A PROJECT REPORT ON

**A DOMAIN SPECIFIC SOCIAL WEBSITE**  
 **“CAR RENTAL”**

Submitted in partial fulfilment of the  
requirements for the award of the degree of

**BACHELOR OF TECHNOLOGY**  
**IN**  
**COMPUTER SCIENCE AND ENGINEERING**

**Under Guidance of**  
**Dr. Avadhesh Kumar Dixit**  
**Assistant Professor, Dept. Of CSE**

**Submitted By**  
Dheeraj Singh (21122)  
Tripti Tiwari (21160)  
Kaushlendra Pandey (21134)

**Department of Computer Science and Engineering**  
**INSTITUTE OF ENGINEERING AND TECHNOLOGY**  
**DR. RAM MANOHAR LOHIA AVADH UNIVERSITY**  
**AYODHYA (UP)**  
**JULY-2025**

**DECLARATION**

We hereby declare that the project work entitled **“Car Rental Site: A Domain Specific Social Website”** submitted to the **Department of Computer Science and Engineering, Institute of Engineering and Technology, Dr. Ram Manohar Lohia Avadh University, Ayodhya (U.P.), India**, is a record of an original work done by us under the guidance of **Dr. Avadhesh Kumar Dixit**, Assistant Professor, CSE.

This project work is submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering.

This project is designed to connect students and alumni of an educational institution.  
It provides seamless communication, community interaction, and resource sharing while ensuring data privacy and security.

We confirm that the results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma, and this work is our own, unless otherwise stated.

**Signature of Students  
1.**

**2.**

**3.**

**Date:**

**ACKNOWLEDGEMENTS**

We would like to express our deep gratitude to **Dr. Avadhesh Kumar Dixit**, our project guide, for his invaluable support, guidance, and encouragement throughout the course of this project.

We are also thankful to **Prof. S.S. Mishra (Director)** of **Dr. Rammanohar Lohia Avadh University, Ayodhya** for providing us the necessary resources and a conducive environment for our project.

Our sincere thanks to all our professors and staff members who have directly or indirectly helped us in this endeavor.

We extend our heartfelt appreciation to our families and friends for their continuous support, patience, and understanding during the project development period. Their encouragement has been a driving force behind our dedication and perseverance.

Lastly, we would like to acknowledge the contributions of our classmates and peers, whose collaboration and discussions have been immensely helpful in overcoming various challenges we faced during this project.

**Dheeraj Singh (21122)  
Tripti Tiwari (21160)  
Kaushlendra Pandey (21134)**

**CERTIFICATION**

This is to certify that the project report entitled **“Car Rental : A Domain Specific Social Website”** submitted by **Dheeraj Singh (21122)  
Tripti Tiwari (21160) Kaushlendra Pandey (21134)** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** of the **Institute of Engineering and Technology, Dr. Ram Manohar Lohia Avadh University, Ayodhya (U.P.), India**, is a record of bona fide work carried out by them under my guidance and supervision.

The results embodied in this report have not been submitted to any other University or Institution for the award of any degree or diploma.

**Date:**

(Signature)

**Dr. Avadhesh Kumar Dixit**  
(Project Guide)  
Department of Computer Science and Engineering  
**Institute of Engineering and Technology**  
**Dr. Ram Manohar Lohia Avadh University**  
 Ayodhya (U.P.), India

**TABLE OF CONTENTS**

1. **Introduction**  
   1.1 Project Overview ............................................... 7  
   1.2 Project Objective ................................................ 8
2. **Technologies Used**  
   2.1 Tools and Technologies ................................. 9
3. **System Design**  
   3.1 Requirements .................................................. 10  
   3.2 Architecture ..................................................... 11
4. **Implementation Details**  
   4.1 Development Approach ............................... 16  
   4.2 Features Implemented ................................. 18
5. **Testing**  
   5.1 Testing Approach ........................................... 21  
   5.2 Test Cases ....................................................... 23
6. **Future Work** ...................................................... 26
7. **Conclusion**  
   7.1 Achievements .................................................. 28  
   7.2 Lessons Learned ............................................. 29
8. **References** .......................................................... 30
9. **Appendices**  
   9.1 Appendix A: Code Snippets ...................... 31  
   9.2 Appendix B: Screenshots ............................. 38
10. **Introduction**

