NATIONAL UNIVERSITY
of Computer & Emerging Sciences

PROJECT PROPOSAL

Project Title: Ride Revive - Bringing your car back to life with doorstep services.

Course: Fundamental of Software Engineering - FSE

Course Instructor: Ms. Sobia Iftikhar

PROJECT MEMBERS:

MUHAMMAD OWAIS RAFIQ (23K-2042)

ABUBAKAR BIN HASSAN (23K-2025)

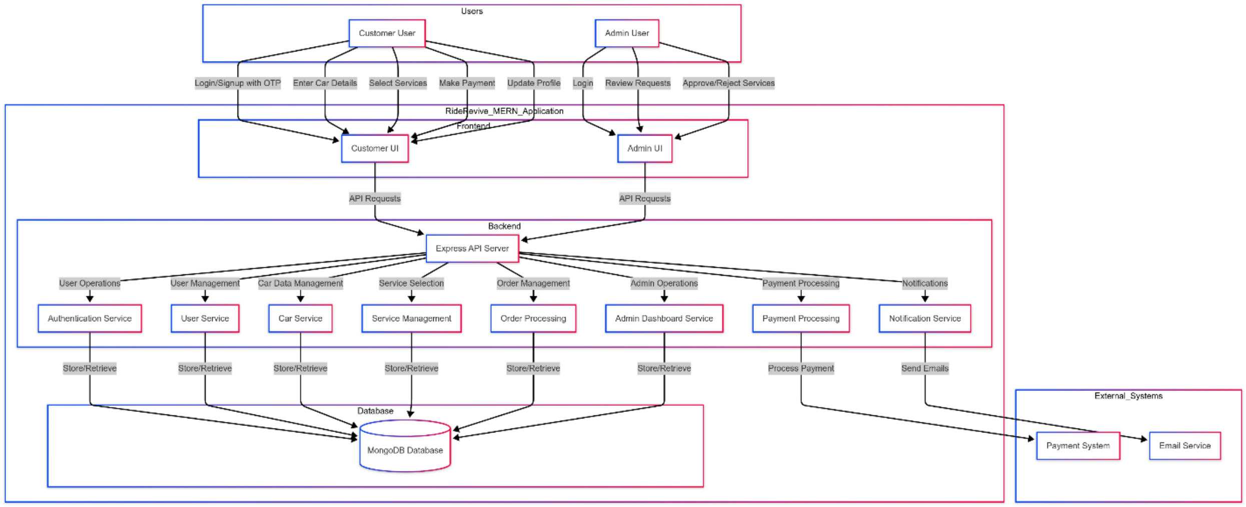
MUHIB ALI ZAIDI (23K-2030)

RideRevive Project Proposal

Web App Car Servicing Application

University Project Proposal | MERN Stack Implementation

Context Diagram



Link:
https://drive.google.com/file/d/19rPVCnpTAlN9DTY9rb8IbyUbg5jQ33YG/view?usp=sharing

---

Executive Summary

RideRevive aims to revolutionize vehicle maintenance and emergency services by providing a comprehensive web app application that brings essential automotive services directly to the customer's doorstep. As vehicle ownership continues to rise among busy professionals, the demand for convenient, time-saving automotive solutions has never been greater. RideRevive bridges this gap with its integrated platform offering fuel delivery, car washing, tire changes, emergency towing, battery services, and engine oil maintenance—all accessible through a streamlined web app interface.

Developed using the industry-standard MERN stack (MongoDB, Express.js, React.js, Node.js), RideRevive demonstrates practical application of full-stack development concepts while solving a real-world problem. This university project showcases database design, authentication systems, payment integration, and administrative functionality within a cohesive application framework.

Project Vision

"Bringing your car back to life with doorstep services."

RideRevive envisions a world where vehicle maintenance no longer disrupts your daily schedule. Our application eliminates the need to visit service stations by delivering professional automotive care whenever and wherever you need it.

Problem Statement

Modern vehicle owners face several challenges in maintaining their vehicles:

- **Time constraints** - Traditional service centers operate during business hours when most people are working
- **Inconvenient locations** - Service centers are often located in industrial areas, requiring significant travel time
- **Emergency situations** - When a vehicle breaks down, finding immediate assistance can be challenging
- **Scheduling conflicts** - Coordinating service appointments with busy personal schedules causes delays in necessary maintenance

RideRevive addresses these pain points by bringing services directly to customers on their schedule.

