Architecture Diagram

1. Overview

The architecture of the Project Diary system has been designed to support both individual and team project tracking needs, with functionality for logging activities, tracking time, analysing data, and facilitating team collaboration. The architecture is structured to optimise project activity management, enhance productivity, and foster transparency through an intuitive, collaborative platform.

2. System Components

The Project Diary system is divided into several core components, each with a unique responsibility and role in facilitating user needs. These components interact with each other to achieve seamless functionality across the application.

2.1. User Interface (UI) Layer

- Responsibilities: This layer is responsible for displaying the graphical interface that allows users to interact with the application. It enables functionalities such as project and activity creation, time tracking, and visualisation of analytics.
- Interaction: Communicates with the Business Logic Layer to pass user requests and retrieve processed information.
- Technologies Used: JavaFX is used for the UI due to its ease of use in building visually rich desktop applications with customizable components and event-handling capabilities.

2.2. Business Logic Layer

- Responsibilities: Acts as the core processing unit, handling project and activity management, data validation, role-based access control, and time tracking functions. It also manages the application's main processes, such as creating projects, tracking progress, and logging user reflections on project challenges and solutions.
- Interaction: Communicates with the Data Access Layer to retrieve and update data as well as with the UI Layer to process user commands and return results for display.
- Technologies Used: Java is chosen for the business logic due to its robustness, strong object-oriented structure, and efficiency in managing various processes and calculations.

2.3. Data Access Layer

- Responsibilities: Provides an interface for data storage and retrieval, managing the interaction between the application and the database. This layer ensures data consistency, manages queries, and secures user data.
- Interaction: It interacts with the Business Logic Layer to store or retrieve information (e.g., user activities, time logs, project details).
- Technologies Used: MySQL database stores all user-related information, project data, time logs, and activity categories. Although MySQL is not available in the lab, it is selected for its scalability, support for complex queries, and wide adoption in Java-based applications.

2.4. Authentication & Role Management

- Responsibilities: Manages user authentication, including login and logout functionality, session management, and role-based access control (e.g., Administrator, Project Manager, Team Member). This component ensures that only authorised users access specific features and data based on their role.
- Interaction: Integrates with the Business Logic Layer to enforce access permissions and with the UI Layer to display user-specific data based on role.
- Technologies Used: Java-based authentication mechanisms supported by a secure session management approach and Java's libraries for password encryption.

2.5. Reporting & Analytics Component

- Responsibilities: Provides data analysis and visualisation features to users, allowing them to generate insights on time spent across projects and activities. This component supports custom report generation, exports, and graphical representations (e.g., charts showing time distribution).
- Interaction: Integrates with the Business Logic Layer to retrieve data on user activities and display analytics results in the UI Layer.

- Technologies Used: JavaFX for front-end data visualisation tools and Java-based libraries for generating data reports.

2.6. Export Module

- Responsibilities: Facilitates data export functionality, allowing users to export activity logs, project information, and analysis results in various formats (e.g., CSV, PDF, Excel).
- Interaction: Operates through the Business Logic Layer to retrieve relevant data for export, integrating with the UI Layer to provide export options to the user.
- Technologies Used: Java-based libraries for file generation in multiple formats.

3. Technology Platform

The Project Diary application is implemented using the following technology stack to ensure a balance of usability, performance, and scalability:

3.1. Programming Language: Java

Chosen for its reliability, multi-threading support, and rich libraries, Java is well-suited for complex logic management and integration with UI and database components.

3.2. UI Development: JavaFX

JavaFX is selected for its flexibility in building interactive and visually appealing desktop applications. It allows easy customization and integration of visualisation components, essential for the Project Diary's analytics features.