The purpose of this document is to outline the functional and non-functional requirements of the Car Rental System, which is designed to facilitate car rentals through an online platform. This system will allow customers to browse available vehicles, make reservations, and complete payments online, while providing administrators with tools to manage bookings, users, and the fleet efficiently.

**1.1 Project Overview**

This project traverses a lot of areas ranging from business concept to computing field, and required to perform several researches. General customers as well as the company’s staff will be able to use the system effectively.

**1.2**  **Objectives**

The Car Rental System is being developed for customers so that they can book their cars. This application takes information from the customers through filling their details. A customer being registered in the website has the facility to book a Car which he requires. The proposed system is completely integrated online systems. It automates manual procedure in an effective and efficient way. This automated system facilitates customer and provides to fill up the details according to their requirements.

1. **TECHNOLOGIES USED**

This section provides an overview of the specific tools and technologies employed in the development of the social media application.

**2.1 Tools and Technologies**

The development of this application leverages a robust tech stack to ensure functionality, scalability, and security:

* **Framework**:
  + **Node-JS**:
* **Frontend Technologies**:
  + **HTML**: Used for structuring the web pages and content presentation.
  + **CSS**: Stylesheets for enhancing the visual presentation and layout of the application.
  + **Tailwind CSS**: A utility-first CSS framework that helps in quickly building custom designs without writing traditional CSS.
* **Real-time Communication**:
  + **Pusher**: Integrated for real-time features like chat functionality and notifications, enhancing user interaction and engagement.
* **Email Handling**:
  + **SMTP**: Manages email functionalities such as user notifications and account verification, ensuring seamless communication with users.
* **Database Management**:
  + **MongoDB**: Chosen for its reliability, scalability, and efficiency in handling structured data, ensuring robust database management for the application.

**3. SYSTEM DESIGN**

This section provides a detailed overview of the design aspects essential for developing the social media application.

**3.1 Requirements**

The system design begins with a thorough analysis of the project requirements, ensuring alignment with user needs and technical specifications.

**3.1.1 Functional Requirements**

The functional requirements outline the core functionalities and capabilities expected from the social media application:

* **User Authentication**:  
  Users should be able to securely register, log in, and manage their accounts using email and password credentials. Password recovery and account verification mechanisms ensure user data integrity and accessibility.
* **User Profile Management**:  
  Enables users to create, update, and personalize their profiles with information such as name, contact details, educational background, and current status (student or alumni). Profile management enhances user engagement and community interaction.
* **Community Chat**:  
  Facilitates real-time messaging capabilities, including one-on-one and group chat functionalities. Users can engage in discussions, share updates, and collaborate within specific community groups.
* **Media Sharing**:  
  Supports the uploading, viewing, and sharing of multimedia content, including images, videos, and documents. Privacy settings allow users to control access to shared media among selected individuals or groups.
* **Knowledge Sharing**:  
  Provides a platform for users to publish and access educational resources, articles, links, and academic materials. A commenting system promotes discussions, feedback, and knowledge exchange among community members.

**3.1.2 Non-Functional Requirements**

Non-functional requirements focus on performance, security, scalability, and usability to ensure the application meets high standards of quality and user satisfaction:

* **Performance**:  
  The application should load within 3 seconds, optimizing user experience and accessibility. Real-time communication features must maintain a latency of less than 500ms to support seamless interaction and responsiveness.
* **Scalability**:  
  Designed to accommodate at least 10,000 concurrent users, with an architecture that supports future growth and increased user engagement. Scalability ensures the application remains robust and responsive under varying user loads.
* **Security**:  
  Implements stringent security measures, including data encryption in transit and at rest, secure authentication protocols, and access control mechanisms. Ensures user privacy, data integrity, and protection against unauthorized access and cyber threats.
* **Usability**:  
  Provides an intuitive and user-friendly interface, enhancing navigation, accessibility, and overall user experience. Accessibility features cater to diverse user needs, promoting inclusivity and ease of use across different devices and platforms.

