Name:	ARYA T V
Roll Number:	7376222AD117
Seat No:	324
Problem Id:	24
Problem Statement:	Invigilation duty automation

Project ID	Module Name	Module Description	Domain
4	Invigilation duty automation	Automate C o E duty for faculty members.	C o E

Context:

Educational institutions often face significant challenges in managing invigilation duties for exams. Assigning invigilators manually is a time-consuming task that is prone to errors and inefficiencies. This can lead to issues such as uneven distribution of workload, scheduling conflicts, and last-minute changes that disrupt the smooth conduct of exams.

Problem:

The manual process of assigning invigilation duties to faculty and staff members is inefficient and error-prone. It often results in:

1. **Inequitable Distribution:** Some staff members may be overburdened while others have fewer duties.
2. **Scheduling Conflicts:** Overlapping schedules and double bookings for invigilation can occur, leading to confusion and last-minute adjustments.

3. **Lack of Flexibility:** Inability to quickly adapt to changes such as unexpected staff unavailability or exam schedule modifications.
4. **Administrative Burden:** Significant time and effort are required to manually plan, coordinate, and communicate invigilation assignments.

Objective:

Develop an automated system to manage invigilation duties efficiently. The system should:

1. **Automate Allocation:** Use algorithms to automatically assign invigilation duties based on predefined criteria such as staff availability, workload balance, and specific qualifications.
2. **Handle Conflicts:** Identify and resolve scheduling conflicts, ensuring no double bookings or overlaps.
3. **Adaptability:** Allow for easy adjustments in case of sudden changes in staff availability or exam schedules.
4. **User-Friendly Interface:** Provide a simple and intuitive interface for administrators to oversee and manage invigilation duties.
5. **Communication:** Automatically notify staff members of their invigilation assignments and any changes in real-time.

Expected Outcomes:

1. **Efficiency**: Reduced time and effort required for managing invigilation duties.
2. **Fairness**: More equitable distribution of invigilation tasks among staff.
3. **Accuracy**: Fewer errors and scheduling conflicts.
4. **Responsiveness**: Improved ability to handle last-minute changes smoothly.

By implementing an invigilation duty automation system, educational institutions can streamline their exam processes, ensuring that exams are conducted smoothly and efficiently with minimal administrative hassle.

Scope:

The Invigilation Duty Automation module aims to streamline and optimize the process of assigning and managing invigilation duties within educational institutions. The scope of this module includes the following components:

1. User Management:

- **Administrator Access:** Create and manage admin accounts with full control over the system.
- **Faculty/Staff Access:** Create and manage faculty and staff accounts with relevant information such as availability, qualifications, and workload preferences.

2. Data Input and Management:

- **Exam Schedule Input:** Input and manage exam schedules, including dates, times, locations, and specific requirements.
- **Faculty Availability:** Input and update faculty availability and preferences for invigilation duties.
- **Qualification and Special Requirements:** Manage and assign special invigilation requirements based on subject expertise or specific qualifications.

3. Automated Duty Allocation:

- **Algorithm Development:** Develop algorithms to automatically allocate invigilation duties based on predefined criteria such as availability, workload balance, and qualifications.

- **Conflict Resolution:** Implement mechanisms to detect and resolve scheduling conflicts, ensuring no double bookings or overlaps.

4. Notification and Communication:

- **Automated Notifications:** Automatically notify faculty and staff of their invigilation assignments via email or other communication channels.

- **Real-Time Updates:** Provide real-time updates for any changes in assignments or schedules.

- **Reminder System:** Set up reminders for upcoming invigilation duties.

5. User Interface:

- **Admin Dashboard:** Develop an intuitive admin dashboard for managing and overseeing the invigilation schedule.

- **Faculty/Staff Portal:** Create a user-friendly portal for faculty and staff to view their assignments, update their availability, and receive notifications.

6. Reporting and Analytics:

- **Duty Reports:** Generate reports on invigilation duties, including assigned duties, attendance, and any issues encountered.

- **Workload Analysis:** Analyze the distribution of invigilation duties to ensure fair and balanced workload among staff.

- **Historical Data:** Maintain historical data for reference and future planning.

7. Adaptability and Flexibility:

- **Dynamic Adjustments:** Allow for easy adjustments in case of sudden changes such as staff unavailability or exam schedule modifications.

- **Customization:** Enable customization of allocation criteria and notification preferences to suit the institution's specific needs.

8. Security and Privacy:

- **Data Protection:** Ensure secure handling of personal data and compliance with relevant data protection regulations.

- **Access Control:** Implement access controls to ensure that only authorized users can manage and view sensitive information.

9. Integration with Existing Systems:

- **System Integration:** Integrate with existing systems such as the institution's Learning Management System (LMS), Human Resource Management System (HRMS), and email communication tools for seamless operation.

10. Training and Support:

- **User Training:** Provide training resources and sessions for administrators and faculty to effectively use the system.

- **Technical Support:** Offer ongoing technical support to address any issues or questions that arise during use.