Target Audience

- **Busy professionals** with limited time for vehicle maintenance
- **Emergency situations** requiring immediate assistance
- **Customers without transportation** alternatives when their vehicle needs service
- **Urban dwellers** with limited access to traditional service centers
- **Vehicle owners** seeking convenience and efficiency in routine maintenance

---

Technical Requirements

System Architecture

The application will be built using the MERN stack:

- **MongoDB**: NoSQL database for storing user profiles, vehicle information, service requests, and order history
- **Express.js**: Backend framework for creating robust API endpoints and handling business logic
- **React.js**: Frontend library for building responsive and dynamic user interfaces
- **Node.js**: Runtime environment for executing server-side JavaScript code

Hardware Requirements

Development Environment

- Computers with minimum 8GB RAM and quad-core processors
- Development environments supporting JavaScript, MongoDB, and web technologies
- Testing devices (various mobile platforms)

Software Requirements

Development Tools

- Visual Studio Code or similar IDE
- Git for version control
- MongoDB Compass for database management
- Postman for API testing

Libraries and Frameworks

- React for Web Application development
- Express.js for API development
- Mongoose for MongoDB object modeling
- JWT for authentication
- Nodemailer for email notifications

Features & Functionality

Core Features

User Authentication System

- Secure signup and login process
- OTP-based verification

Vehicle Management

- Adding multiple vehicles to user profile
- Vehicle specifications storage (make, model, year, etc.)

Service Booking

1. **Fuel Delivery**
   - Fuel type selection
   - Quantity specification
   - Location selection
2. **Car Washing**
   - Interior/exterior/complete options
   - Scheduled or on-demand booking
3. **Tire Services**
   - Tire change
   - Pressure check
   - Puncture repair
4. **Emergency Rescue**
   - Towing services
   - Jump start
5. **Battery Services**
   - Battery replacement
   - Charging services
   - Health diagnostics
6. **Engine Oil Services**
   - Oil change
   - Filter replacement
   - Quality options

Payment Processing

- Invoice generation

Admin Dashboard

- Request management
- Service approval workflow
- Customer management

Notifications System

- Booking confirmations
- Payment receipts

Workflow

User Journey

1. **Account Creation & Authentication**
    - User access the RideRevive application through browser
    - User creates an account with basic information
    - OTP verification confirms email
    - Authentication credentials are stored securely in MongoDB
2. **Vehicle Information**
    - User inputs vehicle details (brand, model, year, etc.)
    - Information is stored in user profile for future service requests
3. **Service Selection**
    - User navigates service categories
    - Selects specific service required (e.g., fuel delivery)
    - Provides service-specific details (e.g., fuel type, quantity in liters)
4. **Location & Scheduling**
    - Current location is detected or user inputs preferred location
    - User selects preferred service window (immediate or scheduled)
5. **Review & Payment**
    - Order summary displays service details and pricing
    - User enters billing information
6. **Admin Approval**
    - Request enters admin queue for review
    - Admin approves valid service requests
7. **Service Confirmation**
    - User receives confirmation email with service details
8. **Service Delivery**
    - Service is performed at the specified location

Admin Workflow

1. **Admin Authentication**
    - Secure login to administrative portal
    - Role-based access control
2. **Request Management**

  - Dashboard displays pending service requests
  - Requests are sorted by priority and time
  - Request details include customer information, service type, and location
3. **Request Processing**
   - Admin reviews request details
   - Validates service feasibility and resource availability
   - Approves or declines request with reason
4. **Notification Triggering**
   - System automatically generates customer notifications
   - Confirmation emails are sent upon approval
   - Explanatory emails for declined requests

Technology Stack

Frontend

- **React.js**: Component-based UI development
- **React Router**: Navigation management
- **Axios**: HTTP client for API requests
- **Regex**: Form validation

Backend

- **Node.js**: Server-side JavaScript runtime
- **Express.js**: Web application framework
- **MongoDB**: NoSQL database
- **Mongoose**: MongoDB object modeling
- **JWT**: Authentication tokens
- **bcrypt**: Password hashing
- **Nodemailer**: Email service integration

Development Tools

- **Git/GitHub**: Version control and collaboration
- **Postman**: API testing
- **MongoDB Compass**: Database management
- **ESLint/Prettier**: Code quality and formatting

