone

1. **Enhanced Validation**:
    - Added robust input checks for percentages, CGPA, email, and mobile number formats.
2. **File Upload Support**:
    - Fully implemented upload forms for certificates with validation.
3. **Dynamic Pre-Filling**:
    - Integrated a mechanism to fetch and pre-fill user data on login.
4. **UI/UX Improvements**:
    - Polished design with better alignment and navigation.
5. **Error Messages**:
    - Improved error messages to guide users effectively.

Challenges Faced

1. **File Upload Validation**:
    - Ensuring all file inputs are mandatory and correctly linked to backend endpoints.
2. **Dynamic Data Fetching**:
    - Handling errors in data retrieval from the backend when regid is missing or incorrect.
3. **Input Validation**:
    - Balancing client-side validation with backend checks to ensure data integrity.

6.2 Frontend Field and Component Table

Section	Field	Component Type	Attributes/Validation
Applicant Details	Full Name	`<input> (text)`	Used to identify the user for the password change request.
	Application No	`<input> (text)`	Placeholder: "Application No", Required: Yes
	Date of Birth	`<input> (text)`	Placeholder: "Application No", Required: Yes
	Admission No	`<input> (text)`	Placeholder: "Admission No", Required: Yes
	Mobile Number	`<input> (text)`	Placeholder: "Mobile", Regex: Indian number format, Required: Yes
	Email	`<input> (email)`	Placeholder: "Email", Regex: Email format, Required: Yes
	Category	`<select>`	Options: General, OBC, SC, ST; Required: Yes
	PwD Status	`<input> (text)`	Placeholder: "PwD Status", Required: Yes
	EWS Status	`<input> (text)`	Placeholder: "EWS Status", Required: Yes
	Gender	`<input> (text)`	Placeholder: "Gender", Required: Yes

Section	Field	Component Type	Attributes/Validation
GATE Details 2021	Rank	`<input> (number)`	Placeholder: "Rank", Required: Yes
	Roll No	`<input> (text)`	Placeholder: "Roll No", Required: Yes
	Score	`<input> (number)`	Placeholder: "Score", Required: Yes
	Discipline	`<input> (text)`	Placeholder: "Discipline", Required: Yes
GATE Details 2022	Rank	`<input> (number)`	Placeholder: "Rank", Required: Yes
	Roll No	`<input> (text)`	Placeholder: "Roll No", Required: Yes
	Score	`<input> (number)`	Placeholder: "Score", Required: Yes
	Discipline	`<input> (text)`	Placeholder: "Discipline", Required: Yes

Section	Field	Component Type	Attributes/Validation
Academic Details	HSSC Exam Date	`<input> (date)`	Placeholder: "Exam Date", Required: Yes
	HSSC Percentage	`<input> (number)`	Placeholder: "Exam Date", Required: Yes
	SSC Exam Date	`<input> (date)`	Range: 0-100, Required: Yes
	SSC Percentage	`<input> (number)`	Range: 0-100, Required: Yes
	SSC Board	`<input> (text)`	Placeholder: "Board", Required: Yes
	Degree Passing Date	`<input> (date)`	Placeholder: "Passing Date", Required: Yes
	Degree Percentage (7th Sem)	`<input> (number)`	Range: 0-100, Required: Yes
	Degree CGPA (7th Sem)	`<input> (number)`	Range: 0-10, Required: Yes
	Degree Percentage (8th Sem)	`<input> (number)`	Range: 0-100, Required: Yes
	Degree CGPA (8th Sem)	`<input> (number)`	Range: 0-10, Required: Yes
	Degree Qualification	`<input> (text)`	Placeholder: "Qualification", Required: Yes
	Degree Institute	`<input> (text)`	Placeholder: "Institute", Required: Yes
	Degree Branch	`<input> (text)`	Placeholder: "Branch", Required: Ye

Certificate Upload	10th Certificate	`<input> (file)`	Required: Yes
	Intermediate Certificate	`<input> (file)`	Required: Yes
	Degree Certificate	`<input> (file)`	Required: Yes
	Caste Certificate	`<input> (file)`	Required: Yes
	Aadhaar Card	`<input> (file)`	Required: Yes

5. Admin Dashboard for Applicant Profile Management and Eligibility Criteria Filtering

Introduction

The continuation of the Bachelor's Thesis Project (BTP) focuses on the ongoing development of an admin dashboard application designed to streamline the management of applicant profiles for the Visvesvaraya PhD scheme. The dashboard includes filtering functionality for eligibility criteria, which allows admins to evaluate and manage applicants based on their educational qualifications, GATE scores, and other personal data.

The key components of the project include:

1. **Eligibility Criteria Filters**: A section that enables the filtering of applicants based on predefined eligibility requirements, such as CPI values for different categories (e.g., IIT graduates, SC/ST applicants, others).
2. **Applicant Profile Management**: Displaying detailed profiles of applicants, including their personal and academic information.
3. **Certificate Management**: Offering a preview and download feature for certificates submitted by applicants, using PDF previews to enhance the user experience.

Objectives of the Project

The main goals for the continuation phase of the project are:

1. **Filtering by Eligibility Criteria**: Implementing an effective filter mechanism based on eligibility rules, such as CPI scores for various categories (IIT, SC/ST, other non-IIT) and GATE qualifications.
2. **Profile Card Layout**: Creating a user-friendly interface that displays comprehensive applicant profiles, including their academic history, personal information, and submitted certificates.
3. **PDF Preview and Download**: Integrating a feature to preview and download certificates in PDF format directly from the dashboard.