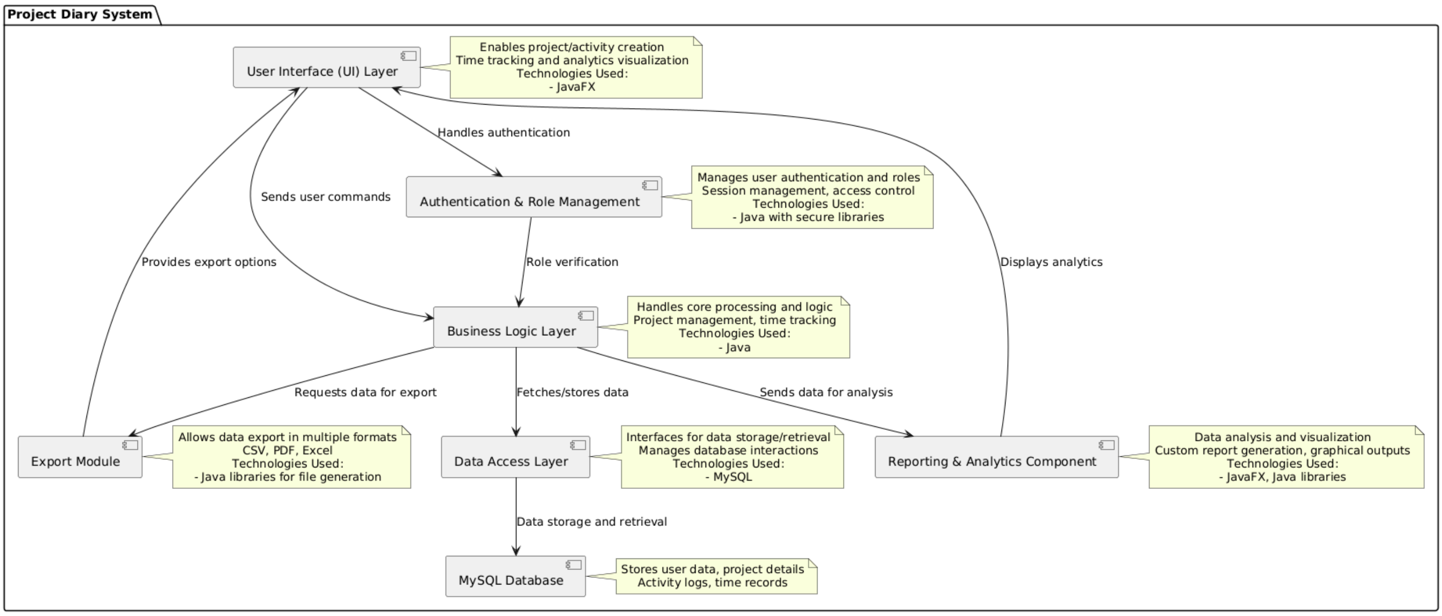
3.3. Database Management: MySQL

MySQL is a popular relational database chosen for its robustness, efficient handling of large data sets, and compatibility with Java-based applications. Despite lab limitations, MySQL will support structured data storage, queries, and data consistency.

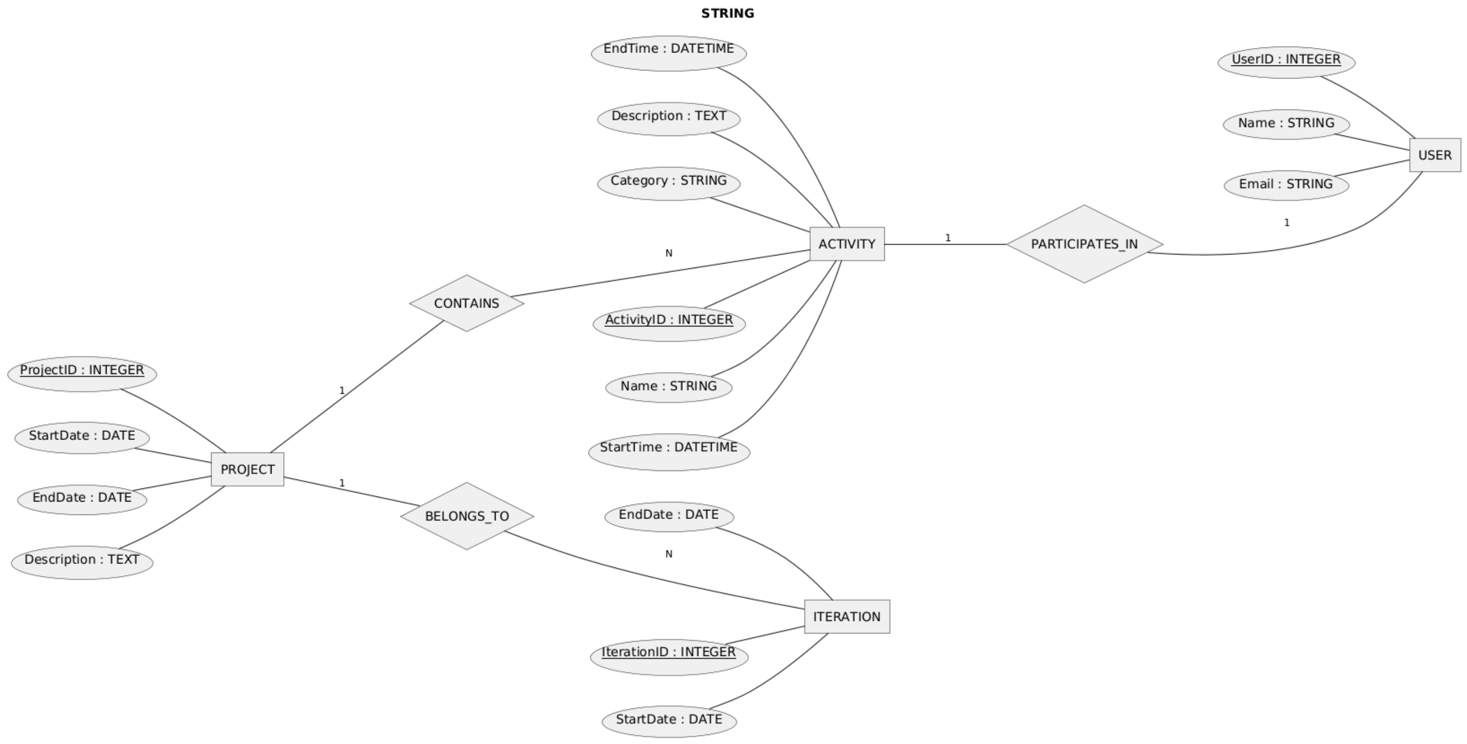
3.4. **Authentication & Security:** Java's built-in libraries for user authentication and encryption methods secure user data, support role-based access, and maintain secure user sessions throughout application usage.

