

MINI GITHUB



A PROJECT REPORT

Submitted by

AGALYA M (2303811710522003)

in partial fulfillment of requirements for the award of the course

EGB1221- DATABASE MANAGEMENT SYSTEMS

in

ELECTRICAL AND ELECTRONICS ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

JUNE - 2025

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)**

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “**MINI GITHUB**” is the Bonafide work of **AGALYA M (2303811710522003)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE

Dr. A.T. SANKARA SUBRAMANIAN, M.E.,phd.,
HEAD OF THE DEPARTMEN

ASSOCIATE PROFESSOR
Department of EEE

K. Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

SIGNATURE

Ms. P. KARTHIKA, M.E.,
SUPERVISOR

ASSISTANT PROFESSOR
Department of CSE

K. Ramakrishnan College of
Technology (Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**MINI GITHUB**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **EGB1221 – DATABASE MANAGEMENT SYSTEMS**.

Signature

AGALYA M

Place: Samayapuram

Date: 04.06.2025

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A.T. SANKARA SUBRAMANIAN, M.E, PhD.**, Head of the department, **ELECTRICAL AND ELECTRONICS ENGINEERING** for providing his encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Ms. P. KARTHIKA M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for her incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To develop globally competent Electrical Engineers with expertise in education and cutting edge research technologies thereby contribute value to their career and society.

MISSION OF DEPARTMENT

M1: Knowledge: To bestow quality education in Electrical and Electronics Engineering and prepare the students for career development and higher studies.

M2: Skill: To Excel in Contemporary core and Interdisciplinary areas with Prime Research Facilities and Industrial collaborations.

M3: Attitude: To augment student competency along with moral and ethical values through academics to serve the society.

PROGRAM EDUCATIONAL OBJECTIVES

PEO 01 – Knowledge

To empower graduates with high standards of technical knowledge making them readily employable or well prepared for pursuing higher education to thrive in their career development.

PEO 02 – Skills

To produce graduates with interdisciplinary skills who can contribute meaningfully to cutting-edge research and innovation in emerging areas, thereby making a significant impact on their respective industries and the global community.

PEO 03 - Attitude

To develop highly competent professionals who are also committed to uphold the highest standards of ethical conduct and moral consciousness in the society.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 01

To demonstrate proficiency in Core Electrical areas such as Electrical Circuits, Electromagnetic fields, Control Engineering, Instrumentation, Electrical Drives, Power System and Power Electronics to Solve Practical Engineering Problems.

PSO 02

To analyze, design and develop Electronic circuits and systems through insights acquired in Integrated circuits, Embedded systems, Analog and Digital Electronics.

PSO 03

To apply technical competency, management and interdisciplinary skills for developing socially acceptable solutions to complex emerging area problems and its applications.

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
- 3. Design/development of solutions:** Design 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
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** 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.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

Mini GitHub is a lightweight web-based platform designed to replicate core functionalities of GitHub. It allows users to create accounts, manage repositories, and collaborate on code projects. Users can upload files, track versions, and view commit histories. The system supports basic Git operations like commit, push, and pull through a user-friendly interface. Collaboration features include issues, pull requests, and comments. Repositories can be public or private based on user preferences. Built with modern web technologies, it ensures responsive and secure interactions. The backend uses version control logic to simulate Git behavior. This project demonstrates an understanding of software development workflows. It serves as a simplified model for learning and prototyping code collaboration tools.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD DATABASE APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
Mini GitHub is a web-based platform designed to facilitate code repository management and version control for developers. Built using PHP and MySQL with XAMPP, it provides essential features such as project creation, file uploads, and collaborative code sharing. The system aims to simplify code tracking and improve teamwork in software development projects.	PO1 -3 PO2 -3 PO3 -3 PO5 -3 PO9 -3 PO10 -3 PO12 -3	PSO1 -1 PSO2 -2 PSO3 -3