Deployment

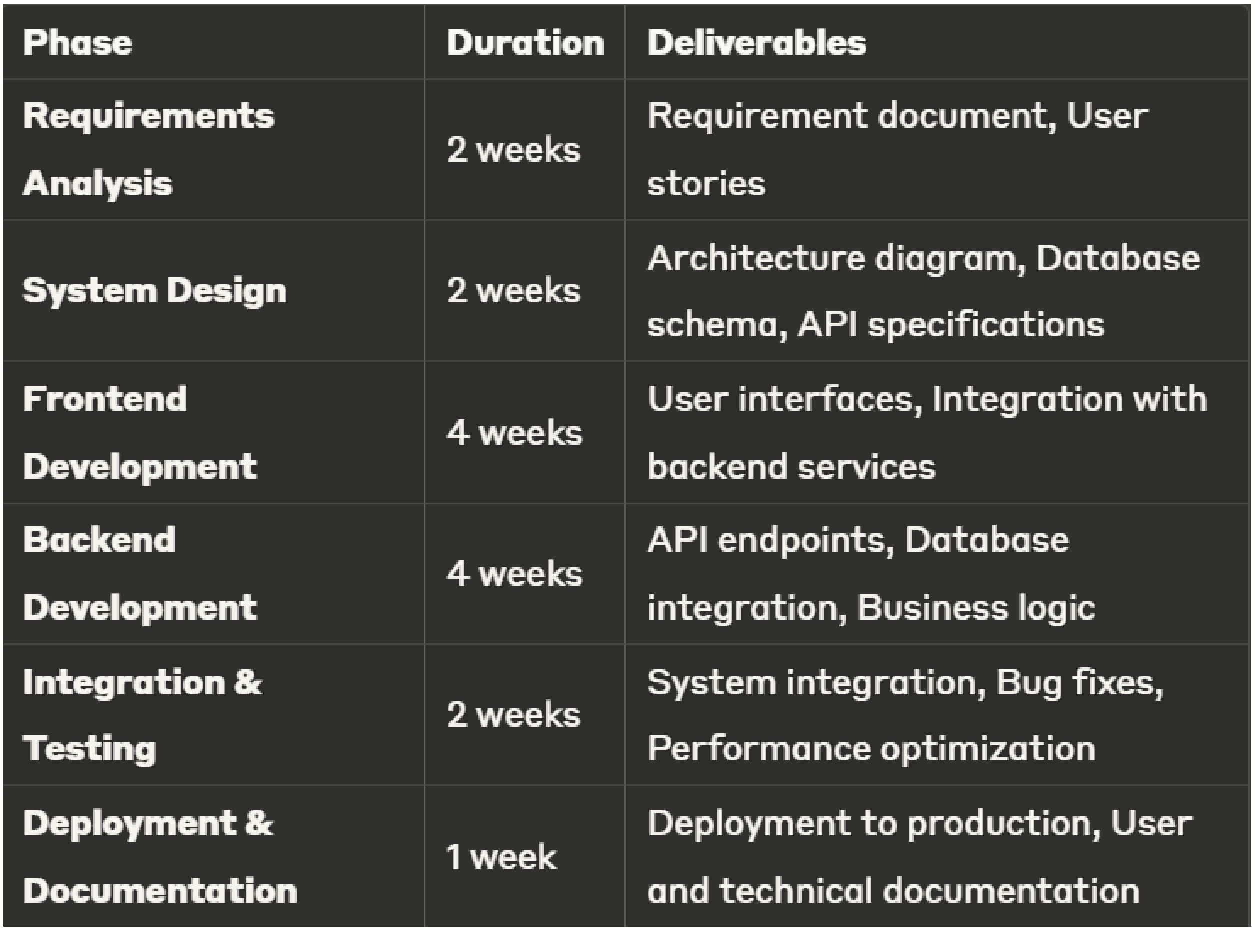
- **Vercel**: Application hosting

- **MongoDB Atlas**: Cloud database
- **GitHub Actions**: CI/CD pipeline

Security Considerations

- **Authentication**: Secure OTP verification and password hashing
- **Authorization**: Role-based access control
- **Password Encryption**: Secure transmission of sensitive information
- **Session Management**: Secure token handling
- **API Security**: Rate limiting and request validation

Project Timeline



Phase	Duration	Deliverables
Requirements Analysis	2 weeks	Requirement document, User stories
System Design	2 weeks	Architecture diagram, Database schema, API specifications
Frontend Development	4 weeks	User interfaces, Integration with backend services
Backend Development	4 weeks	API endpoints, Database integration, Business logic
Integration & Testing	2 weeks	System integration, Bug fixes, Performance optimization
Deployment & Documentation	1 week	Deployment to production, User and technical documentation

Learning Outcomes

Through this university project, team members will gain practical experience in:

- Full-stack development using the MERN stack
- Building secure authentication systems
- Implementing payment processing in web applications
- Designing and implementing database schemas for real-world applications
- Creating and managing an administrative interface
- Developing notification systems using email
- Implementing role-based access control
- Project management and teamwork in software development

Conclusion

RideRevive represents a comprehensive application of modern web development techniques to solve a practical problem in the automotive service industry. By leveraging the MERN stack, the project will demonstrate the ability to create a fully-functional, secure, and user-friendly application that connects vehicle owners with essential services.

The project provides an excellent opportunity to apply theoretical knowledge to a practical scenario while developing valuable skills in full-stack development, database design, user authentication, and project management. Upon completion, RideRevive will serve as both a testament to the team's technical abilities and a potential foundation for future commercial development.