Student Name: MANI SANKAR U

Seat No: 169

PROJECT ID: 9

PEOJECT ID title: Reward Point Calculation Portal for TAC

Problem statement:

Build the portal to automatically compute the final reward points of the TAC IDs when the reviewers submit the review marks.

The RP includes :

1. Initial Submission
2. Final Report
3. Plagiarism of final report
4. Review Marks
5. Team Communication
6. Work Log

TAC ID Overview:

The portal will automatically compute and display the final reward points for TAC IDs based on submissions and reviews. The system will cater to three types of users: administrators, reviewers, and students.

System Components:

Front End : React.js

Back End : Java with Spring Boot

Database : MySQL

API : RESTful API

User Roles:

Admin : Manages TAC ID submission, plagiarism and worklog details.

Reviewer : Submits review marks and team communication scores.

Student : Views final reward points for respective TAC ID.

Functional Requirements:

1. Admin Functions:

- Login and authentication.
- Input and manage marks for Initial Submission, Final Report, Plagiarism, and Work Log.
- Calculate Amin total for each TAC ID.

2. Reviewer Functions:

- Login and authentication.
- Submit marks for ten review categories and six team communication categories.
- System automatically converts and calculates reviewer marks and team communication into reward points.

3. Student Functions:

- Login and authentication.
- View final reward points for their respective TAC ID.

Data Management:

- Marks for initial submission, final report, plagiarism, and work log are stored and managed in MySQL.
- Review marks and team communication scores are also stored and processed.

Security:

- Secure authentication mechanisms for all user types.
- Role-based access control to ensure users can only access relevant functionalities.

Workflow:

Admin Workflow:

1. Login: Authenticate and access the admin dashboard.

2. Select TAC ID: Choose a TAC ID to manage.

3. Enter Marks: Input marks for the selected TAC ID's Initial Submission, Final Submission, Plagiarism, and Work Log.

4. Calculate Admin Total : System calculates the total marks for the TAC ID.

Reviewer Workflow:

1. Login: Authenticate and access the reviewer dashboard.

2. Select TAC ID: Choose a TAC ID to review.

3. Submit Marks: Input marks for review components and team communication.

4. Marks Conversion: System automatically converts total review marks to 60 and team communication to 10.

5. Calculate Reviewer Total: System calculates the total reward points from the converted marks.

Student Workflow:

1. Login: Authenticate and access the student dashboard.

2. View Final Reward Points: Display calculated final reward points for each TAC ID.

API Endpoints:

- POST `/login`: Authenticate users.

- GET `/TAC IDs/{id}/marks`: Retrieve marks for a TAC ID.

- POST `/TAC IDs/{id}/marks`: Submit or update marks for a TAC ID.

- GET `/TAC IDs/{id}/Reward points`: Retrieve final reward points for a TAC ID.

Scope:

The scope of the TAC Reward Points Portal includes the development of a web-based application that automates the computation of final reward points for project TAC IDs. The system will cater to three user roles: administrators, reviewers, and students. Administrators can manage TAC ID details, allocate marks, and view final reward points. Reviewers can submit review marks, which are automatically converted into points. Students can view the final reward points for TAC IDs they are involved in.

Non-Functional Requirements:

Security:

- All user authentication and data transmission must be encrypted to ensure the confidentiality and integrity of user data.
- Role-based access control must be implemented to restrict unauthorized access to sensitive functionalities.

Performance:

- The system must be responsive and able to handle multiple concurrent user sessions without significant degradation in performance.
- Database queries and API responses should be optimized for fast retrieval and processing of data.

Usability:

- The user interface must be intuitive and user-friendly, with clear navigation and informative error messages.
- Forms for inputting marks and submitting reviews should provide validation to prevent erroneous data entry.

Reliability:

- The system should have robust error handling mechanisms to gracefully handle unexpected errors and prevent data loss.
- Regular backups of the database should be performed to ensure data integrity and availability.

Scalability:

- The architecture should be designed to scale horizontally to accommodate a growing number of users and TAC IDs.
- Load balancing and caching mechanisms should be implemented to distribute traffic evenly and improve response times.

Compatibility:

- The application should be compatible with modern web browsers such as Chrome, Firefox, and Safari, across multiple devices including desktops, tablets, and mobile phones.