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	viii
	LIST OF FIGURES	xi
	LIST OF ABBREVIATIONS	xii
1	INTRODUCTION	
	1.1 OBJECTIVE	9
	1.2 OVERVIEW	
	1.3 SQL AND DATABASE CONCEPTS	10
2	PROJECT METHODOLOGY	
	2.1 PROPOSED WORK	11
	2.2 ARCHITECTURE DIAGRAM	13
3	MODULE DESCRIPTION	
	3.1 USER MANAGEMENT	14
	3.2 TASK MANAGEMENT	15
	3.3 COLLABRATION	16
	3.4 CALENDAR AND SCHEDULING	17
	3.5 FILE MANAGEMENT	18
4	CONCLUSION AND FUTURE ENHANCEMENT	19
5	APPENDIX A SOURCE CODE	20
	APPENDIX B SCREEN SHOT	25
	REFERENCES	

LIST OF FIGURES

FIGURE.NO	TITLE	PAGE.NO
2.1 .	MINI GITHUB	13
4.1	ADMIN LOGIN	25
4.2.	REPOSITORIES	25
4.3.	DASHBOARD.	25

LIST OF ABBREVIATIONS

ABBREVIATIONS

API	– APPLICATION PROGRAMMING INTERFACE
DB	– DATABASE
HTTP	– HYPER TEXT TRANSFER PROTOCOL
HTML	– HYPER TEXT MARKUP LANGUAGE
SQL	– STRUCTURED QUERY LANGUAGE
UI	– USER INTERFACE
UX	– USER EXPERIENCE
MVC	– MODEL VIEW CONTROLLER
IDE	– INTEGRATED DEVELOPMENT ENVIRONMENT
JSON	– JAVASCRIPT OBJECT NOTATION
URL	– UNIFORM RESOURCE LOCATOR
VCS	– VERSION CONTROL SYSTEM

CHAPTER 1

INTRODUCTION

1.1 Objective

The primary objective of this project is to design and implement a simple web-based repository management system, named **Mini GitHub**, which allows users to create an account, log in, and manage their repositories in a way similar to GitHub. The platform aims to provide basic functionality for user authentication and repository management entirely on the frontend using HTML, CSS, and JavaScript. By utilizing `localStorage`, this project eliminates the need for a backend server or database, making it suitable for educational purposes and small-scale applications. Through this project, users will be able to register an account by entering their name, email, and password, and then log in using these credentials to access their personalized repository management dashboard. In this dashboard, users can create new repositories and manage them by specifying whether they want them to be public or private. This repository management system is designed to be simple, intuitive, and user-friendly, making it accessible even to those with minimal experience in web development.

1.2 Overview

The Mini GitHub project is a simplified web application designed to replicate basic functionalities found in repository management platforms like GitHub. It allows users to register an account, log in using their credentials, and manage repositories through a user-friendly interface. The system operates entirely on the frontend using HTML, CSS, and JavaScript, with `localStorage` utilized to persist user data and repository information in the browser. Users can create repositories and choose between public or private visibility settings..

1.3 SQL and Database Concepts

SQL (Structured Query Language) is the standard language used to manage and manipulate data in relational databases. It is vital for building scalable and secure backend systems for applications like Mini GitHub. Although this project uses localStorage for data persistence, SQL would typically be used in a full-fledged system to manage user and repository data on the server side.

Key SQL concepts include:

- **Tables:** Data is organized into tables, where each table contains rows (records) and columns (attributes). For example, a users table could store user details (name, email, password), and a Repositories table could store repository information.
- **Primary Key:** Each record in a table must have a unique identifier known as the primary key, which ensures each entry is distinct. In the Users table, the email or user ID could serve as the primary key.
- **Foreign Key:** A foreign key establishes a relationship between two tables. For instance, a Repositories table might have a foreign key linking to the Users table to associate each repository with a specific user.
- **CRUD Operations:** SQL enables the basic operations of creating, reading, updating, and deleting data (CRUD), which are fundamental for interacting with the database. For example, users can be created during registration, their data can be read during login, and repositories can be updated or deleted.
- **Relationships and Normalization:** SQL supports relationships (one-to-many, many-to-many) between tables and normalization to reduce data redundancy and improve efficiency.

CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work for the Mini GitHub project involves the design and development of a simplified version control and collaboration platform inspired by GitHub. The goal is to enable users to create repositories, upload code files, track changes, and collaborate with others in a lightweight environment. The project will be developed using PHP and MySQL with XAMPP.