**3.2 Architecture**

The architecture of the social media application is designed to support scalability, reliability, and efficient data management, leveraging modern technologies and frameworks.

**3.2.1 Architecture Overview**

The system architecture of the social media application is designed to ensure scalability, reliability, and security. The architecture leverages Laravel for the backend, HTML, CSS, and Tailwind CSS for the frontend, and MySQL for database management. SMTP is used for email management. Pusher is used for real-time communication and functionalities.

**3.2.2 System Components**

**a. Client-Side**

* **HTML/CSS/Tailwind CSS**: Used to create a responsive and user-friendly interface.
* **JavaScript**: Enhances user interaction and real-time features.

**b. Server-Side**

* **HTML/CSS/JS**: Serves as the backend framework, handling routing, authentication, and data processing.
* **Node-JS**: Scripting language used to manage server-side logic.
* **Pusher**: Enables real-time functionalities like chat and notifications.

**c. Database**

* **MongoDB**: Manages user data, posts, messages, and other relevant information.

**d. Communication**

* **SMTP**: Handles email notifications and password recovery.
* **Pusher**: Manages real-time communication between clients.

**3.2.3 Architectural Diagram**

The architectural diagram provides a visual representation of the system’s structure, illustrating the components and their interactions:

**3.2.4 Database Design**

The database architecture for our social media application is designed to manage user interactions, content sharing, and communication effectively. Below is a brief summary of the key tables and their relationships:

**Figure: System Architecture Flow**

[Frontend (HTML/CSS/Tailwind CSS/JavaScript)]

↓

[Backend (Node-JS)]

↓

[Database (MongoDB)]

↓

[Real-Time Communication (Pusher)]

↓

[Email Services (SMTP)]

**Key Tables in the Database:**

1. **Users Table**:  
   Stores user information including name, email, and password.
2. **Password Reset Tokens Table**:  
   Manages tokens for password reset functionality.
3. **Favourites Table**:  
   Keeps track of posts favourited by users.
4. **Likes Table**:  
   Records user likes on posts.
5. **Failed Jobs Table**:  
   Logs information about failed background jobs.
6. **Personal Access Tokens Table**:  
   Handles API authentication tokens for users.
7. **Followables Table**:  
   Manages user follow relationships (following and followers).
8. **Posts Table**:  
   Contains user-generated posts with content.
9. **Media Table**:  
   Stores media files associated with posts, such as images and videos.

10**. Comments Table**:  
Holds comments made by users on posts.

11. **Notifications Table**:  
Manages notifications sent to user.

**12. Conversations Table**:  
Represents user conversations for messaging.

**13. Messages Table**:  
Contains individual messages sent within conversations.

**Relationships**

* Users can create multiple Posts, Comments, Likes, and Favourites.
* Posts can have multiple Media files and Comments.
* Users can follow other users, managed through the Followables table.
* Conversations involve multiple Messages sent by users.

This architecture ensures efficient data management and supports various functionalities required for a robust social media platform. The relationships between tables are carefully designed to maintain data integrity and facilitate seamless user interactions.

Car Rental System

Admin

User(Booker)

List cars

**Browse and Book**

Invoice service

One time reg.

Register/Login

Contact us

Book Car

Rent car

**Figure: Table Relationships Diagram**

**3.2.5 Data Flow Diagram (DFD)**

A Data Flow Diagram (DFD) helps visualize the flow of data within the system. Below is an example DFD for the social media application:

[User]---🡪 [User Authentication]---🡪 [Profile Management]----🡪 [Post Creation]--🡪 [Media Sharing]---🡪[ Real-Time Chat]

[Database]

**4. IMPLEMENTATION DETAILS**

Specific Social App" follows a functionality and user experience are

**Technology Stack:**

**Front-End:**

**HTML:** Used for Structuring the web pages, ensuring a semantic and accessible layout. CSS: Utilized for styling the web pages, making the application visually appealing and user-friendly.

