

*Mini-Project Report On*  
**ACTIVITY POINT MANAGEMENT SYSTEM**

*Submitted in partial fulfillment of the requirements for the award of the degree of*

**Bachelor of Technology**  
**in**  
**Computer Science & Engineering**

**By**  
**Shreepad Sumesh Pudiyavalappil (RET20CS193)**  
**Rayyan Fadi (RET20CS160)**  
**Romain Robert (RET20CS168)**  
**Sebastian K Suresh (RET20CS183)**

*Under the guidance of*

**Dr.Renu Mary Daniel**



**Department Of Computer Science & Engineering**  
**Rajagiri School Of Engineering And Technology(Autonomous)**  
**Rajagiri Valley, Kakkanad, Kochi, 682039**

**July 2023**

*Mini-Project Report On*  
**ACTIVITY POINT MANAGEMENT SYSTEM**

*Submitted in partial fulfillment of the requirements for the award of the degree of*

**Bachelor of Technology**  
**in**  
**Computer Science & Engineering**

**By**  
**Shreepad Sumesh Pudiyavalappil (RET20CS193)**  
**Rayyan Fadi (RET20CS160)**  
**Romain Robert (RET20CS168)**  
**Sebastian K Suresh (RET20CS183)**

*Under the guidance of*

**Dr.Renu Mary Daniel**



**Department Of Computer Science & Engineering**  
**Rajagiri School Of Engineering And Technology(Autonomous)**  
**Rajagiri Valley, Kakkanad, Kochi, 682039**

**July 2023**

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING RAJAGIRI SCHOOL  
OF ENGINEERING AND TECHNOLOGY  
(AUTONOMOUS)

RAJAGIRI VALLEY, KAKKANAD, KOCHI, 682039



***CERTIFICATE***

*Certified that Mini Project work entitled “Activity Point Management System” is a bonafide work done by Shreepad Sumesh Pudiyavalappil (RET20CS193), Rayyan Fadi (RET20CS160), Romain Robert (RET20CS168), and Sebastian K Suresh (RET20CS183) of Sixth semester in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the APJ Abdul Kalam Technological University during the academic year 2022-2023*

**Dr. Preetha K.G**

Head of Department

Dept. of CSE

RSET

**Ms. Anita John**

Mini-Project Coordinator

Asst. Professor

Dept. of CSE

RSET

**Dr. Renu Mary Daniel**

Mini-Project Guide

Asst. Professor

Dept. of CSE

RSET

## ACKNOWLEDGEMENT

We are extremely honored to thank **Prof. (Dr). P.S. Sreejith**, Principal, RSET who has always made sure that our ladder to success was leaning against the right wall. We thank him for his constant help and support. We thank **Dr. Preetha K.G**, HoD, Department of Computer Science & Engineering, RSET whose help and guidance has been a major factor in completing our journey.

We express our gratitude to our Project Co-coordinators, **Ms. Anita John**, Asst. Professor, Dept. of Computer Science & Engineering, and **Mr. Sajan Raj**, Assistant Professor, Department of Computer Science and Engineering, for their valuable support and timely help, which have been the major factors in completing our journey.

We extend our sincere and heartfelt thanks to our guide **Dr. Renu Mary Daniel**, Asst. Professor, Dept. of Computer Science & Engineering, RSET for taking the time and effort to review our work and providing valuable advice and feedback from time to time. Last but not least, We would like to express our heartfelt and sincere gratitude to our family and friends for their constant support throughout this entire journey.

**Shreepad Sumesh Pudiyavalappil**

**Rayyan Fadi**

**Romain Robert**

**Sebastian K Suresh**

## **ABSTRACT**

The Activity Point Management System is a web-based application designed to streamline and automate the process of managing activity points in an educational institution. This system offers a convenient and efficient way to track and calculate activity points for student based on their participation in various activities.

The system provides a user-friendly interface where administrators can define different types of activities and assign corresponding point values. Students can then log into the system and record their participation in specific activities. The system maintains a comprehensive activity log, allowing users to track their individual progress and accumulated points.

# Table of Contents

<b>ACKNOWLEDGEMENT</b>	<b>ii</b>
<b>Abstract</b>	<b>iii</b>
<b>List of Figures</b>	<b>v</b>
<b>List Of Abbreviations</b>	<b>vi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 General Background . . . . .	1
1.1.1 Web development . . . . .	1
1.2 Objective . . . . .	1
1.3 Motivation . . . . .	1
1.4 Summary . . . . .	2
<b>2 Literature Survey</b>	<b>3</b>
2.1 New technologies for web development . . . . .	3
2.2 Webapp Service for Booking Handyman Using Mongodb, Express JS, React JS, Node JS . . . . .	3
<b>3 Proposed Method</b>	<b>4</b>
3.1 Problem Definition . . . . .	4
3.2 Scope Of The Work . . . . .	4
3.3 System Architecture . . . . .	4
3.4 Module Division . . . . .	4
3.5 Implementation . . . . .	5
3.5.1 Software Utilized . . . . .	5
<b>4 RESULTS AND DISCUSSIONS</b>	<b>7</b>
<b>5 Conclusion And Future Scope</b>	<b>11</b>
5.1 Challenges . . . . .	11
5.2 Conclusion . . . . .	11
5.3 Scope of Future Work . . . . .	11
<b>References</b>	<b>12</b>
<b>Appendix</b>	<b>13</b>

## List of Figures

1	3.3Architecture Diagram . . . . .	4
2	3.4Module Division . . . . .	5
3	4.1 Signup Page . . . . .	7
4	4.2 Login Page . . . . .	8
5	4.3 User View Page . . . . .	9
6	4.4 Certificate View Page . . . . .	10

## List Of Abbreviations

- HTML      Hyper Text Markup Language
- CSS        Cascading Style Sheets
- IDE        Integrated Development Environment
- APP        Application
- JSON      JavaScript Object Notation



# **1 Introduction**

## **1.1 General Background**

### **1.1.1 Web development**

Web development refers to the process of creating and maintaining websites and web applications. It involves a combination of programming, design, and other technical skills to build and optimize digital experiences on the internet. Web development has become a crucial aspect of our modern digital world, shaping how businesses, organizations, and individuals interact and engage online.

The foundation of web development lies in the use of technologies such as HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), and JavaScript. HTML provides the structure and content of web pages, CSS controls their visual appearance, and JavaScript enables interactivity and dynamic functionality.

## **1.2 Objective**

The objective of web development for the Activity Point Management System is to create a user-friendly web application that automates and streamlines the management and tracking of activity points. The web application aims to replace manual processes with an automated system, allowing participants to easily submit activities and monitor their progress. Administrators will have the ability to review and approve submitted activities and hence calculate activity points based on predefined criteria.