By covering these components, the Invigilation Duty Automation module will provide a comprehensive solution to manage invigilation duties efficiently, ensuring smooth and fair exam administration in educational institutions.

Dependencies:

1. **Web Framework:**

   Django: A high-level Python web framework for building robust and scalable web applications.

2. **Database and ORM:**

   - **Django ORM:** Integrated with Django for database interactions.
   - **Psycopg2:** For connecting to PostgreSQL databases.

3. **Data Manipulation and Analysis:**

- **Pandas:** For data manipulation and analysis.

4. **Background Tasks:**

- **Celery:** For handling asynchronous tasks and job queues.

5. **Authentication and Authorization:**

- **Django Allauth:** For user authentication and authorization

6. **API Development:**

- **Django Rest Framework (DRF):** For building RESTful APIs.

7. **Email and SMS Notifications:**

- **SendGrid:** For sending email notifications.

- **Twilio:** For sending SMS notifications.

8. **Deployment and DevOps:**

- **Gunicorn:** For serving the Django application

- **Docker:** For containerizing the application.

9. **Testing:**

- **PyTest:** For writing and running tests.

10. **Security:**

- **Django Security Middleware:** For security features in Django.

11. **Logging and Monitoring:**
    - **Sentry:** For error tracking and monitoring.

These dependencies cover the core functionality needed for developing, deploying, and maintaining the Invigilation Duty Automation module efficiently.

The Invigilation Duty Automation module aims to streamline the process of assigning and managing invigilation duties within educational institutions. Below is a high-level overview of the system architecture, including its main components and their interactions:

1. Architecture Overview:

Frontend:

- **User Interface (UI):** Built using HTML, CSS, and JavaScript, or a frontend framework like React.js, for interacting with the system.

- **Admin Dashboard:** Provides features for administrators to manage exam schedules, faculty information, and invigilation assignments.

- **Faculty Portal:** Allows faculty members to view their invigilation schedules and update their availability.

Backend:

- **Web Framework:** Django, which handles HTTP requests, business logic, and serves the frontend.

- **REST API:** Built using Django Rest Framework (DRF) to facilitate communication between the frontend and backend.

- **Task Management:** Celery for handling background tasks such as sending notifications.

Database:

- **Relational Database:** PostgreSQL for storing exam schedules, faculty data, availability, assignments, and historical records.

Communication:

- **Email Notifications:** SendGrid for sending automated email notifications to faculty and staff.

- **SMS Notifications:** Twilio for sending SMS alerts for important updates.

Deployment and DevOps:

- **Containerization:** Docker for packaging the application into containers.

- **Web Server:** Gunicorn for serving the Django application.

- **Reverse Proxy:** Nginx for handling HTTP requests and serving static files.

2. Component Breakdown:

Frontend Components:

- **Login and Authentication:** Secure login system for administrators and faculty.
- **Dashboard:** Interface for administrators to create and manage exam schedules, input faculty availability, and assign invigilation duties.
- **Schedule Viewer:** Interface for faculty to view their assigned invigilation duties and update their availability.

Backend Components:

- **Authentication and Authorization:**
  - **Django Allauth:** Manages user authentication and permissions.
- **Scheduling Algorithm:**
  - **Assignment Logic:** Automatically assigns invigilation duties based on predefined criteria like availability and workload balance.
  - **Conflict Resolution:** Detects and resolves scheduling conflicts.
- **API Endpoints:**
  - **Exam Management:** CRUD operations for exam schedules.
  - **Faculty Management:** CRUD operations for faculty data and availability.

    - **Assignment Management:** CRUD operations for invigilation assignments.

- **Notification System:**
    - **Email Notifications:** Sends notifications to faculty about their assignments and updates.
    - **SMS Notifications:** Sends urgent alerts via SMS.

- **Background Task Management:**
    - **Celery:** Manages background tasks such as sending notifications and periodic updates.

Database Schema:

- **Tables:**
    - **Users:** Stores user information and roles.
    - **Exams:** Stores exam schedules and details.
    - **Faculty:** Stores faculty information and availability.
    - **Assignments:** Stores invigilation assignments.

    - **Notifications:** Logs notifications sent to users.

Deployment:

- **Containerization:** Docker containers for application, database, and Celery workers.

- **Continuous Integration/Continuous Deployment (CI/CD):** Automates testing and deployment using tools like GitHub Actions or Jenkins.

- **Monitoring and Logging:**

    - **Sentry:** For error tracking.

    - **Prometheus/Grafana:** For monitoring application performance (optional).

3. Workflow:

1. **Admin Logs In:** Administrator logs into the system using their credentials.

2. **Setup Exam Schedule:** Admin creates and manages exam schedules through the admin dashboard.

3. **Faculty Updates Availability:** Faculty members log in and update their availability for invigilation duties.

4. **Automatic Assignment:** The system runs the scheduling algorithm to assign invigilation duties based on availability and predefined criteria.