**Tailwind CSS:** A utility-first CSS framework that allows for rapid and flexible custom design, ensuring consistency across the application. Adds interactivity to the web pages, enhancing the user experience with dynamic content and features.

**Back-End**: A JS framework chosen for its elegant syntax, extensive libraries, and built-in features that accelerate development and ensure maintainability. PHP: The server-side scripting language that powers the back-end logic, handling data processing, server communication, and business logic implementation.

**MySQL:** A relational database management system used to store and manage user data, posts, comments, and other relevant information efficiently and securely.

Additional Services:

**SMTP:** Used for handling email notifications and verifications, ensuring reliable and secure communication with users.

**Pusher:** Provides real-time notifications and updates, ensuring users receive instant feedback on their activities and interactions within the application.

**Development Methodology:**

**Agile Development:** The Project is developed using agile methodologies, involving iterative cycles of planning, development, testing, and feedback. This approach allows for continuous improvement and adaptation based on user feedback and changing requirements.

**Version Control:** Git is used for version control, facilitating collaboration among team members and maintaining a history of changes made to the codebase. This ensures that all team members can work on different parts of the project simultaneously without conflicts.

**Code Quality**: Emphasis is placed on writing clean, modular, and maintainable code. Regular code reviews and adherence to coding standards ensure the quality of the codebase, making it easier to understand, extend, and debug. Implementation

**Implementation Phases:**

1. **Requirement Analysis**: Gathering and analyzing the requirements to understand the needs of the target users. This phase involves detailed discussions with stakeholders to define the scope, features, and objectives of the application.

2. **System Design**: Designing the architecture of the system, including the database schema, API endpoints, and user interface. This phase also involves creating wireframes and prototypes to visualize the user experience.

3. **Development:** Implementing the features as per the design specifications. This phase includes both front-end and back-end development, ensuring that all components work together seamlessly.

4. **Testing:** Conducting unit testing, integration testing, and user acceptance testing to ensure the application meets the requirements and functions as expected. Automated testing tools and manual testing are used to identify and fix any issues.

5**. Deployment**: Deploying the application to a live server, making it accessible to the users. This phase involves configuring the server environment, setting up databases, and ensuring that all components are correctly integrated.

**6. Maintenance:** Providing ongoing support and making necessary updates based on user feedback and technological advancements. This phase includes monitoring the application's performance, fixin as needed. g bugs, and adding new features 4.2 Features Implemented

**User Registration and Authentication:**

**Registration:** New users can create an account by providing their email, password, and other required information. The registration process includes \_ email verification to ensure the validity of user accounts.

**Login:** Registered users can log in to the application using their email and password. The login process is secured with measures like CAPTCHA and two-factor authentication to prevent unauthorized access.

**Password Reset:** Users can reset their password via email if they forget it. This feature includes security questions and token-based validation to ensure account security.

**User Profile Management**:

**Profile Editing**: Users can update their profile information, including their display name, profile picture, and bio. The profile editing interface is user- friendly and allows for easy customization.

**Privacy Settings:** Users can manage their privacy settings, such as who can view their profile and posts. Options include making the profile public, private, or visible only to followers.

**Post Creation and Interaction:**

» **Create Posts:** Users can create new posts with text, images, and videos. The post creation interface supports rich text formatting, hashtags, and mentions to enhance user engagement.

**Like, Comment, and Share:** Users can interact with posts by liking, and nd sharing them with their followers, Real-time updates ensure that users are notified of interactions immediately. Commenting.

**Messaging System:**

**Send and Receive Messages:**

Users can send and receive direct messages with other users. The messaging system supports text, images, and emojis, providing a rich communication experience.

