Pronto
(Online Delivery & Bike Taxi System)

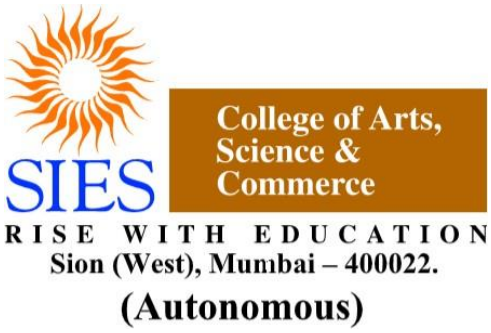
A Project Report
Submitted in partial fulfilment of the
Requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Roshan Jose
TIT2122077

Under the esteemed guidance of
Mrs. Biju Ramesh



DEPARTMENT OF INFORMATION TECHNOLOGY
SIES COLLEGE OF ARTS, SCIENCE & COMMERCE (AUTONOMOUS)
SION (W), MUMBAI, 400022
MAHARASHTRA

2021

PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

Roll No**:** TIT2122077

1.	Name of the Student	Roshan Jose
2.	Title of the Project	Pronto (Online Delivery & Bike Taxi System)
3.	Name of the Guide	Mrs. Biju Ramesh
4.	Teaching experience of the Guide	18 years
5.	Is this your first submission?	Yes ✓ No

Roshan Jose
Signature of the Student

Date: 30/9/21

Ms. Biju Ramesh
Signature of the Guide

Date: 30/9/21

Ms. Sudha. B
Signature of the Coordinator

Date: 30/9/21

SIES COLLEGE OF ARTS, SCIENCE & COMMERCE (AUTONOMOUS)
(Affiliated to University of Mumbai)
SION (W), MUMBAI-400022

DEPARTMENT OF INFORMATION TECHNOLOGY

College of Arts, Science & Commerce
SIES
RISE WITH EDUCATION
Sion (West), Mumbai – 400022.
(Autonomous)

CERTIFICATE

This is to certify that the project entitled, **"Pronto (Online Delivery & Bike Taxi System)"**, is bonafide work of **Roshan Jose** bearing Seat No: **TIT2122077 s**ubmitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY.

Ms. Biju Ramesh
Internal Guide

Ms. Sudha.B
Coordinator

Ms. Biju Ramesh
ExternalExaminer

Date: 05/10/21

ABSTRACT

The website ‘Pronto’ is a reliable express delivery partner for people who need to deliver or receive products and travel from one place to another. Just as the word pronto (early) suggests, the main objective of this site is to provide a fast and secure service to the users. Here in Pronto a person can take the full advantage of the door-to-door courier service which has no weight limits. Also, through this website anyone can enjoy the bike taxi experience and can avoid busy traffic. Pronto is a forum which can understand people’s current requirements and adapt accordingly. Considering this pandemic situation users can use the complete safe and secured shop to home delivery service offered by this site in which a user can order from their local shop online and a cartero (delivery guy) will deliver it to their door step.

ACKNOWLEDGEMENT

Before getting into thickest of things, we would like to thank the personalities who were part of our project in numerous ways, those who gave us outstanding support from the start of the project.

It is our privilege to express sincerest regards to our guide, **Mrs. Biju Ramesh** for her valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of our project. We deeply express our sincere thanks to our Head of Department, **Mrs. Sudha B** for encouraging and supporting us. We take this opportunity to thank all our lecturers who have directly or indirectly helped in the project. We pay our respect and love to our parents and all other family members and friends for their love and encouragement throughout the project.

Last but not the least I am grateful to my project partner for his cooperation, efforts and support without which completing the project in time would not be possible.

DECLARATION

I hereby declare that the project entitled, "**Pronto (Online Delivery & Bike Taxi System)**" done at **SIES COLLEGE OF ARTS, SCIENCE AND COMMERCE (AUTONOMOUS)**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

Roshan Jose

Name and Signature of the Student

TABLE OF CONTENTS

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Background

In recent times the country has been struck by a pandemic, compelling us to explore for innovative ways to adapt the new normal. Authorities had to take rigorous steps as public health became a priority, resulting in restricted public movement. This has created a hostile environment in which people can't step out of their houses unless it is related to some essential work. People are not permitted to use public transportation, making social gatherings impossible. Stopover at any outlet is limited due to frequently evolving virus strains and strict lockdown. As a result, the demand for quick and secure transit-oriented delivery services has soared. Transportation is an important part of our everyday lives and is regarded as the backbone of many businesses that sustain the country's economy. Using a website that offers a reliable and safe express delivery as well as a swift bike taxi ride would not only ensure a person's safety, but also help to reduce the number of cases.

1.2 Objective

The main aim of Pronto is to provide users with a fast and secure service. It primarily focuses on people who are not able to leave their homes due to the lockdown. The courier service provided by this site allows users to send anything to anyone without having to worry about its safety. The bike taxi service of this site will ensure that the user gets a swift ride through the traffic. In addition to this, a person can comfortably sit at home and have their favored or essential items delivered to their doorsteps by using this site.

1.3 Purpose, Scope & Applicability

1.3.1 Purpose

Before the pandemic, daily tasks were much simpler and lives were at peace. However, now the lifestyle has changed and everyday activities are comparatively difficult. This is where Pronto comes in with the main purpose of providing a fast and reliable service that people can trust and use as per their needs. Pronto ensures a safe courier service without any exposure to contagious surroundings. The user can use the bike taxi service as an alternative mode of transport without becoming weary of driving or being concerned about traffic. The home delivery service allows users to order anything they need and have it delivered immediately from their local markets, which is especially important for elderly people who are currently living alone in the pandemic.

1.3.2 Scope

This website is extremely convenient for customers because they don't have to leave their homes during the pandemic. If someone wishes to send items to their friends and family without risking their own lives, they can use this site's secure courier service. The amount of traffic on Indian roads is gradually increasing, making it difficult to save time when commuting. As a result, customers will benefit from the affordable bike taxi service, which will assist them to safely arrive at their destination. Some users are worried about the quality and authenticity of online products, but with the help of this platform, consumers may place orders online and have them delivered from a trustworthy local market.

When it comes to limitations, the availability of a service may be affected if there is a high demand for a specific service from the users. Also, because the initiative is still in its early stages, the initial coverage region for the services will be limited.

1.3.3 Applicability

People nowadays seek homely comfort and thus choose online shopping where user can have their preferred products delivered to their doorsteps. With a wide variety of courier services available today, it is difficult for a user to choose one that meets all of their requirements, such as no weight limit and no type restriction. However, as this site understands people's needs, it adapts accordingly and tries to solve such problems. Car taxi services contributes in traffic congestion, resulting into longer trip times which can be avoided with Pronto's affordable bike taxi services. The online delivery services provided by this site mainly benefits the senior citizens of our community who can't step out of their homes. Thus, by using this site a user can take full advantage of swift bike taxi and express delivery services under one roof.

Chapter 2: Survey of Technologies

Web application is a client-side and server-side software application in which the client runs or request in a web browser. Web applications can be designed for a wide variety of uses and can be used by anyone; from an organization to an individual for numerous reasons. Web applications do not need to be downloaded since they are accessed through a network. Web application have three parts namely, front-end which is the application's user interface, scripting which is the logic and working part and back-end which is the database.

2.1 Front-End Development

Front-end development is the part of web development that codes and creates front-end elements of a website that user sees, touches and experiences. The frontend of an application is less about code and more about how a user will interpret the interface into an experience. Front-end languages and frameworks contribute to the overall front-end development. A front-end developer creates the overall design and aesthetic, in addition to debugging and using static code analysis.