## **1.3 Motivation**

Activity Point Management Systems (APMS) provide a structured framework for managing activity points and offer numerous benefits over the current system as it has a more streamlined approach. The current system often relies on manual processes, which can be time-consuming, error-prone, and lack transparency. Therefore, implementing an APMS offers several motivations:

- **Streamlined Approach:** This streamlined approach in UI improves efficiency and saves valuable time for both students and administrators.
- **Enhanced Accuracy:** Manual systems are prone to errors, whether in data entry or calculation. APMS minimizes the risk of human error by automating the tracking and calculation of activity points.
- **Improved Transparency:** With APMS, the evaluation and allocation of activity points become transparent and fair. Students and faculty members can easily understand the criteria and guidelines for earning activity points, fostering a sense of clarity and equity within the institution. Transparency also enables students to track their progress and encourages active participation in various activities.

## **1.4 Summary**

The primary goal of this report explains the processes and efforts that went into developing a Web Application that helps students submit their certificates and the administrator to efficiently grant them activity points for the same. The first section deals with general background of the project its objective and the motivation behind it. The second part deals with an analysis of the different papers that were referred throughout the project. The third section describes the problem definition, the system architecture, the scope of the project and a detailed explanation about the implementation of the project and the technologies used. The fourth chapter contains the final results and discussions associated with the project while the last chapter outlines the conclusion of the entire project and its future scope.

## **2 Literature Survey**

### **2.1 New technologies for web development**

-Jakus, Grega Jekovec, Matija Tomazic, Saso Sodnik(2010)[2].

The paper gives an overview of the new features of web technologies. The general idea of the new version of HTML (Hyper Text Markup Language), i.e. HTML5, and other tools presented in this paper is the formal specification and the establishment of uniform solutions for technologies and functionalities which have already been in use through various hacks and plug-ins proposed by web developers. Many of these functionalities will now be implemented in browsers. The applications can access these functionalities through newly defined application programming interfaces. The latter include support for multimedia, dynamic graphic rendering, geolocation, multithreading, local data storage etc. HTML5 also introduces semantic markup, which can be used for marking the document structure as well as its elements and data. The new version of HTML enforces strict separation of the page content from its style. The styling can only be done using CSS (Cascading Style Sheets) language. The new CSS version, i.e. CSS3, has a modular structure, in which different modules define different styling features. The development cycles of the individual modules are independent as well as their support and implementation in various browsers.

### **2.2 Webapp Service for Booking Handyman Using Mongodb, Express JS, React JS, Node JS**

-K. Saundariya, M. Abirami, K. R. Senthil, D. Prabakaran, B. Srimathi and G. Nagarajan (2021)[1]

this paper gives an overview on how online website makes it easier to book your own workers at the correct time and cost, it makes the workers available in just one click at your doorstep. Handyman workers have a separate login to showcase themselves by adding the works and skills they have. It also helps the professionals to gain opportunities and money based on their work. There are several categories and services, on the time of users' login for the need for the services, the workers are listed based on location and cost with their name and contact. Creating a website using React JS makes it faster, boosts productivity, and is SEO friendly. MongoDB is a schema-less database and with ease of scale-out, hence it is easier to manage data. With this, a user can avoid delay and difficulty.

## 3 Proposed Method

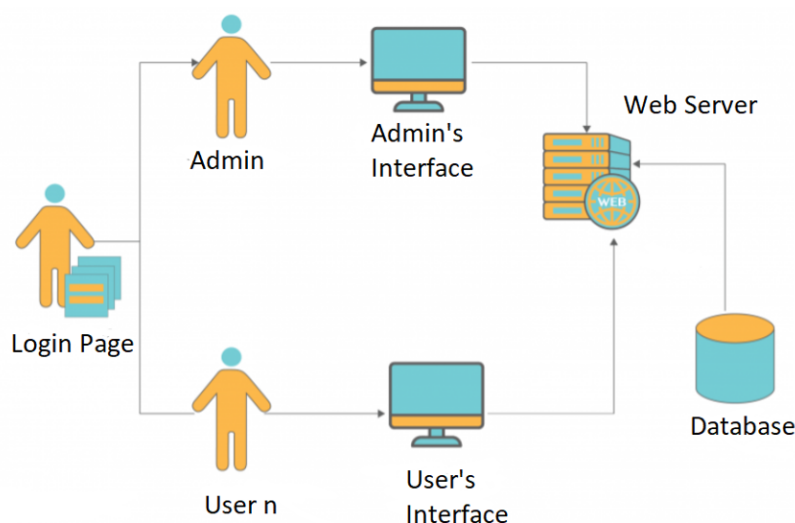
### 3.1 Problem Definition

To develop a web-based activity point management system that is easy to use for both students and faculty.

### 3.2 Scope Of The Work

The project helps students to easily submit their certificates through a user-friendly web application, which can then be accessed by the faculty in-charge to assign activity points if the submission is valid

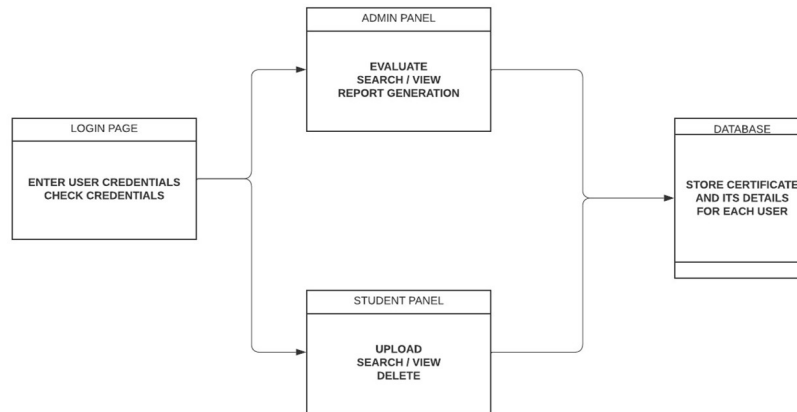
### 3.3 System Architecture



**Figure 1:** 3.3Architecture Diagram

### 3.4 Module Division

In the Login Module, you are able to log into your profile using a valid user id and a password of at least 8 characters. After sign in, based on the credential the user is directed to the student module or the admin module. The student module allows the student to upload their certificates with the necessary details. It also allows the user to view all uploaded certificates and whether any points were gained for the certificate. The admin module allows the admin to view all the certificates submitted by the students and to grant them activity points based on the validity of the uploaded certificates



**Figure 2: 3.4Module Division**

## 3.5 Implementation

### 3.5.1 Software Utilized

The various software utilized in our report include

- **VSCode**

Visual Studio Code (VSCode) is a free, open-source source code editor developed by Microsoft. It offers a customizable user interface with support for various programming languages. VSCode provides features such as syntax highlighting, code completion, linting, and intelligent code suggestions. It has an extensive ecosystem of extensions that allow users to customize and enhance the editor's functionality. The integrated terminal enables users to execute commands and run scripts directly within the editor. VSCode supports debugging with features like breakpoints and variable inspection. It also integrates well with version control systems like Git

- **Operating System:Windows 8 or above**

Windows 8 is a personal computer operating system that is part of the Windows NT family. Windows 8 introduced significant changes to the Windows operating system and its user interface (UI), targeting both desktop computers and tablets. It is a touch-optimized platform based on the modern Metro design architecture, which specifies how applications are delivered and rendered in the UI. Along with having a much different look and feel from its predecessor, Windows 7, Windows 8 also boasted faster startup times and better performance.

