A red text on a white background

Description automatically generated

|  |  |
| --- | --- |
| Project Title | **Timesheet Dashboard** |
| Project Estimated Start Date | **30/9/2024** |
| Project Estimated End Date | **18/10/2024** |
| Candidate Name | **Dhananjay Singh** |
| Mentor Name | **Subodh Tiwari** |

**GEBS GET Training Project Document**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Doc Version | Submitted Date | Reviewed By and Date | Comment/Remarks |
|  |  |  |  |

Reviewer(s)

|  |  |
| --- | --- |
| Name | Title |
|  |  |

Approver(s)

|  |  |
| --- | --- |
| Name | Title |
|  |  |

Document Reference(s)

|  |
| --- |
| Releated Documents : |
| 1. **https://mui.com/material-ui/** |

1.Introduction

1.1 Scope

Scope of the project is to develop a web application that provides users a timesheet, raising leave and bug tickets functionality, where users will also receive alerts. Dashboard for managers and team leads that provides them a quick view to important business parameters and ability to manage team and projects. This project will also contain User Authentication, Real-Time Data Updates, Audit Logs features.

1.2 Technology Stack

**Software Requirements**

* ReactJs for frontend
* Fastapi for backend server
* SQL Server as database

1.3 Glossary or Terminology

// New terms you come across as you research your design or terms you may suspect your readers/stakeholders not to know

* AJAX Requests: These requests allow web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes, without reloading the entire page.
* Authorization: Authorization is the process of determining whether a user has permission to access a resource or perform an action.
* JWT-token: Is a compact, URL-safe token used for securely transmitting information between parties as a JSON object, often used for authentication and authorization.
* Object-Relational Mapper (ORM): This component allows developers to interact with databases using Python objects and methods, rather than writing raw SQL queries.

1.4 High Level Design

A diagram of a database

Description automatically generated

**App Process flow**

A diagram of a diagram

Description automatically generated

**System Architecture**

Figma Page URL: <https://www.figma.com/design/QFUn0kPtTzfnRhGCVH4TfJ/Timesheet-Wireframe?node-id=0-1&m=dev&t=3fllME0L9WOtnBRf-1>

1.5 Programming Standards

// Programming standard used with few Code snippet

* Modularity: Entire codebase is kept modular for separating different logical components and keeping code easier to maintain and understand.A screen shot of a computer program

  Description automatically generated
* Exception Handling: Exception handling is done to avoid system crash due to runtime errors.A screen shot of a computer screen

  Description automatically generated
* Client-Server Architecture: Frontend and backend is run separately for improved performance and scalability.

1.6 Components

//Components Description

1. Frontend: Frontend of this application is made with ReactJS and most of the component of the application uses MUI components, chart.js and plotly are used to create different charts for the dashboard of the application, axios is used to make AJAX call to backend to fetch essential information.
2. Backend: Backend of this application is made with FASTAPI which contains sqlalchemy models for all the table of database, multiple endpoints to send and aggregate data required in frontend, sqlalchemy is also used to connect to database and execute sql queries, Pydantic is used to validate requests objects from frontend and jwt authentication in also performed to send token to the logged in user.
3. Database: SQL SERVER is the database of my application which is a structured database that contains all the data required in my application.

1.7 Pre-requisite

// Any pre-requisite which user/stakeholder needs to know before going through the project

1. Should have login credentials like Employee ID, Password to use the app.
2. Past experience of using any management and dashboard app (But not mandatory as app is user friendly) .

1.8 User Guide

//User manual to go through the project

Every user will require to login with their employee id and password and will have access to the app only after proper authentication.

**Types of users inside app:**

1. Manager
2. Employee (Not Manager)

**Interface:**

1. Manager
   1. Dashboard: Manager will have a dashboard which shows useful information of its team and will have option to adjust the dashboard according to dates.
   2. Timesheet page: Manager will be able to fill and send his own timesheet along with feature to approve timesheets of his team.
   3. Leave: Manager will be able to apply for leave and also approve leave request of his team.
   4. Ticket page: Any user will have option of creating and closing tickets to report any issue with his colleagues.
   5. Project Page: Manager could add new project for the team with the option of selecting employees for each project.
   6. Employee page: Manager will have general information of hist team on his page and option of getting data with date range.
2. Employee (Not Manager):
   1. Timesheet page: Any employee will have option to fill their own timesheet and send this timesheet to manager for approval.
   2. Leave page: Any employee will have option to raise leave request to their manager.
   3. Ticket page: Any user will have option of creating and closing tickets to report any issue with his colleagues.

2.Testing

2.1 Test Scenario(s)

//Use Cases with Actual and Expected Results

Note : Multiple test scenario is required