A Project Report on

WAL PULSE

Submitted by

U.Vishnuvardhan(R170516)

Submitted to

IIIT RK Valley,

Idupulapaya, Vempalli, YSR Kadapa,

Andhra Pradesh, India PIN 516330.

Under the supervision of

Ms M.Himabindu

Assistant Professor

RGUKT, RK Valley

as a part of partial fulfilment of the degree of Bachelor of Technology in Computer Science and Engineering.

April 2023

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING, RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES, RK VALLEY

CERTIFICATE

This is to certify that the report entitled "WAL PULSE" submitted by U.Vishnuvardhan(R170516) in partial fulfillment of the requirement for the award of Bachelor of Technology in Computer Science Engineering is a bonafide work carried out by her under supervision and guidance.

The report hasn't been submitted previously in part or in full to this or any other university or institution for the award of any degree. Under the Guidance of **M.Himabindu** (Assistant Professor, Computer Science &Engineering, RGUKT, R.K Valley).

M.Himabindu,
Project Internal Guide,
Computer Science and Engineering,
R.K Valley, RGUKT.

N.Satyanandaram,
Head of the Department,
Computer Science and Engineer
R.K Valley, RGUKT

DECLARATION

I U.Vishnuvardhan (R170516) here by declare that this report entitled “WAL PULSE” submitted by me under the guidance and supervision of Ms M.Himabindu, is a bonafide work. I also declare that it has not been submitted previously in part or in full to this university or other university or institution for the award of any degree or diploma.

Date :01-05-2023 **ID NO**

Place : RK Valley U. Vishnuvardhan(R170516)

ACKNOWLEDGEMENT

I would like to express our sincere gratitude to **Ms M.Himabindu**, our project internal guide for valuable suggestions and keen interest throughout the progress of our course of research .

We are grateful to **Mr N.Satyanandaram** HOD CSE , for providing excellent computing facilities and congenial atmosphere for progressing with our project .

With sincere regards,
U.Vishnuvardhan(R170516)

ABSTRACT

In today's fast-paced business environment, maintaining an organization's software systems is crucial for ensuring that the organization remains competitive and meets the needs of its customers. Full-stack development has emerged as a popular approach for building and maintaining software systems, as it allows developers to work on both the front-end and back-end components of an application.

This project aims to explore the use of full-stack development in maintaining an organization's software systems. The project will involve developing a maintenance system using popular full-stack development frameworks such as React, and Node.js. The system will be designed to address common maintenance issues such as bug fixing, feature updates, and security patches.

The project will involve several phases, including requirements gathering, design, development, testing, and deployment. Throughout these phases, best practices for full-stack development will be followed, including agile development methodologies, continuous integration and deployment, and automated testing.

Table of Contents

Introduction

In today's digital age, maintaining an organization's software systems has become a critical aspect of business operations. As software systems become more complex and the demands of customers increase, organizations need to ensure that their software systems are running smoothly, are up to date with the latest features and security patches, and can adapt to changing market needs.

Full-stack development has emerged as a popular approach to building and maintaining software systems. It involves developing both the front-end and back-end components of an application, using a combination of technologies such as HTML, CSS, JavaScript, and various frameworks such as React, Angular, and Node.js. Full-stack development offers several advantages, including improved efficiency, faster time-to-market, and greater control over the entire development process.

The aim of this project is to explore the use of full-stack development in maintaining an organization's software systems. The project will involve developing a maintenance system that incorporates the latest technologies and best practices in full-stack development. The system will address common maintenance issues such as bug fixing, feature updates, and security patches, and will be designed to be scalable, flexible, and easy to maintain.

Throughout this project, we will follow best practices in full-stack development, including agile development methodologies, continuous integration and deployment, and automated testing. The outcome of this project will be a fully functional maintenance system that can be used by organizations to manage their software systems more efficiently.

Overall, this project aims to contribute to the advancement of software maintenance practices and help organizations stay competitive in today's fast-paced business environment.