Dependencies:

Spring Boot Stack (Java):

Dependency management and inversion of control provided by Spring Boot facilitate rapid development of backend functionalities.

React.js (Front End):

React.js enables the creation of interactive and dynamic user interfaces, improving user experience and engagement.

MySQL (Database):

Uses MySQL as the relational database management system for storing all user data, TAC ID marks, and system logs efficiently.

The choice of MySQL ensures compatibility with a wide range of hosting environments and offers robust support and extensive community resources.

RESTful API:

The RESTful API architecture allows for seamless communication between the frontend and backend components of the application.

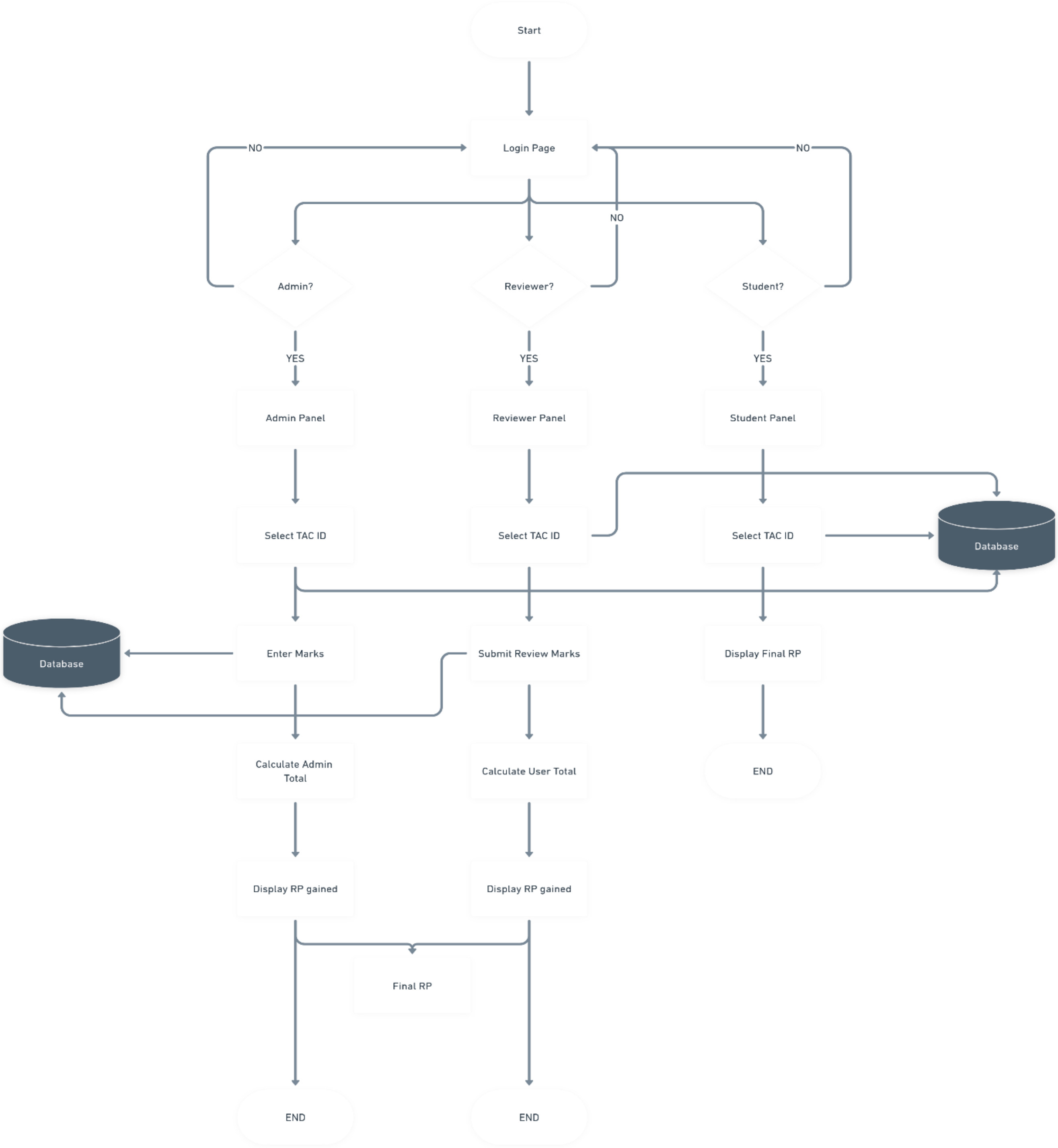
Security Libraries:

Libraries such as Spring Security for backend authentication and authorization, and client-side encryption libraries for frontend security, are essential dependencies to ensure system security.

