SYNOPSIS

PROJECT TITLE : LabTracker : Automating Lab Management using GitHub

DEPARTMENT OF COMPUTER SCIENCE

ALIGARH MUSLIM UNIVERSITY

SUBMITTED TO:

PROF. AASIM ZAFAR

PROF. SWALEHA ZUBAIR

MS. PRITI BALA

DR. ASIF IRSHAD KHAN

SUBMITTED BY:

MOHD SAUD

23CAMSA107

GK0827

MCA III

Project Title: LabTracker : *Automating Lab Management using* **GitHub**

1. Introduction

1.1 Background

Managing lab exercises and coding assignments in academic settings traditionally involves labor-intensive and time-consuming manual processes, such as tracking student progress with paper-based indexes and generating individual files for each lab submission. In programming courses, these tasks are particularly challenging due to the need for frequent updates, evaluations, and detailed documentation of student submissions. Moreover, manually checking each student's GitHub repository for submission verification is a tedious and error-prone task, especially with large class sizes.

By integrating platforms like GitHub into the lab management workflow, there is an opportunity to streamline these processes and improve overall efficiency and accuracy.

1.2 Objective

LabTracker aims to provide a comprehensive web-based platform that integrates seamlessly with GitHub to facilitate the tracking of students' weekly lab coding exercises. The platform will automate the submission verification process, generate essential documentation such as progress indexes, detailed lab files, weekly reports, class reports, and specific student reports. Additionally, it will equip educators with tools to monitor and manage student performance effectively. By doing so, LabTracker seeks to enhance the efficiency, accuracy, and overall quality of lab management in academic settings.

2. Problem Statement

2.1 Problem Description

Current lab tracking processes lack integration with modern coding platforms, leading to inefficiencies in submission tracking, verification, and feedback provision. Traditional methods are time-consuming and fail to leverage the benefits of automation offered by platforms like GitHub.

2.2 Importance of the Problem

Automating lab tracking and documentation generation is crucial for delivering timely feedback, maintaining academic standards, and enhancing the learning experience for students.

LabTracker addresses these issues by streamlining the entire process through integration with GitHub.

3. Study of the Existing System

3.1 Overview of the Existing System

Current lab management systems address various aspects of lab tracking but fall short in integrating seamlessly with version control systems like GitHub. These tools lack comprehensive integration for tracking and documenting progress, resulting in inefficient lab management processes. They are limited in automating lab-specific documentation and verifying student submissions, which affects their overall effectiveness.

3.2 Limitations of the Existing System

- Limited integration with GitHub for real-time tracking and updates.
- Manual processes for documentation generation that require significant effort.

3.3 Comparative Analysis

LabTracker distinguishes itself by addressing the gap left by existing systems through its seamless integration with GitHub. By leveraging the GitHub API, LabTracker enables real-time tracking of coding submissions and automates the generation of necessary lab documentation. Unlike existing tools, which rely on manual processes and lack version control integration, LabTracker's approach ensures accurate and efficient management of student submissions.

4. Proposed Solution

4.1 Overview

LabTracker is a web-based application that integrates with GitHub to automate the tracking and management of students' weekly lab exercises. It verifies GitHub usernames and repository structures during signup, tracks student progress through GitHub API interactions, and automates the generation of required documentation, including progress indexes and lab files.

4.2 Key Features

- GitHub Integration: Verifies repository structure and tracks weekly submissions to ensure compliance with course requirements.
- Progress Tracking: Monitors commit history, subdirectories for weekly exercises, and file submissions.

- Automated Documentation: Generates lab files with week numbers, problem descriptions, solutions, and output images based on data fetched from GitHub.
- Teacher Dashboard: Allows teachers to add and edit problems, generate reports, and monitor student progress.

4.3 Innovation

LabTracker innovates by automating verification and documentation processes using GitHub API, significantly reducing manual work and potential errors. While the system does not include code verification, it mitigates this by generating files based on GitHub API responses.