3.5. **Export Functionality:** Java's file handling and export libraries enable flexible data export in various formats, meeting users' requirements for generating reports and sharing data.

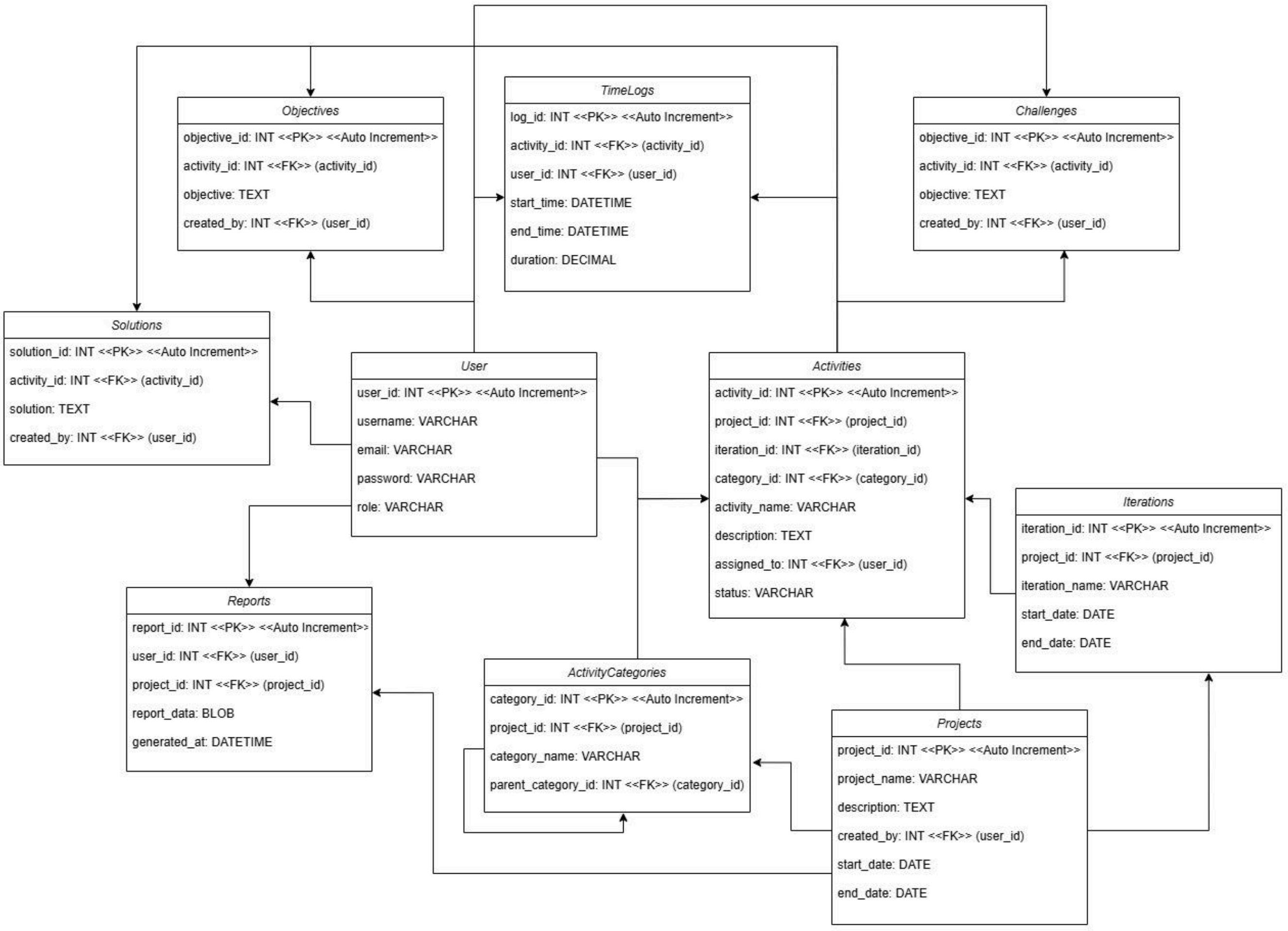
This technology stack aligns with the Project Diary's objectives by providing a scalable, user-friendly, and robust platform for tracking, analysing, and managing project activities effectively.