2.1.1 Front-End Development Languages

Front end languages are programming languages used by developers on the front end of a site. With different languages used front end, the web developer's role would be able to create a theme in line with a set style of the site.

- **Hyper Text Markup Language (HTML)**

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behaviour (JavaScript). "Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. HTML uses "markup" to annotate text, images, and other content for display in a web browser.

Advantages

- It’s easy to learn and use.
- It is a popular language, it’s widely used.
- It can integrate easily with other languages
- Every browser supports HTML language.
- It is a free and open-source markup language.
- It is light weighted and fast to load.

Disadvantages

- Dynamic Pages - Creating dynamic pages are hard. By default, all HTML pages are static.
- Complexity - A lot of code can be complex to handle
- Not Centralized- Each page should be programmed separately.
- Limitations - Alone HTML does not have many capabilities, it alone can create basic web pages nothing much.

- **Cascading Style Sheets (CSS)**

Cascading Style Sheets (CSS) is a standard (or language) that describes the formatting of markup language pages. CSS enables developers to separate content and visual elements for greater page control and flexibility. A CSS file is normally attached to an HTML file by means of a link in the HTML file. In December 1998, the World Wide Web Consortium (W3C) published the first CSS specification (CSS1).

Advantages

- CSS saves time by reusing the same sheet in multiple HTML pages.
- It improves page loading speed.
- It has multiple device compatibility.

Disadvantages

- Sometimes CSS can be messy and can create complications in code.
- In CSS, there is no interaction with databases.
- CSS has cross-browser issues. It has multiple levels like CSS1, CSS2, CSS3, that are sometimes confusing for beginners.

- **JavaScript**

JavaScript is a programming language commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites. While JavaScript is influenced by Java, the syntax is more similar to C and is based on ECMAScript, a scripting language developed by Sun Microsystems.

Advantages

- Simplicity - JavaScript's syntax was inspired by Java's and is relatively easy to learn compared to other popular languages like C++.
- Popularity - JavaScript is everywhere on the web, and with the advent of Node.js, is increasingly used on the backend.
- Server Load - JavaScript is client-side, so it reduces the demand on servers overall, and simple applications may not need a server at all.
- Rich interfaces - JavaScript can be used to create features like drag and drop and components such as sliders, all of which greatly enhance the user interface and experience of a site.

Disadvantages

- Client-Side Security - Since JavaScript code is executed on the client-side, bugs and oversights can sometimes be exploited for malicious purposes. Because of this, some people choose to disable JavaScript entirely.
- Browser Support - While server-side scripts always produce the same output, different browsers sometimes interpret JavaScript code differently.
- Single Inheritance - JavaScript only supports single inheritance and not multiple inheritance. Some programs may require this object-oriented language characteristic.

2.1.2 Front-End Frameworks

Front-End Frameworks are packages containing pre-written, standardized code in files and folders which give users a base to build on while still allowing flexibility with the final design. It has a grid which makes it simple to organize the design elements of your website along with pre-built website components like side panels, buttons, and navigation bars.

- **React**

React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces or UI components. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications.

Advantages

- It facilitates the overall process of writing components
- It boosts productivity and facilitates further maintenance
- It ensures faster rendering
- It guarantees stable code

Disadvantages

- Poor Documentation - React technologies updating and accelerating so fast that there is no time to make proper documentation.
- High pace of development - React is still a quite new technology and it evolves blazingly fast. Therefore, it may be hard for some to keep up this pace because of new features coming out and old ones being forgotten.
- Incompleteness - React provides only the View part of the MVC model. Because of that, you will have to rely on other technologies, too.

- **AngularJS**

AngularJS is a JavaScript-based open-source front-end web framework for developing single-page applications. It is maintained mainly by Google and a community of individuals and corporations. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel (MVVM) architectures, along with components commonly used in web applications and progressive web applications.

Advantages

- Two way data binding: Two-way data binding in Angular will help users to exchange data from the component to view and from view to the component

- Use of typescript: Angular is written using TypeScript, which is a superset for JavaScript. It fully complies to JavaScript and also helps spot and eliminate common mistakes while coding.
- Dependency injection: it is one of the most important application design patterns and angular has its own dependency injection framework.

Disadvantages

- Too many versions: Angular has been evolving. It started out as a JavaScript framework AngularJS and now the current version is Angular 9 with a total of 6 major releases in between. This can cause confusion especially for beginners.
- Decline in popularity: With the advent of newer frameworks like VueJS and ReactJS, Angular has seen a downfall in its popularity.
- Steep learning curve: Another glaring con of using the Angular framework is that it can be quite difficult to learn as it has complex web of modules, coding languages, integrations and customizing capabilities.

- **Vue.js**

Vue.js is an open-source model–view–viewmodel front end JavaScript framework for building user interfaces and single-page applications. It was created by Evan You, and is maintained by him and the rest of the active core team members. Vue.js features an incrementally adaptable architecture that focuses on declarative rendering and component composition. The core library is focused on the view layer only. Advanced features required for complex applications such as routing, state management and build tooling are offered via officially maintained supporting libraries and packages.

Advantages

- Approachable: Vue is quite natural for the even fresher developer, having basic knowledge of HTML, CSS and JavaScript can start a project on Vue platform.
- Versatile: Vue.js is an incrementally adoptable ecosystem, helping developers with versatile or adaptable nature
- Performance: Vue makes development absolutely easy as the production-ready project which weighs 20KB resulting in faster runtime.
- Ease of integration: Being an approachable, versatile, and performant nature, Vue helps developers create fantastic applications which are maintainable and testable.

Disadvantages

- Less Plugins/Components: Common plugins are useful as they work with various other tools to make development easy. Vue.js does not have most of the common plugins
- Issues with iOS & Safari: If you are using Vue.js app on older version iOS and Safari, then you are supposed to get some problems, though they are fixable.
- Vue.js Evolving Fast: What is working in Vue today may be outdated tomorrow.

2.2 Back-End Development

The backend (server-side) is the portion of the website you don't see. It's responsible for storing and organizing data, and ensuring everything on the client-side actually works. The backend communicates with the front-end, sending and receiving information to be displayed as a web page. When a user fills out a contact form, type in a web address or make a purchase, the browser sends a request to the server-side, which returns information in the form of frontend code that the browser can interpret and display.

2.2.1 Server-side scripting languages

Server-side scripting languages create the scripts that run on the server and hence minimize the workload of a browser. The functionality of your website is written in those scripting languages. One of the commonly used server-side scripting languages are Python, Ruby, JavaScript.

- **NodeJS**

Node.js is an open-source, cross-platform, back-end, JavaScript runtime environment that executes JavaScript code outside a web browser. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications. Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server- and client-side scripts.

Advantages

- Easy scalability for modern applications.
- Cost-effective with Full Stack JS community.
- Support to simplify development.

- Improves app response time and boosts performance.
- Reduces loading time by quick caching.
- Helps in building cross-platform applications

Disadvantages

- Reduces performance when handling heavy computing tasks.
- Heavy code changes due to Unstable API.
- Node.js asynchronous programming model makes it difficult to maintain code.
- Lack of library support can endanger your code.
- High demand with a few experienced Node.js developers.

- **Ruby**

Ruby is a dynamic, reflective, object-oriented, general-purpose programming language. Ruby is a pure Object-Oriented language developed by Yukihiro Matsumoto. Everything in Ruby is an object except the blocks but there are replacements too for it i.e. procs and lambda. The objective of Ruby's development was to make it act as a sensible buffer between human programmers and the underlying computing machinery.

Advantages

- Highest standard of safety: it is considered a very secure technology with built in securities and functionalities
- Ruby is a dynamic programming language. It is also very close to spoken languages.
- Well suited for web development, system administration, scripting, back-end development.
- Has a large variety of libraries and plugins to work with.