Deployment Tools:

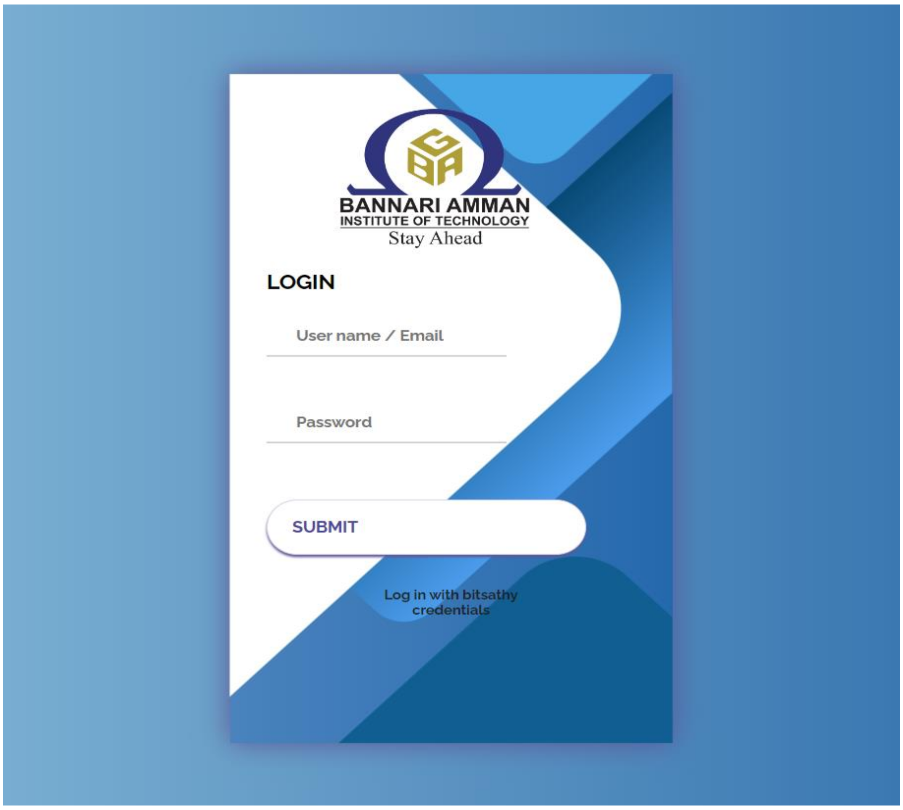
Tools like Docker and Kubernetes for containerization and orchestration, and CI/CD pipelines for automated testing and deployment, are necessary dependencies for efficient deployment and maintenance of the application.

Workflow:

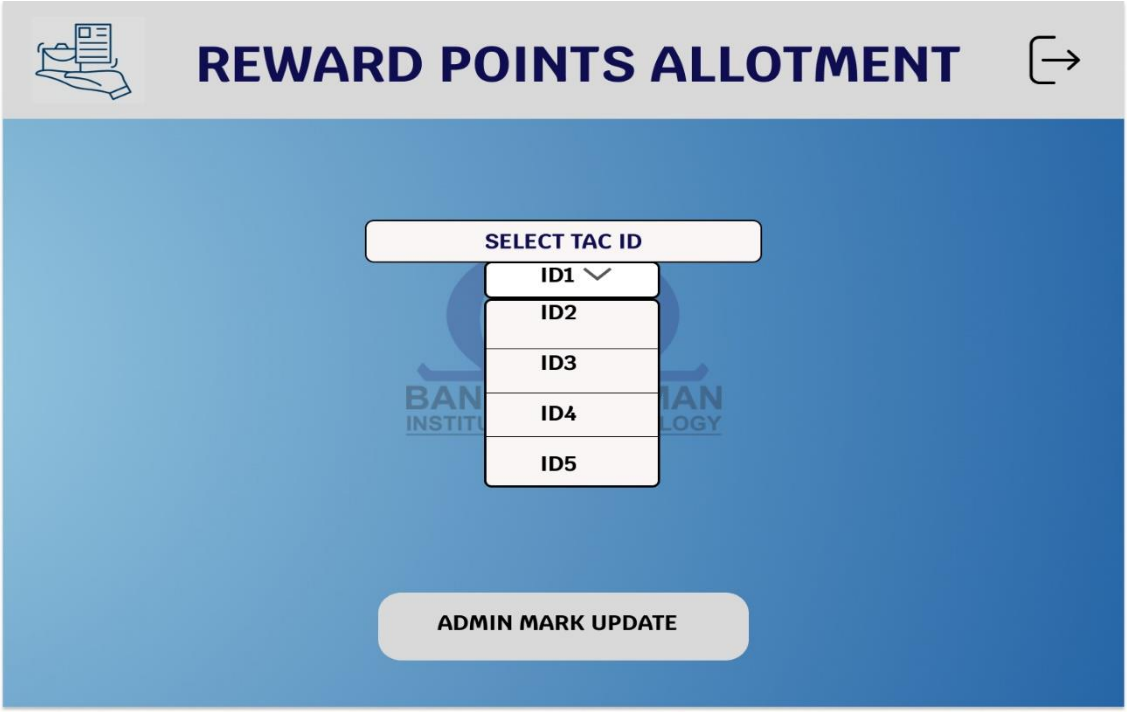

Start
Login Page
NO
NO
NO
Admin?
Reviewer?
Student?
YES
YES
YES
Admin Panel
Reviewer Panel
Student Panel
Select TAC ID
Select TAC ID
Select TAC ID
Database
Database
Enter Marks
Submit Review Marks
Display Final RP
Calculate Admin Total
Calculate User Total
END
Display RP gained
Display RP gained
Final RP
END
END

Prototype of the TAC ID :

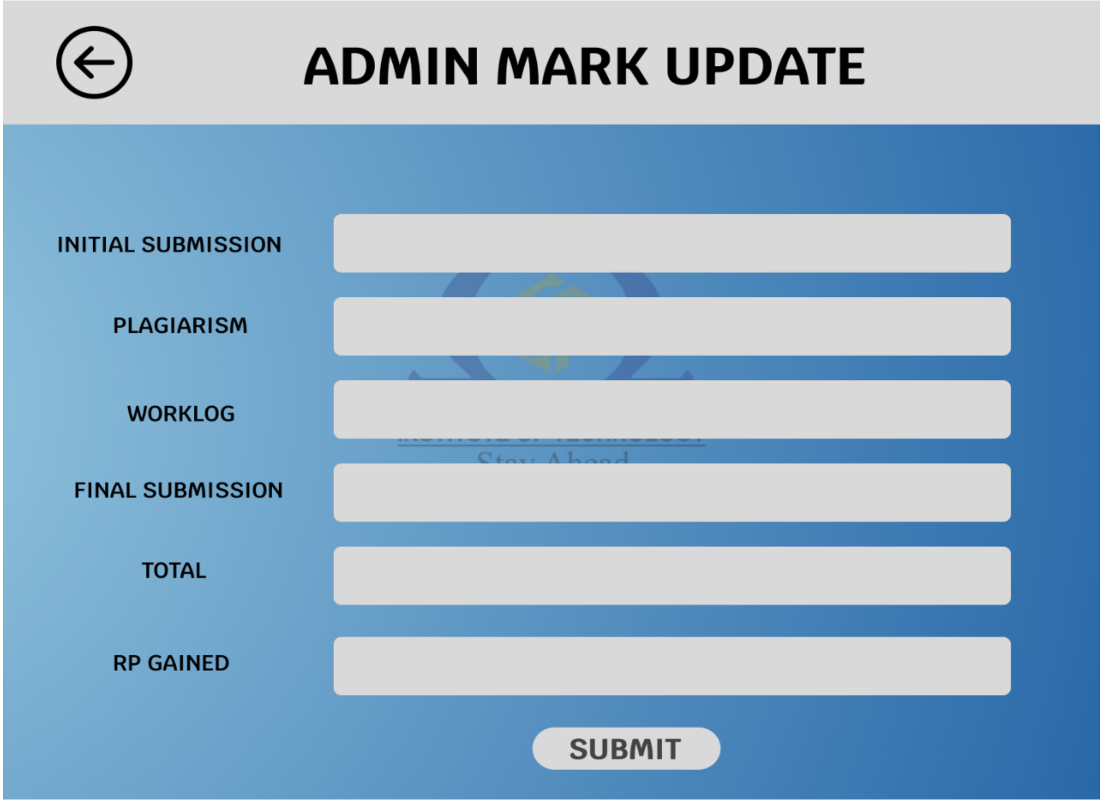
1.LOGIN FORM:



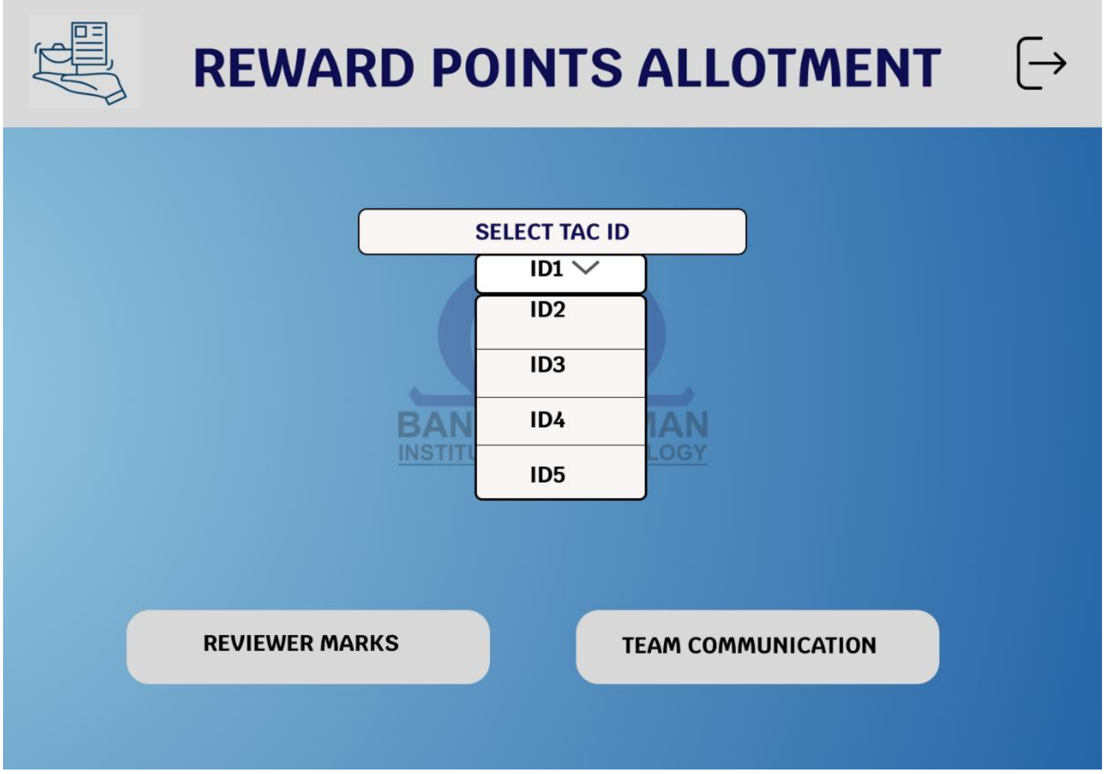
2.ADMIN PANEL:



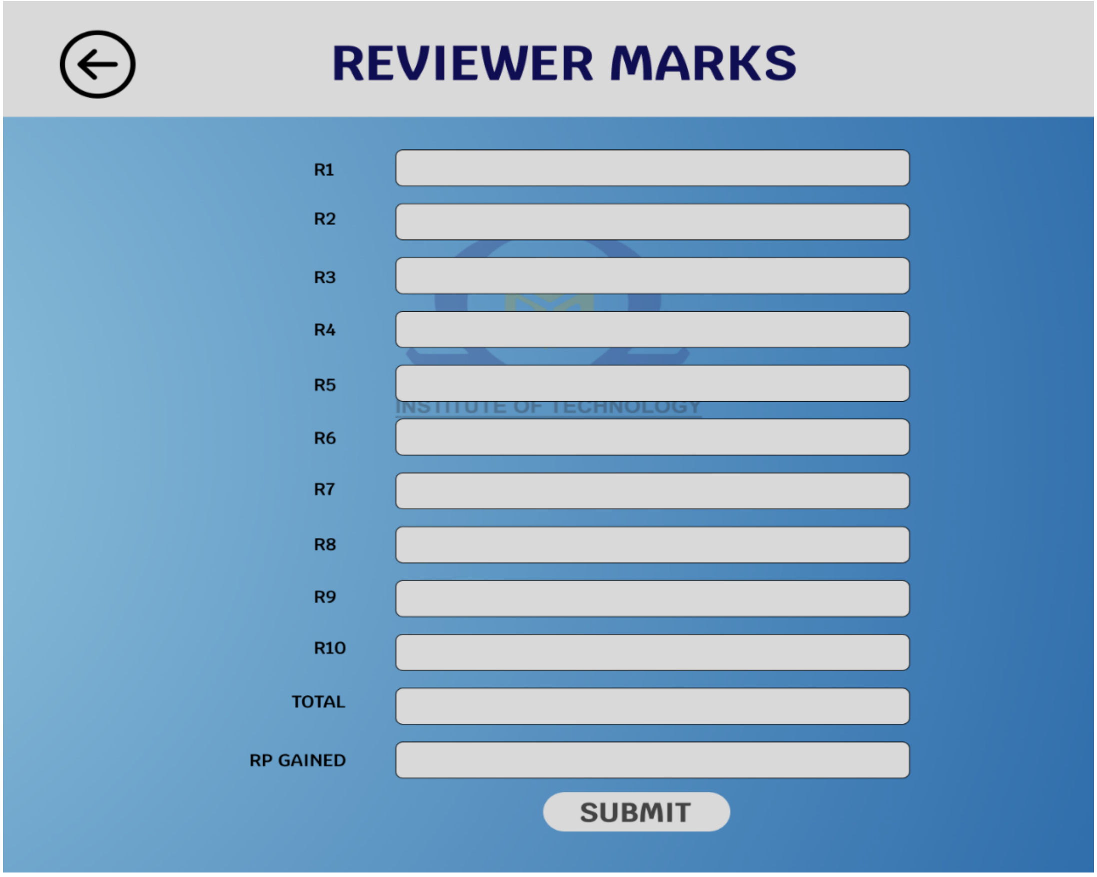
3.ADMIN MARK UPDATE:



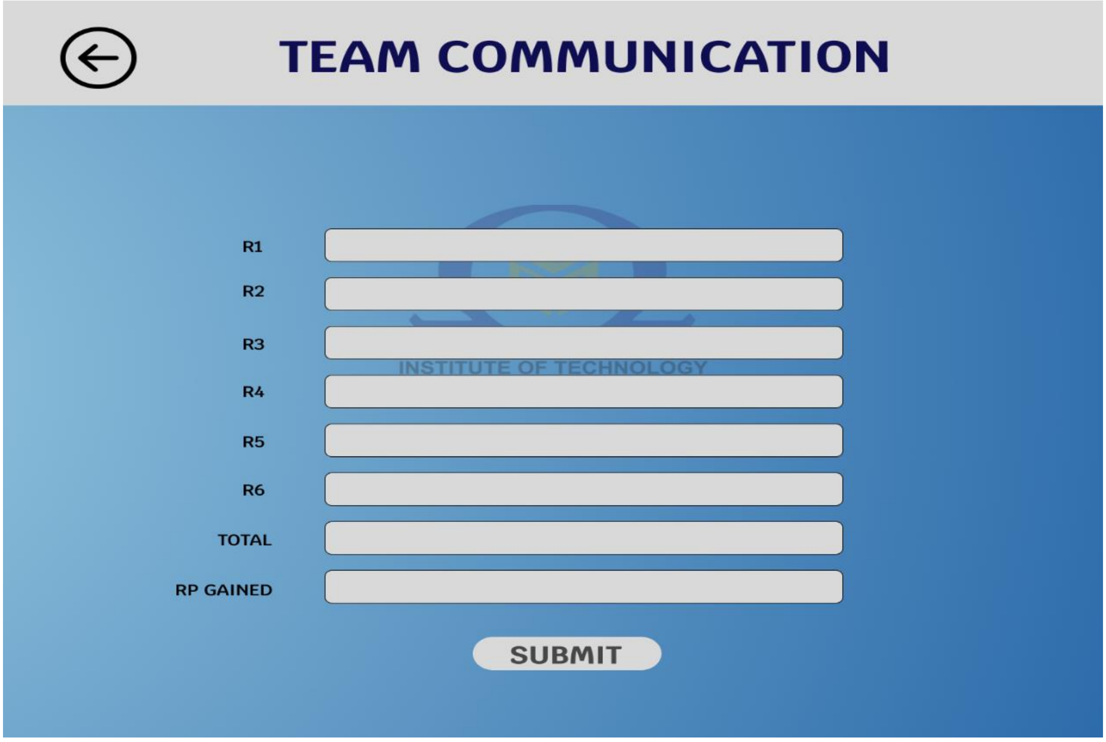
4.REVIEWER PANEL:



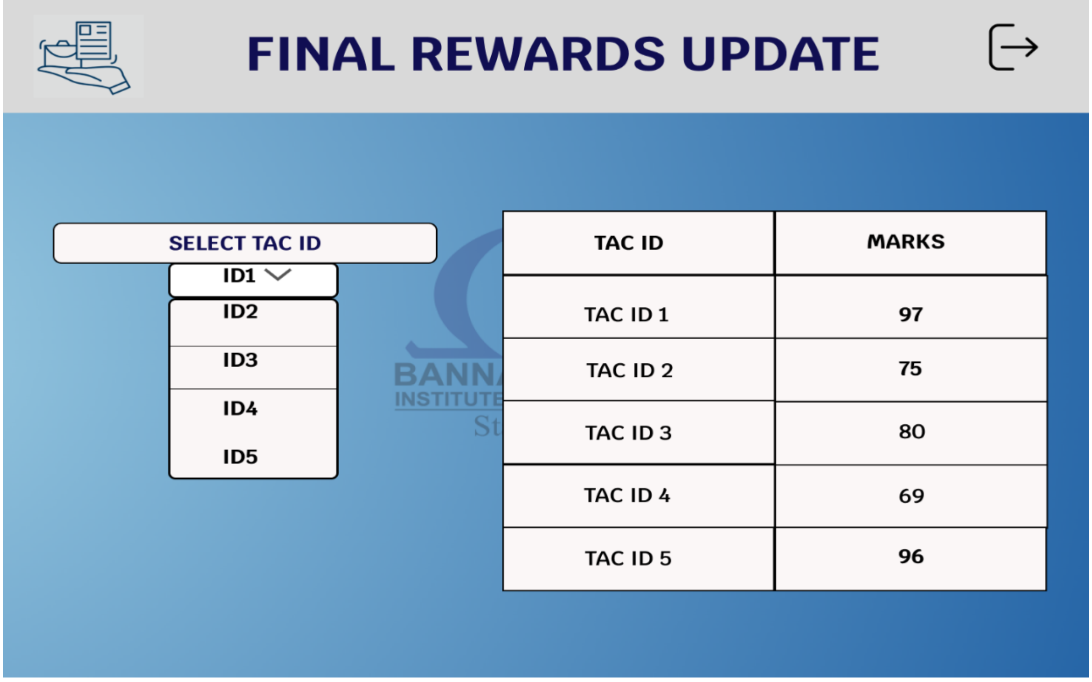
5.REVIEWER MARKS:



6.TEAM COMMUNICATION:



7.STUDENT PANEL:



TAC ID	MARKS
TAC ID 1	97
TAC ID 2	75
TAC ID 3	80
TAC ID 4	69
TAC ID 5	96

Conclusion:

This revised documentation addresses the key aspects of the TAC Reward Points Portal, taking into account the updated technology stack featuring MySQL. It outlines the requirements and dependencies essential for developing a secure, reliable, and efficient system to compute and display TAC TAC ID reward points effectively.