Project Diary System
User Interface (UI) Layer
Enables project/activity creation
Time tracking and analytics visualization
Technologies Used:
- JavaFX
Handles authentication
Sends user commands
Authentication & Role Management
Manages user authentication and roles
Session management, access control
Technologies Used:
- Java with secure libraries
Provides export options
Role verification
Displays analytics
Business Logic Layer
Handles core processing and logic
Project management, time tracking
Technologies Used:
- Java
Requests data for export
Fetches/stores data
Sends data for analysis
Export Module
Allows data export in multiple formats
CSV, PDF, Excel
Technologies Used:
- Java libraries for file generation
Data Access Layer
Interfaces for data storage/retrieval
Manages database interactions
Technologies Used:
- MySQL
Reporting & Analytics Component
Data analysis and visualization
Custom report generation, graphical outputs
Technologies Used:
- JavaFX, Java libraries
Data storage and retrieval
MySQL Database
Stores user data, project details
Activity logs, time records

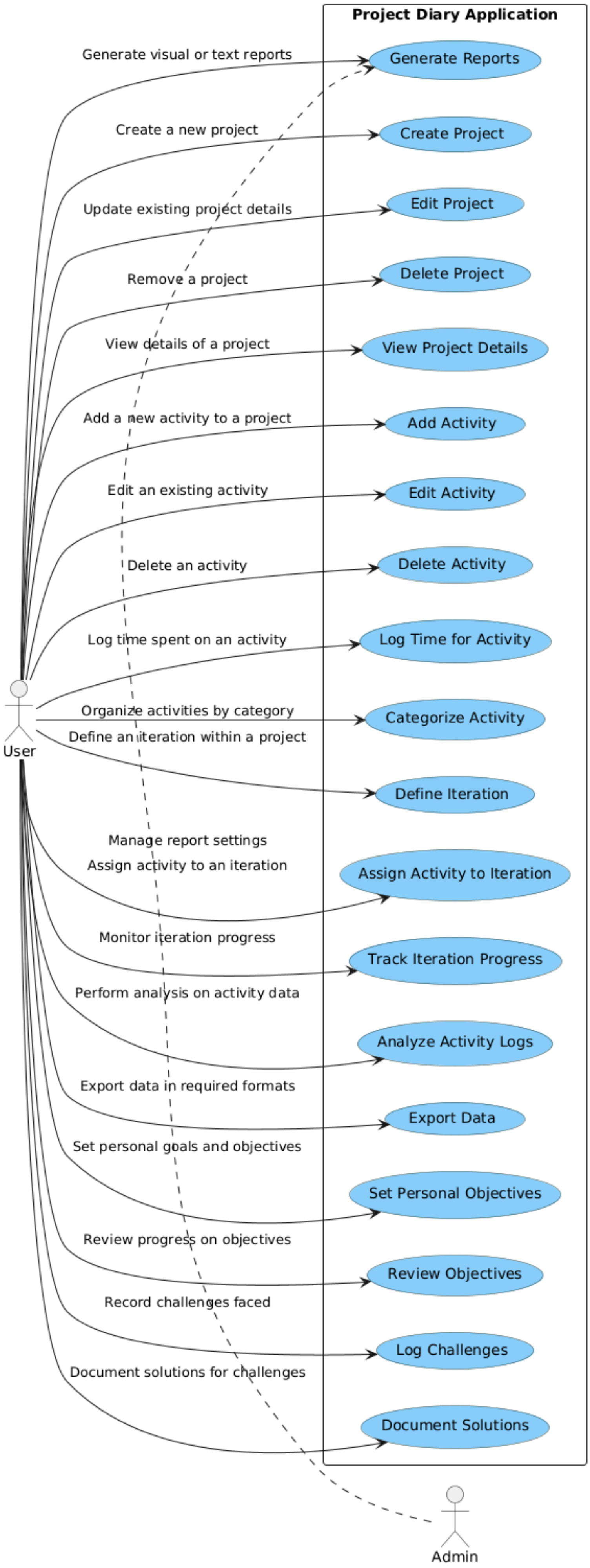
ER Diagram



UML Class Diagram


Objectives
objective_id: INT <<PK>> <<Auto Increment>>
activity_id: INT <<FK>> (activity_id)
objective: TEXT
created_by: INT <<FK>> (user_id)
TimeLogs
log_id: INT <<PK>> <<Auto Increment>>
activity_id: INT <<FK>> (activity_id)
user_id: INT <<FK>> (user_id)
start_time: DATETIME
end_time: DATETIME
duration: DECIMAL
Challenges
objective_id: INT <<PK>> <<Auto Increment>>
activity_id: INT <<FK>> (activity_id)
objective: TEXT
created_by: INT <<FK>> (user_id)
Solutions
solution_id: INT <<PK>> <<Auto Increment>>
activity_id: INT <<FK>> (activity_id)
solution: TEXT
created_by: INT <<FK>> (user_id)
User
user_id: INT <<PK>> <<Auto Increment>>
username: VARCHAR
email: VARCHAR
password: VARCHAR
role: VARCHAR