Disadvantages

- It is flexibility means it can be difficult to debug,
- It has few use cases other than web development,
- Slower than some other languages; it depends requirements.
- It's been declining in popularity.

- **Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

Advantages

- Easy to Read, Learn and Write: Python is a high-level programming language that has English-like syntax. This makes it easier to read and understand the code.
- Improved Productivity: Python is a very productive language. Due to the simplicity of Python, developers can focus on solving the problem.
- Interpreted Language: Python is an interpreted language which means that Python directly executes the code line by line.
- Dynamically Typed: Python doesn't know the type of variable until we run the code.

Disadvantages

- Slow Speed: We discussed above that Python is an interpreted language and dynamically-typed language. The line-by-line execution of code often leads to slow execution.
- Not Memory Efficient: To provide simplicity to the developer, Python has to do a little trade-off. The Python programming language uses a large amount of memory
- Weak in Mobile Computing: Python is generally used in server-side programming. Python is not memory efficient and it has slow processing power as compared to other languages.
- Database Access: Programming in Python is easy and stress-free. But when we are interacting with the database, it lacks behind.

2.2.2 Backend Frameworks

A framework is the base of an easy and structured way for developers to create applications**.** Backend frameworks are the libraries of server-side languages that aid in building the server configuration of any website. These frameworks are constituted by a variety of elements that allows developers to work more rapidly.

- **ExpressJS**

Express.js is a free and open-source web application framework for Node.js. It is used for designing and building web applications quickly and easily. Web applications are web apps that you can run on a web browser. Since Express.js only requires javascript, it becomes easier for programmers and developers to build web applications and API without any effort. It is lightweight and helps to organize web applications on the server-side into a more organized MVC architecture.

Advantages

- Allows you to define routes of your application based on HTTP methods and URLs.
- Easy to integrate with different template engines like Jade, Vash, EJS etc.
- Allows you to define an error handling middleware.
- Easy to serve static files and resources of your application.
- Allows you to create REST API server.

Disadvantages

- A beginner won't know the optimal way to structure their server code and can easily write code that slows down their REST API unnecessarily, or worse.
- Express doesn't dictate how security is implemented - that's entirely up to the app.
- Error messages are usually unhelpful and it’s hard to debug.

- **Ruby on Rails**

Ruby on Rails is software code built on top of Ruby. Technically, it is defined as a package library called RubyGem, installed using the command line interface of the operating system. Ruby on Rails is an open-source web development framework, which provides Ruby developers a timesaving alternative to develop code. It is a collection of code libraries, which offer a ready-made solution for repetitive tasks like developing tables, forms or menus on the website.

Advantages

- It’s time-efficient - Ruby on Rails contains many ready-made plugins and modules, which allow developers not to waste time on writing boilerplate code.
- It’s consistent - Developers follow standardized file storage and programming conventions that keep a project structured and readable.

- It provides excellent quality and promotes bug-free development - The Minitest tool built into the Rails core is a comprehensive test suite that provides many useful testing features including expectation syntax, test benchmarking, and mocking.

Disadvantages

- Runtime Speed and Performance
- Lack of Flexibility
- Multithreading – Rails supports multithreading, though some of the IO libraries do not, as they keep hold of the GIL (Global Interpreter Lock).

- **ASP.NET**

ASP.NET is a unified web development model integrated with .NET framework, designed to provide services to create dynamic web applications and web services. It is built on the Common Language Runtime (CLR) of the .NET framework and includes those benefits like multi-language interoperability, type safety, garbage collection and inheritance. ASP.NET uses the code-behind model to generate dynamic pages based on Model-View-Controller architecture.

Advantage

- One of the best things about .NET is that it is based on object-oriented programming (OOP). This is where the software is divided into smaller chunks, which then allows developers to work on them one at a time
- Easy to Deploy and Maintain - The deployment of applications and their maintenance could not be done easier with the help of the .NET family of development tools.
- Developing applications using the .NET family means not having to redevelop the same applications for each new platform.

Disadvantage

- Licensing cost - Many aspects of the .NET family will cost money in terms of licensing fees and they can stack up. The more demanding the project, the more expensive it can get.
- Not so many supported platforms like Java. For example, Java can run on IBM mainframes, .NET cannot.
- Vendor lock in - It’s an unfortunate fact that since the .NET bundle is under Microsoft, any changes or limitations that the company might impose will inevitably impact projects done under the framework.

2.2.3 Database Management languages

Database Languages are the set of statements, that are used to define and manipulate a database. A Database language has Data Definition Language (DDL) which is used to construct a database, Data Manipulation Language (DML) which is used to access a database, Data Control Language (DCL) which is used to retrieve the stored or saved data and Transaction Control Language (TCL) which is used to run the changes made by the DML statement.

- **MongoDB**

MongoDB is a cross-platform and open-source document-oriented database, a kind of NoSQL database. As a NoSQL database, MongoDB shuns the relational database's table-based structure to adapt JSON-like documents that have dynamic schemas which it calls BSON. This makes data integration for certain types of applications faster and easier. MongoDB is built for scalability, high availability and performance from a single server deployment to large and complex multi-site infrastructures.

Advantages

- Schema less − MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- Conversion/mapping of application objects to database objects not needed.
- Uses internal memory for storing the (windowed) working set, enabling faster access of data.

Disadvantages

- Joins not Supported - MongoDB doesn't support joins like a relational database. Yet one can use joins functionality by adding by coding it manually. But it may slow execution and affect performance.
- High Memory Usage - MongoDB stores key names for each value pairs. Also, due to no functionality of joins, there is data redundancy. This results in increasing unnecessary usage of memory.

- Limited Data Size - You can have document size, not more than 16MB.
- Limited Nesting - You cannot perform nesting of documents for more than 100 levels.

- **MySQL**

MySQL is a freely available database system. However, there are several paid editions also available with which you can use advanced functionality. MySQL is easy to use as compared to other database software such as Microsoft SQL Server and Oracle database etc. It can be used with any programming language, but is largely used with PHP. MySQL can run on multiple platforms such as Linux, Windows, Unix, and an information schema to define and manage your metadata. It is a really flexible, scalable, fast, and reliable solution.

Advantages

- MySQL is a Relational Database Management System or RDBMS which means that it stores and presents data in tabular form, organized in rows and columns.
- MySQL is more secure as it consists of a solid data security layer to protect sensitive data from intruders and passwords in MySQL are encrypted.
- MySQL is compatible with most of the operating systems, including Windows, Linux, NetWare, Novell, Solaris and other variations of UNIX.
- MySQL provides the facility to run the clients and the server on the same computer or on different computers, via internet or local network.

Disadvantages

- MySQL is not very efficient in handling very large databases.
- MySQL doesn't have as good a developing and debugging tool as compared to paid databases.
- MySQL versions less than 5.0 do not support COMMIT, stored procedure and ROLE.
- MySQL is prone to data corruption as it inefficient in handling transactions.

- **PostgreSQL**

PostgreSQL also known as Postgres, is a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance. It was originally named POSTGRES, referring to its origins as a successor to the Ingres database developed at the University of California, Berkeley. PostgreSQL features transactions with Atomicity, Consistency, Isolation,

Durability (ACID) properties, automatically updatable views, materialized views, triggers, foreign keys, and stored procedures.

