Project Report

on

Carpooling System (CapX)

Submitted to

Sant Gadge Baba Amravati University

In partial Fulfillment of the Requirement

For the Degree of

Bachelor of Engineering in

Computer Science and Engineering

Submitted by:

Pakhi Mujmer

Yash Dalal

Tanay Shah

Thavar Sethiya

Sumit Kumar Sethi

Under the Guidance of

Prof. V.S. Mahalle

Department of Computer Science and Engineering

Shri Sant Gajanan Maharaj College of Engineering,

Shegaon – 444 203 (M.S.)

2022-2023

SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING,

SHEGAON – 444 203 (M.S.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that **Ms. Pakhi Mujmer, Ms. Siddhi Mehta, Mr. Yash Dalal, , Mr. Tanay Shah Mr. Thavar Setiya,** and **Mr. Sumit Sethi,** students of final year B.E. in the year 2020-21 of Computer Science and Engineering Department of this institute has completed the project work entitled **"Carpooling System(CapX)"** based on syllabus and has submitted a satisfactory account of his work in this report which is recommended for the partial fulfillment of degree of Bachelor of Engineering in Computer Science and Engineering.

[illegible]
31/5/2023

Prof. V. S. Mahalle
Project Guide

[illegible]
31/5/23

Dr. S. B. Patil
Head of Department

[illegible]

Dr. S. B. Somani
Principal

SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING, SHEGAON – 444 203 (M.S.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled "**Carpooling System(CapX)**" submitted by **Ms. Pakhi Mujmer, Mr. Yash Dalal, Mr. Tanay Shah, Ms. Siddhi Meheta, Mr. Thavar Sethiya, Mr. Sumit Kumar Sethi** students of final year B.E. in the year 2020-21 of Computer Science and Engineering Department of this institute, is a satisfactory account of his work based on syllabus which is recommended for the partial fulfillment of degree of Bachelor of Engineering in Computer Science and Engineering.

Internal Examiner	**External Examiner**
Date:	**Date:**

ABSTRACT

Carpooling (also known as car sharing, ride sharing and lift sharing), is the sharing of car journeys so that more than one person travels in a car. Carpooling reduces each person’s travels costs such as fuel costs, tolls, and the stress of driving. Carpooling is one method that can be easily instituted and can help resolve a variety of problems that continue to plague urban areas, ranging from energy demands and traffic congestion to environmental pollution. Authorities often encourage carpooling, especially during high pollution periods and high fuel prices. We intent on making an ANDROID based application that will enable to let people know if vehicles are available for carpool in their desired path they can sign in for it.

This will enable people using this application to share expense, not worry about hiring a cab and making new connections. People having this application on their cell phone with advance facilities can easily carpool with unacquainted people without worrying about security. It will also helpful for blind or lack of knowledge of using gadgets such a people they can operate this application using speech reorganization technique. It will show the accurate time requires to reach at particular location. It gives a better way for pooling a car with a very efficient environment that is easy to use. This is a Web-based collaboration, communications, and content delivery framework.

Keywords — carpool, traffic, route, travel, private, public, location.

ACKNOWLEDGEMENT

The real spirit of achieving a goal is through the way of excellence and lustrous discipline. I would have never succeeded in completing our task without the cooperation, encouragement and help provided to me by various personalities.

We would like to take this opportunity to express our heartfelt thanks to my guide ***Prof. V.S. Mahalle****, for his esteemed guidance and encouragement, especially through difficult times. His suggestions broaden our vision and guided us to succeed in this work. We are also very grateful for his guidance and comments while studying part of our project and we learnt many things under his leadership.*

We extend our thanks to ***Dr. S. B. Patil,*** *Head of Computer Science & Engineering Department, Shri Sant Gajanan Maharaj College of Engineering, and Shegaon for the invaluable support that made us consistent performer.*

We also extend our thanks to ***Dr. S. B. Somani****, Principal, Shri Sant Gajanan Maharaj College of Engineering, and Shegaon for his valuable support.*

Also, we would like to thanks to all teaching and non-teaching staff of the department for their encouragement, cooperation and help. Our greatest thanks are to all who wished us success especially our parents, our friends whose support and care makes us stay on earth.

Ms. Pakhi Mujmer
Ms. Siddhi Meheta
Mr. Yash Dalal
Mr. Tanay Shah
Mr. Thavar Sethiya
Mr. Sumit Kumar Sethi

Session 2022-23

CONTENTS

List of Figures and Tables

List of Snapshots

Abbreviations

CNN	Convolutional Neural Network
DNN	Deep Neural Network
PSAI	Proctoring System using AI
BP	Back Propagation
GD	Gradient Descent
MSE	Mean Square Error
RL	Reinforcement Learning
tanh	Hyperbolic Tangent
ReLU	Rectified linear Unit

CHAPTER 1
INTRODUCTION

1.1 Preface

This carpooling project, known as CapX, leverages the power of the MERN Stack technology. MERN, an acronym for Mongo DB, Express.js, React.js, and Node.js, offers a robust and efficient framework for developing web applications. With CapX, our primary objective is to create a comprehensive carpooling app that enables convenient ride-sharing among commuters. By providing a user-friendly platform, we aim to simplify the process of registering as either a driver or a passenger. Users will be able to create profiles, specify their travel preferences, and connect with other individuals traveling on the same route. To enhance the overall user experience, CapX incorporates various features. One such feature is a secure payment system that ensures smooth and efficient transactions between users. This eliminates the need for cash exchanges and streamlines the financial aspect of the carpooling experience. Furthermore, CapX prioritizes user safety by implementing a rating system and user verification mechanisms. The rating system allows users to provide feedback on their experience with drivers and passengers, promoting accountability and trust within the community. User verification adds an extra layer of security by confirming the identities of individuals participating in the carpooling system. By promoting carpooling, CapX aims to address common commuting challenges while contributing to a more sustainable future. By reducing the number of private vehicles on the road, we can mitigate traffic congestion, decrease carbon emissions, and promote environmentally friendly transportation alternatives. Overall, CapX represents a comprehensive solution that combines technology, convenience, and environmental consciousness to improve the commuting experience for users and contribute to a greener world.

1.2 PROBLEM STATEMENT

The problem of traffic congestion and its associated challenges, such as increased travel time, fuel consumption, and environmental pollution, has become a critical issue in many urban areas. Private vehicles, often carrying only one or a few passengers, contribute significantly to the problem. Carpooling, the practice of sharing rides with others who are traveling in the same direction, has emerged as a potential solution to address these challenges and promote sustainable transportation.

1.3 AIM and OBJECTIVE

AIM:

The aim of the carpooling System project is to develop a user-friendly, efficient, and secure platform for ride-sharing that promotes sustainable and eco-friendly transportation, reduces traffic congestion, and enhances the overall commuting experience for users.

OBJECTIVES:

- To develop a carpooling app that enables users to register as drivers or passengers and provides a user-friendly interface for creating and managing ride requests and offers.
- To incorporate features that ensure the safety and security of all users.
- To provide a scalable and robust platform that can handle a large user base and high volume of ride requests efficiently, while ensuring optimal server performance and data management.
- To continuously improve and innovate the app by gathering user feedback, analyzing performance metrics, and incorporating new features and technologies that enhance the carpooling experience for users.
- By achieving these objectives, the Carpooling System project aims to provide a sustainable and efficient transportation solution that benefits commuters, reduces traffic congestion and environmental pollution, and contributes to the development of smart and live able cities.

1.4 ORGANIZATION OF PROJECT

The purpose of the introduction chapter is to provide an overview of the entire thesis, summary of the project and its future scope. It also tells about the basic architecture and implementation of the project.

- It contains the methodology of the system, tells about the existing system and our proposed system.
- It contains the aspects of system analysis and feasibility study which tells about the how much the developed application has met requirements.
- It explain about the different modules involved in the system and its basic overview.
- It elaborates the design process used for the system.
- It shows class diagram of the proposal system gives the brief explanation of hardware, Software and application requirements.
- It deals with implementation part of the system.
- It contains the various Frameworks implemented in our application.
- It concludes our project work and gives an idea about future scope of the project.

CHAPTER 2
LITERATURE SURVEY

2. LITERATURE SURVEY

This project includes various technologies like Mongo DB, Express.js, React.js, and Node.js. So, for detailed study of these technologies and algorithms following papers are reviewed:

Sr. No.	Title	Author and Year of the paper	Limitations
1	Peer-to-peer car sharing system,	Rutuja Pharande, Prof. Neha Sharma, Shubhangi Gunjal and Abhishek Mahale Year: 2022(Dec)	This System Focused on a decentralized Peer-to-peer automobile sharing which is implemented to aid with transportation issues in urban areas by reducing traffic congestion. Two kinds of stakeholders are identified for this D-App i.e., the driver and the rider. Each user has different functions and responsibilities that are offered using various dashboards of the D-App.
2	UCarpooling: decongesting traffic through carpooling using automatic pairings,	Alejandro Lugo, Nathalie Aquino, Magalí González, Luca Cernuzzi Year: 2021(July)	This proposed System UCarpooling: decongesting traffic through carpooling using automatic focuses on pairings which Both the back end and the front end are included in the design and modelling of such systems; however, in this study, authors primarily concentrated on the back end while leaving the front end for future work. make it easier for those who

			frequent the same institution or are coworkers to carpool.
3	real-time demand-aware ridesharing service	Yueshen Xu, Yuqiao Liao, Jianbin Huang and Ying L Year: 2021	a real-time demand-aware ridesharing service which is developed with the aim to give the quality-of-service. When users submit a request in devices, it will find a car on the user's way only. It is also made to find the most appropriate route.
4	VISHWA-CONNECT: A Ride Sharing Mobile Application for Campus Students	Kaushalya Thopate, Mahesh Sathe, Saurav Gujar, Sarthak Honmute, Pushkraj Savji, Vinayak Sawandkar, Satvik Vishnoi Year: 2022(Nov)	This mobile application is designed for Campus Students using React native, JavaScript, Firebase and VS Code. Carpooling is a great way to travel. Also it has more benefits. Also provide more comfortable and time saving way to travel. Ride sharing also leads to lesser pollution. It is best for college students/employees to travel.This application is limited for some geographical area.

