BANNARI AMMAN INSTITUTE OF TECHNOLOGY
Stay Ahead

BANNARI AMMAN INSTITUTE OF TECHNOLOGY

An Autonomous Institution Affiliated to Anna University - Chennai, Accredited by NAAC with A+ Grade

Sathyamangalam - 638401 Erode District, Tamil Nadu, India

Name	Karthikeyan S N
Roll no	7376221CS188
Seat no	241
Project ID	1
Problem Statement	Placement Registration Form

Technical Components

Component	Tech Stack
Backend	Spring Boot
Frontend	React js
Database	MySQL
API	REST Ful API

1. Introduction:

- The primary objective of this document is to delineate the requirements essential for the development of a Placement Registration System.
- The system aims to streamline the process of posting job roles, enabling eligible students to apply, and facilitating the management of student academic details by administrators.
- By providing detailed requirements, this document serves as a blueprint for the development and implementation of the Placement Registration System, ensuring its effectiveness and functionality.

2. System Overview:

- The Placement Registration System comprises a front-end interface developed using React.js, ensuring a modern and interactive user experience.
- On the back end, the system is powered by Java with Spring Boot, providing robustness and scalability to handle various functionalities seamlessly.
- Data storage and retrieval are managed by a MySQL database, ensuring efficiency and reliability in managing student information and job postings.
- Communication between the front end and back end is facilitated through RESTful APIs, enabling seamless interaction and data exchange between different system components.

3. Functional Requirements:

3.1 Admin Module:

Secure Login: The system should provide a secure login mechanism for administrators to access the system.

Job Posting: Admins should be able to post job roles on the forum, specifying job details such as title, description, and eligibility criteria.

Eligibility criteria may include minimum CGPA, 10th & 12th marks, or other academic requirements.

Academic Details Management: Admins should have the capability to update student academic details, including CGPA, on a regular basis to ensure accurate information.

Profile Management: The system should enable admins to manage student profiles, including functionalities such as viewing, editing, and deleting profiles as needed.

Admins should be able to view detailed information about each student, including their academic history, contact details, and application status.

3.2 Student Module:

Job Role Browsing:

- The system should provide an accessible platform for students to browse and view job roles posted on the forum.
- Students should be able to search for job roles based on various criteria such as job title, company, or eligibility requirements.

Job Application:

- Students should have the ability to apply for job roles based on the eligibility criteria set by the admin.
- The system should validate student eligibility before allowing them to submit their applications.

Profile Management:

- Optionally, students may be provided with functionality to update their profile information as needed.
- This may include updating contact details, academic information, or adding additional skills and experiences to their profile.

3.3 Automated Offer Distribution:

Automated System Implementation:

- The system should implement an automated mechanism to filter and distribute placement offers to eligible students based on the criteria defined by the admin.

- This automation should streamline the process of offer distribution, reducing manual effort and ensuring timely communication with students.

Eligibility Verification:

- Before sending out placement offers, the system should verify that students meet the specified eligibility criteria set by the admin.
- Only students who meet these criteria should receive placement offers, maintaining fairness and transparency in the process.

Efficient Offer Distribution:

- By automating the offer distribution process, the system can ensure efficient and prompt communication with eligible students, minimizing delays and improving overall system responsiveness.
- This automated approach enhances the efficiency of the placement process, benefiting both students and administrators.

4. Non-Functional Requirements:

4.1 Performance:

Low Latency and Rapid Response Times:

- The system should be designed to exhibit low latency and provide rapid response times to user actions, ensuring a smooth and seamless user experience.
- This includes minimizing the time taken for page loading, form submissions, and other interactions within the system.

Scalability Considerations:

- The system should be architected with scalability in mind to accommodate a potentially large volume of concurrent users.
- This includes implementing strategies such as horizontal scaling, load balancing, and caching to handle increased traffic without compromising performance.
- The system should be able to dynamically scale resources up or down based on demand, ensuring optimal performance under varying usage patterns.