Advantages

- PostgreSQL's write-ahead logging makes it a highly fault-tolerant database
- PostgreSQL source code is freely available under an open-source license. This allows you the freedom to use, modify, and implement it as per your business needs.
- PostgreSQL supports geographic objects so you can use it for location-based services and geographic information systems
- Low maintenance and administration for both embedded and enterprise use of PostgreSQL

Disadvantages

- Postgres is not owned by one organization. So, it has had trouble getting its name out there despite being fully featured and comparable to other DBMS systems
- Changes made for speed improvement requires more work than MySQL as PostgreSQL focuses on compatibility
- Many open-source apps support MySQL, but may not support PostgreSQL
- Lack of index-oriented tables makes certain workloads such as job queues difficult.

- **Technologies used in this project**

The website Pronto is developed using MERN stack that is basically MongoDB, Express JS, React and NodeJS along with HTML CSS and JavaScript. MERN is one of the best stack development suites with MongoDB in backend for increased scalability, Express JS for speed improvements and JavaScript language for end-to-end development. When it comes to front-end, React is the best since it gives speedier code development. React is better than Angular in terms of UI rendering and performance since it allows users to freely design applications and organise code. Since Pronto is using full stack development approach, it aids in cost effectiveness, speed of delivery and easy maintenance of the website.

Chapter 3: Requirement and Analysis

3.1 Problem Definition

In today's world our lives revolve around technology. It is clear that technology has improved from start to finish considering the emergence of new technologies, making people's lives more comfortable than in the past. Today majority of tasks can be completed while sitting comfortably at home. This was not the case a few years ago; people had to put in a lot of extra work to complete simple chores like buying groceries, searching for best transit system, delivering items to friends. All these tasks were time consuming and required lot of efforts, but by using modern technologies and services such tasks can be completed quickly with minimal effort. Many online sites have sprung up recently where one can order items and have them delivered to their doorsteps, but such services too have some limitations. That's why Pronto was created, with the goal of providing the best service to its users while also serving as an all-in-one platform.

3.1.1 Problems with Traditional Delivery and Transportation service

Less variety of choices

A few decades ago, the market had a limited number of products. People were forced to choose from the available option without having a choice to customize as per their convenience. In order to avail more options, a user had to invest more time, manpower, and money, with no assurance of receiving better service. Lack of choices led to inefficiency of transport and delivery services resulting in minimal benefits.

Lack of tracking facility

Previously, because of lack in technology people were not able to track orders. In order to know the status of the item, the customer had to frequently visit the courier service office. Users had difficulty determining where the package had arrived and how much longer it would take to reach its destination. There may be instances where the package is delivered to the incorrect address or is lost in transit.

Time Consuming

Technology plays an essential role in saving usage time, but this was not the case few years ago. When technology was not the part of our lives, ordinary chores that now take only a few minutes would take hours or even a day to complete, resulting in time waste. Traveling to a common centre to purchase goods that were located in a distant location also took more time.

Manual Labour

Using courier service as an example, a person had to put in a lot of effort just to find out the status of their courier, which not only wastes time but also necessitates a significant amount of labour to complete the task. On the contrary, thanks to technological advancements, the task can now be completed quickly, in a short amount of time, just a few clicks away.

3.1.2 Problems with Existing Delivery and Transportation service

Different sites for different services

In current scenario due to lack of all-in-one services, users have to surf through many websites which results in time consumption and sloppiness. User have to register in different websites which sometimes becomes tedious and risky. It becomes difficult to manage and remember all the login credentials of multiple accounts. Simultaneous use of different websites sometimes results in inconveniency for the users.

Unauthenticated users

As technology advances, an increasing amount of data is collected. When a large number of users register in website, it becomes challenging to handle it appropriately. These unhandled data lead to high risk of authenticity. Such unauthenticated users can cause unauthenticated attack searches for vulnerabilities on a network system without actually having to log in as an authorized user.

Few Bike Taxi services

When a person travelling alone calls for a taxi, he/she wishes to find the most convenient option to reach their destination but ends up choosing a cab or auto rickshaw which results into underutilization of services being provided. In such cases bike taxi can be efficiently used as it can

overcome traffic and high fare problems. Hence more bike taxi services should be introduced in the market.

Weight and Size limit

Many existing courier services restrict weight and size of the package which is to be delivered. Such restrictions tend to decrease the number of users preferring courier service. If size and weight limit on a package is not applied then there would be a drastic increase in demand for this service. For the type restriction applied on packages for security purposes, there can be individual examination of each packet so weight and size limit are not necessary.

3.2 Requirement Specification

Pronto primarily focuses on user convenience by offering services such as home-to-home courier service, dependable bike taxi service, and online delivery service. Since Pronto is a web-based application, users can access it from any location without any installation. The easy-to-use interface allow seamless navigation throughout the site. A user can utilise Pronto's services by providing minimal information and utilizing maximum benefit from it.

A user can conveniently sit at home and take full advantage of online delivery service by using this feature-rich platform. The website allows users to use the courier service, which delivers items to the recipient's location by only providing the address and contact information. The growing number of automobiles causes longer travel times and traffic congestion, but by using Pronto's bike taxi service, customers can avoid traffic and arrive at their destination much faster. The website will request the user's pickup and destination points, as well as their contact information. Once the item or ride has been booked, the admin can view the order details and validate it as needed.

The user interface of the website has a deep understanding of users and their needs, as well as what they value and their constraints. It evaluates the services and tries to bring the best out of them as a result of confirming all the user's requirements.

3.3 Planning and Scheduling

3.3.1 Planning

Proper implementation of a plan will have a significant impact on an organization's success. Even the most basic website requires a significant amount of preparation and research before any design or development can begin.

The user-friendly web application, Pronto, can be viewed in two ways, one which is specified for the user and the other which is for the administrators. In user view, users can surf through the website and use the services offered by Pronto. The admin view can be subdivided further into two views; one is for the delivery partners in which they can see the pending orders and their details and another is for the actual admins in which they login and inspect all the pending or completed order details as well as view and resolve any issues raised by the user.

The users can either register themselves to create an account on the website or can start using the website's services directly without registering. However, in either case, the user must submit certain information such as their address, contact information, and so on. This application allows the user to book any service; once a service is booked, the admin is notified and the booking is confirmed. The user may also view the status of their purchase in order status section. In the event of a problem, user can raise an issue within the website and the administrator will look into it. However, a person who registers in as an administrator will be able to validate, process, and view any issues raised by the users. They can also view the user's service request and can evaluate the bookings depending on the availability.

3.3.2 Scheduling

Gantt Chart

A gantt chart is a horizontal bar chart used in project management to visually represent a project plan over time. Modern gantt charts typically show you the timeline and status—as well as who’s responsible—for each task in the project. In other words, a gantt chart is a super-simple way to communicate what it will take to deliver a project on time and budget. That means it’s a whole lot easier to keep your project team and stakeholders on the same page from the get-go. Reading a gantt chart really comes down to understanding how the different elements come together to make a gantt chart work.

To summarize, Gantt charts will allow you to:

- Know what the different activities are
- When activities need to start, and when they need to finish.
- How long your activities were primarily scheduled to last.
- How much your activities cost, and how much your project is likely to cost.
- Which activities depend on others to be effectively completed.
- Current progress status.
- The duration of the entire project.
- Know who is responsible for each task
- Find and take care of any problems associated with the project

The below Gantt chart shows the time taken to complete each chapter.