5	mT-Share: A Mobility-Aware Dynamic Taxi Ridesharing System	Zhidan Liu, Zengyang Gong, Jiangzhou Li, Kaishun Wu Year: 2022	This platform uses information of taxis and ride requests to get efficient sequence of taxis requests and better passenger–taxi matching, It uses of both geographical information and travel directions to match taxis and ride requests and provides the shortest path. It also gives a novel payment model to share the ridesharing benefits among the taxi driver and passengers.
6	UiTM Share Ride: Requirements Validation, Design and Development of a Campus Ride-Sharing Mobile Application	Kamaruddin, Kamalia Azma, and Nur Rozliana Mohd Rozlis. Year: 2020(April)	This platform is designed for reducing the number of cars on college campuses, which will improve parking spaces issues, also it is cost saving for students and reduce fuel emissions for cleaner and healthier environment too. In order to address this urgent problem on many campuses, the creation of a campus ride-sharing smartphone application will be a good medium for university personnel and students.

7	TMATCH: A New Framework for Supporting Car-Sharing Under Carpooling Model	Xiufeng Xia, Zhenchang Hu, Rui Zhu, Jiajia Li, Chuanyu Zong, Xiangyu Liu, Year: 2021	This paper studies the problem of distributed route matching over road network. To send tasks to different servers , we had created a random server selection algorithm. We had made a novel index to maintain tasks in every server. Last of all, we discuss how to execute tasks matching over road network.
8	VoIP integration for mobile ride-sharing application	Amasyali, Mustafa Burak, and Ensar Gul Year: 2017(Nov)	Car pooling or ride sharing applications have become feasible due to advances in mobile technology. There is a need to enable communication between riders and drivers by keeping their personal details secure. This paper provides a solution named VoIP, which provides a platform for a voice calls between drivers and riders by hiding their mobile number that is secure too
9	A Dynamic Carpooling System with Social Network Based	Sasikumar C, Jaganathan S Year : 2017	The system outlines the usage and usefulness of carpooling and also makes it more attractive by enabling users to share their ride with their friends and relatives and restricts others. This feature

	Filtering		encourages carpooling in less developed areas of the world where possession of a car is not very common.
10	B-ride: Ride sharing with privacy-preservation, trust and fair payment atop public blockchain	Baza, Mohamed, Noureddine Lasla, Mohamed MEA Mahmoud, Gautam Srivastava, and Mohamed Abdallah. Year: 2019	The majority of existing ride-sharing services rely on a central third to organize the service. This makes them subject to single point of failure and privacy concerns by both internal and external attackers. This paper gives public blockchain based ride sharing service named B-ride which offers ride sharing services independent of third party.

CHAPTER 3
ANALYSIS

3.1 PROBLEM ANALYSIS

To conduct a problem analysis for the Carpooling System (CapX), let's identify some potential challenges or issues that users or stakeholders may encounter:

1. Limited User Base: One of the main challenges for a carpooling system is establishing a critical mass of users. Initially, attracting a sufficient number of users in a specific area can be difficult. Without an active user base, it may be challenging to match drivers with passengers effectively.

2. Trust and Safety Concerns: Trust is a significant factor in carpooling. Users may have concerns about riding with strangers or sharing personal information. Ensuring the safety and security of users is crucial, and implementing user verification, rating systems, and secure communication channels can help address these concerns.

3. Coordination and Scheduling: Coordinating pick-up and drop-off locations, schedules, and preferences among users can be complex. Finding mutually convenient times and locations for multiple users can present a challenge, particularly in densely populated areas with high traffic.

4. Reliability and Punctuality: Users rely on carpooling systems to reach their destinations on time. However, unforeseen circumstances such as traffic congestion or delays can affect the punctuality of rides. Ensuring a reliable and predictable carpooling experience requires effective communication and contingency planning.

5. Payment and Transaction Management: Managing payments between drivers and passengers can be a potential challenge. Handling payment calculations, processing transactions securely, and addressing issues related to fare disputes or payment discrepancies require a robust payment system.

6. User Engagement and Retention: To maintain an active and engaged user base, it is important to provide incentives and a positive user experience. Encouraging user feedback, implementing gamification elements, and offering rewards or loyalty programs can help promote user engagement and retention.

7. Legal and Regulatory Compliance: Carpooling services may need to comply with local regulations, such as licensing requirements, insurance coverage, or liability issues. Adhering to relevant legal frameworks and ensuring compliance can be a challenge, especially when operating in different jurisdictions.

8. Scalability and Technical Infrastructure: As the user base grows, the system should be able to handle increased traffic and data processing requirements. Ensuring the scalability and stability of the technical infrastructure, including servers, databases, and network resources, is essential for providing a smooth user experience.

Conducting a thorough problem analysis helps identify potential obstacles and challenges that the Carpooling System (CapX) may face. Addressing these issues through appropriate strategies, technologies, and user-centric solutions will contribute to the success and effectiveness of the carpooling platform.

3.2. REQUIREMENT SPECIFICATION

3.2.1 SOFTWARE REQUIREMENTS

- VS CODE EDITOR
- GOOGLE CROME BROWSER
- MONGO DB DATABASE
- INSPECT TOOL

3.2.2 HARDWARE REQUIREMENTS

- Minimum 8 GB RAM
- Minimum Intel i5 processor
- Minimum 80GB HDD

3.2.3 TECHNOLOGIES REQUIREMENTS

- Mongo DB
- Express JS
- React.js

- Node JS

3.3 FUNCTIONAL REQUIREMENTS

1. User Registration: Allow users to create accounts and provide necessary information to join the carpooling system.

2. Ride Request and Offer: Enable users to request rides when they need transportation or offer rides when they have available seats in their vehicles.

3. Matching and Scheduling: Match riders with suitable drivers based on factors like pickup and drop-off locations, preferred routes, time of travel, and user preferences.

4. Ride Management: Provide features to manage rides, including accepting or declining ride requests, confirming pickups and drop-offs, and updating ride status.

5. Payment System: Facilitate financial transactions between riders and drivers, allowing for fare splitting or cost sharing according to predetermined rules.

6. Ratings and Reviews: Allow users to rate and provide feedback about their experience with drivers and riders to build a reliable and trustworthy community.

7. Notifications and Alerts: Send notifications and alerts to users regarding ride requests, ride confirmations, updates, cancellations, and other relevant information.

8. Route Optimization: Optimize routes to minimize travel time and distance, considering multiple pickups and drop-offs within a single ride.

9. Safety and Security Measures: Implement safety features such as identity verification, user profiles, and driver background checks to ensure user security.

10. Communication Platform: Provide an in-app messaging or chat system for riders and drivers to communicate and coordinate logistics before and during the ride.

11. User Support: Offer customer support channels to address user inquiries, resolve issues, and provide assistance when needed.

12. Reporting and Analytics: Generate reports and gather analytics on ride statistics, user feedback, and system performance to improve the carpooling experience.

These requirements cover the essential functionalities of a carpooling system, enabling users to efficiently find, share, and manage rides while promoting safety, convenience, and cost-effectiveness.

3.4 NON- FUNCTIONAL REQUIREMENTS

The nonfunctional requirement elaborates a performance characteristic of the system, nonfunctional requirements of this project are:

Accessibility: the system is easily accessible
Availability: the system is available 24*7 and is accessible anytime
Recoverability: the system is easily recovered
Maintainability: the system is easy to be understood and maintain
Simplicity: the system has an easy to interact user interface
Efficiency: the system is very efficient
Robustness: the system is strong

CHAPTER 4

DESIGN

4.1 DESIGN GOAL

Design is a meaningful engineering representation of something that is to be build. It can be trace to a customer. Requirements and at the same time assessed for quality against a set of predefined criteria
For good design. In a software engineering context, design focuses on four major areas of concern: data, architecture, interfaces, and components the design process translate requirement into representation of software that can be accessed for a quality before code generation. Design is a process in which requirements are translated to blue print for constructing into software. Initially the blue print depicts the holistic view of software. This is a design represented at the high level of abstraction.

During various stages of system development and design of following goals have been set up for complete architecture

- Analysis
- Design
- Development
- Testing
- Deployment

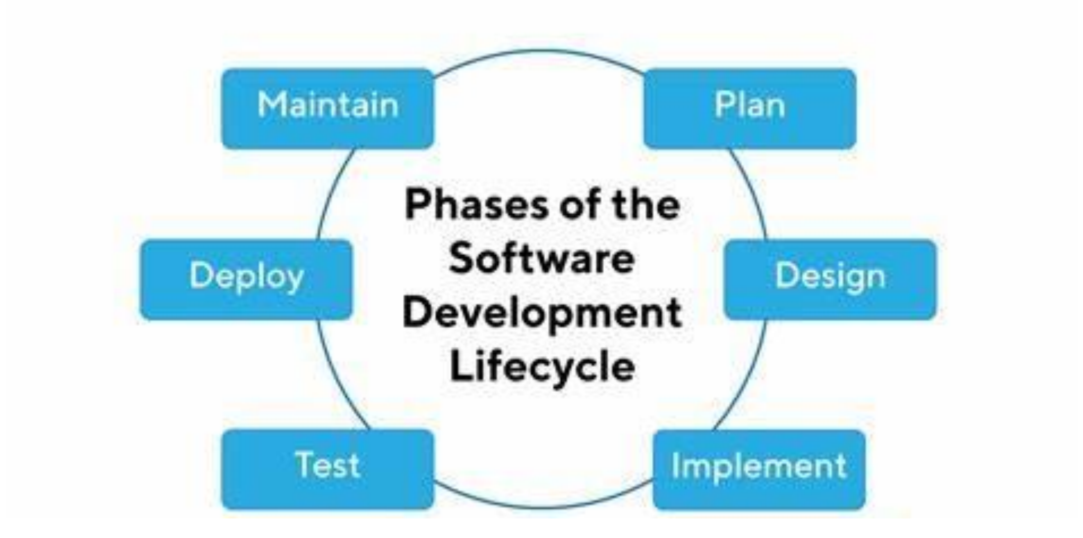


Fig. 4.1 Software Development Lifecycle

4.2 DESIGN STRATEGY

Design is the structure of software. Design strategy is implementation keep in mind the design goals of systems. The most creative and challenging face of life cycle is the system design. They may be define as the process of various technique and principle of various purpose of defining device, the process or system with sufficient details to permit in physical realizations. The importance of software design strategy can be stated in a single word "QUALITY". The strategy provides us with the representation of the software that can accessed for quality. Without a proper design strategy we risk building unstable system that might fail if small changes are made. It may be difficult to test or could be one whose quality can't be tested.