PURPOSE

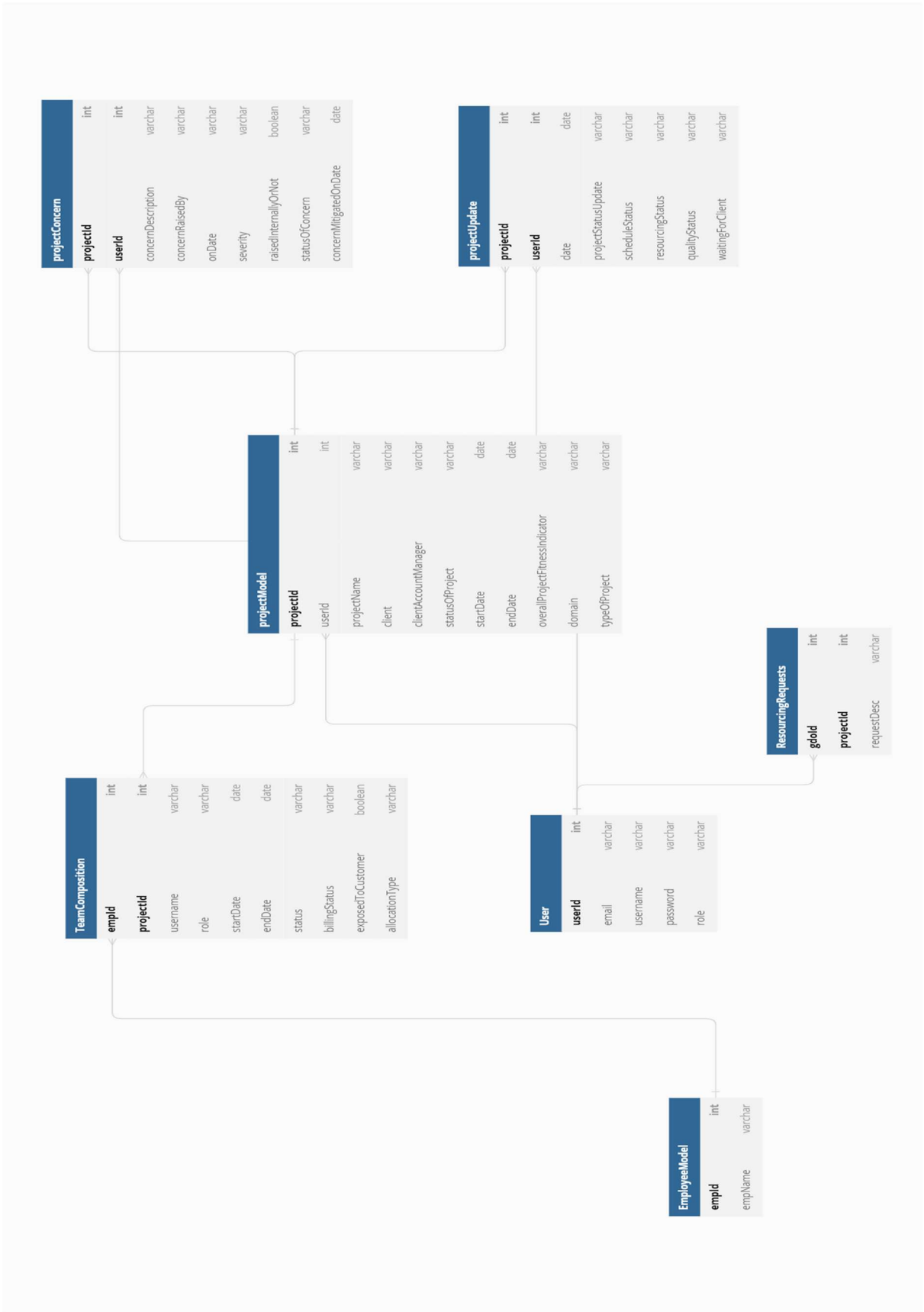
The purpose of this project is to develop a maintenance system for an organization using full-stack development. The maintenance system will address common maintenance issues such as bug fixing, feature updates, and security patches, and will be designed to be scalable, flexible, and easy to maintain.

The primary goal of this project is to demonstrate the advantages of full-stack development in maintaining an organization's software systems. By using a full-stack development approach, we can take advantage of the latest technologies and frameworks, and develop a system that is efficient, effective, and adaptable to changing business needs.