- **JavaScript**

JavaScript is a versatile and widely-used programming language primarily used for developing dynamic and interactive web applications. It is supported by all modern web

browsers, making it an essential component of front-end web development. JavaScript is a high-level, interpreted programming language that adds interactivity and dynamic behavior to web pages.

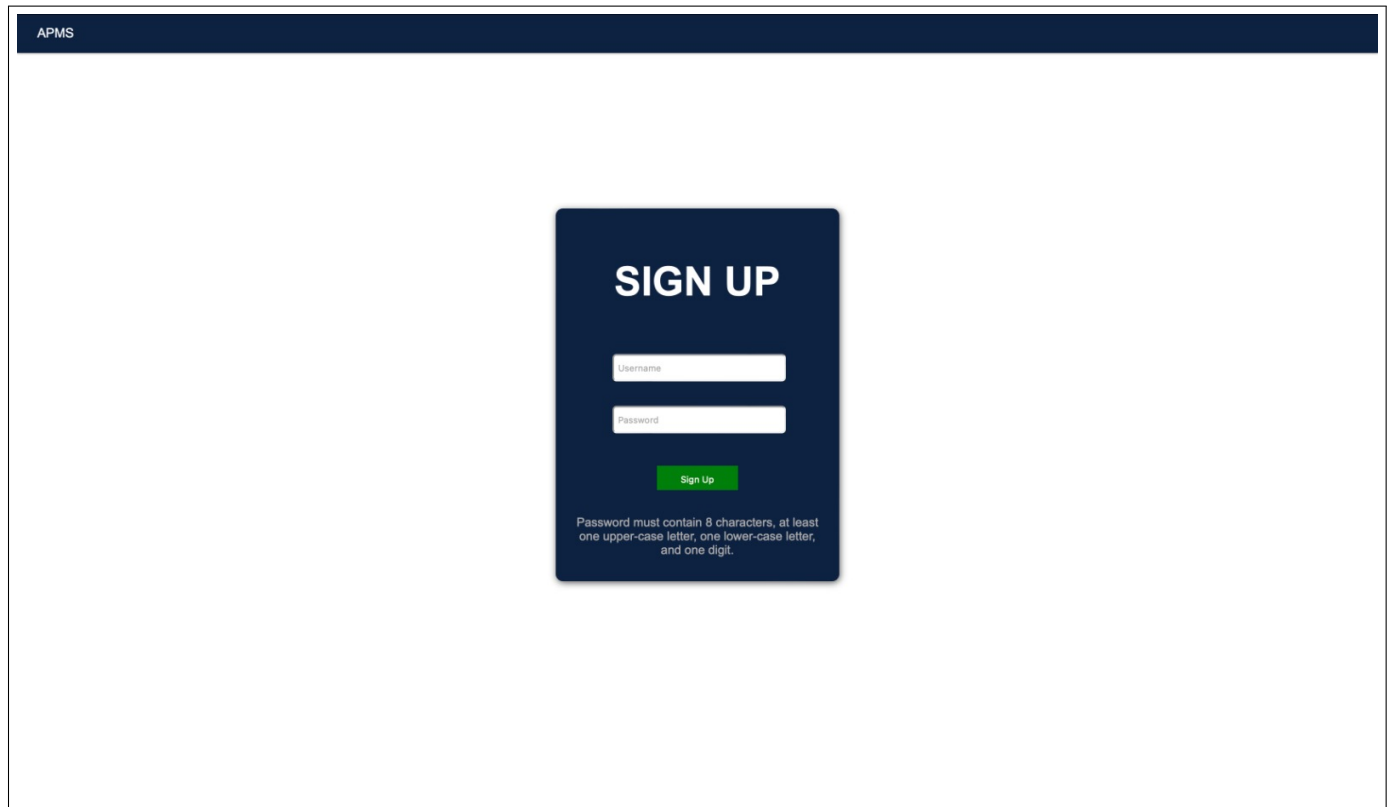
- **MongoDb**

MongoDB is a popular, open-source NoSQL database that provides a flexible and scalable solution for storing and managing data. It is designed to handle large volumes of unstructured or semi-structured data and offers high performance and availability. Unlike traditional relational databases, MongoDB uses a document-based data model. It stores data in flexible, JSON-like documents, allowing for easy and dynamic schema evolution. Documents within MongoDB collections can have varying structures, providing the flexibility to store data of different types and formats.

- **GitHub**

GitHub is a web-based platform and hosting service that allows developers to collaborate on software projects using the Git version control system. It provides a wide range of features and functionalities that facilitate project management, code collaboration, and community engagement. GitHub serves as a central repository for code hosting, version control, and collaboration. It enables developers to create and maintain Git repositories for their projects. Developers can push their code changes to a GitHub repository, track revisions, and collaborate with others seamlessly.