5. Scope of the Project

5.1 Inclusions

- Integration with GitHub: The system will integrate with GitHub to facilitate the tracking of student coding submissions, enabling real-time updates, version control, and progress monitoring directly from GitHub repositories.
- User Authentication and Verification: Implementation of secure user authentication to verify the identities of both students and educators. The system will validate GitHub usernames, repository structures, and access permissions to ensure proper alignment with course requirements and submission guidelines.
- Automated Documentation Generation: The platform will automatically generate lab-specific documentation, indexes, and summaries using data pulled from GitHub repositories. This will reduce the administrative workload for educators by automating the creation of standardized reports and submission logs.

5.2 Exclusions

- Comprehensive Code Verification: The system will not perform a detailed analysis or verification of the submitted code's correctness or quality. It will only verify the presence of required files as an indicator of submission completion.
- Advanced Analytical Features: The project will not include features like AI-driven code analysis, automated feedback on code quality, or plagiarism detection, as these are beyond the scope of the initial development phase.
- Offline Functionality: The platform will require internet connectivity for accessing GitHub and other online resources; offline capabilities are not within the project's scope.

- Third-Party Integrations Beyond GitHub: The system will not support integration with other version control platforms (e.g., GitLab, Bitbucket) or third-party learning management systems (LMS) in this phase.

6. Preliminary System Design

6.1 High Level Architecture

The system follows a client-server architecture with Django as the backend, interacting with GitHub APIs, and a frontend developed with HTML, CSS, and JavaScript for user interactions.

6.2 Major Components

Student Module : The Student Module enables students to sign up by providing their enrollment details and GitHub username, which the system verifies along with the repository's existence based on course and semester. The Progress Dashboard then displays weekly problem details, tracking subdirectories (Week1 to Week14) and updating progress based on file submissions. Additionally, the module generates and offers downloadable indexes and code files in formats such as .docx, .pdf, and spreadsheets using libraries like jsPDF and exceljs.

Teacher Module : The Teacher Module allows teachers to add and edit problems, generate various reports (weekly, course-wide, and individual student reports), and view recent activity. It also provides functionality to update student progress and assignment data using the GitHub API, and offers recent activity logs to track changes and interactions.

6.3 Data Flow

User inputs are processed by the frontend and sent to the backend, which interacts with the GitHub API to fetch relevant data. The backend processes the data, stores it in the database, and updates the frontend with real-time information on student progress and generated documentation.

7. Feasibility Analysis

7.1 Technical Feasibility

The project is technically feasible due to its reliance on well-established and widely used technologies. Django, a high-level Python web framework, provides a robust backend infrastructure that is scalable, secure, and efficient. The integration with GitHub's API facilitates seamless communication between the platform and the version control system, enabling real-time tracking and updates. The use of other supporting technologies, such as PostgreSQL for the database ensures that the system is performant. Additionally, the extensive documentation and large communities around these technologies provide ample resources for development and troubleshooting.

7.2 Economic Feasibility

The project is economically feasible as it leverages open-source tools and technologies, significantly reducing the costs associated with software licensing and development. By using free and open-source frameworks like Django and public APIs like GitHub's, the project minimizes upfront and ongoing expenses, making it particularly suitable for educational institutions with constrained budgets. The low-cost infrastructure, which can be hosted on affordable cloud platforms or existing institutional servers, further enhances the economic viability of the project. The potential for long-term savings in administrative and manual processing costs also contributes to its economic attractiveness.

7.3 Operational Feasibility

LabTracker's user-friendly interface, combined with its integration with widely-used platforms like GitHub, ensures that it is easy to adopt and operate. The system is designed with a minimal learning curve, requiring limited training for educators and students who are already familiar with GitHub. Moreover, the platform offers streamlined workflows and automation for tasks such as documentation generation, which reduces the administrative burden on educators and students. The system's adaptability to various course structures and assignment types makes it suitable for a wide range of programming curricula.

8. Tools and Technologies

8.1 Programming Languages

- Python: Used for backend development, handling server-side logic and integration with external services.
- JavaScript: Employed for frontend development to create interactive and dynamic user interfaces.

8.2 Development Tools

- Django: A Python web framework used for backend development, managing database interactions, and user authentication.
- PyCharm Professional: IDE for Python development, offering advanced debugging and code management.