1. User Management

- User Registration and Login system
- Secure authentication with hashed passwords
- Profile management (name, email, bio)

2. Repository Management

- Create and delete repositories
- Set repository visibility: public or private
- View list of repositories for each user

3. File Management

- Upload source code files to repositories
- Display files and folders within each repository
- Enable file updates (simulate versioning)

4. Commit History (Basic Version Control)

- Save commit messages with each file update
- Track file update timestamps

5. Collaboration Features

- Add collaborators to repositories
- Allow collaborators to upload and update files

6. Repository Viewer

- Display code files with syntax highlighting (optional)
- Allow public users to view public repositories

7. Admin Dashboard (Optional)

- View all users and repositories
- Manage (delete/block) users or repos

8. Technology Stack

- Frontend: HTML, CSS, JavaScript
- Backend: PHP
- Database: MySQL
- Server: Apache (XAMPP)

2.2 Architecture Diagram

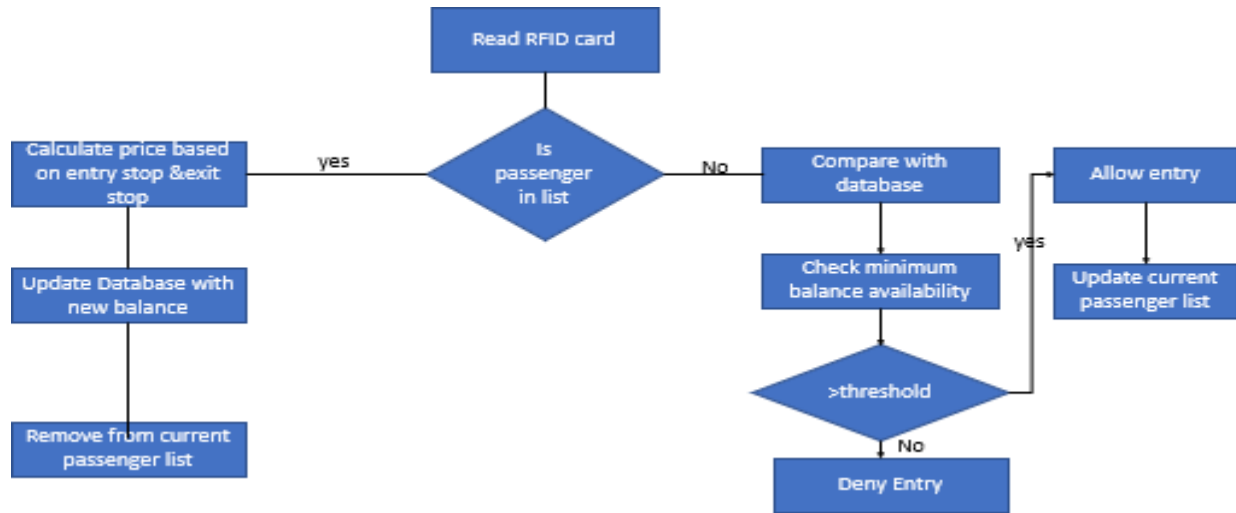


Fig: 2.1 Mini Github

CHAPTER 3

MODULE DESCRIPTION

3.1 User Interface Module

This module manages user registration, login, and logout to control secure access to the Mini GitHub platform. It ensures only authorized users can access their accounts by verifying credentials securely. Passwords are stored using encryption techniques to protect sensitive information. The module also manages user sessions to maintain login state.

Key Features

- User registration with validation of input data
- Secure login with encrypted passwords
- Session management and automatic logout
- Password reset and user account management

Functional Flow

- User accesses login or registration page and submits credentials
- System validates user information and authenticates the user
- On successful login, user session is initiated
- User can logout to terminate the session securely

Outcome

- Secure authentication prevents unauthorized platform access
- Protects user credentials and sensitive data
- Maintains active user sessions to manage state
- Enhances overall security and trust in the system

3.2 Registration Module

This module provides an intuitive and interactive front-end interface allowing users to access Mini GitHub features easily. It includes pages for login, dashboards, repository management, profile settings, and collaboration. The UI is designed to be responsive and user-friendly, ensuring smooth navigation and interaction.