## 4 RESULTS AND DISCUSSIONS



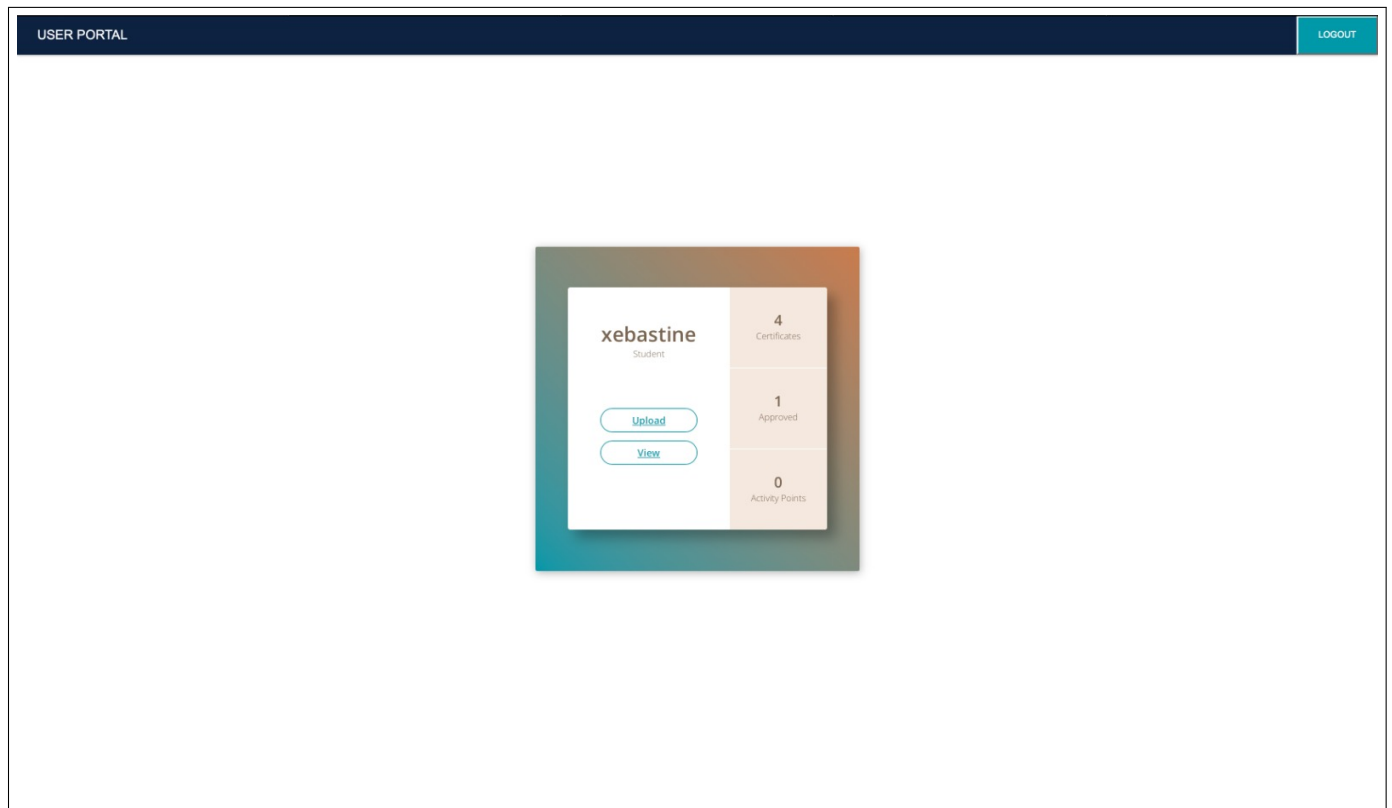
The image shows a web application interface for a signup page. At the top, there is a dark blue header bar with the text "APMS" in white. The main content area is white and contains a dark blue rectangular card with rounded corners. The card has the text "SIGN UP" in white, bold, uppercase letters. Below this, there are two white input fields: the first is labeled "Username" and the second is labeled "Password". Below the input fields is a green button with the text "Sign Up" in white. At the bottom of the card, there is a small line of text in white that reads: "Password must contain 8 characters, at least one upper-case letter, one lower-case letter, and one digit."

**Figure 3:** 4.1 Signup Page

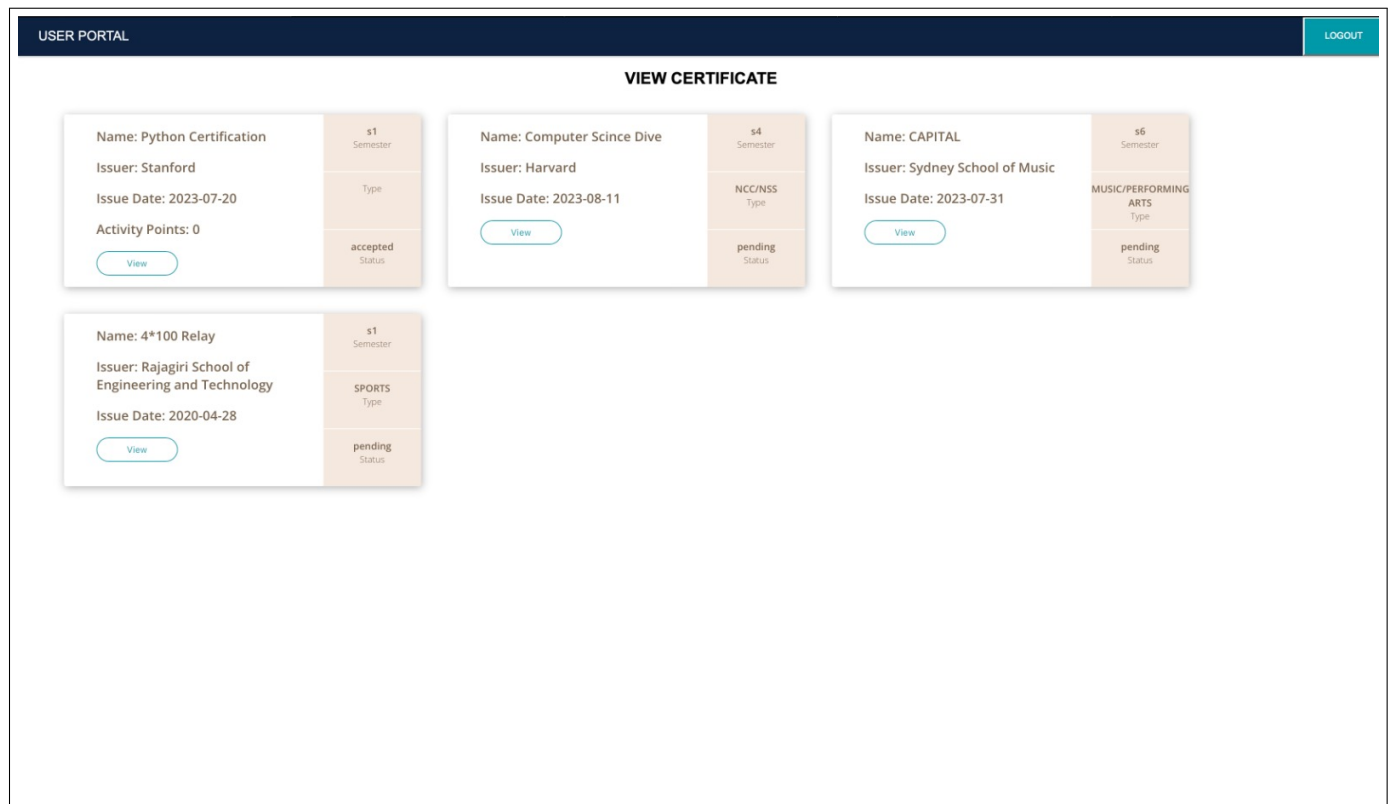
The image shows a web application login page. At the top, there is a dark blue horizontal header bar containing the text "APMS" in white. The main content area has a light gray background. In the center of this area is a dark blue rectangular box with rounded corners. Inside this box, the word "LOGIN" is written in large, white, bold, uppercase letters. Below the title, there are two white input fields: the first is labeled "Username" and the second is labeled "Password". Both labels are in a small, dark gray font. Below the password field is a green rectangular button with the word "Login" in white text.

**Figure 4:** 4.2 Login Page





**Figure 5:** 4.3 User View Page



**Figure 6:** 4.4 Certificate View Page

## **5 Conclusion And Future Scope**

### **5.1 Challenges**

We faced several realistic challenges for the perfect implementation of the project in the given time frame. This includes ensuring security of user information as well as accommodating a large number of users without system failure or lag.

### **5.2 Conclusion**

The Activity Point Management System is a valuable tool that automates the tracking and management of activity points within an educational institution. With the Activity Point Management System, students can easily submit their certificates and documentation for various activities, such as participation in events, workshops, or community service. Administrators and coordinators can use the system to review and approve activity submissions, keeping track of the accumulated points for each student or faculty member. The system provides an organized and centralized platform to manage activity points, eliminating the need for manual paperwork and reducing the chances of errors or discrepancies. Overall, the Activity Point Management System serves as a comprehensive solution for submitting certificates, ensuring validity and effectively managing activity points.