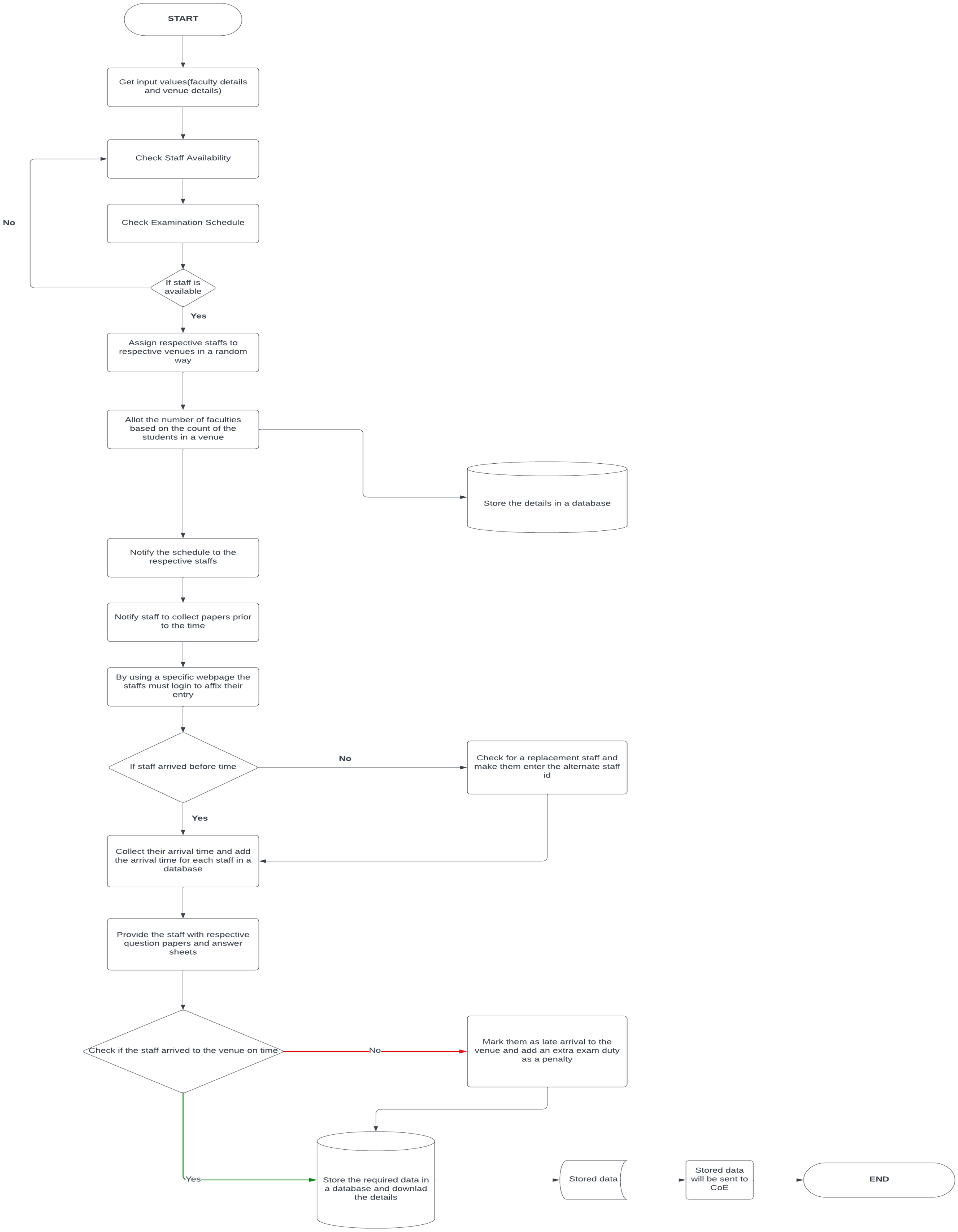
5. **Notification Dispatch:** Celery tasks send email and SMS notifications to faculty members about their assignments.

6. **Real-Time Updates:** Any changes to the schedule or assignments trigger real-time notifications to affected faculty members.

7. **Monitoring:** Admins monitor the schedule, resolve conflicts, and make adjustments as necessary.

By integrating these components and workflows, the Invigilation Duty Automation module ensures efficient, fair, and transparent management of invigilation duties within educational institutions.

Mind Map:


START
Get input values(faculty details and venue details)
Check Staff Availability
No
Check Examination Schedule
If staff is available
Yes
Assign respective staffs to respective venues in a random way
Allot the number of faculties based on the count of the students in a venue
Store the details in a database
Notify the schedule to the respective staffs
Notify staff to collect papers prior to the time
By using a specific webpage the staffs must login to affix their entry
If staff arrived before time
No
Check for a replacement staff and make them enter the alternate staff id
Yes
Collect their arrival time and add the arrival time for each staff in a database
Provide the staff with respective question papers and answer sheets
Check if the staff arrived to the venue on time
No
Mark them as late arrival to the venue and add an extra exam duty as a penalty
Yes
Store the required data in a database and downlad the details
Stored data
Stored data will be sent to CoE
END