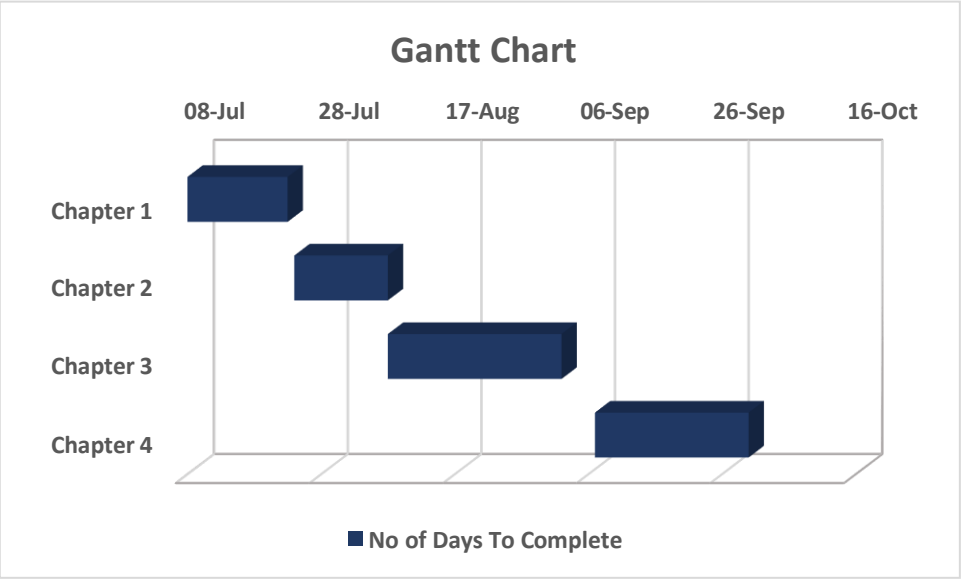


Figure 3.1: Gantt Chart

PERT Chart

A PERT chart is a visual project management tool that's useful for mapping out project tasks and planning the overall project schedule. Many people confuse PERT with PERT chart, so the best way to provide a comprehensive PERT chart definition is to start with clarifying both of these terms. PERT stands for Program Evaluation Review Technique. PERT is the actual technique that is used to create a PERT chart. Meanwhile, a PERT chart is the visual diagram that results from using PERT. Think of PERT as the process and the PERT chart as the outcome. PERT charts allow project managers to see essential scheduling details such as task dependencies, task duration estimates, and the minimum amount of time a project can be completed within. But, they're not the most user-friendly or well-understood tool.

The below PERT diagram shows the schedule of the following chapters.

Chapter 1: A **Chapter 2: B** **Chapter 3: C** **Chapter 4: D**

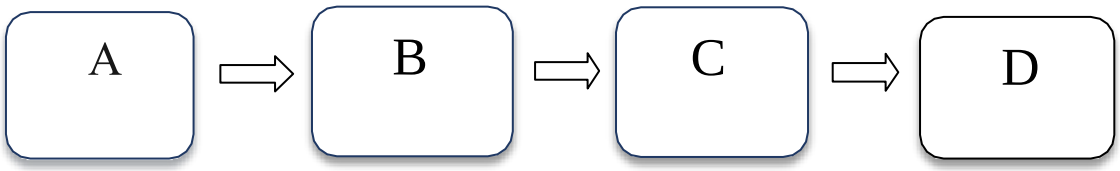


Figure 3.2: Activity

Activity	Precedence	Duration(weeks)
A	**-**	**2**
B	**A**	**2**
C	**B**	**4**
D	**C**	**3**

Table 3.1: Critical Activity

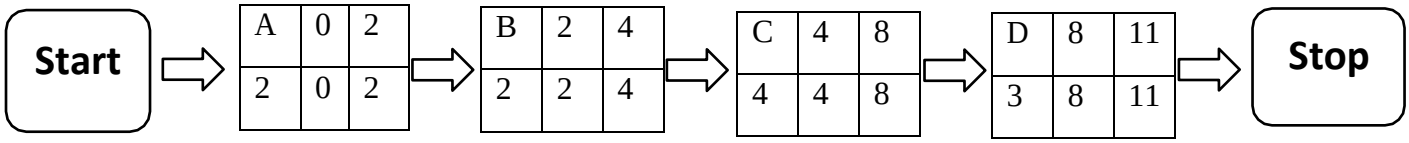


Figure 3.3: Critical Activity

Slack = 0 for all the activities.