Another goal of this project is to promote best practices in software maintenance. We will follow agile development methodologies, continuous integration and deployment, and automated testing to ensure that the maintenance system is of high quality, robust, and reliable.

Overall, the purpose of this project is to provide organizations with a maintenance system that can help them stay competitive in today's digital landscape. By using the latest technologies and following best practices in software maintenance, we aim to develop a system that can help organizations manage their software systems more efficiently, reduce downtime, and improve overall customer satisfaction.

Class Diagram



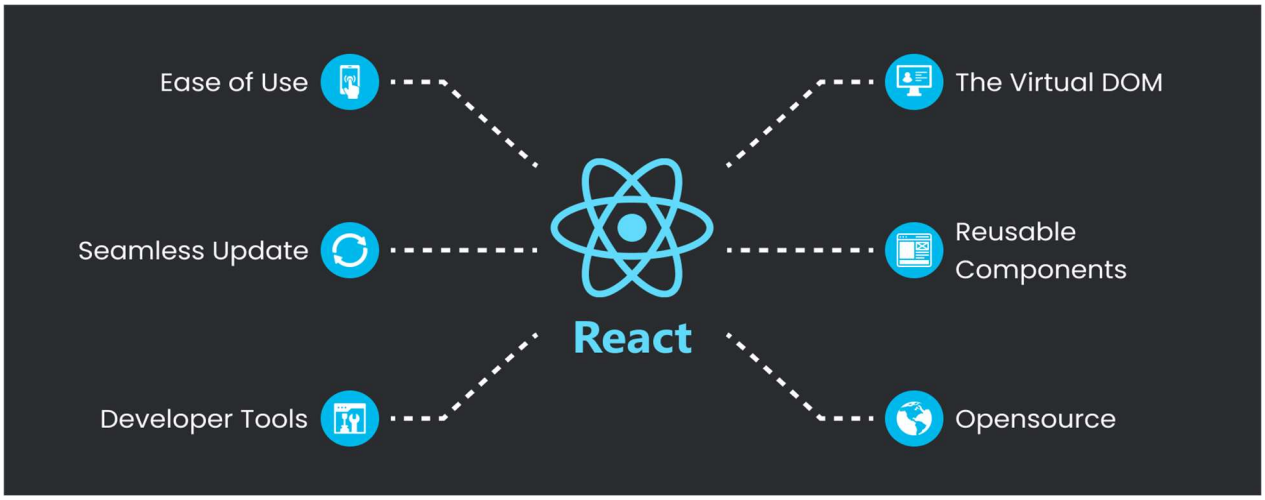
Software Requirements

FrontEnd:

I used React js for FrontEnd development. React.js, commonly referred to as React, is an open-source JavaScript library that is widely used for building user interfaces. Developed by Facebook, React was first released in 2013 and has since become one of the most popular libraries for front-end development.

React is based on the concept of building components, which are reusable building blocks that can be combined to create complex user interfaces. Each component has its own state and can be updated based on user interactions, data changes, or other events.

One of the key features of React is its virtual DOM (Document Object Model) implementation. Instead of directly manipulating the DOM, React uses a virtual representation of the DOM that is faster and more efficient. This allows for quick and seamless updates to the user interface, without the need for a full page reload.



React is also known for its flexibility and modularity. It can be easily integrated with other libraries and frameworks, such as Redux for state management or Next.js for server-side rendering. Additionally, React can be used to build mobile applications using React Native, which shares many of the same concepts and components as React.

Overall, React has revolutionized front-end development by providing a powerful, efficient, and flexible tool for building complex user interfaces. Its popularity continues to grow as more and more developers adopt it for their projects.

Backend:

I used NodeJs as Backend development to develop this project. Node.js is an open-source server-side JavaScript runtime environment that was developed by Ryan Dahl in 2009. It allows developers to build scalable and high-performance applications using JavaScript, a language that was previously only used for client-side development.

Node.js is built on the V8 JavaScript engine, which was originally developed for Google Chrome, and uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. This allows Node.js to handle a large number of

simultaneous connections without blocking other requests.