Activities
activity_id: INT <<PK>> <<Auto Increment>>
project_id: INT <<FK>> (project_id)
iteration_id: INT <<FK>> (iteration_id)
category_id: INT <<FK>> (category_id)
activity_name: VARCHAR
description: TEXT
assigned_to: INT <<FK>> (user_id)
status: VARCHAR
Iterations
iteration_id: INT <<PK>> <<Auto Increment>>
project_id: INT <<FK>> (project_id)
iteration_name: VARCHAR
start_date: DATE
end_date: DATE
Reports
report_id: INT <<PK>> <<Auto Increment>>
user_id: INT <<FK>> (user_id)
project_id: INT <<FK>> (project_id)
report_data: BLOB
generated_at: DATETIME
ActivityCategories
category_id: INT <<PK>> <<Auto Increment>>
project_id: INT <<FK>> (project_id)
category_name: VARCHAR
parent_category_id: INT <<FK>> (category_id)
Projects
project_id: INT <<PK>> <<Auto Increment>>
project_name: VARCHAR
description: TEXT
created_by: INT <<FK>> (user_id)
start_date: DATE
end_date: DATE

UML Class Diagram


Project Diary Application
Generate visual or text reports
Generate Reports
Create a new project
Create Project
Update existing project details
Edit Project
Remove a project
Delete Project
View details of a project
View Project Details
Add a new activity to a project
Add Activity
Edit an existing activity
Edit Activity
Delete an activity
Delete Activity
Log time spent on an activity
Log Time for Activity
Organize activities by category
Categorize Activity
Define an iteration within a project
Define Iteration
User
Manage report settings
Assign activity to an iteration
Assign Activity to Iteration
Monitor iteration progress
Track Iteration Progress
Perform analysis on activity data
Analyze Activity Logs
Export data in required formats
Export Data
Set personal goals and objectives
Set Personal Objectives
Review progress on objectives
Review Objectives
Record challenges faced
Log Challenges
Document solutions for challenges
Document Solutions
Admin