Therefore, Critical Path = A-B-C-D

3.4 Software and Hardware Requirements

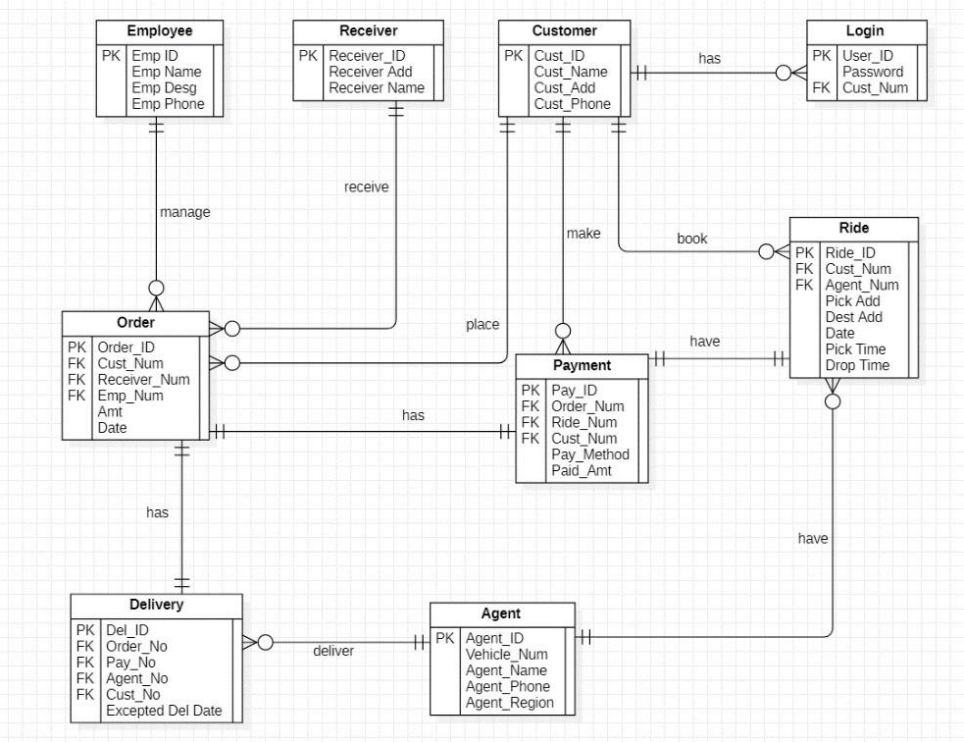
3.4.1 Software Requirements

- Adobe Photoshop
- Visual Code
- Mongo DB
- Node JS

3.4.2 Hardware Requirements

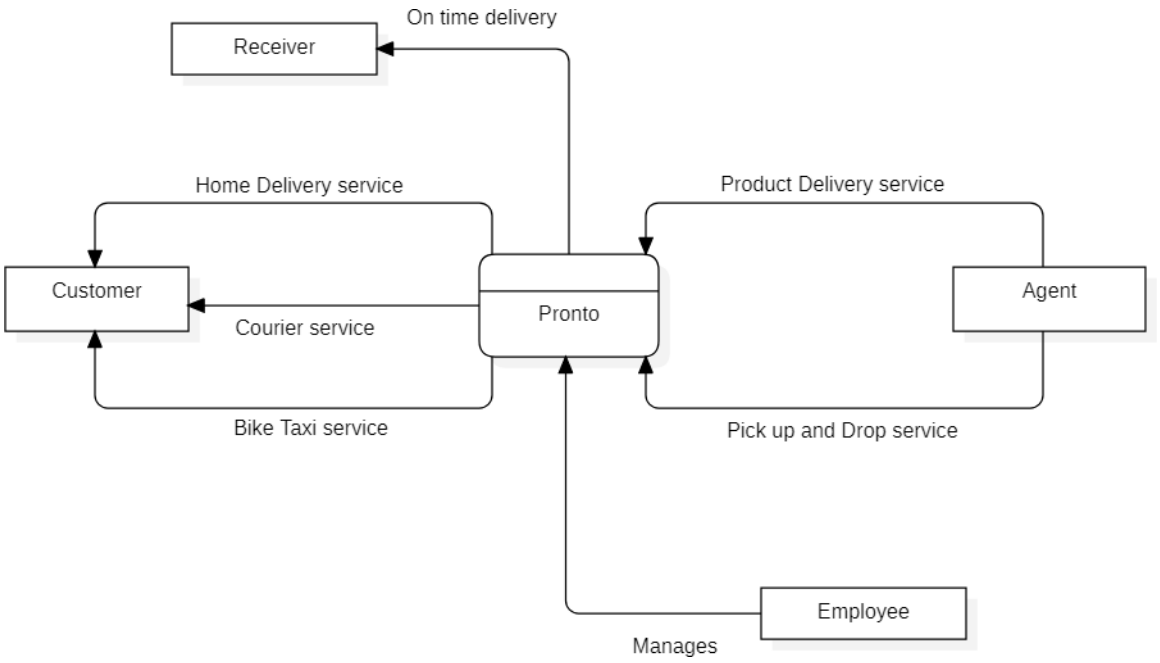
- CPU: Intel Core i5-8259U or higher
- RAM: 6GB or Higher
- Storage: 10GB
- Direct X: Version 11

3.5 Conceptual Models

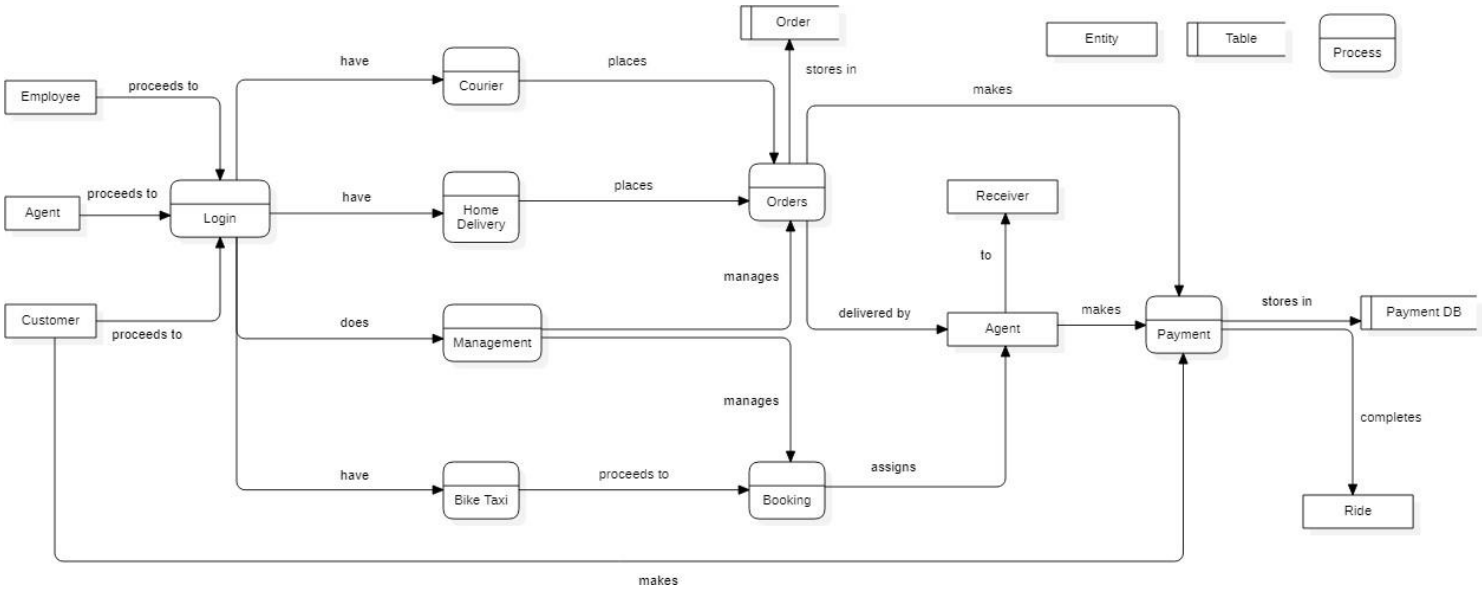


3.5.1 ER Diagram

Level 0



Level 1



3.5.2 Data Flow Diagram

Chapter 4: System Design

4.1 Basic Modules:

Including the homepage there are in total 9 modules:

1. **Homepage**

This page will have an introduction to the website. Through this page, they would straight away know what we are offering them. There will also be a footer where the user will be able to see the social media icons which the site's social media accounts will be linked to and contact information to the company.

2. **Registration module**

This module is for those users who are new to this website. They have to submit their details which will be stored in the database in order to utilize all the services efficiently.

3. **Login module**

This module will have a login form. The registered users will enter their login credentials like user id and password.

4. **Order module**

This page will contain all the orders placed by the user. The user can review their orders and can get frequent updates. The employee will manage all the orders. The agents can view the pending orders and assigned orders.

5. **Employee module**

The employee will have the right to verify and assign orders according to resources available. In case any issues are raised by the customers or agents the employee will be able to see and solve the issues.

6. **Customer module**

The customers can either login or directly view this module. They can place order and review orders, raise any issues, and make payments.

7. **Agent module**

The agent needs to login by giving username and password. The agents can view the assigned orders, completed orders, pending orders and details related to order**.**

8. **Bike Taxi module**

In this module the customer can book bike taxi services and pay for the same, and agents can view the booking details and proceed accordingly.

9. **Delivery Module**

This module comprises of two services, courier service and home delivery. The customer can choose any of the services as per their needs and place order and make payment.

4.2 Scheme Design

Primary Key	Emp ID	VarChar(50)
	Emp Name	VarChar(50)
	Emp Desg	VarChar(50)
	Emp phone	Number

4.2.1 Employee Details

Primary Key	Cust_Id	VarChar(50)
	Cust_Name	VarChar(50)
	Cust_Add	VarChar(50)
	Cust_Email	VarChar(50)
	Cust_Phone	Number

4.2.2 Customer Details

Primary Key	Receiver Id	VarChar(50)
	Receiver Add	VarChar(50)
	Receiver Name	VarChar(50)

4.2.3 Receiver Details

Primary Key	Order_id	VarChar(50)
Foreign Key	Cust_num	VarChar(50)
Foreign Key	Receiver_num	VarChar(50)
Foreign Key	Emp_num	Number
	Amt	Number
	Date	Date

4.2.4 Order Details

Primary Key	User_id	VarChar(50)
	Password	VarChar(50)
Foreign Key	Cust_ID	VarChar(50)

4.2.5 Login Details

Priamry Key	Pay_id	VarChar(50)
Foreign Key	Order_num	VarChar(50)
Foreign Key	Ride_num	VarChar(50)
Foreign Key	Cust_num	VarChar(50)
	Pay_method	Number
	Paid_amt	Number

4.2.6 Payment Details

Primary Key	Ride_id	VarChar(50)
Foreign Key	Cust_num	VarChar(50)
Foreign Key	Agent_num	VarChar(50)
	Pick Add	VarChar(50)
	Dest Add	VarChar(50)
	Vehicle_no	VarChar(50)
	Date	Date
	Pick Time	Time
	Drop Time	Time