**Group Conversations:** Users can create and participate in group conversations. Group chats include features like adding/removing members, . customizing group names, and sharing media files.

**Notification System:** Real-Time Notifications: Users receive real-time notifications for activities such as likes, comments, new followers, and direct messages. The notification “system ensures that users are always aware of important activities.

**Email Notifications:** Users receive email notifications for significant activities and updates. Email notifications include customization options for frequency and types of notifications.

**Follow System:**

**Follow/Unfollow Users**: Users can follow or unfollow other users to see their posts in their feed. The follow system includes suggestions for new users to follow based on interests and interactions.

**Followers and** **Following Lists:** Users can view lists of their followers and the users they are following. The lists include options to manage followers and view mutual connections.

**Security and Privacy:**

**Data Encryption:** Sensitive user data, such as passwords, is encrypted to ensure security. The encryption methods used comply with industry standards and best practices.

**Secure Authentication**: Authentication processes are secured to prevent unauthorized access. Security measures include token-based authentication, SSL certificates, and regular security audits. Data Privacy: User data is handled in compliance with data privacy regulations to ensure user confidentiality. The application includes features for users to download and delete their data. These features are implemented with a focus on providing a seamless and intuitive user experience while maintaining high standards of security and performance. The application is designed to be scalable and adaptable, allowing for future enhancements and integrations.

**5. TESTING**

Testing phase is crucial to ensure that the application. The following testing approach is adopted to achieve a high-quality and reliable.

**Types of Testing:**

**1. Unit Testing:**

Objective: Verify the correctness of individual components or units of code. Method: Each function, method, or class is tested in isolation using automated test scripts. This ensures that the smallest parts of the application are functioning correctly.

2. **Integration Testing:**

Objective: Ensure that different modules or components of the application work together as expected. Method: Multiple units are combined and tested as a group. This helps identify issues related to the interaction between different parts of the system.

3. **System Testing:**

Objective: Validate the complete and integrated software to ensure it meets the specified requirements. Method: The entire application is tested as a whole in an environment that mimics production. This includes testing all functionalities, user interfaces, and interactions.

4**. User Acceptance Testing (UAT):** Objective: Confirm that the application meets the business requirements and is ready for deployment. Method: End-users test the application in real-world scenarios to ensure it performs as expected. Feedback from users is collected and used to make final adjustments.

5. **Performance Testing:**

**Test Execution**: Execute the test cases

Objective: the application's Performance under vari conditions. Method: The application is tested for load, stress, and scalability. This ensures the application can handle high traffic and data processing loads without degradation in performance

**6. Security Testing**: o Objective: Identify vulnerabilities and ensure the application is secure from threats.

Method: Penetration testing, vulnerability scanning, and code reviews are conducted to identify and mitigate security risks. This includes testing for SQL injection, XSS, CSRF, and other common vulnerabilities.

**Testing Tools:** .

-Unit Testing: Jest, PHPUnit

Integration Testing: Cypress, Postman

System Testing: Selenium, TestRail

Performance Testing: JMeter, LoadRunner

Security Testing:

**Testing Process:**

**Requirement Analysis:** Understand the requirements and define the scope of testing. Identify the testable features and functionalities.

**Test Planning**: Develop atest plan that outlines the testing strategy, resources, schedule, and deliverables.

**Test Case Design**: Create detailed test cases based on the requirements and design documents. Each test case includes input data, execution steps, and expected results.

**Test Environment Setup**: Prepare the testing environment, including hardware, software, and network configurations. Ensure the environment . simulates the production environment and record the results.

**Test Execution:** Identify and report defects using a defect tracking tool. resolved defects to ensure they are fixed,

**7. Test Closure**: Conduct a final review of the testing process, Prepare a test ummary report that includes the test results, defects, and lessons.