4.2 Security:

Safeguarding User Data:

- The system should employ robust security measures to safeguard user data against unauthorized access, tampering, or theft.
- This includes implementing encryption protocols, secure transmission channels, and strict access controls to protect sensitive information.

Secure Authentication and Authorization:

- The system should utilize secure authentication mechanisms, such as multi-factor authentication or OAuth, to verify the identity of users accessing the system.
- Authorization mechanisms should be implemented to ensure that users are only granted access to resources and functionalities appropriate to their role and privileges.

Encryption of Sensitive Data:

- Sensitive data, including passwords and academic details, should be encrypted both in transit and at rest to prevent unauthorized access.
- Strong encryption algorithms and key management practices should be employed to mitigate the risk of data breaches and unauthorized disclosure of information.

4.3 Reliability:

High System Uptime:

- The system should adhere to reliability standards that ensure high uptime, minimizing downtime and ensuring availability to users.
- This includes implementing measures such as redundant systems, failover mechanisms, and proactive monitoring to mitigate the risk of system failures.

Resilience to Failures:

- The system should be designed to be resilient to failures, including hardware failures, software bugs, and network disruptions.
- Fault-tolerant architectures, such as microservices or distributed systems, should be employed to ensure that failures in one component do not affect the overall system operation.

Effective Error Handling:

- The system should implement effective error handling mechanisms to gracefully manage exceptions and errors that may occur during operation.
- Error messages should be informative and user-friendly, providing clear guidance on how to resolve issues and recover from errors.
- Logging and monitoring systems should be in place to track system errors and facilitate troubleshooting and debugging efforts.

4.4 Usability:

Intuitive and User-Friendly Interface:

- The system should adhere to design requirements that prioritize an intuitive and user-friendly interface.
- Interface elements should be organized logically, with clear navigation paths and intuitive layouts that facilitate ease of use for both administrators and students.

Ease of Navigation and Interaction:

- Navigation within the system should be straightforward and intuitive, allowing users to easily access desired functionalities and information.
- Interaction with interface elements, such as buttons, forms, and menus, should be intuitive and responsive, minimizing user confusion and frustration.

Informative Feedback:

- The system should provide informative feedback to users during various interactions, such as form submissions, error messages, and confirmation dialogs.
- Feedback messages should be clear, concise, and contextually relevant, guiding users through the interaction process and helping them understand the outcome of their actions.

Accessibility Considerations:

- The system should be designed with accessibility in mind, ensuring that all users, including those with disabilities, can effectively navigate and interact with the interface.
- Accessibility features such as screen reader compatibility, keyboard navigation, and color contrast adjustments should be implemented to enhance usability for all users.

5. External Interfaces:

Interaction with MySQL Database:

The system will interact with a MySQL database for efficient storage and retrieval of data, including student profiles, job postings, and eligibility criteria.

Utilization of RESTful APIs:

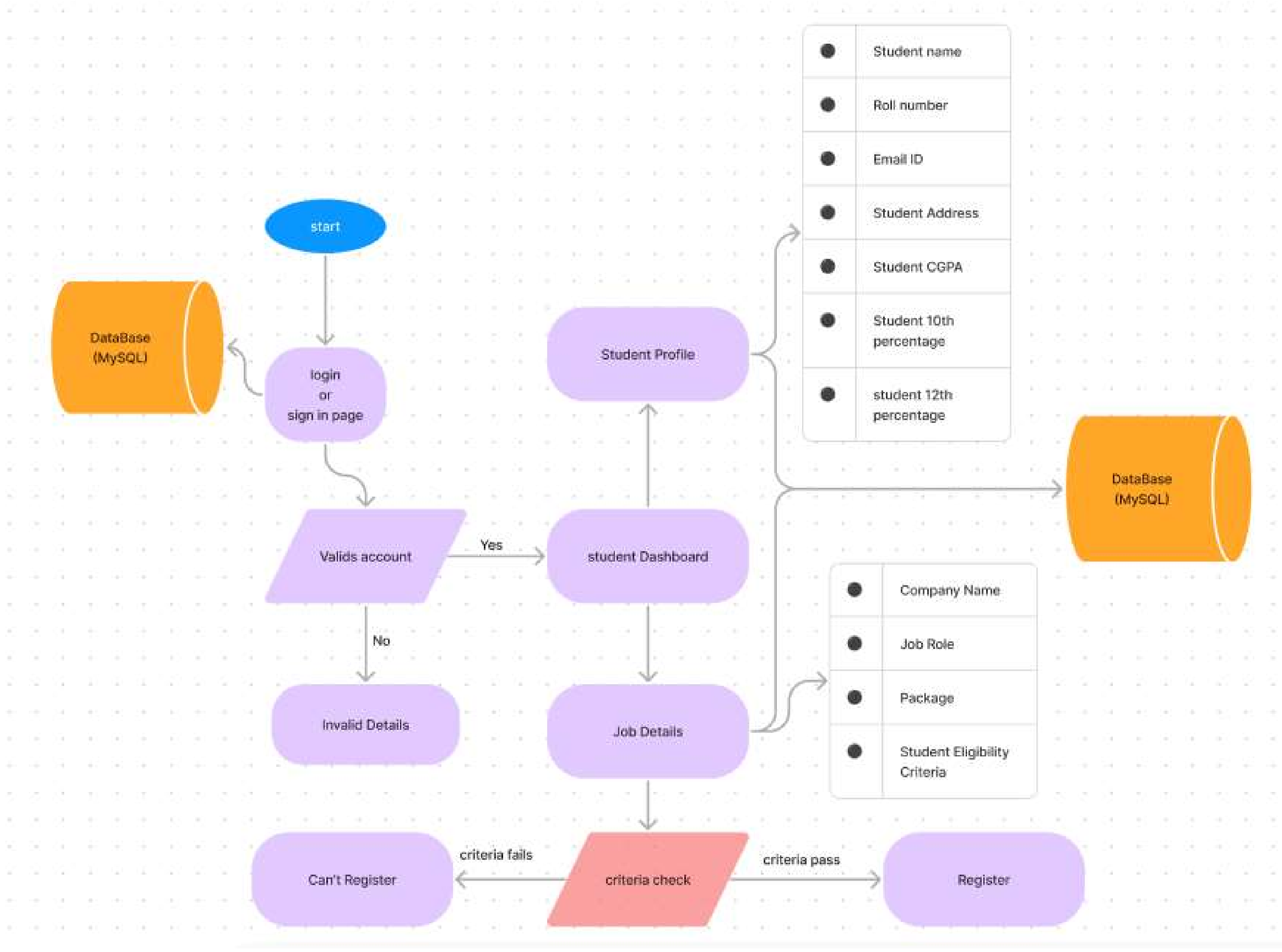
- RESTful APIs will be utilized to enable seamless communication between the front end and back end components of the system.
- These APIs will facilitate data exchange and functionality execution, ensuring smooth integration between different system layers.