Key Features

- Responsive dashboard showing user repositories and activities
- Navigation bar for quick access to key features
- Interactive forms for repository and profile management
- Real-time notifications and feedback alerts

Functional Flow

- User logs in and views the personalized dashboard
- Navigates through repositories, profiles, and collaboration sections
- Forms dynamically update and submit data to the backend
- Receives immediate feedback on actions via notifications

Outcome

- Provides easy and efficient navigation for users
- Connects backend data with visual elements effectively
- Enhances user satisfaction through clear and responsive design
- Minimizes user errors with validation and prompt alerts

3.3 Repository Management Module

This module enables users to create, organize, and manage repositories, which serve as containers for project code. Users can define repository visibility (public/private) and maintain control over their projects. It also allows adding collaborators for shared project work.

Key Features

- Create, rename, and delete repositories
- Set repository visibility (public/private)
- List and search user repositories
- Add and manage collaborators with roles

Functional Flow

- User creates or edits repository details such as name and visibility
- Repository data is stored and reflected in the user dashboard
- User can delete repositories or modify details as needed
- Collaborators can be invited or removed by the owner

Outcome

- Enables efficient project organization and management
- Provides control over repository accessibility
- Facilitates collaboration through shared repositories
- Simplifies repository lifecycle and user interaction

3.4 File and Version Control Module

This module manages source code files within repositories and tracks their changes through a basic version control system. Users can upload, update files, and add commit messages to document changes. It maintains a history of file versions with timestamps for traceability.

Key Features

- Upload and update source code files
- Commit changes with descriptive messages
- Maintain version history with timestamps and user info
- View commit logs and past versions

Functional Flow

- User uploads or modifies a file in a repository
- Adds a commit message explaining the change
- System saves the new version and updates commit history
- User can review previous versions or revert if necessary

Outcome

- Tracks and manages code changes effectively
- Improves project accountability and progress visibility
- Allows rollback to earlier versions when needed
- Simplifies version control without complex commands

3.5 Collaboration Module

This module supports collaborative development by allowing repository owners to add other users as collaborators. It manages permissions, granting collaborators either read-only or read-write access, and tracks contributor activities to facilitate teamwork.

Key Features

- Add or remove collaborators by username or email
- Assign permission levels (read/write) for collaborators
- Display collaborator list per repository
- Notify users about collaborator activity and updates

Functional Flow

- Owner invites collaborators with specific permissions
- Collaborators access the repository according to assigned rights
- Collaborators make changes, which are tracked by the version control system
- Owner can modify or revoke collaborator access anytime

Outcome

- Encourages team-based development and sharing
- Controls access to code with permission management
- Improves productivity through coordinated collaboration
- Maintains accountability and role clarity

CHAPTER 4

CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION:

This project demonstrates the creation of a simple web-based repository system with user authentication and repository management using only frontend technologies (HTML, CSS, JavaScript) and localStorage for data persistence. The system offers a foundational look into how a repository platform can work. However, it's limited to the client-side and does not feature advanced functionalities such as collaboration, version control, or database interaction, which would typically be seen in a full-fledged version of GitHub.

FUTURE ENHANCEMENT:

To enhance this project in the future, the following improvements could be made:

Backend Integration: Replace localStorage with a server-side database like MySQL, PostgreSQL, or MongoDB for better data management, security, and scalability.

Authentication & Security: Implement secure user authentication using encryption techniques (e.g., hashing passwords) and token-based authentication (JWT).

Version Control: Introduce basic version control functionality to repositories, allowing users to track changes and revert to previous versions.

User Interface Improvements: Improve the user interface for better usability, responsiveness, and aesthetics. Add features like drag-and-drop for repository creation or file uploads.

Collaboration Features: Allow users to share repositories with others, similar to how GitHub enables collaboration.