### **5.3 Scope of Future Work**

We hope to extend the app to accommodate a larger database as well to include options for automatic data collection from the certificate uploaded, a mobile application and to also increase the overall security.

## References

- [1] K. Saundariya, M. Abirami, K. R. Senthil, D. Prabakaran, B. Srimathi and G. Nagarajan, "Webapp Service for Booking Handyman Using MongoDB, Express JS, React JS, Node JS," 2021 3rd International Conference on Signal Processing and Communication (ICPSC), Coimbatore, India, 2021, pp. 180-183, doi: 10.1109/ICSPC51351.2021.9451783.
- [2] Jakus, Grega Jekovec, Matija Tomazic, Saso Sodnik, J.. (2010). New technologies for web development. 77.
- [3]B. Carter, "HTML Educational Node.js System (HENS): An Applied System for Web Development," 2014 Annual Global Online Conference on Information and Computer Technology, Louisville, KY, USA, 2014, pp. 27-31, doi: 10.1109/GOCICT.2014.25.
- [4]Y. Gu, S. Shen, J. Wang and J. -U. Kim, "Application of NoSQL database MongoDB," 2015 IEEE International Conference on Consumer Electronics - Taiwan, Taipei, Taiwan, 2015, pp. 158-159, doi: 10.1109/ICCE-TW.2015.7216831.
- [5]Y. Wu, "Design of User Database Resource Management System Based on Web," 2017 International Conference on Computer Technology, Electronics and Communication (ICCTEC), Dalian, China, 2017, pp. 364-367, doi: 10.1109/ICCTEC.2017.00084.

# **Appendix A: Sample Code**

Source Code for App.js :-

```
import React, { useState, useEffect } from "react";

import {
  BrowserRouter as Router,
  Routes,
  Route,
  Link,
  Outlet,
  useNavigate,
} from "react-router-dom";

import { useParams } from "react-router-dom";

import bcrypt from "bcryptjs";

import axios from "axios";

import "./App.css";

import { ToastContainer, toast } from "react-toastify";
import "react-toastify/dist/ReactToastify.css";

import $ from "jquery";

import HomePage from "./scenes/homePage";

import ribbon from "./assets/ribbon.png";

function App() {
  const [loggedInUser, setLoggedInUser] = useState(
    JSON.parse(localStorage.getItem("loggedInUser")) || null
  );

  const navigate = useNavigate();

  const handleSignUp = async (username, password) => {
    try {
```

```

const checkUsername = await axios.post(
  "http://localhost:3080/api/checkUsername",
  { username }
);

if (checkUsername.data.message === "Username already exists") {
  toast.error("Username already exists", {
    position: "top-center",
    autoClose: 5000,
    hideProgressBar: false,
    closeOnClick: true,
    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    theme: "colored",
  });
  return;
}

const saltRounds = 10;
const hashedPassword = await bcrypt.hash(password, saltRounds);
const newUser = await axios.post("http://localhost:3080/api/signup", {
  username,
  password: hashedPassword,
});
alert(newUser.data.message);
} catch (error) {
  console.error(error);
  toast.error("Error creating user", {
    position: "top-center",

```

```

    autoClose: 5000,
    hideProgressBar: false,
    closeOnClick: true,
    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    theme: "colored",
  });
}
};

const handleLogin = async (username, password, onSuccess, onFailure) => {
  try {
    const user = await axios.post("http://localhost:3080/api/login", {
      username,
      password,
    });
    if (user.data.error) {
      onFailure();
      toast.error("User not found or invalid password.", {
        position: "top-center",
        autoClose: 5000,
        hideProgressBar: false,
        closeOnClick: true,
        pauseOnHover: true,
        draggable: true,
        progress: undefined,
        theme: "colored",
      });
    }
  }
};

```



```

const { remainingAttempts, lockoutTime } = user.data;
if (remainingAttempts === 0) {
  const minutes = Math.ceil(lockoutTime / 60000); // Convert milliseconds to minutes
  toast.info(`You are locked out. Please try again in 5 minutes.`, {
    position: "top-center",
    autoClose: 5000,
    hideProgressBar: false,
    closeOnClick: true,
    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    theme: "colored",
  });
}

return;
}

localStorage.setItem("loggedInUser", JSON.stringify(user.data));

setLoggedInUser(user.data);
onSuccess();
if (user.data.isAdmin) {
  navigate("/admin");
} else {
  navigate(`/user/${username}`);
}
} catch (error) {

```

```

console.error(error);
toast.error("Error logging in", {
  position: "top-center",
  autoClose: 5000,
  hideProgressBar: false,
  closeOnClick: true,
  pauseOnHover: true,
  draggable: true,
  progress: undefined,
  theme: "colored",
});
}
};

const handleLogout = () => {
  localStorage.removeItem("loggedInUser");

  setLoggedInUser(null);
  navigate("/");
  window.location.reload();
};

return (
  <div>
    <ToastContainer />
    <div className="page">
      <nav className="page__menu menu">
        <div className="menu__wrapper">
          <ul className="menu__list r-list">

```

```

{loggedInUser ? null : (
  <li className="menu__group">
    <Link to="/" className="menu__link r-link text-underlined">
      APMS
    </Link>
  </li>
)}
{loggedInUser ? (
  <>
    {loggedInUser.isAdmin ? (
      <li className="menu__group">
        <Link
          to="/admin"
          className="menu__link r-link text-underlined"
        >
          Admin Portal
        </Link>
      </li>
    ) : (
      <li className="menu__group">
        <Link
          to={` /user/${loggedInUser.username}`}
          className="menu__link r-link text-underlined"
        >
          User Portal
        </Link>
      </li>
    )}
  </>
)}

```

```

    ) : null}
  </ul>
  {loggedInUser && (
    <button
      onClick={handleLogout}
      className="menu__link r-link text-underlined logout-button"
    >
      Logout
    </button>
  )}
</div>
</nav>
</div>

```

```

<Routes>
  <Route path="/" element={<Home />} />
  <Route
    path="/admin/*"
    element={<AdminPortal loggedInUser={loggedInUser} />}
  />
  <Route
    path="/user/:username"
    element={<UserPortal loggedInUser={loggedInUser} />}
  />
  <Route
    path="/user/:username/upload-certificate"
    element={<UploadCertificate loggedInUser={loggedInUser} />}
  />
  <Route

```

```

        path="/user/:username/view-certificate"
        element={<ViewCertificate loggedInUser={loggedInUser} />}
    />
    <Route
        path="/login"
        element={<LoginForm handleLogin={handleLogin} />}
    />
    <Route
        path="/signup"
        element={<SignUpForm handleSignUp={handleSignUp} />}
    />

    <Route
        path="/admin/user/:username"
        element={<UserPage navigate={navigate} loggedInUser={loggedInUser} />}
    />
    <Route path="*" element={<NotFound />} />
</Routes>
</div>
);
}