Design strategy is a discipline which helps firms determine what to make and do, why do it and how to innovative contextually, both immediately and over the long-term. This process involves the interplay between design and business strategy. While not always required, design strategy often use social research methods to help ground the results and mitigate the risk any course of action. Design is the bridge between system analysis and system implementation. Some of the essential fundamental concepts involved in the design of application software are:

4.2.1 ABSTRACTION

Abstraction is used to construct solutions to problem without having to take account of the intricate details of the various component sub problems. Abstraction allows system designer to make step-wise refinement, which at each stage of the design may hide, unnecessary details associated with representation or implementation from the surrounding environment.

4.2.2 MODULARITY

Modularity is concerned with decomposing of main module into well-defined manageable units with well-defined interfaces among the units. This enhances design clarity, which in turn eases implementation, Debugging, Testing, Documenting and Maintenance of the software product. Modularity viewed in this sense is a vital tool in the construction of large software projects.

4.2.3 VERIFICATION

Modularity is concerned with decomposing of main module into well-defined manageable units with well-defined interfaces among the units. This enhances design clarity, which in turn eases implementation, Debugging, Testing, Documenting and Maintenance of the software product. Modularity viewed in this sense is a vital tool in the construction of large software projects.

Verification is fundamental concept in software design. A design is verifiable if it can be demonstrated that the design will result in implementation that satisfies the customer's requirements. Verification is of two types namely.

- Verification that the software requirements analysis satisfies the customer's needs.
- Verification that the design satisfies the requirement analysis.

Some of the important factors of quality that are to be considered in the design of application software are:

Reliability:

The software should behave strictly according to the original specification and should Function smoothly under normal conditions.

Extensibility:

The software should be capable of adapting easily to changes in the specification.

Reusability: The software should be developed using a modular approach, which permits modules to be reused by other application, if possible.

Preliminary Design: Preliminary design is basically concerned with deriving an overall picture of the system. Deriving entire system into modules and sub-modules while keeping Cohesion and Coupling factors in mind. Tools, which assist in preliminary design process, are Data Flow Diagrams.

Code design: The purpose of code is to facilitate the identification and retrieval for items of information. A code is an ordered collection of symbols designed to provide unique identification of an entity or attribute. To achieve unique identification there must be only one place where the identified entity or the attribute can be entered in the code; conversely there must be a place in the code for everything that is to be identified. This mutually exclusive feature must be built into any coding system.

The codes for this system are designed with two features in mind. Optimum humanoriented use and machine efficiency. Length of the code range from length of one to length of five characteristics:

- The code structure is unique; ensuring that only one value of the code with a single meaning may be correctly applied to a given entity or attributes.
- The code structure is expansible allowing for growth of its set of entities and attributes.
- The code is concise and brief for recording, communication, and transmission and storage efficiencies.
- They have a uniform size and format.
- The codes are simple so that the user can easily understand it.
- The codes are also versatile i.e., it is easy to modify to reflect necessary changes in condition, chart eristic and relationships of the encode entities.
- The codes are also easily storable for producing reports in a predetermined. The codes are also stable and do not require being frequently updated order of format.
- The codes are also meaningful.

4.2.4 CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagram are widely used in modelling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a structural diagram.

The class diagram consists of the following:

Register: This class represents a basic detail of a participants in car-pooling system. It has attributes such as Name, email, phone, password and gender. It can be associated with a Driver who owns the car and Rider who offers a ride.

Passenger: This class represents a passenger who wants to request a ride. It has attributes like name and pickup Location. A Passenger can be associated with multiple

Rides.

Driver: This class represents a driver who offers rides. It has attributes like name, number, date, time, place, price, and gender. A Driver can be associated with a Register and Information.

Rider: This class represents a specific rider detail. It has attributes such as the name, date, name, place and gender driver. A Rider is associated with a Driver and can have multiple Passengers.

The class diagram illustrates the relationships between these classes. For example, a Ride has an association with a Driver and can have multiple Passengers. The Driver class has a composition relationship with the Car class, indicating that a Driver owns

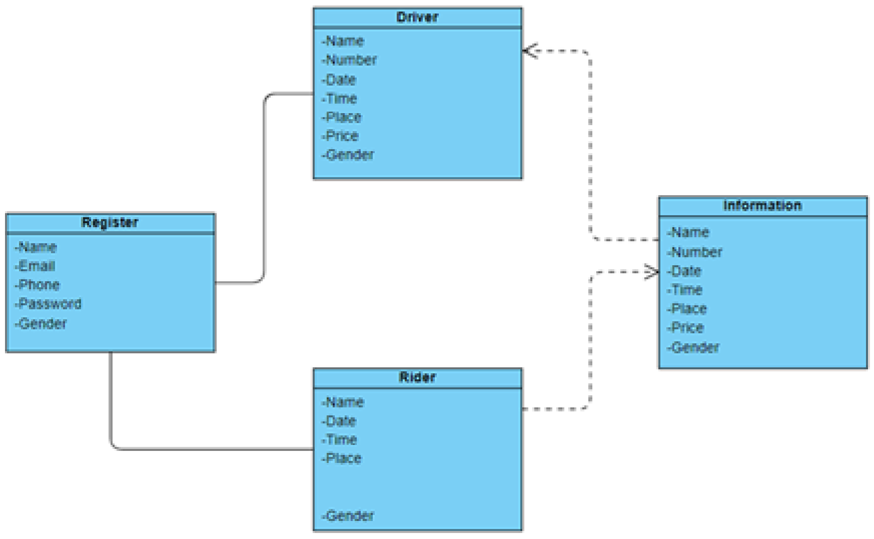


Fig. 4.1 Class Diagram

CHAPTER 5
METHODOLOGY

5.1 What is MERN Stack?

MERN stack is a popular web development stack that consists of four key technologies: Mongo DB, Express.js, React.js, and Node.js. Each component of the MERN stack serves a specific purpose in building dynamic and full-stack web applications.

Here's a breakdown of the technologies in the MERN stack:

1. Mongo DB: Mongo DB is a NoSQL document-oriented database that stores data in a flexible, JSON-like format called BSON (Binary JSON). It provides scalability, high performance, and a flexible schema, allowing developers to work with dynamic and evolving data structures.
2. Express.js: Express.js is a minimalist web application framework for Node.js. It simplifies the process of building web applications by providing a set of robust features and middleware. Express.js handles routing, request handling, and enables the creation of RESTful APIs.
3. React.js: React.js is a JavaScript library for building user interfaces. It allows developers to create reusable UI components and efficiently update and render components as the application state changes. React.js follows a component-based architecture, making it easy to build complex and interactive user interfaces.
4. Node.js: Node.js is a server-side JavaScript runtime built on Chrome's V8 JavaScript engine. It allows developers to execute JavaScript code outside of a web browser, enabling server-side scripting. Node.js provides a scalable and event-driven architecture, making it ideal for building fast and efficient web applications.

When combined, these technologies form the MERN stack, which enables developers to build end-to-end web applications using JavaScript throughout the entire development process. Node.js powers the backend server, Express.js handles routing and middleware, MongoDB serves as the database, and React.js handles the frontend user interface.

The MERN stack is particularly popular for building single-page applications (SPAs) and is known for its flexibility, scalability, and ease of development. It allows developers to leverage JavaScript on both the client-side and server-side, making it a cohesive and efficient technology stack for modern web application development.

5.1.1 Mongo DB: Cross-platform Document-Oriented Database

Mongo DB is a NoSQL database where each record is a document comprising of key-value pairs that are similar to JSON (JavaScript Object Notation) objects. Mongo DB is flexible and allows its users to create schema, databases, tables, etc. Documents that are identifiable by a primary key make up the basic unit of Mongo DB. Once Mongo DB is installed, users can make use of Mongo shell as well. Mongo shell provides a JavaScript interface through which the users can interact and carry out operations (e.g.: querying, updating records, deleting records)

Why use Mongo DB?

- Fast – Being a document-oriented database, easy to index documents. Therefore a faster response.
- Scalability – Large data can be handled by dividing it into several machines.
- Use of JavaScript – Mongo DB uses JavaScript which is the biggest advantage.
- Schema Less – Any type of data in a separate document.
- Data stored in the form of JSON –

1. Objects, Object Members, Arrays, Values, and Strings
2. JSON syntax is very easy to use.
3. JSON has a wide range of browser compatibility.
4. Sharing Data: Data of any size and type (video, audio) can be shared easily.

- Simple Environment Setup – It’s really simple to set up Mongo DB.
- Flexible Document Model – Mongo DB supports document-model (tables, schemas, columns & SQL) which is faster and easier.

Creating a database:

Simply done using a “use” command:

```
Use database name;
```

- **Creating a table: If the collection/table doesn’t exist then a new collection/table will be created:**

```
db.createCollection("collection name");
```

- **Inserting records into the collection:**

```
db.collection_name.insert

(

   {

      "id" : 1,
        "Name" : "Klaus",
                "Department": "Technical",
                "Organization": "Geeks For Geeks"
    }
);
```

Querying a document:

```
db.collection_name.find({Name : "Klaus"}).for Each(printjson);
```

5.1.2 Express: Back-End Framework:

Express is a Node.js framework. Rather than writing the code using Node.js and creating loads of Node modules, Express makes it simpler and easier to write the back-end code. Express helps in designing great web applications and APIs. Express supports many middlewares which makes the code shorter and easier to write.

Why use Express?

