UBER

Data Analysis for Price Prediction

A Comprehensive Full Stack Development.

Team Members

Umesh

Deepak Sangeetham

Harsha Vardhan

Sai Varshith

Vijaya Bhanu

Introduction

We covered essential aspects of modern software development, from frontend implementation to backend design, API development, security considerations, CI/CD practices, and proper version control workflows for UBER Price prediction.

Project Statement and UX Design

Problem

Analyzed Uber data for price prediction. Develop a full-stack application supporting data analysis. Ensure scalability, security, and performance.

UX Design

Designed user-friendly interface for Uber Fair Prediction. Implemented CRUD operations. Integrate features for filtering, sorting data.

Implementation

FRONT-END

React.Js

BACK-END

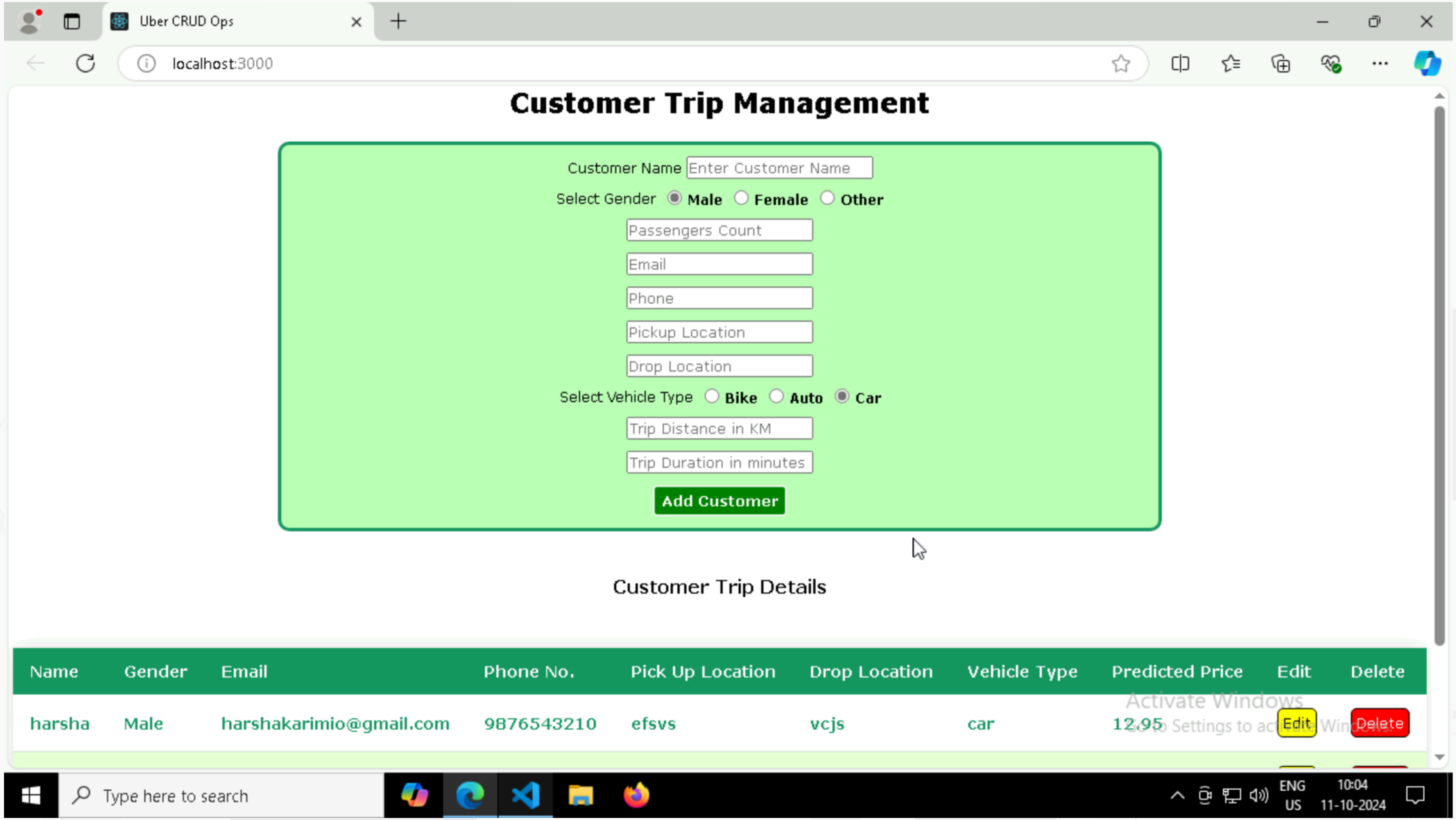
Express.Js,Flask

DATABASE

No SQL
(MongoDB)

Frontend Implementation

- Designed a user-friendly interface using HTML, CSS, and JavaScript
- Implemented a responsive design for desktop.
- Used library’s like React for building the frontend.


Uber CRUD Ops
localhost:3000
Customer Trip Management
Customer Name
Enter Customer Name
Select Gender
Male
Female
Other
Passengers Count
Email
Phone
Pickup Location
Drop Location
Select Vehicle Type
Bike
Auto
Car
Trip Distance in KM
Trip Duration in minutes
Add Customer
Customer Trip Details
Name
Gender
Email
Phone No.
Pick Up Location
Drop Location
Vehicle Type
Predicted Price
Edit
Delete
harsha
Male
harshakarimio@gmail.com
9876543210
efsvs
vcjs
car
12.95
Edit
Delete
Activate Windows
Type here to search
ENG
US
10:04
11-10-2024

Backend Architecture

- Choose MongoDB for storing Uber data.
- Created an efficient schema for Uber ride information.
- Implemented indexing for faster data retrieval.
- Implemented API gateways for secure API access.

API Design

- Included API endpoints, request/response formats, and error handling
- Used API keys or authentication for secure API access
- Design RESTful APIs for data retrieval and manipulation.

CI/CD Pipeline

- Used a CI/CD tool like GitLab CI/CD.
- Implement unit testing and integration testing for the application.
- Used Dockerization for containerization and deployment.

GitLab Branching

- Used a CI/CD tool like GitLab CI/CD.
- Used feature branches for development and testing.
- Used merge requests for code reviews and approvals.
- Used the main branch for production deployment.

Performance Optimization

Performance Optimization

Implemented code splitting, lazy loading, and efficient state management. Optimized API and database queries.
Used CDNs for static asset delivery.
Monitored real-time performance metrics.

Security Measures

Conducted regular vulnerability assessments.
Implemented secure coding practices, HTTPS, and proper authentication.
Kept dependencies updated and used security monitoring tools.

Continuous Monitoring

Set up alerts for performance degradation and security issues.
Regularly reviewed logs and metrics.
Implemented automate security scans in the CI/CD pipeline.

User Data Protection

Implemented encryption for sensitive data at rest and in transit.
Followed data protection regulations like GDPR.
Regular backup and test data recovery procedures.

Conclusion

In this comprehensive full stack implementation, we have designed and implemented a robust and scalable application for Uber data analysis and price prediction. The application includes a user-friendly frontend, a robust backend API, and a secure and performant database. The CI/CD pipeline ensures that the application is built, tested, and deployed automatically, while the security measures ensure that the application is secure and protected from unauthorized access.

Thank You