8.3 Databases

- SQLite: Used for initial development and testing due to its simplicity and ease of setup.
- PostgreSQL: Chosen for production to handle larger datasets and ensure scalability.

8.4 API's

- GitHub API for retrieving repository structures, commit histories, and file data for tracking and documentation purposes.

8.5 Libraries

- jsPDF: JavaScript library for generating PDF documents on the client side.
- exceljs: JavaScript library for creating and manipulating Excel spreadsheets.
- python-docx: Python library for generating and updating Microsoft Word documents.

9. Expected Outcomes

9.1 Deliverables

- A web-based platform with GitHub integration to streamline lab tracking and documentation.
- Automated generation of progress indexes and lab files based on GitHub data.
- User guides and system documentation to support educators and students.

9.2 Success Criteria

- Effective automation of lab tracking and documentation processes.
- Improved efficiency and accuracy in managing lab exercises.
- High satisfaction rates from users due to the system's ease of use and time-saving features.

10. Risks and Challenges

10.1 Potential Risks

- Dependence on GitHub API, which may change and affect data retrieval processes.
- Scalability challenges if user load increases significantly.
- Potential inaccuracies in progress tracking due to the absence of code verification.

10.2 Mitigation Strategies

- Regular updates and testing of GitHub API integration to handle changes effectively.
- Planning for scalable architecture to manage increased usage.

- Clear communication to users about the limitation of the verification process, where file presence is used as a completion indicator without verifying the content, thus simplifying the evaluation of submissions.

11. References

- GitHub API Documentation (https://docs.github.com/en/rest?apiVersion=2022-11-28)
- Django Official Documentation. (https://docs.djangoproject.com/)
- Studies on coding exercise tracking and management systems