- Asynchronous and Single-threaded.
- Efficient, fast & scalable
- Has the biggest community for Node.js
- Express promotes code reusability with its built-in router.
- Robust API
- Create a new folder to start your express project and type below command in the command prompt to initialize a package.json file. Accept the default settings and continue.

  ```
  npm init
  ```
- Then install express by typing the below command and hit enter. Now finally create a file inside the directory named index.js.
- `npm install express --save`
- Now type in the following in index.js to create a sample server.

```
const express = require('express'),

http = require('http');

const hostname = 'localhost';

const port = 8080;

const app = express();



app.use((req, res) => {

  console.log(req.headers);

  res.statusCode = 200;

  res.setHeader('Content-Type', 'text/html');

  res.end('<html><body><h1>This is a test server</h1></body></html>');



});

const sample_server = http.createServer(app);



sample_server.listen(port, hostname, () => {

  console.log(`Server running at http://${hostname}:${port}/`);
});
```

- Update the “scripts” section in package.json file
- Then to start the server by running the below command
  Npm start
- Now you can open the browser and get the output of the running server.

5.1.3 React: Front-End Library

React is a JavaScript library that is used for building user interfaces. React is used for the development of single-page applications and mobile applications because of its ability to handle rapidly changing data. React allows users to code in JavaScript and create UI components.

Why use React?

- Virtual DOM – A virtual DOM object is a representation of a DOM object. Virtual DOM is actually a copy of the original DOM. Any modification in the web application causes the entire UI to re-render the virtual DOM. Then the difference between the original DOM and this virtual DOM is compared and the changes are made accordingly to the original DOM.
- JSX – Stands for JavaScript XML. It is an HTML/XML JavaScript Extension which is used in React. Makes it easier and simpler to write React components.
- Components – ReactJS supports Components. Components are the building blocks of UI wherein each component has a logic and contributes to the overall UI. These components also promote code reusability and make the overall web application easier to understand.
- High Performance – Features like Virtual DOM, JSX and Components makes it much faster than the rest of the frameworks out there.

- Developing Android/Ios Apps – With React Native you can easily code Android-based or IOS-Based apps with just the knowledge of JavaScript and ReactJS.
- You can start your react application by first installing “create-react-app” using npm or yarn.

  ```
  npm install create-react-app --global
  ```

  OR

  ```
  yarn global add create-react-app
  ```

- After that you can create a new react app by using.

  ```
  create-react-app app_name
  ```

- Then navigate into the “app_name” folder and type **yarn start** or **npm start** to start your application.

5.1.4 Node.js: JS Runtime Environment

Node.js provides a JavaScript Environment which allows the user to run their code on the server (outside the browser). Node pack manager i.e. npm allows the user to choose from thousands of free packages (node modules) to download.

Why use Node.JS?

- Open-source JavaScript Runtime Environment
- Single threading – Follows a single-threaded model.
- Data Streaming
- Fast – Built on Google Chrome’s JavaScript Engine, Node.js has a fast code execution.
- Highly Scalable
- Initialize a Node.js application by typing running the below command in the command window. Accept the standard settings.

  ```
  npm init
  ```

- Create a file named index.js.

  Example:

  A basic Node.js example to compute the perimeter & area of a rectangle.

```
var rectangle = {
    perimeter: (x, y) => (2*(x+y)),
    area: (x, y) => (x*y)
};


function Rectangle(l, b) {
    console.log("A rectangle with l = " + l + " and b = " + b);


    if (l <= 0 || b <= 0) {
        console.log("Error! Rectangle's length & breadth should be greater than 0:  l = "
               + l + ",  and b = " + b);
    }
    else {
        console.log("Area of the rectangle: " + rectangle.area(l, b));
        console.log("Perimeter of the rectangle: " + rectangle.perimeter(l, b));
    }
```

```
}




Rectangle(1, 8);

Rectangle(3, 12);

Rectangle(-6,
```

- Run the node application by running the below command in the command window.
  npm start

5.2 APIs

5.2.1 MAPBOX PLACES API:

Mapbox Places API is a location data service that provides developers with access to extensive point of interest (POI) data, geocoding, and reverse geocoding functionality. It allows developers to add location search, autocomplete, and suggestions to their applications and websites.
With Mapbox Places API, developers can access data on millions of places worldwide, including addresses, landmarks, businesses, and other points of interest. They can also use the API to search for and retrieve detailed information on specific locations, such as a place's name, address, phone number, website, hours of operation, and more.
n addition, Map box Places API provides geocoding and reverse geocoding functionality, which allows developers to translate addresses or place names into geographic coordinates (latitude and longitude) and vice versa. This can be useful for a variety of location-based applications, such as mapping, routing, and navigation.
Overall, Mapbox Places API is a powerful tool for developers looking to integrate location-based features into their applications and websites.

5.2.2 MAPBOX DIRECTION API:

Mapbox Directions API is a routing and directions service that provides developers with accurate, turn-by-turn directions for driving, walking, cycling, and public transit routes. It allows developers to integrate route planning and navigation functionality into their applications and websites.

With Mapbox Directions API, developers can access data on millions of roads and paths worldwide, and use it to generate optimized driving, walking, or cycling routes between two or more locations. They can also specify route parameters such as start and end points, waypoints, travel modes, and optimization criteria, and receive detailed instructions and estimated travel times for each segment of the route.

Mapbox Directions API also supports public transit routing, allowing developers to retrieve detailed information on transit stops, schedules, and fares, and provide users with route options that include buses, trains, and other forms of public transportation.

5.3 Work flow of model

The workflow of a carpooling system typically involves several steps and interactions between users. Here's a general outline of the workflow:

1. Registration: Users sign up and create an account on the carpooling platform. They provide necessary information such as name, contact details, and possibly additional verification steps for security purposes.
2. Profile Creation: Users complete their profiles by adding relevant information, such as their location, preferred commuting routes, car details (for drivers), and any specific preferences they may have.
3. Searching and Offering Rides: Users can search for available rides or offer their own rides. They specify their starting point, destination, and preferred time of travel. The system matches users based on compatible routes and schedules.
4. Ride Selection and Confirmation: Users review the available rides or ride requests and select the most suitable option. They can view details about the driver/passenger, ratings, and any additional information provided. Once a selection is made, the system confirms the ride.
5. Communication and Coordination: Users communicate with each other through the platform to discuss specific pickup/drop-off locations, any preferences, or any

necessary coordination details. The system may provide messaging or chat functionality to facilitate communication.

6. Ride Execution: On the scheduled day and time, the driver picks up the passenger(s) from the agreed-upon location, and they travel together to the destination. The system may provide real-time updates, such as estimated time of arrival or tracking of the ride.

7. Payment and Transaction: After the ride is completed, the payment is processed through the secure payment system integrated into the platform. The system calculates the fare based on predefined criteria (e.g., distance, duration) and deducts it from the passenger's account. In some cases, the system may handle cashless transactions or utilize digital payment methods

8. Rating and Feedback: Both the driver and passenger have the opportunity to rate and provide feedback on their experience with each other. This helps maintain accountability and build trust within the carpooling community.

9. User Support and Issue Resolution: The system offers customer support channels for users to report any issues or resolve conflicts that may arise during the carpooling process. Support staff or automated systems handle these inquiries and ensure a satisfactory resolution.

The workflow may vary slightly depending on the specific features and design of the carpooling system. However, the core steps mentioned above provide a general overview of how a typical carpooling system operates.

5.4 FLOWCHART

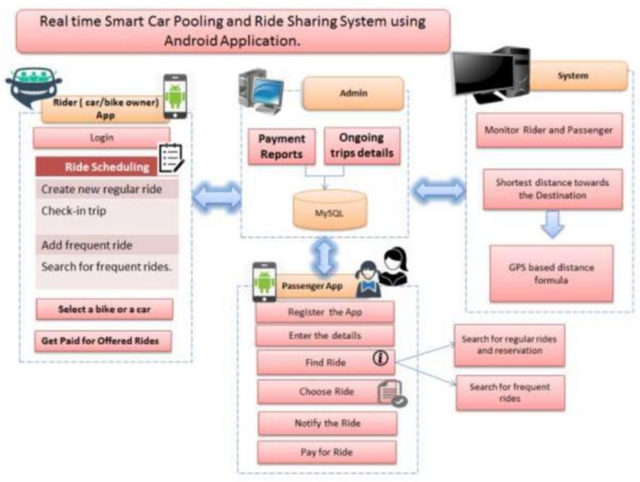

Real time Smart Car Pooling and Ride Sharing System using Android Application.
Rider (car/bike owner) App
Login
Ride Scheduling
Create new regular ride
Check-in trip
Add frequent ride
Search for frequent rides.
Select a bike or a car
Get Paid for Offered Rides
Admin
Payment Reports
Ongoing trips details
MySQL
System
Monitor Rider and Passenger
Shortest distance towards the Destination
GPS based distance formula
Passenger App
Register the App
Enter the details
Find Ride
Choose Ride
Notify the Ride
Pay for Ride
Search for regular rides and reservation
Search for frequent rides

Figure: - Proposed System Architecture.

Fig 5.4 Flowchart

CHAPTER 6
IMPLEMENTATION

6.1 IMPLEMENTAION STATERGY:

Step 1. Design your application: Start by creating a wireframe or prototype of your application. This will help you visualize how your application will look and function.
Step 2. Set up your development environment: Install the necessary software and tools you need to build your application, including Node.js, Mongo DB, and a code editor.
Step 3. Create the backend with Node.js and Express: Begin by building the backend of your application using Node.js and Express. Set up the necessary routes and endpoints to handle user authentication, data retrieval, and data storage in Mongo DB.
Step 4. Implement user authentication: Add user authentication to your application to ensure that only authorized users can access certain features of your application.
Step 5. Set up your database: Create a Mongo DB database and set up the necessary collections to store your data.
Step 6. Create the frontend with React: Once you have the backend set up, create the frontend of your application using React. Use React components to build out the user interface and connect them to the backend API.
Step 7. Implement CRUD operations: Implement Create, Read, Update, and Delete (CRUD) operations to allow users to create, view, update, and delete data in your application.
Step 8. Integrate third-party APIs: If you plan to integrate third-party APIs, such as Mapbox Maps APIs or payment processing APIs, implement the necessary code to integrate them with your application.
Step 9. Test your application: Test your application to ensure that it functions correctly and there are no bugs or errors.
Step 10. Deploy your application: Once you have tested your application, deploy it to a server or hosting platform so that it can be accessed by users.