4.2.7 Ride Details

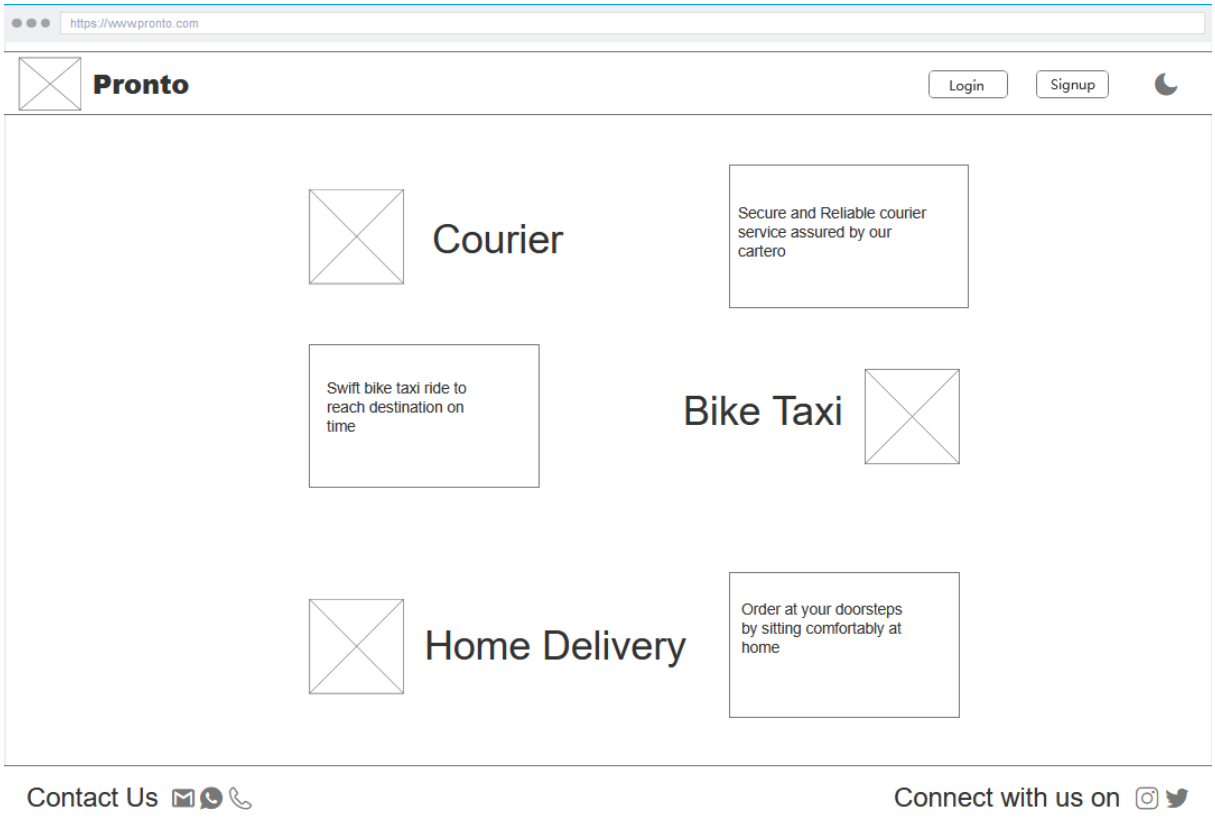
Primary Key	Agent_id	VarChar(50)
	Agent_name	VarChar(50)
	Agent_phone	Number
	Agent_region	VarChar(50)
	Vehicle_num	VarChar(50)

4.2.8 Agent Details

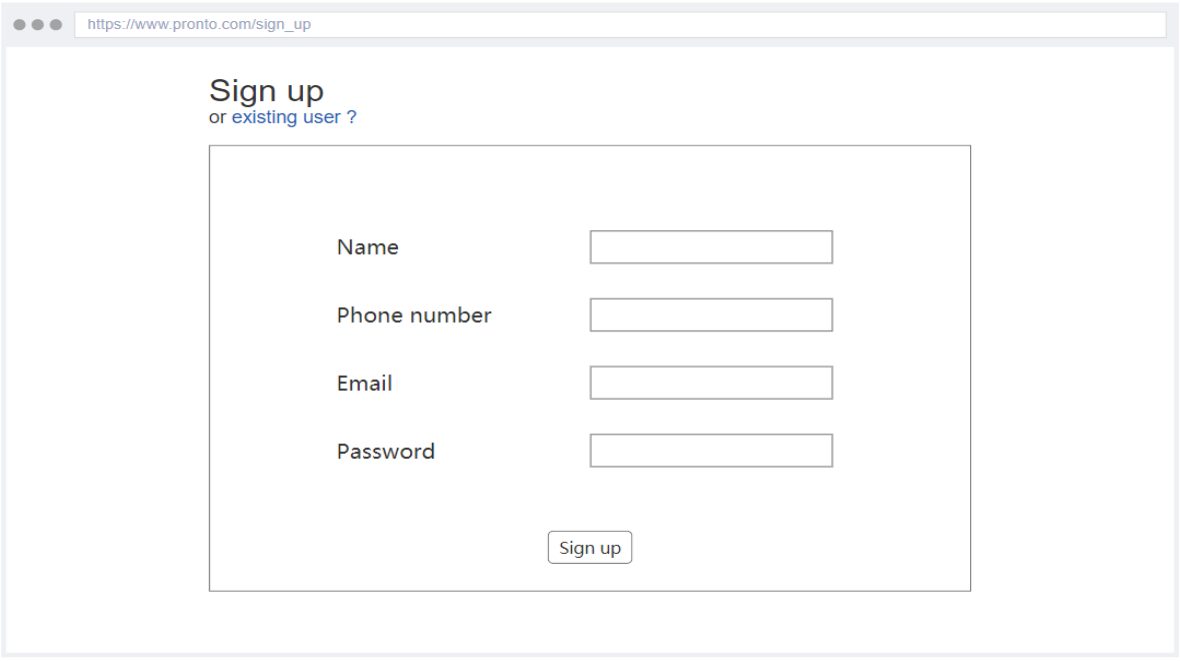
Priamry Key	Del_id	VarChar(50)
Foreign Key	Order_no	VarChar(50)
Foreign Key	Pay_no	VarChar(50)
Foreign Key	Agent_no	VarChar(50)
Foreign Key	Cust_no	VarChar(50)
	Expected Date	Date