One of the key features of Node.js is its package manager, npm, which provides access to a vast library of modules and packages that can be easily installed and

integrated into a project. This makes it easy for developers to leverage existing code and avoid reinventing the wheel for common functionality.

Node.js is also highly flexible and can be used for a wide range of applications, including web servers, command-line tools, desktop applications, and more. It is widely used by companies such as Netflix, Uber, and LinkedIn, and has a thriving community of developers who contribute to its ongoing development and improvement.

Overall, Node.js has revolutionized server-side development by providing a fast, scalable, and efficient platform for building high-performance applications. Its popularity continues to grow as more and more developers adopt it for their projects, and it remains one of the most important technologies in modern web development.

DataBase:

I used MySQL database for data storing and accessing. MySQL is an open-source relational database management system (RDBMS) that was first released in 1995. It is one of the most popular databases used by web applications, and is widely used by developers and businesses of all sizes.

MySQL is based on the SQL (Structured Query Language) standard, and provides a robust and scalable platform for storing and retrieving data. It supports a wide range of data types, including integers, floats, strings, and more, and includes a

powerful set of features for managing databases, tables, and indexes.

One of the key benefits of MySQL is its performance and scalability. It is designed to handle large amounts of data and can support high levels of traffic without sacrificing performance or reliability. This makes it a popular choice for web applications that require fast, efficient access to data.

MySQL is also highly customizable and flexible, and can be easily integrated with other technologies such as PHP, Python, and Node.js. It is widely supported by a range of platforms and operating systems, and can be used on both Windows and Linux.

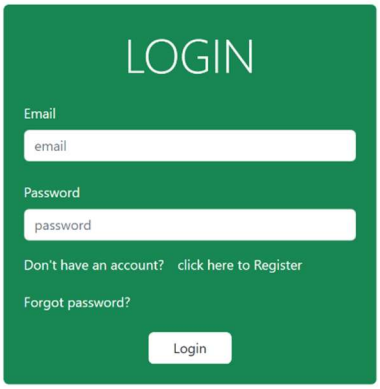
Overall, MySQL has become an essential technology for web development, providing a reliable, scalable, and efficient platform for storing and retrieving data. Its popularity continues to grow as more and more developers and businesses adopt it for their projects.