6.2 HARDWARE PLATFORM USED:

- Intel i5 Processor or above
- 4GB RAM
- Monitor , Mouse, Keyboard

6.3 SOFTWARE PLATFORM USED:

- Windows 10 operating System
- Google Chrome Updated Browser
- VS code editor

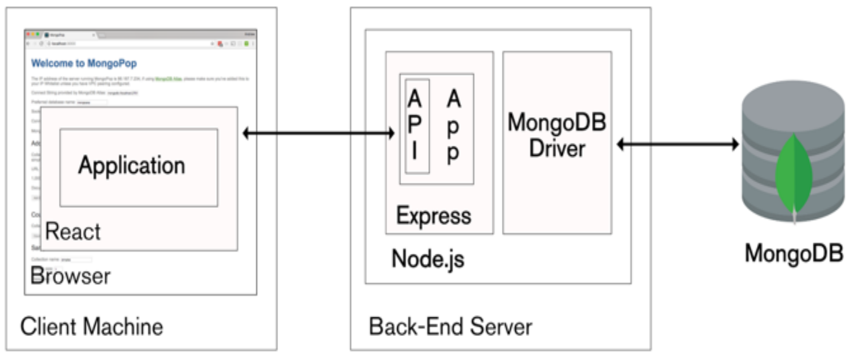
6.4 TECHNOLOGY USED:

• Mongo DB

• Express JS

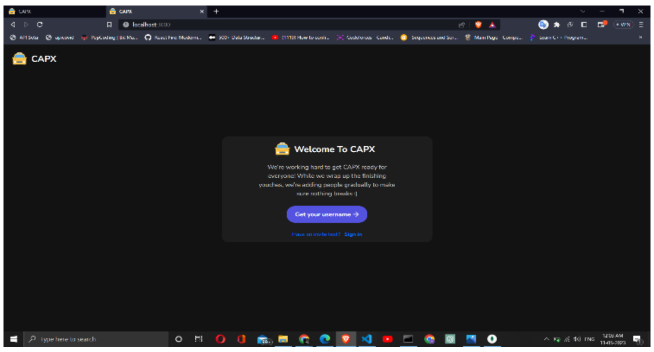
• React JS

• Node JS



6.1 Concept map of AIPS

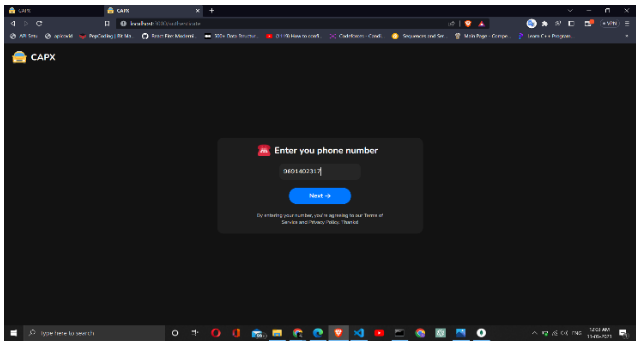
CHAPTER 7
OUTPUT



Welcome To CAPX

We're working hard to get CAPX ready for everyone! While we wrap up the finishing touches, we're adding people gradually to make sure nothing breaks :)

Fig. 7.1 Home Page

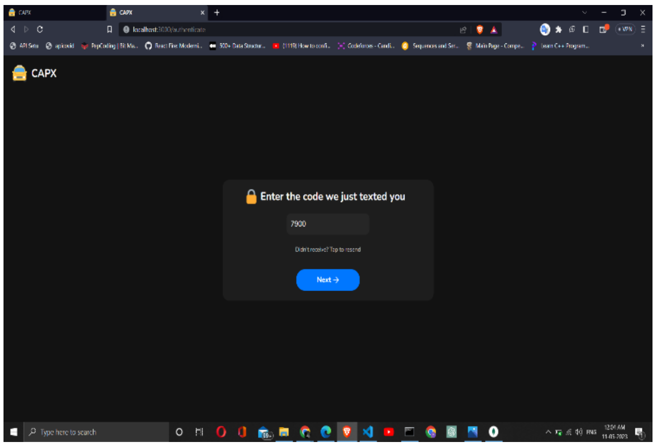


Enter you phone number

9691402317

By entering your number, you're agreeing to our Terms of Service and Privacy Policy. Thanks!

Fig. 7.2   Sign Up



Enter the code we just texted you

7900

Didn't receive? Tap to resend

Fig. 7.3 User Verification

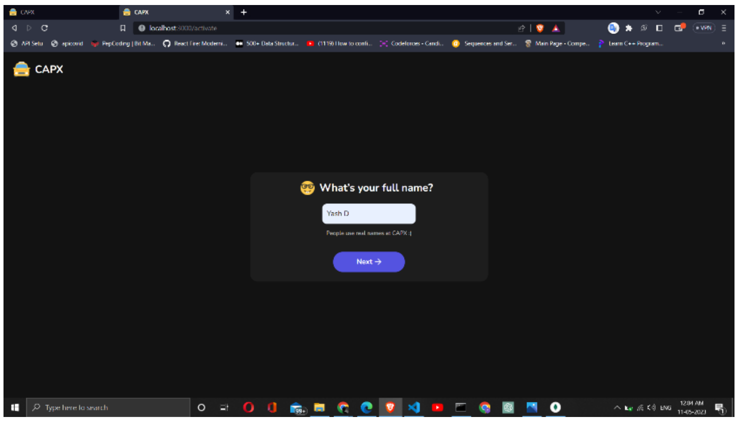


Fig. 7.4 Enter Personal Information

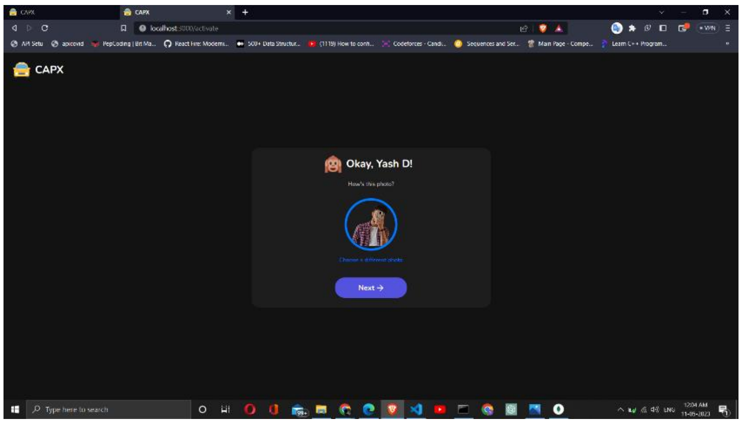


Fig. 7.5 Profile Created

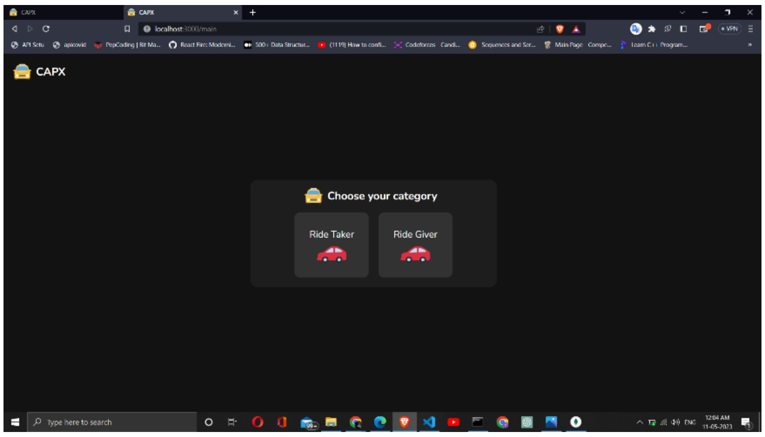
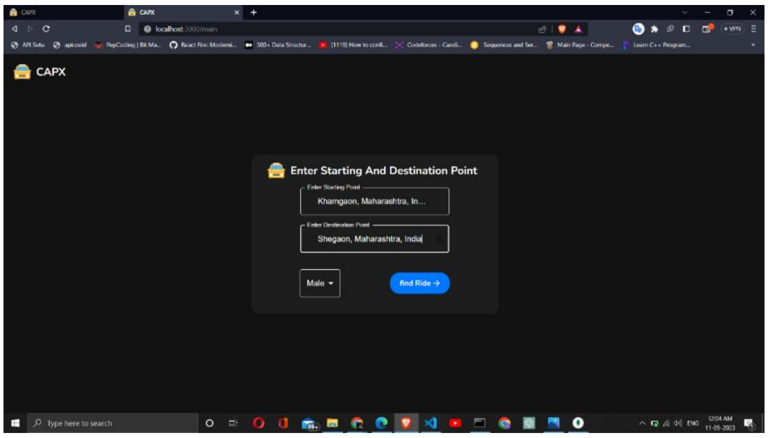
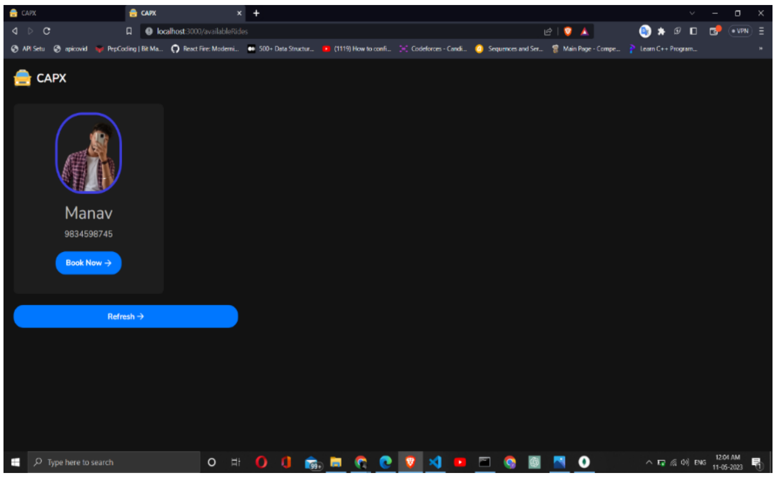