```

```

function Home() {
    const [loggedInUser] = useState(
        JSON.parse(localStorage.getItem("loggedInUser")) || null
    );
    return (
        <div>
            <HomePage />

```

```

<div className="homeButton">
  {!loggedInUser ? (
    <>
      <Link to="/login" className="link">
        Login
      </Link>
      <Link to="/signup" className="link">
        Sign Up
      </Link>
    </>
  ) : null}
</div>
</div>
);
}

```

```

function AdminPortal({ loggedInUser }) {
  const [users, setUsers] = useState([]);
  const [filteredUsers, setFilteredUsers] = useState([]);
  const [searchTerm, setSearchTerm] = useState("");
  const [usersWithPendingImageCount, setUsersWithPendingImageCount] = useState([]);

  const navigate = useNavigate();

  useEffect(() => {
    if (!loggedInUser || !loggedInUser.isAdmin) {
      navigate("/");
    }
    fetchUsers();
  });
}

```

```
}, [loggedInUser, navigate]);
```

```
const fetchUsers = async () => {  
  try {  
    const response = await axios.get("http://localhost:3080/api/users");  
    const filteredUsers = response.data.users.filter((user) => !user.isAdmin);  
    setUsers(filteredUsers);  
    setFilteredUsers(filteredUsers);  
  } catch (error) {  
    console.error(error);  
  }  
};
```

```
useEffect(() => {  
  fetchUsersWithPendingImageCount();  
}, []);
```

```
const fetchUsersWithPendingImageCount = async () => {  
  try {  
    const response = await axios.get("http://localhost:3080/api/users/pendingImageCount");  
    setUsersWithPendingImageCount(response.data.users);  
  } catch (error) {  
    console.error('Error fetching users with pending image count:', error);  
  }  
};
```

```
const handleSearch = (e) => {  
  const searchTerm = e.target.value;  
  setSearchTerm(searchTerm);  
};
```





```

<main className="leaderboard__profiles">
  {filteredUsers.map((user) => (
    <Link to={` /admin/user/${user.username}`} key={user._id}>
      <article className="leaderboard__profile">
        <span className="leaderboard__name">{user.username}</span>
        <span className="leaderboard__value">{user.pendingImageCount}</span>
      </article>
    </Link>
  ))}
</main>
</article>
</div>
</div>
);
}

```

```

function UserPage({ navigate, loggedInUser }) {
  const { username } = useParams();
  const [images, setImages] = useState([]);

  useEffect(() => {
    fetchImages();
  }, []);

  const fetchImages = async () => {
    try {
      const response = await axios.get(
        `http://localhost:3080/api/user/${username}`
      );
    }
  }
}

```

```

    const { imageData } = response.data;
    setImages(imageData || []);
  } catch (error) {
    console.error(error);
  }
};

const handleBack = () => {
  navigate("/admin/users");
};

const handleImageClick = (imageName) => {
  window.open(`http://localhost:3080/api/image/${username}/${imageName}`);
};

const handleStatusChange = async (imageName, status) => {
  try {
    await axios.put(
      `http://localhost:3080/api/user/${username}/image/${imageName}`,
      { status }
    );
    fetchImages(); // Fetch the updated images after status change
  } catch (error) {
    console.error(error);
    toast.error("Error updating status", {
      position: "top-center",
      autoClose: 5000,
      hideProgressBar: false,
      closeOnClick: true,
    });
  }
};

```

```

    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    theme: "colored",
  });
}
};

return (
  <div>
    <h2>User: {username}</h2>
    <button onClick={handleBack}>Back</button>
    {images.length ? (
      <ul>
        {images.map((image, index) => (
          <li key={index}>
            <p>
              Image Name:{" "}
              <span
                className="image-link"
                onClick={() => handleImageClick(image.imageName)}
              >
                {image.imageName}
              </span>
            </p>
            <p>Dropdown 1: {image.dropdown1}</p>
            <p>Dropdown 2: {image.dropdown2}</p>
            <p>Certificate Details:</p>
            <p>Name: {image.certificateDetails.name}</p>

```

```

    <p>Issue Date: {image.certificateDetails.issueDate}</p>
    <p>Issuer: {image.certificateDetails.issuer}</p>
    <p>Status: {image.status}</p>
    <p>Activity Points: {image.activityPoints}</p>

    {loggedInUser && loggedInUser.isAdmin && (
    <div>
        <label htmlFor={`status-select-${index}`}>Status:</label>
        <select
            id={`status-select-${index}`}
            value={image.status}
            onChange={(e) =>
                handleStatusChange(image.imageName, e.target.value)
            }
        >
            <option value="pending">Pending</option>
            <option value="accepted">Accepted</option>
            <option value="rejected">Rejected</option>
        </select>
    </div>
    )}
</li>
)}}
</ul>
):(
    <p>No images found.</p>
    )}
</div>
);

```

```
}
```

```
function UserPortal({ loggedInUser }) {  
  const navigate = useNavigate();  
  const username = loggedInUser ? loggedInUser.username : "";  
  const [totalActivityPoints, setTotalActivityPoints] = useState(0);  
  const [totalCertificates, setTotalCertificates] = useState(0);  
  const [approvedCertificates, setApprovedCertificates] = useState(0);  
  
  useEffect(() => {  
    if (!loggedInUser || !loggedInUser.isAdmin) {  
      navigate("/");  
    } else {  
      fetchUserData();  
    }  
  }, [loggedInUser, navigate]);  
  
  const fetchUserData = async () => {  
    try {  
      const response = await axios.get(  
        `http://localhost:3080/api/user/${username}/data`  
      );  
      const { totalActivityPoints, totalCertificates, approvedCertificates } =  
        response.data;  
      setTotalActivityPoints(totalActivityPoints);  
      setTotalCertificates(totalCertificates);  
      setApprovedCertificates(approvedCertificates);  
    } catch (error) {  
      console.error(error);  
    }  
  }  
}
```

```

    }
};

return (
    <div className="frame">
        <div className="center">
            <div className="profile">
                <div className="name">{username}</div>
                <div className="job">Student</div>

                <div className="actions">
                    <Link className="btn" to={` /user/${username}/upload-certificate`} >
                        Upload
                    </Link>
                    <Link className="btn" to={` /user/${username}/view-certificate`} >
                        View
                    </Link>
                </div>
            </div>

            <div className="stats">
                <div className="box">
                    <span className="value">{totalCertificates}</span>
                    <span className="parameter">Certificates</span>
                </div>
                <div className="box">
                    <span className="value">{approvedCertificates}</span>
                    <span className="parameter">Approved</span>
                </div>
            </div>
        </div>
    </div>
);

```

```

    <div className="box">
      <span className="value">{totalActivityPoints}</span>
      <span className="parameter">Activity Points</span>
    </div>
  </div>
</div>
</div>
);
}