APPENDIX A (CODE):

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Mini GitHub</title>

  <style>

    body { font-family: Arial, sans-serif; margin: 20px; }

    nav a { margin-right: 15px; cursor: pointer; color: blue; text-decoration: underline; }

    section { display: none; margin-top: 20px; }

    section.active { display: block; }

    form { margin-top: 10px; }

    input, select, button { margin: 5px 0; display: block; }

  </style>

</head>

<body>

  <h1>Mini GitHub</h1>

  <nav>

    <a onclick="showSection('home')">Home</a>

    <a onclick="showSection('register')">Register</a>

    <a onclick="showSection('login')">Login</a>

    <a onclick="showSection('repositories')">Repositories</a>

  </nav>

  <section id="home" class="active">

    <h2>Welcome to Mini GitHub</h2>

    <p>This is a simple code repository system (frontend only) using localStorage.</p>

  </section>

  <section id="register">
```



```

<h2>User Registration</h2>

<form onsubmit="registerUser(event)">

  <label>Name: <input type="text" id="regName" required></label>

  <label>Email: <input type="email" id="regEmail" required></label>

  <label>Password: <input type="password" id="regPass" required></label>

  <button type="submit">Register</button>

</form>

</section>

<section id="login">

  <h2>User Login</h2>

  <form onsubmit="loginUser(event)">

    <label>Email: <input type="email" id="loginEmail" required></label>

    <label>Password: <input type="password" id="loginPass" required></label>

    <button type="submit">Login</button>

  </form>

  <p id="loginStatus" style="color: red;"></p>

</section>

<section id="repositories">

  <h2>Your Repositories</h2>

  <div id="repoList"></div>

  <form onsubmit="createRepo(event)">

    <h3>Create Repository</h3>

    <label>Name: <input type="text" id="repoName" required></label>

    <label>Visibility:

      <select id="repoVisibility">

        <option value="public">Public</option>

        <option value="private">Private</option>

      </select>

    </label>

```

```

    <button type="submit">Create</button>

</form>

</section>

<script>

let currentUser = null;

function showSection(id) {

    document.querySelectorAll('section').forEach(sec => sec.classList.remove('active'));

    document.getElementById(id).classList.add('active');

    if(id === 'repositories') loadRepositories();

}

function registerUser(event) {

    event.preventDefault();

    const name = document.getElementById('regName').value;

    const email = document.getElementById('regEmail').value;

    const pass = document.getElementById('regPass').value;

    const users = JSON.parse(localStorage.getItem('users') || '[]');

    if (users.find(user => user.email === email)) {

        alert('User already exists!');

        return;

    }

    users.push( { name, email, password: pass, repos: [] } );

    localStorage.setItem('users', JSON.stringify(users));

    alert('Registration successful!');

    showSection('login');

}

function loginUser(event) {

    event.preventDefault();

    const email = document.getElementById('loginEmail').value;

    const pass = document.getElementById('loginPass').value;

```

```

const users = JSON.parse(localStorage.getItem('users') || '[]');
const user = users.find(u => u.email === email && u.password === pass);
if (user) {
  currentUser = user.email;
  sessionStorage.setItem('currentUser', currentUser);
  document.getElementById('loginStatus').textContent = "";
  showSection('repositories');
} else {
  document.getElementById('loginStatus').textContent = 'Invalid credentials!';
}
}

function loadRepositories() {
  const repoDiv = document.getElementById('repoList');
  repoDiv.innerHTML = "";
  const users = JSON.parse(localStorage.getItem('users') || '[]');
  const user = users.find(u => u.email === sessionStorage.getItem('currentUser'));
  if (!user) {
    repoDiv.innerHTML = '<p>Please login to view repositories.</p>';
    return;
  }
  if (user.repos.length === 0) {
    repoDiv.innerHTML = '<p>No repositories found.</p>';
  } else {
    const ul = document.createElement('ul');
    user.repos.forEach(repo => {
      const li = document.createElement('li');
      li.textContent = `${repo.name} (${repo.visibility})`;
      ul.appendChild(li);
    });
  }
}

```

```

        repoDiv.appendChild(ul);
    }
}

function createRepo(event) {
    event.preventDefault();

    const repoName = document.getElementById('repoName').value;
    const visibility = document.getElementById('repoVisibility').value;
    const users = JSON.parse(localStorage.getItem('users') || '[]');
    const userIndex = users.findIndex(u => u.email === sessionStorage.getItem('currentUser'));

    if (userIndex === -1) {
        alert('User not logged in!');
        return;
    }

    users[userIndex].repos.push({ name: repoName, visibility });
    localStorage.setItem('users', JSON.stringify(users));
    document.getElementById('repoName').value = '';
    loadRepositories();
}

window.onload = () => {
    const savedUser = sessionStorage.getItem('currentUser');
    if (savedUser) {
        currentUser = savedUser;
        showSection('repositories');
    }
};
</script>