Fig. 7.6 Choose Category



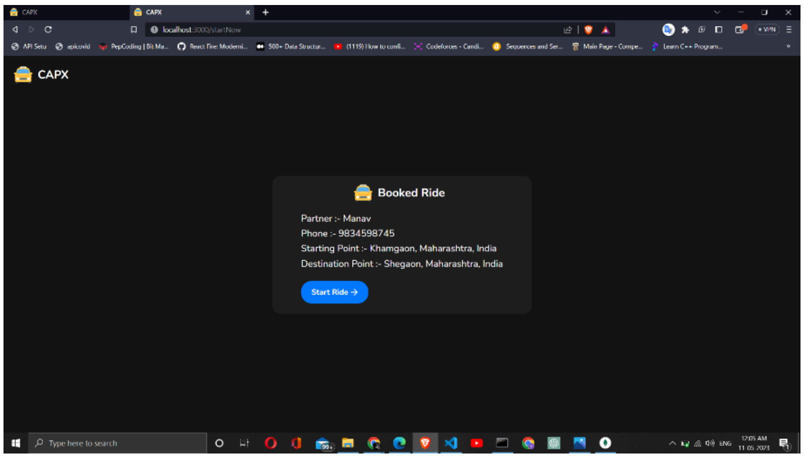
Enter Starting Point: Khamgaon, Maharashtra, In...
Enter Destination Point: Shegaon, Maharashtra, India
Male

Fig. 7.7 Add details



Manav
9834598745

Fig. 7.6 Rider Profile



Partner :- Manav
Phone :- 9834598745
Starting Point :- Khamgaon, Maharashtra, India
Destination Point :- Shegaon, Maharashtra, India

Fig. 7.7 Ride Booked

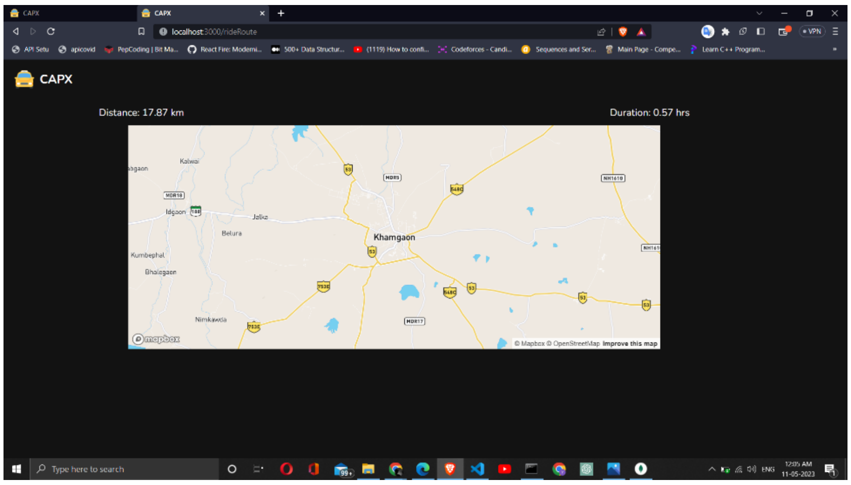


Fig. 7.8 Maps Ride Booked

CHAPTER 8
CONCLUSION

In conclusion, the Carpooling System (CapX) developed using the MERN stack technology aims to revolutionize the commuting experience by providing a user-friendly platform for ride-sharing and carpooling. By leveraging MongoDB, Express.js, React.js, and Node.js, CapX offers a robust and efficient solution for connecting drivers and passengers traveling on the same routes.

With CapX, users can easily register as drivers or passengers, create profiles, and search for compatible rides. The system facilitates communication and coordination between users, ensuring a smooth pickup and drop-off process. A secure payment system ensures hassle-free transactions, eliminating the need for cash exchanges.

To prioritize user safety, CapX implements a rating system and user verification mechanisms, fostering trust and accountability within the community. Users can provide feedback on their experiences, enhancing transparency and improving the overall carpooling ecosystem.

By promoting carpooling, CapX contributes to an eco-friendly and cost-effective solution for commuting problems. By reducing the number of private vehicles on the road, CapX aims to alleviate traffic congestion, lower carbon emissions, and promote a greener future.

Overall, the Carpooling System (CapX) developed using the MERN stack combines convenience, safety, and sustainability. It offers a comprehensive solution to address commuting challenges while improving the overall commuting experience for users.

CHAPTER 9

Future Scope

The carpooling system made with MERN stack technology has a lot of potential for future development and improvement. Here are some potential future scopes:

Integration with public transportation: The carpooling system can be integrated with public transportation systems to create a seamless and efficient transportation network. This will allow users to easily plan and book rides that combine both public transportation and carpooling.

Real-time tracking and notifications: Real-time tracking and notifications can be added to the carpooling system to provide users with accurate information about the status of their ride. This will include information such as the location of the driver, estimated time of arrival, and any updates about delays or changes in the ride schedule.

Intelligent matching algorithms: The carpooling system can use intelligent matching algorithms to match riders and drivers based on their location, destination, and preferences. This will improve the efficiency of the system and make it easier for users to find compatible rides.

Integration with smart city technology: The carpooling system can be integrated with smart city technology to create a more sustainable and efficient transportation system. This can include features such as dynamic pricing based on traffic congestion, rewards for using eco-friendly vehicles, and incentives for carpooling during peak hours.

Expansion to new markets: The carpooling system can be expanded to new markets, both domestically and internationally. This will allow the system to reach more users and provide a more comprehensive transportation solution for different regions and cultures.

REFERENCES

[1] Rutuja Pharande, Prof. Neha Sharma, Shubhangi Gunjal and Abhishek Mahale. Peer-to-peer car sharing system, IRJMETS, e-ISSN: 2582-5208, Volume:04/Issue:12/December-2022.

[2] Alejandro Lugo, Nathalie Aquino, Magalí González, Luca Cernuzzi, UCarpooling: decongesting traffic through carpooling using automatic pairings, CLEI electronic journal, Volume 24, Number 2, Paper 10, July 2021.

[3] Yueshen Xu, Yuqiao Liao, Jianbin Huang, Ying Li, \"A Constraint-aware Ridesharing Service Guaranteeing Quality-of-Service for Smart Cities\", 2021 IEEE International Conference on Services Computing (SCC), pp.154-164, 2021.

[4] Kaushalya Thopate, Mahesh Sathe, Saurav Gujar, Sarthak Honmute, Pushkraj Savji, Vinayak Sawandkar, Satvik Vishnoi, Vishwa-connect: a ride sharing mobile application for campus students, IJRASET, ISSN : 2321-9653

[5] Zhidan Liu, Zengyang Gong, Jiangzhou Li, Kaishun Wu, \"mT-Share: A Mobility-Aware Dynamic Taxi Ridesharing System\", IEEE Internet of Things Journal, vol.9, no.1, pp.182-198, 2022.

[6] Kamaruddin, Kamalia Azma, and Nur Rozliana Mohd Rozlis. \"UiTM Share Ride: Requirements Validation, Design and Development of a Campus Ride-Sharing Mobile Application.\" In 2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS), pp. 1-6. IEEE, 2019.

[7] Xiufeng Xia, Zhenchang Hu, Rui Zhu, Jiajia Li, Chuanyu Zong, Xiangyu Liu, \"TMATCH: A New Framework for Supporting Car-Sharing Under Carpooling Model\", 2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (Smart CNS), pp.153-158, 2019.

[8] Amasyali, Mustafa Burak, and Ensar Gul. \"VoIP integration for mobile ride-sharing application.\" In 2017 7th International Conference on Communication Systems and Network Technologies (CSNT), pp. 44-47. IEEE, 2017.

[9] Sasikumar C, Jaganathan S. A Dynamic Carpooling System with Social Network Based Filtering. Research J. Engineering and Tech. 2017; 8(3): 263-267. doi: 10.5958/2321-581X.2017.00044.7

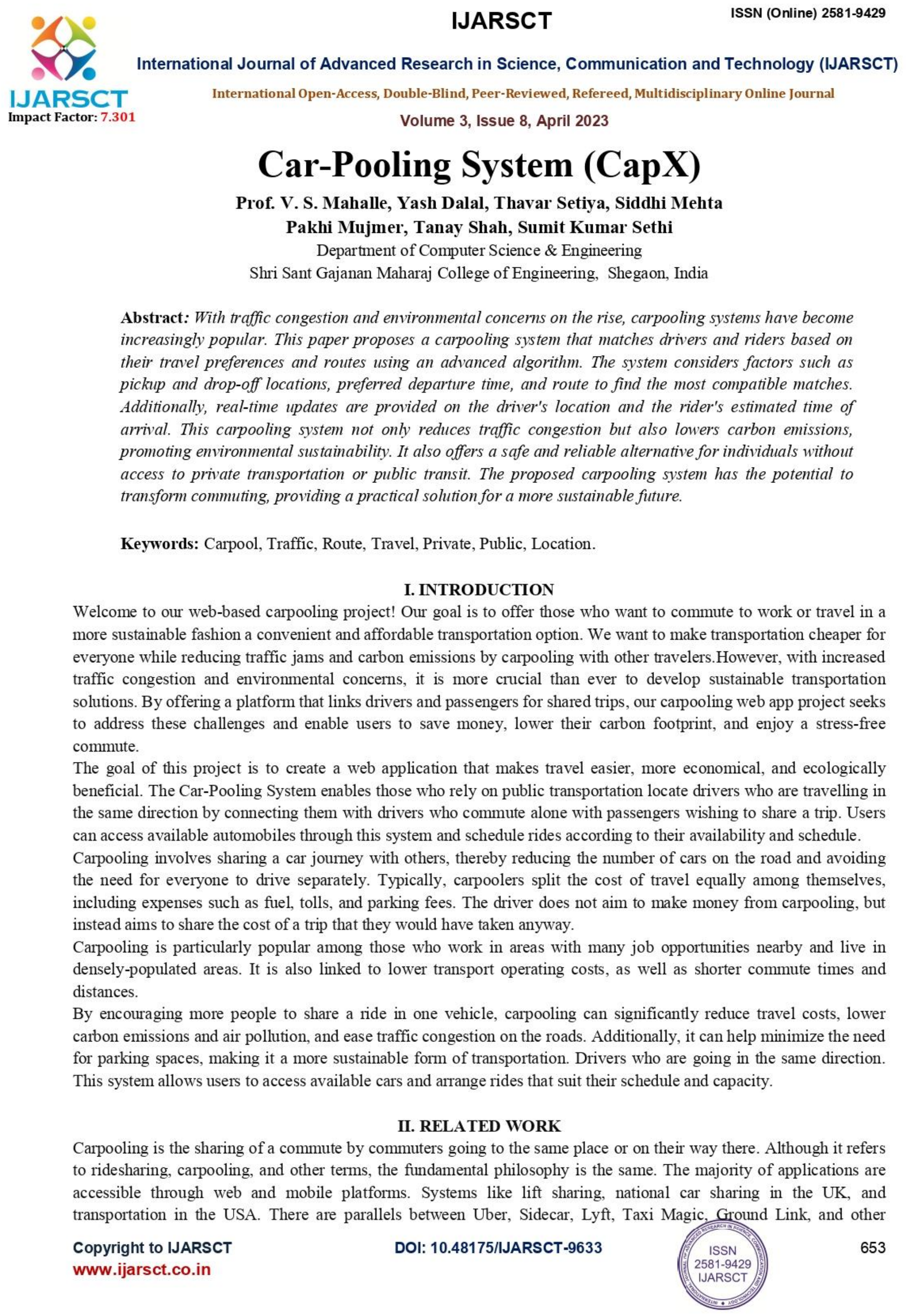
[10] Baza, Mohamed, Noureddine Lasla, Mohamed MEA Mahmoud, Gautam Srivastava, and Mohamed Abdallah. \"B-ride: Ride sharing with privacy-preservation, trust and fair payment atop public blockchain.\" IEEE Transactions on Network Science and Engineering 8, no. 2 (2019): 1214-1229.