12. Appendices

Figure 1 **: Use Case Diagram**

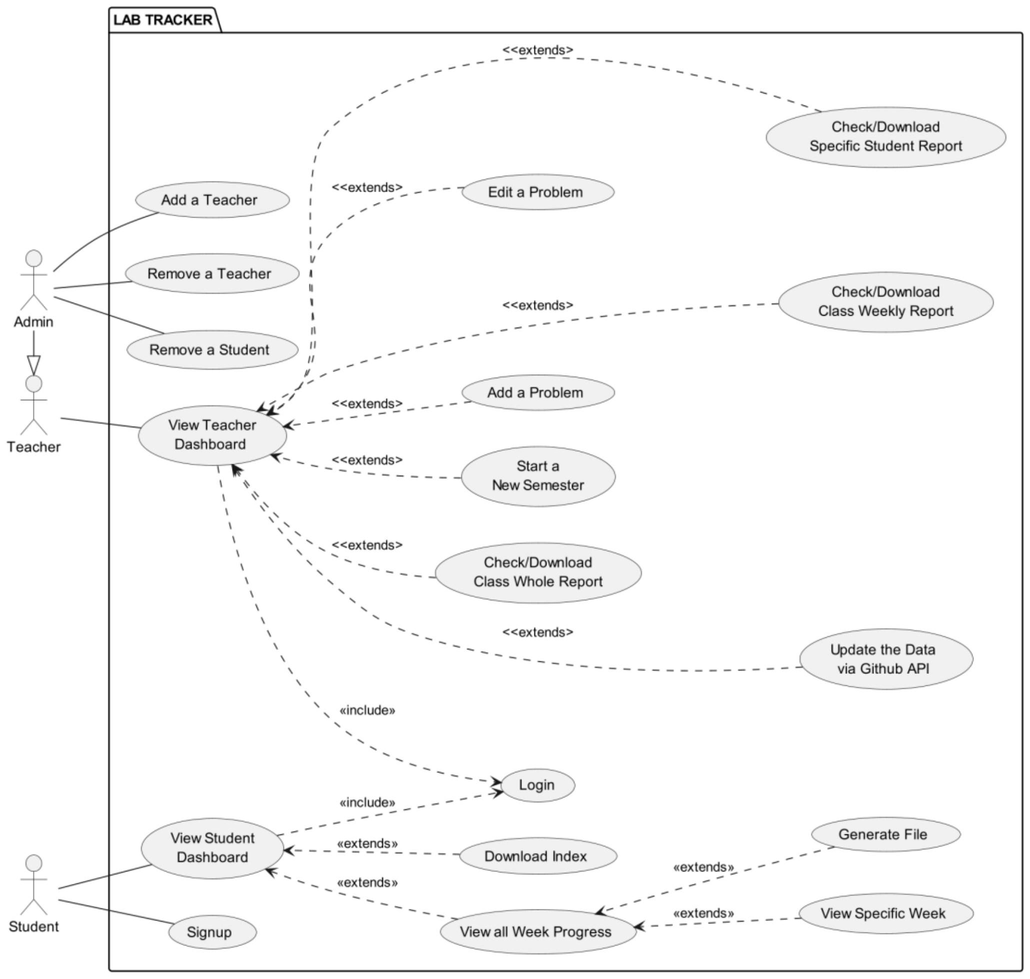


Figure 2 : **Activity Diagram (Students)**

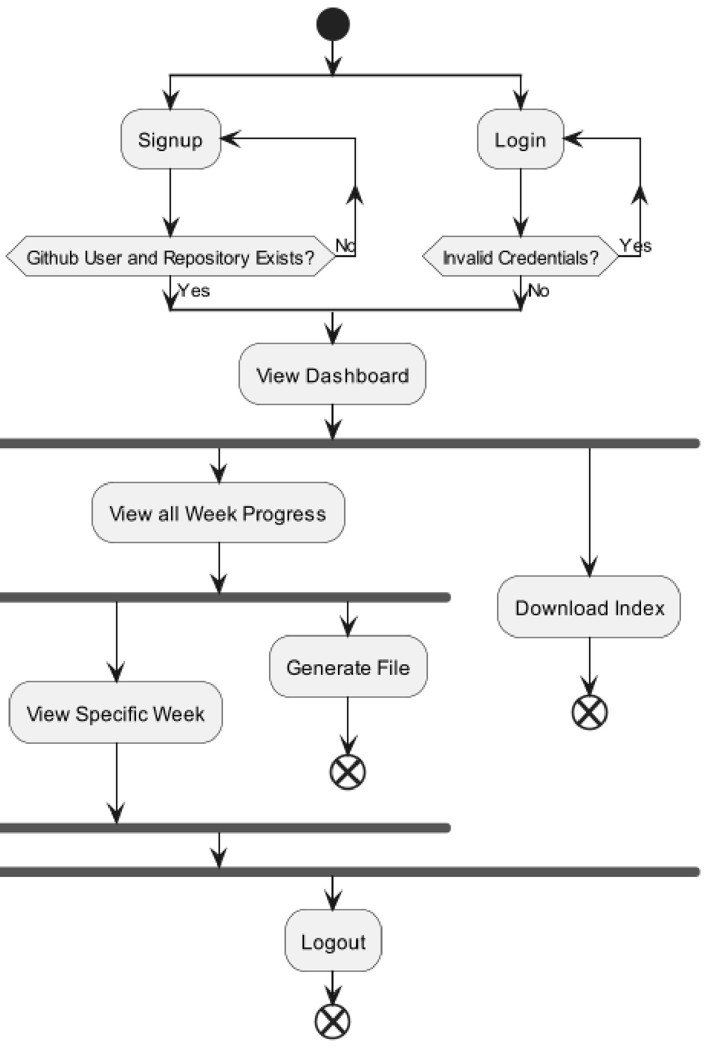


Figure 3 **: Activity Diagram(Teacher)**

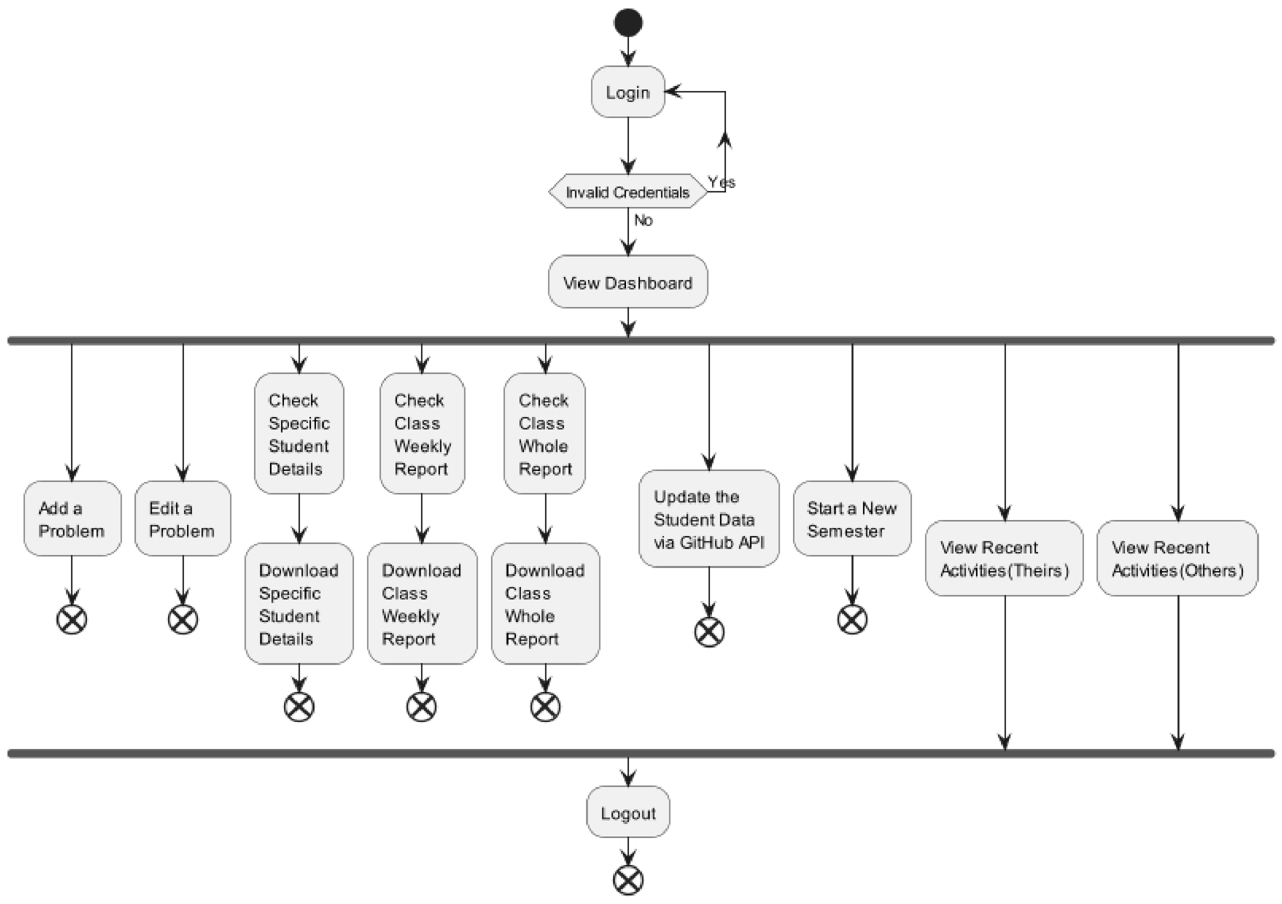
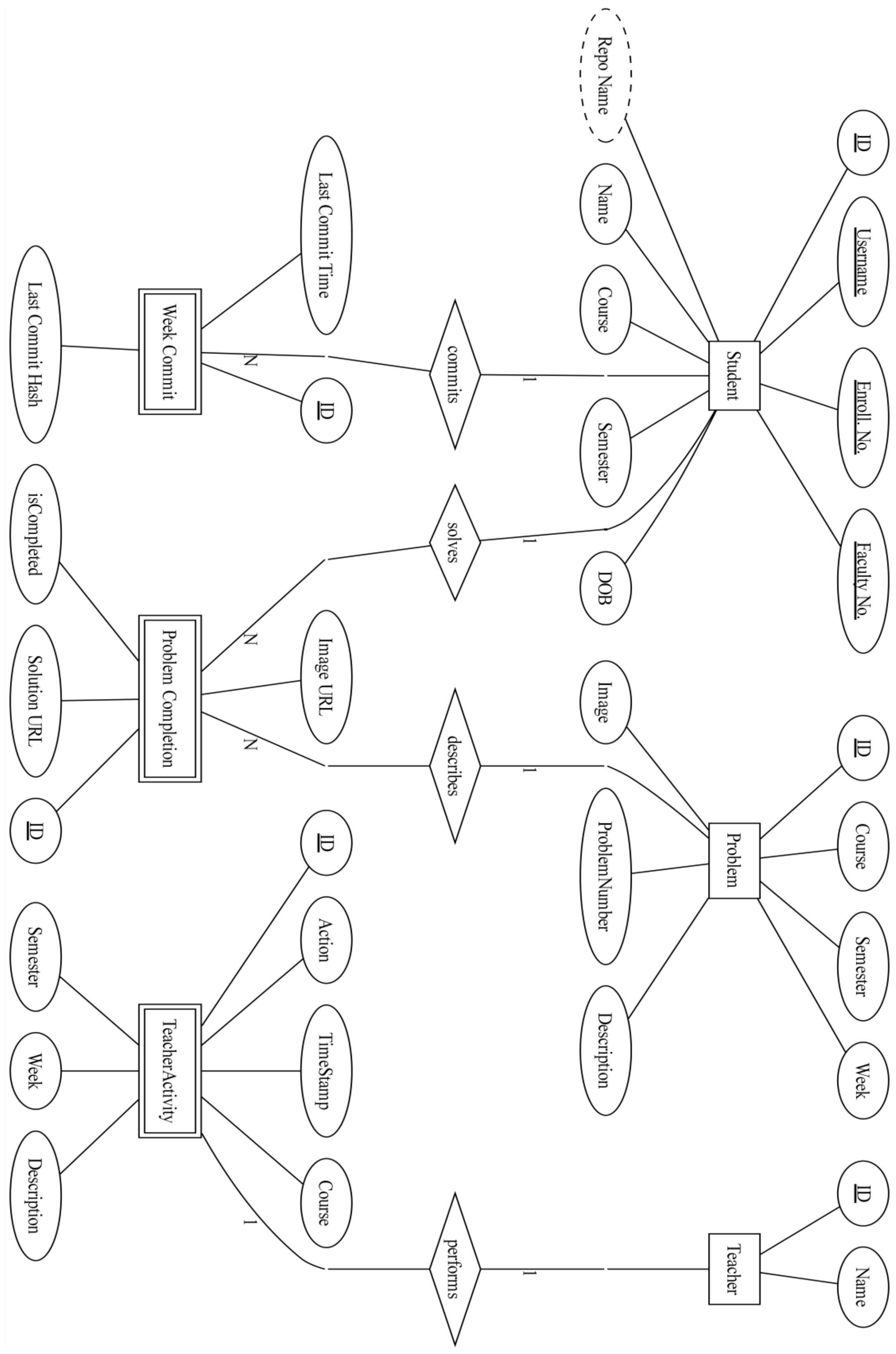

Login
Invalid Credentials
Yes
No
View Dashboard
Add a Problem
Edit a Problem
Check Specific Student Details
Download Specific Student Details
Check Class Weekly Report
Download Class Weekly Report
Check Class Whole Report
Download Class Whole Report
Update the Student Data via GitHub API
Start a New Semester
View Recent Activities(Theirs)
View Recent Activities(Others)
Logout

Figure 4 **: ER Diagram**


Repo Name
ID
Username
Enroll. No.
Faculty No.
Name
Course
Semester
DOB
Student
commits
1
N
Week Commit
Last Commit Time
Last Commit Hash
ID
solves
1
N
Problem Completion
isCompleted
Solution URL
Image URL
ID
describes
1
N
Problem
Image
ID
Course
Semester
Week
ProblemNumber
Description
TeacherActivity
ID
Action
TimeStamp
Course
Semester
Week
Description
performs
1
1
Teacher
ID
Name