```

```

function UploadCertificate({ loggedInUser }) {
  const [image, setImage] = useState(null);
  const [activityPoints, setActivityPoints] = useState(0);

  const [dropdownValues, setDropdownValues] = useState({
    dropdown1: "s1",
    dropdown2: "",
  });

  const [certificateData, setCertificateData] = useState({
    name: "",
    issueDate: "",
    issuer: "",
  });

  const handleOptionChange = (event) => {
    const { name, value } = event.target;
    setDropdownValues((prevValues) => ({
      ...prevValues,
      [name]: value,
    }));
  };
}

```

```
}});
```

```
let points = 0;  
if (name === "dropdown2") {  
  if (value === "NCC/NSS") {  
    points = 50;  
  } else if (value === "SPORTS") {  
    points = 60;  
  } else if (value === "MUSIC/PERFORMING ARTS") {  
    points = 70;  
  }  
}
```

```
setActivityPoints(points);  
};
```

```
const handleCertChange = (event) => {  
  const { name, value } = event.target;  
  setCertificateData((prevData) => ({  
    ...prevData,  
    [name]: value,  
  }));  
};
```

```
const handleImageUpload = (e) => {  
  const file = e.target.files[0];  
  setImage(file);  
};
```



```

const handleSubmit = async (e) => {
  e.preventDefault();

  if (!loggedInUser) {
    toast.error("User not logged in", {
      position: "top-center",
      autoClose: 5000,
      hideProgressBar: false,
      closeOnClick: true,
      pauseOnHover: true,
      draggable: true,
      progress: undefined,
      theme: "colored",
    });
    return;
  }

  if (!image) {
    toast.warn("Please select an image", {
      position: "top-center",
      autoClose: 5000,
      hideProgressBar: false,
      closeOnClick: true,
      pauseOnHover: true,
      draggable: true,
      progress: undefined,
      theme: "colored",
    });
    return;
  }

```

```
}
```

```
const formData = new FormData();
formData.append("image", image);
formData.append("username", loggedInUser.username);
formData.append("dropdown1", dropdownValues.dropdown1);
formData.append("dropdown2", dropdownValues.dropdown2);
formData.append("name", certificateData.name);
formData.append("issueDate", certificateData.issueDate);
formData.append("issuer", certificateData.issuer);
formData.append("activityPoints", activityPoints); // Add activityPoints to form data

try {
  await axios.post("http://localhost:3080/api/uploadImage", formData, {
    headers: { "Content-Type": "multipart/form-data" },
  });
  toast.success("Image uploaded successfully", {
    position: "top-center",
    autoClose: 2000,
    hideProgressBar: false,
    closeOnClick: true,
    pauseOnHover: true,
    draggable: true,
    progress: undefined,
    theme: "colored",
  });
  setImage(null);
} catch (error) {
  console.error(error);
}
```

```

toast.error("Error uploading image", {
  position: "top-center",
  autoClose: 5000,
  hideProgressBar: false,
  closeOnClick: true,
  pauseOnHover: true,
  draggable: true,
  progress: undefined,
  theme: "colored",
});
}
};

return (
  <div className="certcard">
    <img src={ribbon} />
    <h2 className="cert-heading">Certificate</h2>
    <div className="CertificateForm">
      <form onSubmit={handleSubmit}>
        <select
          name="dropdown1"
          value={dropdownValues.dropdown1}
          onChange={handleOptionChange}
        >
          <option value="s1">s1</option>
          <option value="s2">s2</option>
          <option value="s3">s3</option>
          <option value="s4">s4</option>
          <option value="s5">s5</option>
        </select>
      </form>
    </div>
  </div>
);

```

```

<option value="s6">s6</option>
<option value="s7">s7</option>
<option value="s8">s8</option>
</select>
<select
  name="dropdown2"
  value={dropdownValues.dropdown2}
  onChange={handleOptionChange}
>
  <option value="">Select an option</option>
  <option value="NCC/NSS">NCC/NSS</option>
  <option value="SPORTS">SPORTS</option>
  <option value="MUSIC/PERFORMING ARTS">MUSIC/PERFORMING ARTS</option>
</select>
<div>
  <label htmlFor="name">Name:</label>
  <input
    type="text"
    id="name"
    name="name"
    value={certificateData.name}
    onChange={handleCertChange}
    required
  />
</div>
<div>
  <label htmlFor="issuer">Issuer:</label>
  <input
    type="text"

```

```

        id="issuer"
        name="issuer"
        value={certificateData.issuer}
        onChange={handleCertChange}
        required
    />
</div>
<div>
    <label htmlFor="issueDate">Date:</label>
    <input
        type="date"
        id="issueDate"
        name="issueDate"
        value={certificateData.issueDate}
        onChange={handleCertChange}
        required
    />
</div>
<br />
<input type="file" accept="image/jpeg" onChange={handleImageUpload} />
<br />
    <button type="submit">Upload</button>
</form>
</div>
</div>
);
}

```

```

function ViewCertificate({ loggedInUser }) {

```

```

const [imageData, setImageData] = useState([]);

useEffect(() => {
  fetchCertificate();
}, []);

const fetchCertificate = async () => {
  try {
    const response = await axios.get(
      `http://localhost:3080/api/user/${loggedInUser.username}`
    );
    const { imageData } = response.data;
    setImageData(imageData || []);
  } catch (error) {
    console.error(error);
  }
};

const handleImageClick = (imageName) => {
  window.open(
    `http://localhost:3080/api/image/${loggedInUser.username}/${imageName}`
  );
};

return (
  <div className="viewcertificate">
    <center>
      <h2>VIEW CERTIFICATE</h2>
    </center>
  </div>
);

```

```

{imageData.length ? (
  <ul>
    <div className="certviewcard-container">
      {imageData.map((image, index) => (
        <div className="certviewcard">
          <ul key={index}>
            {/* <p>Image Name: <span className="image-link" onClick={() =>
handleImageClick(image.imageName)}>{image.imageName}</span></p> */}
            <p>Name: {image.certificateDetails.name}</p>
            <p>Issuer: {image.certificateDetails.issuer}</p>
            <p>Issue Date: {image.certificateDetails.issueDate}</p>
            {/* <p>Status: {image.status}</p>
            <p>Semester: {image.dropdown1}</p>
            <p>Type: {image.dropdown2}</p> */}

            {image.status === "accepted" && (
              <p>Activity Points: {image.activityPoints}</p>
            )}
            <button type="View" onClick={() => handleImageClick(image.imageName)}
>View</button>
          </ul>
        <div className="viewcertificatestatus">
          <div className="vcbox">
            <span className="value">{image.dropdown1}</span>
            <span className="parameter">Semester</span>
          </div>
          <div className="vcbox">
            <span className="value">{image.dropdown2}</span>
            <span className="parameter">Type</span>
          </div>

```

```

        <div className="vcbox">
            <span className="value">{image.status}</span>
            <span className="parameter">Status</span>
        </div>
    </div>
</div>
    )}
</div>
</ul>
): (
    <p>No images found.</p>
    )}
</div>
);
}