DISSEMINATION OF WORK

PUBLICATION DETAILS

PAPER TITLE	CONFERENCE NAME	PUBLICATION DATE	ISSN NUMBER
Carpooling System (CapX)	International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)	8, April 2023	ISSN (Online) 2581-9429

Research Paper



IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 8, April 2023

Car-Pooling System (CapX)

Prof. V. S. Mahalle, Yash Dalal, Thavar Setiya, Siddhi Mehta
Pakhi Mujmer, Tanay Shah, Sumit Kumar Sethi
Department of Computer Science & Engineering
Shri Sant Gajanan Maharaj College of Engineering, Shegaon, India

Abstract*: With traffic congestion and environmental concerns on the rise, carpooling systems have become increasingly popular. This paper proposes a carpooling system that matches drivers and riders based on their travel preferences and routes using an advanced algorithm. The system considers factors such as pickup and drop-off locations, preferred departure time, and route to find the most compatible matches. Additionally, real-time updates are provided on the driver's location and the rider's estimated time of arrival. This carpooling system not only reduces traffic congestion but also lowers carbon emissions, promoting environmental sustainability. It also offers a safe and reliable alternative for individuals without access to private transportation or public transit. The proposed carpooling system has the potential to transform commuting, providing a practical solution for a more sustainable future.*

Keywords: Carpool, Traffic, Route, Travel, Private, Public, Location.

I. INTRODUCTION

Welcome to our web-based carpooling project! Our goal is to offer those who want to commute to work or travel in a more sustainable fashion a convenient and affordable transportation option. We want to make transportation cheaper for everyone while reducing traffic jams and carbon emissions by carpooling with other travelers.However, with increased traffic congestion and environmental concerns, it is more crucial than ever to develop sustainable transportation solutions. By offering a platform that links drivers and passengers for shared trips, our carpooling web app project seeks to address these challenges and enable users to save money, lower their carbon footprint, and enjoy a stress-free commute.

The goal of this project is to create a web application that makes travel easier, more economical, and ecologically beneficial. The Car-Pooling System enables those who rely on public transportation locate drivers who are travelling in the same direction by connecting them with drivers who commute alone with passengers wishing to share a trip. Users can access available automobiles through this system and schedule rides according to their availability and schedule.

Carpooling involves sharing a car journey with others, thereby reducing the number of cars on the road and avoiding the need for everyone to drive separately. Typically, carpoolers split the cost of travel equally among themselves, including expenses such as fuel, tolls, and parking fees. The driver does not aim to make money from carpooling, but instead aims to share the cost of a trip that they would have taken anyway.

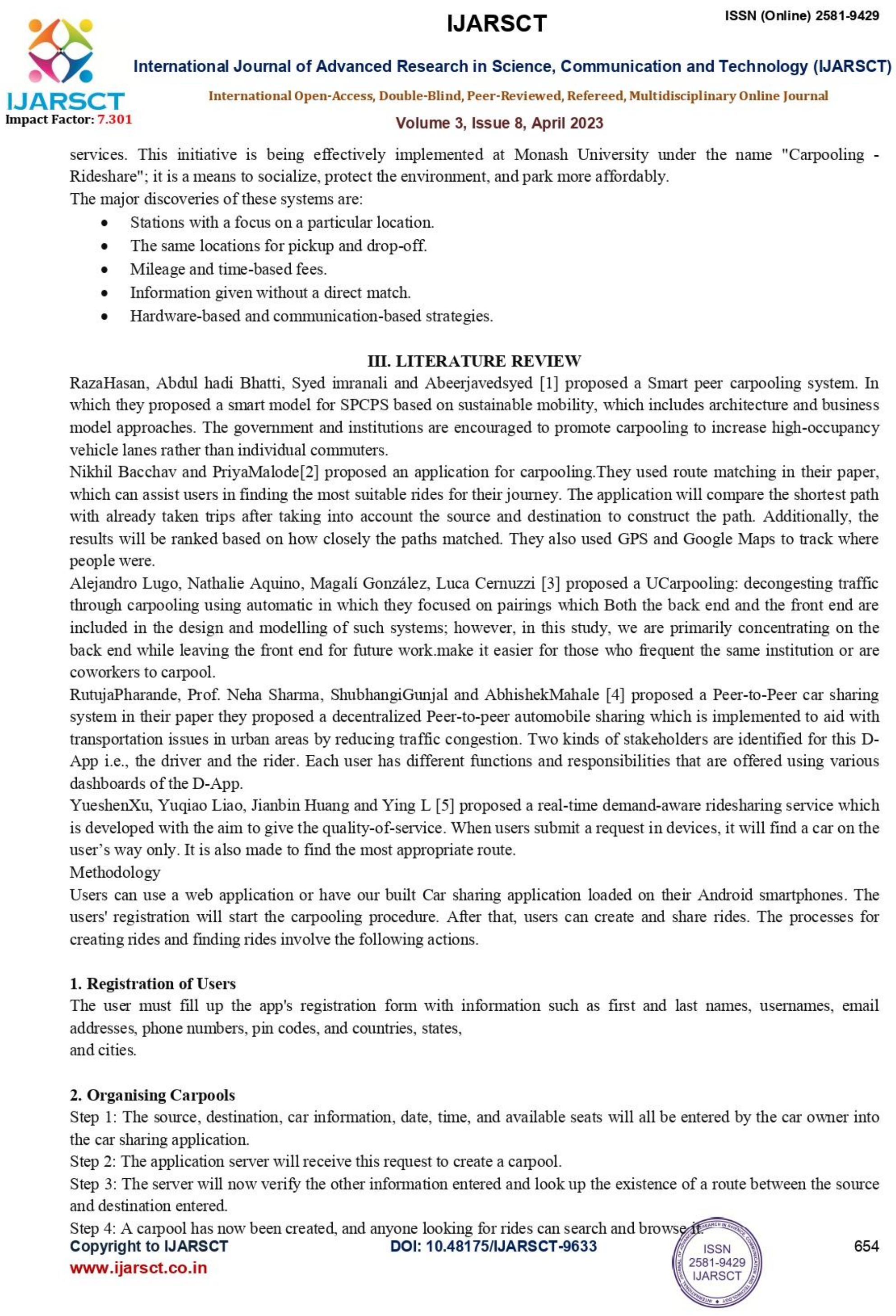
Carpooling is particularly popular among those who work in areas with many job opportunities nearby and live in densely-populated areas. It is also linked to lower transport operating costs, as well as shorter commute times and distances.

By encouraging more people to share a ride in one vehicle, carpooling can significantly reduce travel costs, lower carbon emissions and air pollution, and ease traffic congestion on the roads. Additionally, it can help minimize the need for parking spaces, making it a more sustainable form of transportation. Drivers who are going in the same direction. This system allows users to access available cars and arrange rides that suit their schedule and capacity.

II. RELATED WORK

Carpooling is the sharing of a commute by commuters going to the same place or on their way there. Although it refers to ridesharing, carpooling, and other terms, the fundamental philosophy is the same. The majority of applications are accessible through web and mobile platforms. Systems like lift sharing, national car sharing in the UK, and transportation in the USA. There are parallels between Uber, Sidecar, Lyft, Taxi Magic, Ground Link, and other

ISSN
2581-9429
IJARSCT



IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 8, April 2023

services. This initiative is being effectively implemented at Monash University under the name "Carpooling - Rideshare"; it is a means to socialize, protect the environment, and park more affordably.
The major discoveries of these systems are:

- Stations with a focus on a particular location.
- The same locations for pickup and drop-off.
- Mileage and time-based fees.
- Information given without a direct match.
- Hardware-based and communication-based strategies.