6. Constraints:

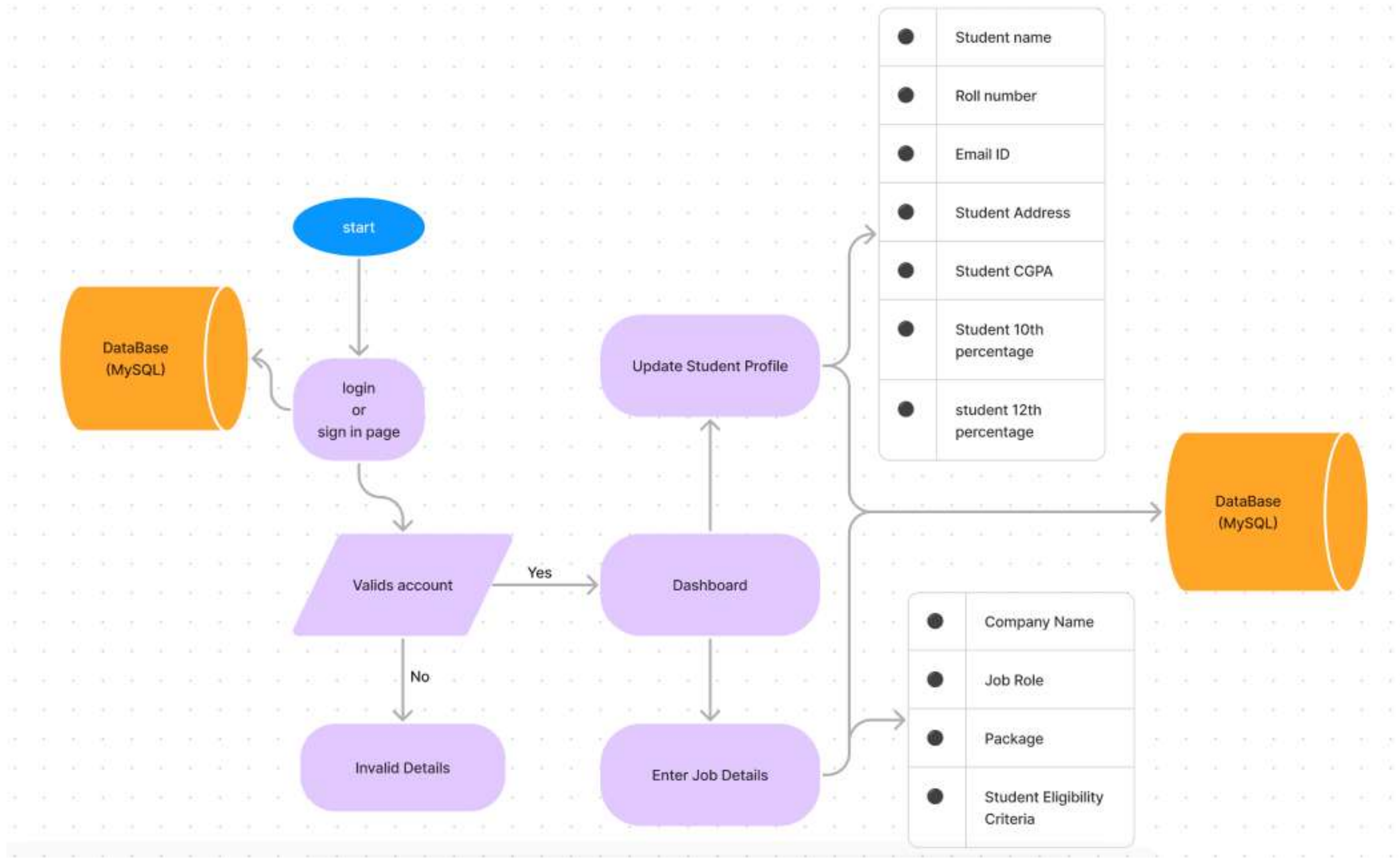
Placed Student Constraints:

- Placed students will be prohibited from applying for a second job opportunity until the completion of their current placement tenure.
- The system will implement mechanisms to enforce this constraint, preventing ineligible students from receiving further placement offers until they are eligible.

USER’S INTERFACE



ADMIN INTERFACE


start
DataBase (MySQL)
login or sign in page
Valids account
Yes
No
Invalid Details
Dashboard
Update Student Profile
Enter Job Details
Student name
Roll number
Email ID
Student Address
Student CGPA
Student 10th percentage
student 12th percentage
DataBase (MySQL)
Company Name
Job Role
Package
Student Eligibility Criteria