User Interfaces

User Login and Registration Interface

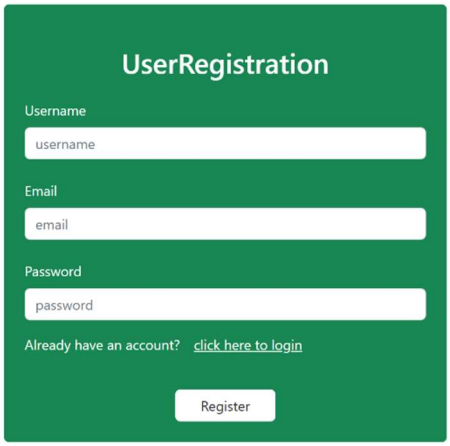
Project Pulse

A project tracking tool

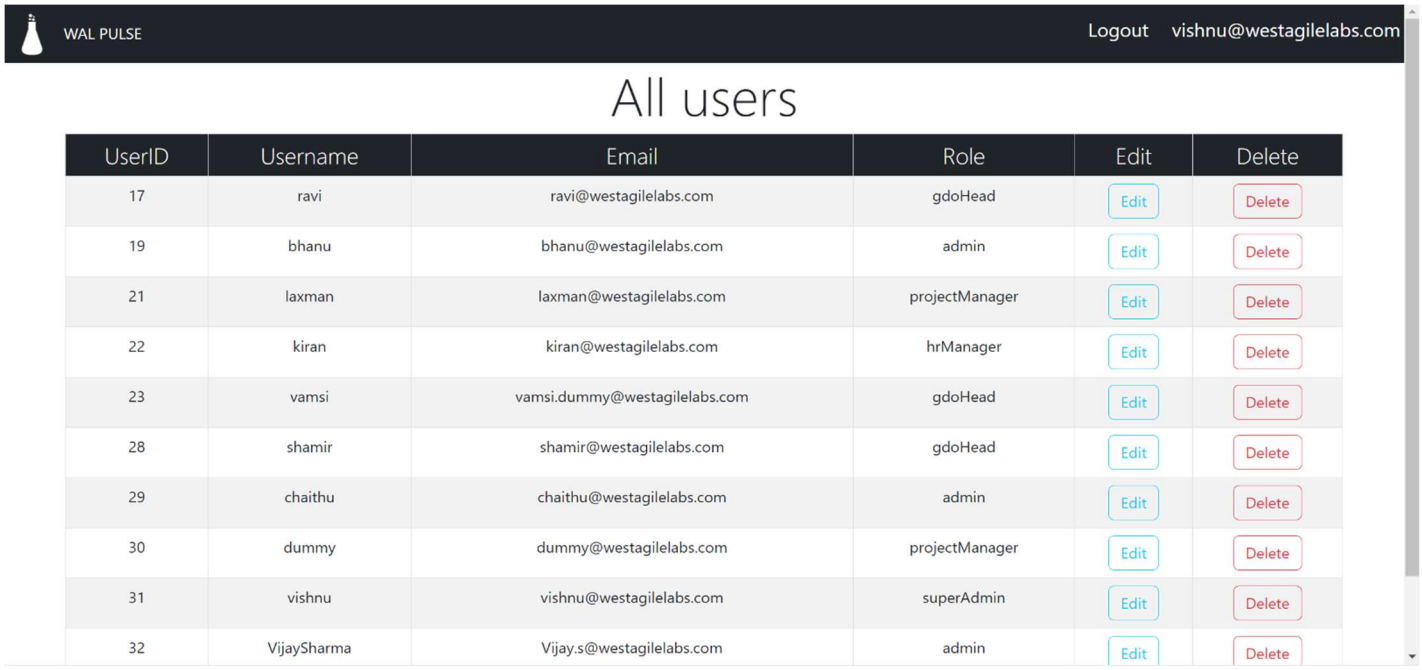


Project Pulse

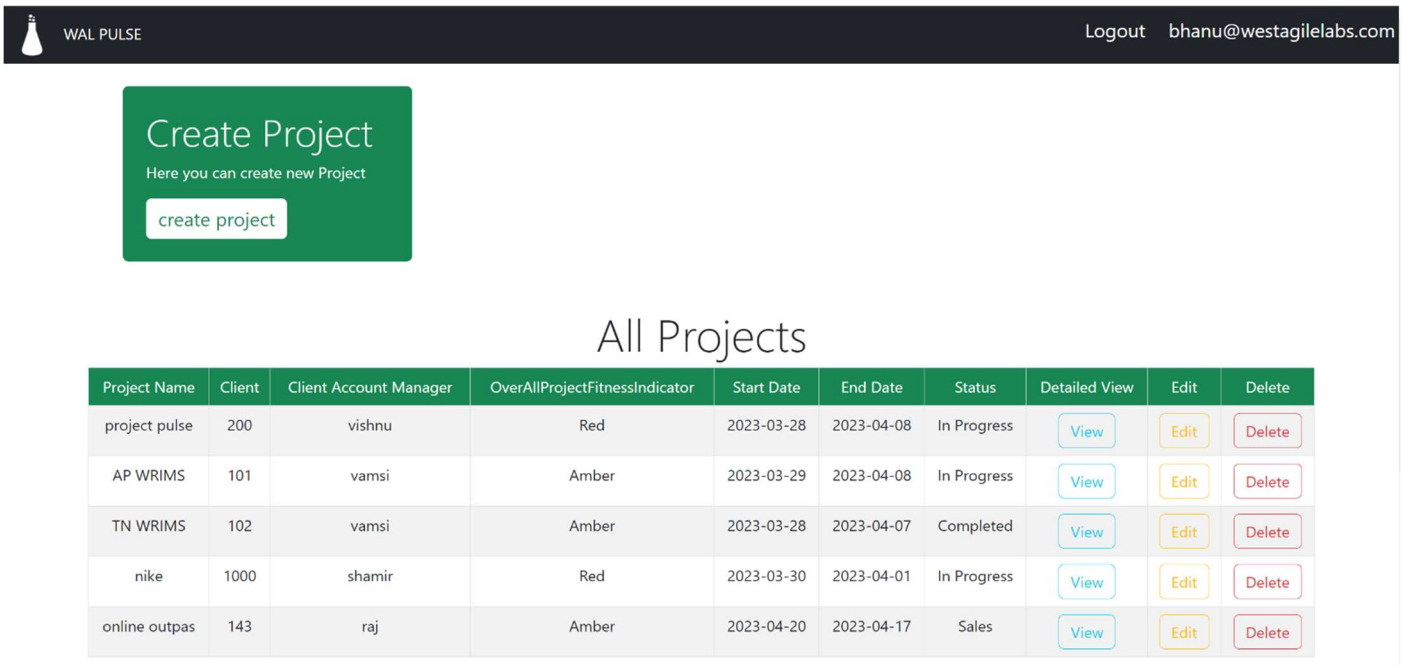
A project tracking tool



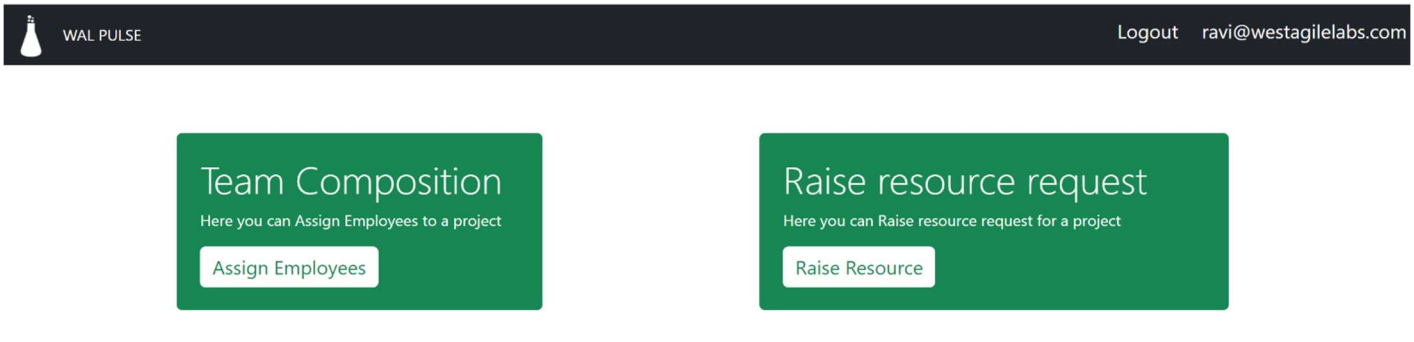
SuperAdmin Interface



Admin Interface



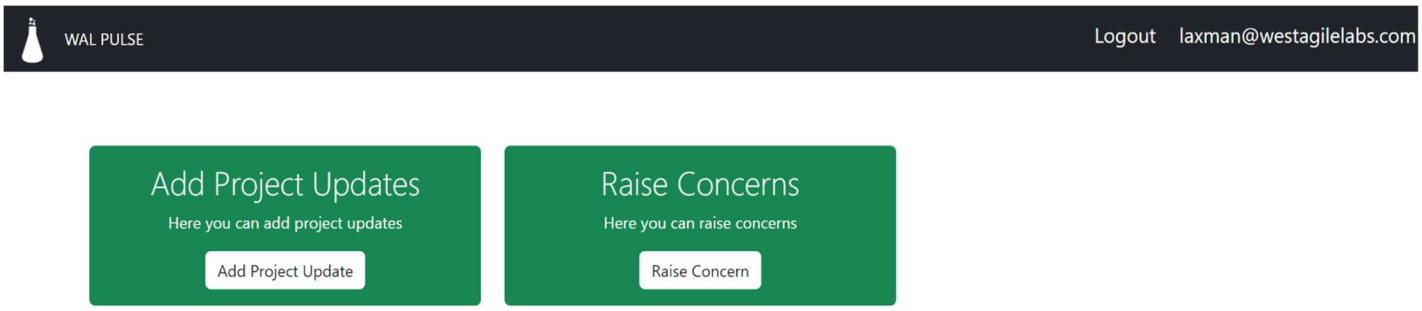
GDO Interface



All Projects

Project Name	Client	Client Account Manager	OverAllProjectFitnessIndicator	Start Date	End Date	Status	Detailed View
project pulse	200	vishnu	Red	2023-03-28	2023-04-08	In Progress	View
AP WRIMS	101	vamsi	Amber	2023-03-29	2023-04-08	In Progress	View
nike	1000	shamir	Red	2023-03-30	2023-04-01	In Progress	View

Manager Interface



All Projects

Project Name	Client	Client Account Manager	OverAllProjectFitnessIndicator	Start Date	End Date	Status	Detailed View
project pulse	200	vishnu	Red	2023-03-28	2023-04-08	In Progress	View
AP WRIMS	101	vamsi	Amber	2023-03-29	2023-04-08	In Progress	View
TN WRIMS	102	vamsi	Amber	2023-03-28	2023-04-07	Completed	View
nike	1000	shamir	Red	2023-03-30	2023-04-01	In Progress	View
online outpas	143	raj	Amber	2023-04-20	2023-04-17	Sales	View