III. LITERATURE REVIEW

RazaHasan, Abdul hadi Bhatti, Syed imranali and Abeerjavedsyed [1] proposed a Smart peer carpooling system. In which they proposed a smart model for SPCPS based on sustainable mobility, which includes architecture and business model approaches. The government and institutions are encouraged to promote carpooling to increase high-occupancy vehicle lanes rather than individual commuters.
Nikhil Bacchav and PriyaMalode[2] proposed an application for carpooling.They used route matching in their paper, which can assist users in finding the most suitable rides for their journey. The application will compare the shortest path with already taken trips after taking into account the source and destination to construct the path. Additionally, the results will be ranked based on how closely the paths matched. They also used GPS and Google Maps to track where people were.
Alejandro Lugo, Nathalie Aquino, Magalí González, Luca Cernuzzi [3] proposed a UCarpooling: decongesting traffic through carpooling using automatic in which they focused on pairings which Both the back end and the front end are included in the design and modelling of such systems; however, in this study, we are primarily concentrating on the back end while leaving the front end for future work.make it easier for those who frequent the same institution or are coworkers to carpool.
RutujaPharande, Prof. Neha Sharma, ShubhangiGunjal and AbhishekMahale [4] proposed a Peer-to-Peer car sharing system in their paper they proposed a decentralized Peer-to-peer automobile sharing which is implemented to aid with transportation issues in urban areas by reducing traffic congestion. Two kinds of stakeholders are identified for this D-App i.e., the driver and the rider. Each user has different functions and responsibilities that are offered using various dashboards of the D-App.
YueshenXu, Yuqiao Liao, Jianbin Huang and Ying L [5] proposed a real-time demand-aware ridesharing service which is developed with the aim to give the quality-of-service. When users submit a request in devices, it will find a car on the user's way only. It is also made to find the most appropriate route.
Methodology
Users can use a web application or have our built Car sharing application loaded on their Android smartphones. The users' registration will start the carpooling procedure. After that, users can create and share rides. The processes for creating rides and finding rides involve the following actions.

1. Registration of Users
The user must fill up the app's registration form with information such as first and last names, usernames, email addresses, phone numbers, pin codes, and countries, states,
and cities.

2. Organising Carpools
Step 1: The source, destination, car information, date, time, and available seats will all be entered by the car owner into the car sharing application.
Step 2: The application server will receive this request to create a carpool.
Step 3: The server will now verify the other information entered and look up the existence of a route between the source and destination entered.
Step 4: A carpool has now been created, and anyone looking for rides can search and browse it.

ISSN 2581-9429 IJARSCT

IJARSCT ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Impact Factor: 7.301 Volume 3, Issue 8, April 2023

3. Look up Carpool

To search and browse for rides, the ride seeker must first enter the source and destination into the application.
Step 2: The carpool server will receive this search information.
Step 3: The server will now verify every input that the user has provided.
Step 4: The server will display available rides to the ride seeker after validation. Any of these ride makers may receive a request from a ride seeker.

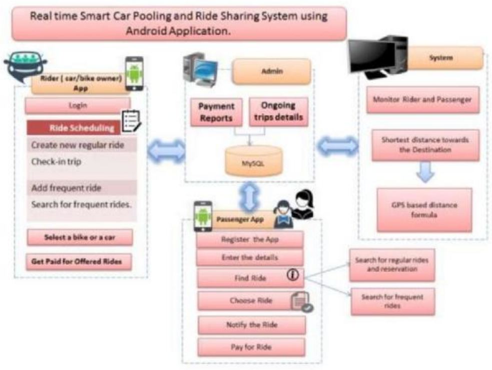

Real time Smart Car Pooling and Ride Sharing System using Android Application.
Rider (car/bike owner) App
Login
Ride Scheduling
Create new regular ride
Check-in trip
Add frequent ride
Search for frequent rides.
Select a bike or a car
Get Paid for Offered Rides
Admin
Payment Reports
Ongoing trips details
MySQL
System
Monitor Rider and Passenger
Shortest distance towards the Destination
GPS based distance formula
Passenger App
Register the App
Enter the details
Find Ride
Choose Ride
Notify the Ride
Pay for Ride
Search for regular rides and reservation
Search for frequent rides

Figure: - Proposed System Architecture.

IV. CONCLUSION

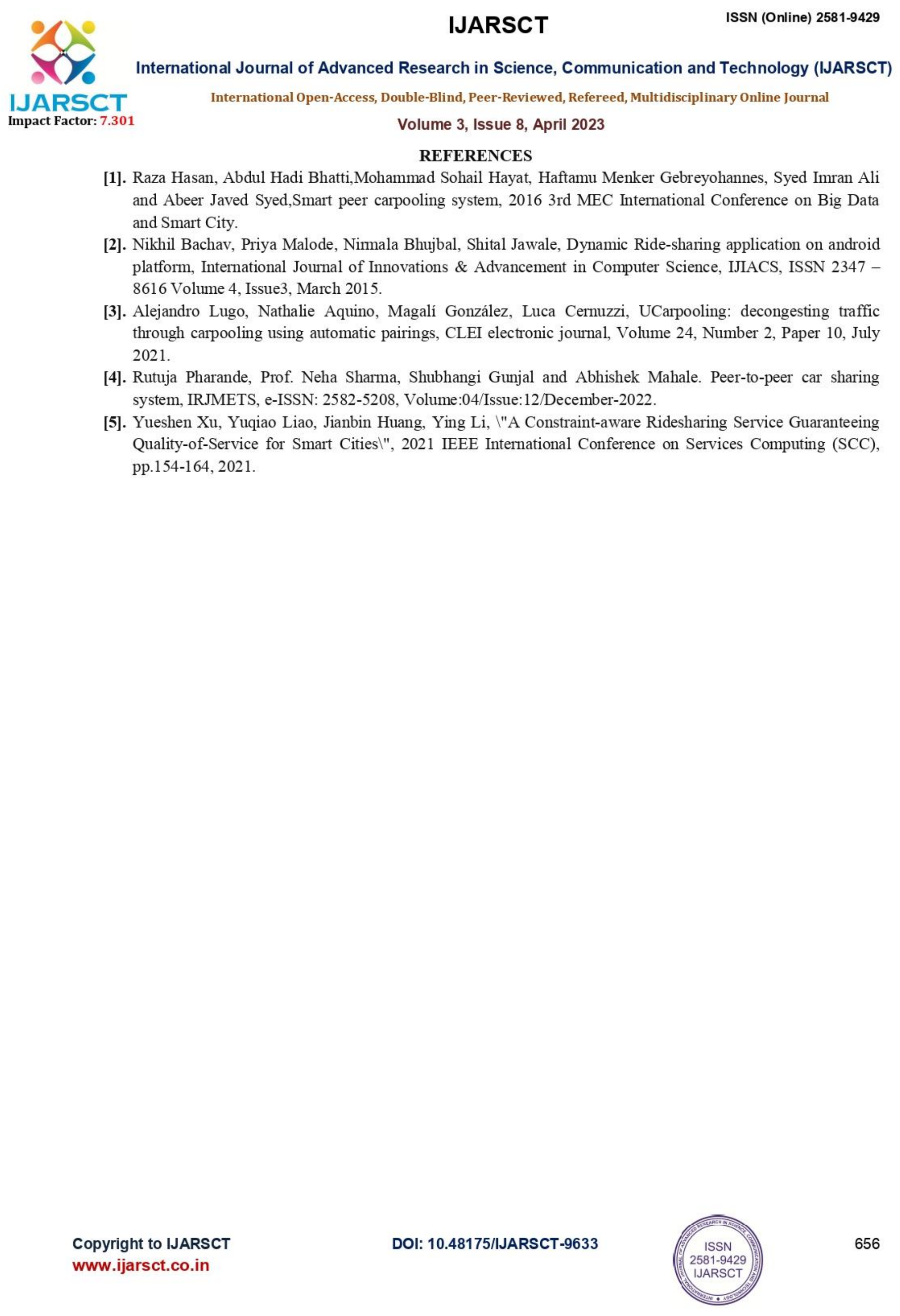
In this study, we presented a useful web application for carpooling that tracks user location. In today's world, carpooling is much more effective because it can lessen traffic and environmental issues. By promoting car/bike sharing, this initiative, Car Pooling System, hopes to minimise traffic on the roads as well as the consumption of gasoline, our most important non-renewable resource. Therefore, it is a social application that is favorable to the environment and aids in reducing travel time. This application has successfully completed computation and testing using "test cases". It is user-friendly and has the necessary options that the user can use to carry out the desired activities. Express, React, Node.js, and MongoDB are used to create the application's front end and back end, respectively.
The web application succeeds in achieving the following objectives: Instant access. productivity gains. optimal use of the available resources. efficient record management. operations are made simpler. speedier information retrieval and processing. friendly to users. Scalable and portable for future improvement.

V. FUTURE WORK

The aforementioned design allows for the development of both web-based and mobile applications. The programme can leverage free APIs like Google Maps to direct the user. Having both drivers and passengers allows for at least two passenger to be carried in each vehicle. To motivate commuters, a system based on credit points could be implemented. This study can be improved further to pave the way for the future generation of vehicles and aid in the development of intelligent transportation systems.

VI. ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in America isThe authors are thankful to Mr. V.S. Mahalle and Head of Department Mr. S. B. Patil for their support and encouragement to carry out this study.



IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Volume 3, Issue 8, April 2023

IJARSCT
Impact Factor: 7.301

REFERENCES

[1]. Raza Hasan, Abdul Hadi Bhatti,Mohammad Sohail Hayat, Haftamu Menker Gebreyohannes, Syed Imran Ali and Abeer Javed Syed,Smart peer carpooling system, 2016 3rd MEC International Conference on Big Data and Smart City.

[2]. Nikhil Bachav, Priya Malode, Nirmala Bhujbal, Shital Jawale, Dynamic Ride-sharing application on android platform, International Journal of Innovations & Advancement in Computer Science, IJIACS, ISSN 2347 – 8616 Volume 4, Issue3, March 2015.

[3]. Alejandro Lugo, Nathalie Aquino, Magalí González, Luca Cernuzzi, UCarpooling: decongesting traffic through carpooling using automatic pairings, CLEI electronic journal, Volume 24, Number 2, Paper 10, July 2021.

[4]. Rutuja Pharande, Prof. Neha Sharma, Shubhangi Gunjal and Abhishek Mahale. Peer-to-peer car sharing system, IRJMETS, e-ISSN: 2582-5208, Volume:04/Issue:12/December-2022.

[5]. Yueshen Xu, Yuqiao Liao, Jianbin Huang, Ying Li, \"A Constraint-aware Ridesharing Service Guaranteeing Quality-of-Service for Smart Cities\", 2021 IEEE International Conference on Services Computing (SCC), pp.154-164, 2021.