</body>
</html>

```

APPENDIX B (OUTPUT):



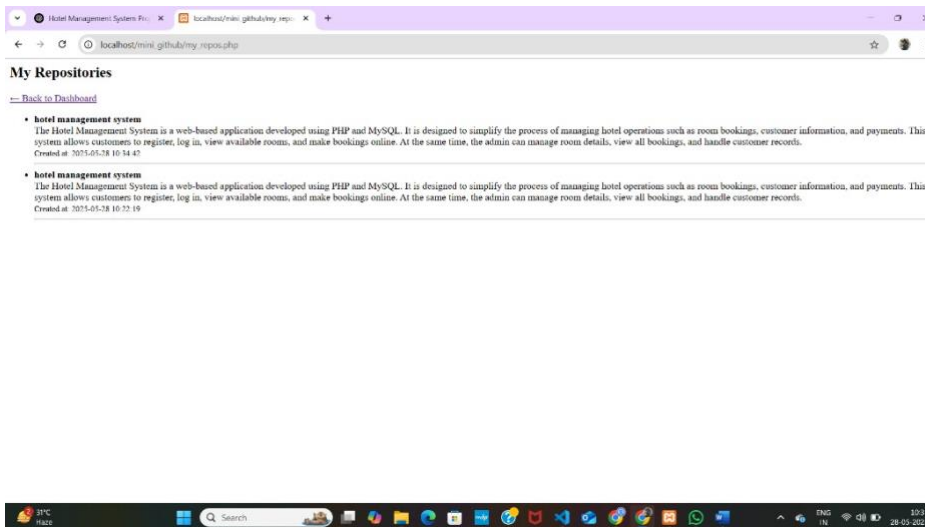
User Registration

Username:

Email:

Password:

Fig 4.1 Admin Login



My Repositories

[Back to Dashboard](#)

- hotel management system**
The Hotel Management System is a web-based application developed using PHP and MySQL. It is designed to simplify the process of managing hotel operations such as room bookings, customer information, and payments. This system allows customers to register, log in, view available rooms, and make bookings online. At the same time, the admin can manage room details, view all bookings, and handle customer records.
Created at: 2025-05-28 10:14:47
- hotel management system**
The Hotel Management System is a web-based application developed using PHP and MySQL. It is designed to simplify the process of managing hotel operations such as room bookings, customer information, and payments. This system allows customers to register, log in, view available rooms, and make bookings online. At the same time, the admin can manage room details, view all bookings, and handle customer records.
Created at: 2025-05-28 10:12:19

Fig 4.2 Repositories



Index of /mini_github/

Name	Last modified	Size	Description
Parent Directory	-	-	-
add_repo.php	2025-05-28 10:17	1.0K	
dashboard.php	2025-05-28 10:15	312	
db.php	2025-05-28 09:22	394	
login.php	2025-05-28 09:52	1.1K	
mv_repos.php	2025-05-28 10:23	1.0K	
register.php	2025-05-28 09:24	888	
repo.php	2025-05-28 10:30	1.0K	
test_db.php	2025-05-28 09:23	72	

Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.0.30 Server at localhost Port 80

Fig 4.3 Dashboard

REFERENCES:

1. **W3Schools – PHP, MySQL, and Web Development Tutorials –**
<https://www.w3schools.com>
2. **GeeksforGeeks – PHP & DBMS Programming Resources –**
<https://www.geeksforgeeks.org>
3. **PHP Manual – Official Documentation –**
<https://www.php.net/manual/en/>
4. **MySQL Documentation – Official MySQL Reference Manual –**
<https://dev.mysql.com/doc/>
5. **Bootstrap Documentation – CSS & JavaScript Framework –**
<https://getbootstrap.com/docs/>
6. **GitHub Docs – Understanding Repositories, Commits, and Git –**
<https://docs.github.com>
7. **Stack Overflow – PHP, Git, and Web Development Q&A –**
<https://stackoverflow.com>