```

```

function LoginForm({ handleLogin }) {
    const navigate = useNavigate();
    const [username, setUsername] = useState("");
    const [password, setPassword] = useState("");
    const [failedAttempts, setFailedAttempts] = useState(0);

    const handleSubmit = (e) => {
        e.preventDefault();
        handleLogin(username, password, handleLoginSuccess, handleLoginFailure);
    };

    const handleLoginSuccess = () => {
        setFailedAttempts(0);
    };
}

```



```

};

const handleLoginFailure = () => {
  setFailedAttempts((prevAttempts) => prevAttempts + 1);

  if (failedAttempts + 1 >= 3) {
    toast.error("Maximum login attempts exceeded. Please try again later.", {
      position: "top-right",
      autoClose: false,
      hideProgressBar: false,
      closeOnClick: true,
      pauseOnHover: true,
      draggable: true,
      progress: undefined,
      theme: "colored",
    });
    navigate("/");
  }
};

return (
  <div className="card">
    <h2 className="card-heading">LOGIN</h2>
    <form className="LoginPage" onSubmit={handleSubmit}>
      <input
        type="text"
        placeholder="Username"
        value={username}
        onChange={(e) => setUsername(e.target.value)}

```

```

    />
    <br />
    <input
      type="password"
      placeholder="Password"
      value={password}
      onChange={(e) => setPassword(e.target.value)}
    />
    <br />
    <button type="submit">Login</button>
  </form>
</div>
);
}

function SignUpForm({ handleSignUp }) {
  const [username, setUsername] = useState("");
  const [password, setPassword] = useState("");

  const handleSubmit = (e) => {
    e.preventDefault();
    if (password.length < 8) {
      toast.warn("Password must be at least 8 characters long.", {
        position: "top-right",
        autoClose: 2500,
        hideProgressBar: false,
        closeOnClick: true,
        pauseOnHover: true,
        draggable: true,

```

```

        progress: undefined,
        theme: "colored",
    });
    return;
}

if (!/(?=.*[a-z])(?=.*[A-Z])/).test(password)) {
    toast.warn(
        "Password must contain at least one uppercase letter and one lowercase letter.",
        {
            position: "top-right",
            autoClose: 2500,
            hideProgressBar: false,
            closeOnClick: true,
            pauseOnHover: true,
            draggable: true,
            progress: undefined,
            theme: "colored",
        }
    );
    return;
}

if (!/(?=.*\d)/.test(password)) {
    toast.warn("Password must contain a digit.", {
        position: "top-right",
        autoClose: 2500,
        hideProgressBar: false,
        closeOnClick: true,
        pauseOnHover: true,
        draggable: true,

```

```

        progress: undefined,
        theme: "colored",
    });
    return;
}
handleSignUp(username, password);
};

return (
    <div className="card">
        <h2 className="card-heading">SIGN UP</h2>
        <form className="LoginPage" onSubmit={handleSubmit}>
            <input
                type="text"
                placeholder="Username"
                value={username}
                onChange={(e) => setUsername(e.target.value)}
            />
            <br />
            <input
                type="password"
                placeholder="Password"
                value={password}
                onChange={(e) => setPassword(e.target.value)}
            />
            <br />
            <button type="submit">Sign Up</button>
        </form>
    <div className="conditions">

```

Password must contain 8 characters, at least one upper-case letter, one lower-case letter, and one digit.

```
</div>
```

```
</div>
```

```
);
```

```
}
```

```
function NotFound() {
```

```
  return (
```

```
    <div>
```

```
      <h1>404 Not Found</h1>
```

```
      <p>Oops! The page you're looking for does not exist.</p>
```

```
    </div>
```

```
  );
```

```
}
```

```
export default App;
```

## **Appendix B: CO-PO And CO-PSO Mapping**

## Course Outcome

SI No.	Description	Blooms Taxonomy Level
CSD334.1	Think innovatively on the development of components, products, processes or technologies in the engineering field.	Knowledge (Level 1) Analyse (Level 4)
CSD334.2	Apply knowledge gained in solving real life engineering problems.	Evaluate (Level 2) Understand (Level 5)

## CO-PO Mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CSD334.1	-	3	-	2	-	-	1	-	3	2	1	3
CSD334.2	3	2	3	2	2	-	-	2	3	2	-	1

## CO-PSO Mapping

	PSO1	PSO2	PSO3
CSD334.1	3	-	1
CSD334.2	3	3	2

### Justifications for CO-PO/PSO Mapping

Mapping	Low/Medium/High	Justification
CSD334.1–PO4	M	Conduct investigations of complex problems : I used research based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
CSD334.1–PO7	L	Environment and sustainability : I understood the impact of the professional engineering solutions in societal and environmental contexts, and demonstrated the knowledge of- and the need for- sustainable developments.
CSD334.1–PO9	H	Individual: We were able to function effectively as an individual, in multi-disciplinary settings.
CSD334.1–PO10	M	Communication : We were able to communicate effectively on complex Engineering activities with the Engineering Community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
CSD334.1–PO11	L	Project Management and finance : Demonstrated knowledge and understanding of the Engineering and management principles and apply these to ones own work, to manage projects and in multi-disciplinary environments.
CSD334.1–PO12	H	Life-long learning : Recognized the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



CSD334.1–PSO1	H	Computer Science Specific Skills : Was able to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas by understanding the core principles and concepts of computer science.
CSD334.1–PSO3	L	Professional Skills : Was able to apply the fundamentals of computer science to formulate competitive research proposals and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur.
CSD334.2–PO1	H	Engineering Knowledge : Applied the knowledge of Mathematics, Science, Engineering fundamentals, and an Engineering discipline to the solution of complex engineering problems.
CSD334.2–PO2	M	Problem analysis : We were able to identify, formulate, review research literature, and analyze complex Engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and Engineering sciences.
CSD334.2–PO3	H	Design/Development of solutions : Designed solutions for complex Engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
CSD334.2–PO4	M	Conduct investigations of complex problems : Used research based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

CSD334.2–PO5	L	Modern Tool usage : Created, selected, and applied appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex Engineering activities with an understanding of the limitations.
CSD334.2–PO8	M	Ethics : Applied ethical principles and commit to professional ethics and responsibilities and norms of the Engineering practice.
CSD334.2–PO9	H	Individual: We were able to function effectively as an individual, and in multi-disciplinary settings.
CSD334.2–PO10	M	Communication : Communicated effectively on complex Engineering activities with the Engineering Community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
CSD334.2–PO12	L	Life-long learning : Recognized the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.
CSD334.2–PSO1	H	Computer Science Specific Skills : We were able to identify, analyze and design solutions for complex engineering problems in multi-disciplinary areas by understanding the core principles and concepts of computer science.
CSD334.2–PSO2	H	Programming and Software Development Skills : Acquired programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products.

---

CSD334.2–PSO3	M	Professional Skills : Applied the fundamentals of computer science to formulate competitive research proposals and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur.
---------------	---	--