Sample code

FrontEnd: React JS

```
import "./App.css";
import { createBrowserRouter, RouterProvider } from "react-router-dom";
import HomePage from "./components/homepage/HomePage";
import UserRegistration from "./components/userRegistration/UserRegistration";
import Login from "./components/login/Login";
import ForgotPassword from "./components/forgotPassword/ForgotPassword";
import RootLayout from "./components/rootLayout/RootLayout";
import SuperAdmin from "./components/superAdmin/SuperAdmin";
import GetUsers from "./components/superAdmin/GetUsers";
import AdminRootLayout from "./components/admin/AdminRootLayout";
import GetProjects from "./components/getProjects/GetProjects";
import Admin from "./components/admin/Admin";
import GdoRootLayout from "./components/gdo/GdoRootLayout";
import Gdo from "./components/gdo/Gdo";
import ProjectManager from "./components/projectManager/ProjectManager";
import ErrorPage from "./components/errorPage/ErrorPage";

function App() {
  // configuration of routes
  let browserRoutes = createBrowserRouter([
    {
      path: "/",
      element: <RootLayout />,
      errorElement: <ErrorPage />,
      children: [
        {
```

```
      path: "",
      element: <Login />,
    },

    {
      path: "/register",
      element: <UserRegistration />,
    },
  ],
},
{
  path: "/forgot-password",
  element: <ForgotPassword />,
},
{
  path: "/super-admin",
  element: <SuperAdmin />,
},
{
  path: "/admin",
  element: <AdminRootLayout />,
  children: [
    {
      path: "",
      element: <Admin />,
    },
    {
      path: "project/:projectId",
      element: <ProjectDetailedViewAdmin />,
    },
  ],
```

```
  },
  {
    path: "/gdo",
    element: <GdoRootLayout />,
    children: [
      {
        path: "",
        element: <Gdo />,
      },
      {
        path: "project/:projectId",
        element: <ProjectDetailedViewAdmin />,
      },
    ],
  },
  {
    path: "/project-manager",
    element: <ProjectManagerRootLayout />,
    children: [
      {
        path: "",
        element: <ProjectManager />,
      },
      {
        path: "project/:projectId",
        element: <ProjectDetailedViewAdmin />,
      },
    ],
  },
]);
```

```
  return (
    <div className="App">
      <RouterProvider router={browserRoutes} />
    </div>
  );
}

export default App;
```