Current Progress

1. **Eligibility Filtering**:
    - The functionality to apply filters based on CPI values has been successfully implemented. This allows the admin to enter minimum CPI requirements for different categories (e.g., IIT graduates, SC/ST applicants, etc.), and the system filters applicants

accordingly. This feature is key to narrowing down the applicant pool for further processing.

2. **Applicant Profiles Display**:
    - A comprehensive profile card layout has been designed to present the key information of each applicant, such as their name, application number, educational qualifications, GATE scores, and more. These profiles are dynamically populated based on the filtered criteria and show all relevant details for each applicant in a user-friendly format.
3. **Certificate Management**:
    - The certificate section of the dashboard has been enhanced with features allowing the admin to preview the certificates before downloading them. Certificates are shown as clickable thumbnail images, and admins can view them in full size in a modal window. Additionally, there is a download button for each certificate to allow for easy access.

Challenges and Solutions

1. **Complexity in Filtering**:
    - Initially, there were challenges in implementing complex filter criteria, especially when dealing with multiple eligibility conditions for different applicant categories. However, these challenges were overcome by organizing the filtering criteria into structured input fields and using them in combination with backend filtering logic.

3. Report on Flask-Based Backend for COAP Registration and Management System

1. Introduction

The backend of the COAP Registration and Management System is designed using Flask, a lightweight web framework for Python. The application facilitates user registration, login, password management, file uploads, and data retrieval for applicants and administrators. The system interacts with a PostgreSQL database for storing and retrieving user and applicant information, ensuring secure data processing and validation.

This report provides a detailed overview of the Flask application structure, routes, database interactions, and features available within the system.

2. System Features

2.1. User Registration

The registration feature allows new users to register for the COAP process by providing necessary details such as name, date of birth, mobile number, email, and COAP registration ID. Upon successful registration:

- The password is hashed using the SHA-256 algorithm for enhanced security.
- User information is stored in the database, and a success message is returned.
- If a user with the same registration ID already exists, the system prevents duplicate registration.

2.2. User Login

The login route allows users to authenticate by entering their COAP registration ID and password. The system:

- Retrieves the user's hashed password from the database.
- Compares the entered password (after hashing) with the stored password.
- Returns a message indicating whether the login was successful or if the credentials are incorrect.

2.3. Change Password

The system allows registered users to change their password. After validating the user’s registration ID:

- The new password is hashed and updated in the database.
- A success message is returned if the password change is successful.

2.4. Store Applicant Details

Users can upload several important documents (such as certificates and Aadhaar card) as part of their application. The system:

- Receives files and stores them as binary data in the database.
- Calculates the applicant's shortlist status based on their qualifications and category.
- Stores the applicant's data and associated files, including 10th, intermediate, degree certificates, caste certificate, and Aadhaar card.

If the applicant's details already exist in the database, the system updates their information; otherwise, it inserts a new record.

2.5. Shortlist Calculation

The shortlist status of applicants is determined based on the following conditions:

- For IIT graduates with a BTech degree, a CPI (Cumulative Performance Index) of 8.0 or higher is required for shortlisting.
- For non-IIT graduates, the requirements are adjusted based on the applicant's category (SC/ST or other) and degree qualifications.

The shortlist status is automatically calculated and stored in the database based on the input criteria.

2.6. Retrieve Applicant Details

The system provides an API to retrieve an applicant’s details based on their COAP registration ID. The route:

- Fetches detailed applicant information from the database.

- Converts any binary data (like certificates) to Base64 format for easy display or further processing.
- Returns the applicant's details in a structured format.

2.7. Admin Dashboard

The admin dashboard route allows administrators to access and manage applicant data:

- The administrator can filter applicants based on CPI thresholds for various categories (IIT graduates, SC/ST, or others).
- The dashboard displays a list of applicants who meet the criteria, including their personal information, academic records, and shortlisted status.

2.8. Admin Login

The admin login route allows administrators to authenticate using a username and password. The system:

- Verifies the admin's credentials against the database.
- Returns a success or failure message based on the authentication result.

3. Database Interaction

3.1. PostgreSQL Database

The application uses PostgreSQL as its database system. It handles the following tasks:

- **User Authentication:** Stores encrypted user passwords for secure login.
- **Applicant Details:** Stores comprehensive information about applicants, including personal details, academic history, certificates, and application status.
- **Admin Data:** Stores administrator credentials for managing the system.

3.2. Database Tables

The system interacts with two main tables in the PostgreSQL database:

- **users**: Contains user details like name, COAP registration ID, and password.

- **applicant_detail**: Stores detailed information about applicants, including their personal data, academic records, and uploaded certificates.
- **admin**: Stores admin login credentials for authentication.

4. Security Measures

- **Password Encryption:** The system uses SHA-256 hashing to encrypt user passwords before storing them in the database, ensuring that even administrators cannot access plaintext passwords.
- **File Handling:** Uploaded files are stored as binary data in the database to ensure data integrity and security.
- **Database Queries:** All user inputs are sanitised using parameterized queries to prevent SQL injection attacks.

5. Conclusion

This Flask-based backend for the COAP Registration and Management System is designed to provide a secure, efficient, and scalable solution for managing user registrations, login, file uploads, and data retrieval. The system ensures that sensitive data such as passwords and certificates are handled securely, while also providing administrators with an easy-to-use interface to manage applicants. The integration with PostgreSQL allows for reliable and robust data storage and retrieval, supporting the system's core functionalities effectively.