**5.2 Test Cases**

**User Registration and Authentication:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Profile Management:**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Post Creation and Interaction:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**6. Future Works:**

As "CONNECTIFY" continues to grow, there are several key areas for future improvements to enhance the platform’s functionality and user experience. Below are some suggestions:

6.1 Mobile Application Development

* Objective: To make "CONNECTIFY" accessible on mobile devices.
* Details: o Develop mobile applications for both iOS and Android. o Ensure that the mobile app syncs seamlessly with the web version.

6.2 Advanced Analytics

* Objective: To provide insights into user activity and engagement.
* Details: o Implement dashboards to track metrics such as active users and popular posts. o Enable reporting tools for administrators to monitor community engagement.

6.3 Enhanced Personalization

* Objective: To tailor content to individual user preferences.
* Details: o Use algorithms to recommend posts and connections based on user behavior. o Offer personalized feeds and suggestions.

6.4 Integration with Other Platforms

* Objective: To connect "CONNECTIFY" with other popular platforms.
* Details: o Allow users to link their profiles with social media accounts like LinkedIn.

6.5 Gamification

* Objective: To increase user engagement through game-like features.
* Details: o Introduce badges, points, and leaderboards to reward active participation. o Design challenges and interactive events to encourage community involvement.

6.6 Enhanced Security

* Objective: To protect user data and privacy.
* Details: o Update security protocols regularly. o Implement advanced encryption and multi-factor authentication. o Conduct regular security audits to ensure data protection.

6.7 Expanding Community Features

* Objective: To foster a supportive community environment.
* Details: o Introduce mentorship programs and career advice forums. o Enable tools for organizing events like webinars and reunions. o Develop project spaces for collaborative work.

By focusing on these areas, "CONNECTIFY" can continue to evolve, providing a valuable platform for connecting students and alumni while meeting the changing needs of its users.

**7 CONCLUSION**

The development and implementation of **"Car Rental System: A Domain Specific Social App"** have provided valuable insights and experiences. This section highlights the achievements and lessons learned throughout the project.

**7.1 Achievements**

* **User-Friendly Interface**:
  + Developed an intuitive and easy-to-navigate interface that enhances user experience.
  + Ensured accessibility for all users, including those with limited technical skills.
* **Feature-Rich Platform**:
  + Implemented core features such as community chat, media sharing, and knowledge resource access.
  + Integrated advanced functionalities like real-time notifications and privacy controls.
* **Technological Integration**:
  + Successfully utilized technologies like Laravel, HTML, CSS, Tailwind CSS, PHP, SMTP, Pusher, and MySQL.
  + Achieved a scalable and high-performance application architecture.
* **Community Building**:
  + Fostered a sense of community among students and alumni, facilitating networking and collaboration.
  + Enabled users to share knowledge and resources, promoting continuous learning.
* **Data Privacy and Security**:
  + Implemented robust security measures to ensure data privacy and protection.
  + Adhered to best practices in secure coding and data handling.
* **Scalability and Performance**:
  + Designed the application to handle growing user numbers and data volumes.

**7.2 Lessons Learned**

* **Project Management**:
  + Gained experience in managing a full-stack development project from concept to deployment.
  + Learned the importance of setting realistic timelines and milestones.
* **Team Collaboration**:
  + Improved communication and coordination within the development team.
  + Recognized the value of diverse skill sets and perspectives in achieving project goals.
* **Technical Challenges**:
  + Overcame various technical challenges related to integrating multiple technologies.
  + Enhanced problem-solving skills through debugging and optimizing code.
* **User Feedback**:
  + Understood the importance of incorporating user feedback into the development process.
  + Adapted features and functionalities based on user needs and preferences.
* **Continuous Learning**:
  + Realized the need for ongoing learning and adaptation to new technologies and trends.
  + Committed to staying updated with the latest advancements in web development and security.
* **Documentation and Testing**:
  + Emphasized the significance of thorough documentation and comprehensive testing.
  + Ensured that all aspects of the application were well-documented and rigorously tested.

**References**

**Appendices**