BackEnd: NODE JS

```
// import express
const express = require("express");

// import helmet
const helmet = require("helmet")

// import cors
const cors = require("cors");

// create express obj application
const app = express();

app.use(helmet())

// cors
app.use(cors());
// dotenv
require("dotenv").config();

// import sequelize
```

```
const sequelize = require("./databases/db.config");

// import adminApp
const adminApp = require("./routes/admin.route");

// import userApp
const userApp = require("./routes/user.route");

// import superUserApp
const superAdminApp = require("./routes/superAdmin.route");

// import gdoApp
const gdoApp = require("./routes/gdo.route");

// import projectManagerApp
const projectManagerApp = require("./routes/projectManager.route");
const expressAsyncHandler = require("express-async-handler");
const { User } = require("./models/user.model");

// check sequelize connection
sequelize
  .authenticate()
  .then(() => console.log("DB CONNECTED..."))
  .catch((err) => console.log("DB FAILED...."));

// sync the sequelize
sequelize.sync();

// body parser
app.use(express.json());
```

```
// path middleware for superUser
app.use("/superAdmin-api", superAdminApp);

//path middleware for admin login
app.use("/admin-api", adminApp);

//path middleware for user
app.use("/user-api", userApp);

// path middleware for gdoApp
app.use("/gdo-api", gdoApp);

// path middleware for projectManager
app.use("/projectManager-api", projectManagerApp);

//get gdo details to display as select list while creating a project
app.get(
  "/gdo",
  expressAsyncHandler(async (req, res) => {
    let gdoRecord = await User.findAll({
      where: {
        role: "gdoHead",
      },
    });
    res.send({ message: "All GDO's", payload: gdoRecord });
  })
);

// get all projectManager details
app.get(
  "/projectManager",
```

```
  expressAsyncHandler(async (req, res) => {
    let projectManagerRecord = await User.findAll({
      where: {
        role: "projectManager",
      },
    });
    res.send({ message: "All project Manager", payload: projectManagerRecord });
  })
);

// get all hrManager details
app.get(
  "/hrManager",
  expressAsyncHandler(async (req, res) => {
    let hrManagerRecord = await User.findAll({
      where: {
        role: "hrManager",
      },
    });
    res.send({ message: "all Hr managers", payload: hrManagerRecord });
  })
);

//get employees for team composition
app.get(
  "/employees",
  expressAsyncHandler(async (req, res) => {
    let [employeeRecord] = await sequelize.query("select * from employees");
    res.send({ message: "All employees", payload: employeeRecord });
  })
);
```

```
// invalid path
app.use("*", (req, res) => {
  res.send({ message: "Invalid path" });
});
// error handling middleware
app.use((err, req, res, next) => {
  res.send({ message: err.message });
});

// import app
module.exports = app;
```

Future Scope

We can add additional feature in future that will be helpful to improve the efficiency of this project. I developed this project in west agile labs under the guidance of trainer provided by the organization. By this same concept we can apply to universities where HOD's and internal guides can track their student projects and hod's can track all students projects. We can extend this project to Universities that will be helpful to track of students projects and stoing the projects.

Conclusion

In conclusion, project maintenance for an organization using full stack development is a crucial aspect of ensuring the smooth operation and continued success of web projects. By utilizing a full stack approach, organizations can benefit from a comprehensive understanding of both the front-end and back-end technologies involved in their applications.

Effective project maintenance involves various tasks such as bug fixing, performance optimization, security updates, and feature enhancements. It requires a diligent and proactive approach to identify and address issues promptly, ensuring that the application remains stable, secure, and up-to-date.

Adopting the right software technologies and tools for project maintenance, such as version control systems, code editors, testing frameworks, and project management tools, can greatly streamline the maintenance process and improve collaboration among team members.

The future of project maintenance in full stack development holds great potential for automation, artificial intelligence, and continuous integration and delivery. With the growing emphasis on DevOps practices, cloud computing, and microservices, organizations can expect increased efficiency, scalability, and flexibility in their project maintenance efforts.

It is crucial for organizations to stay up-to-date with the latest trends and advancements in web development and invest in ongoing training and skill development for their maintenance teams. By embracing emerging technologies, adhering to best practices, and prioritizing quality and user experience, organizations can ensure the long-term success of their web projects.

References

JavaScript: https://developer.mozilla.org/en-US/docs/Web/JavaScript
Node Js: https://nodejs.dev/en/learn/
React js : https://react.dev/learn
Mysql : https://dev.mysql.com/doc/
TypeScript: https://www.typescriptlang.org/