4.2.9 Delivery Details

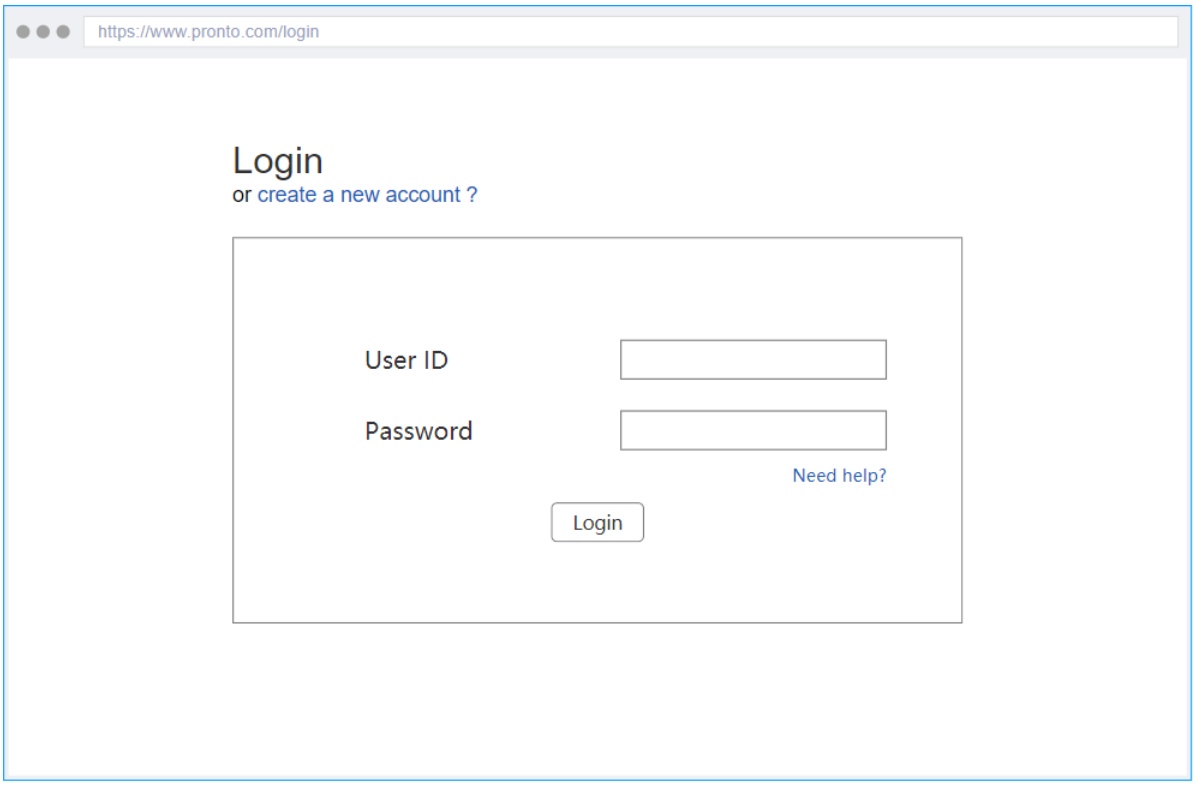
4.3 User Interface Designs



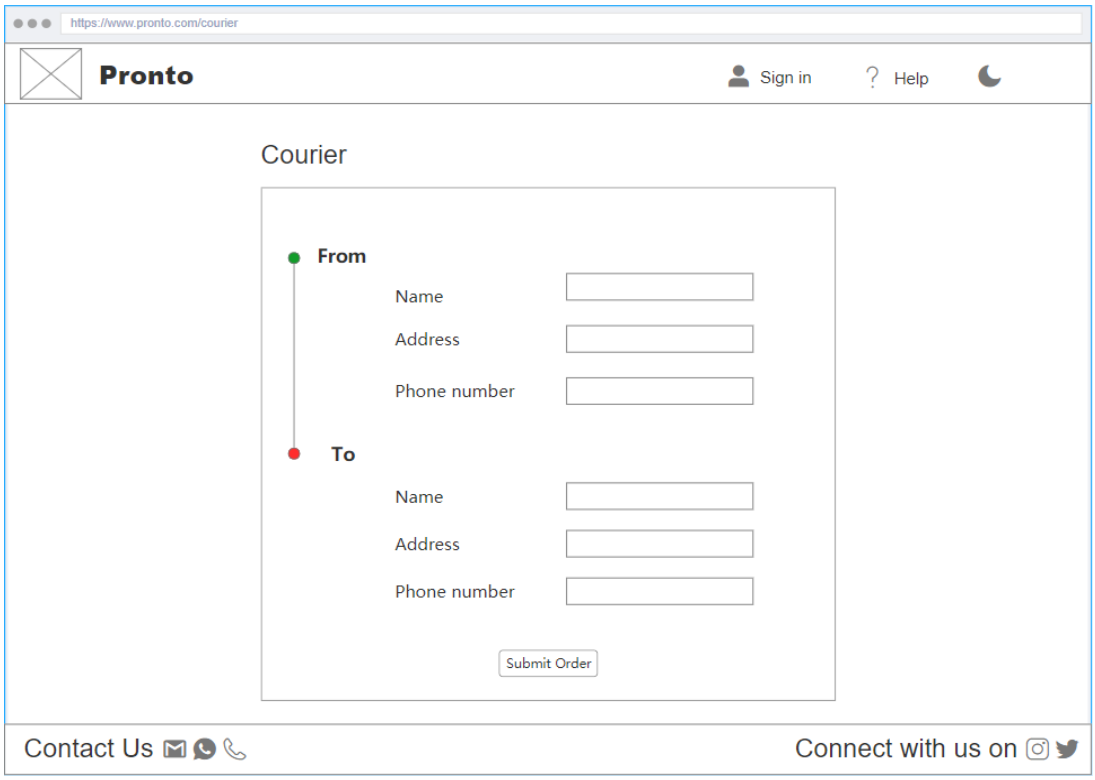
4.3.1 Homepage



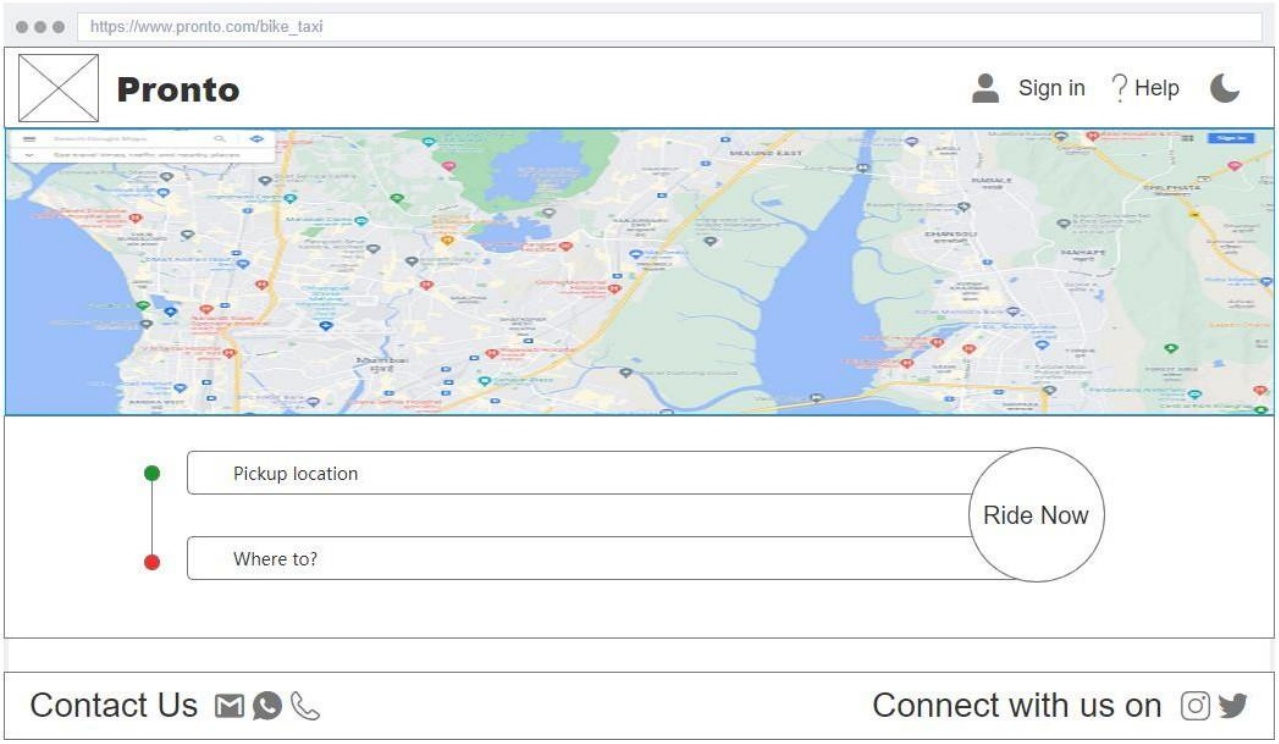
4.3.2 Sign up



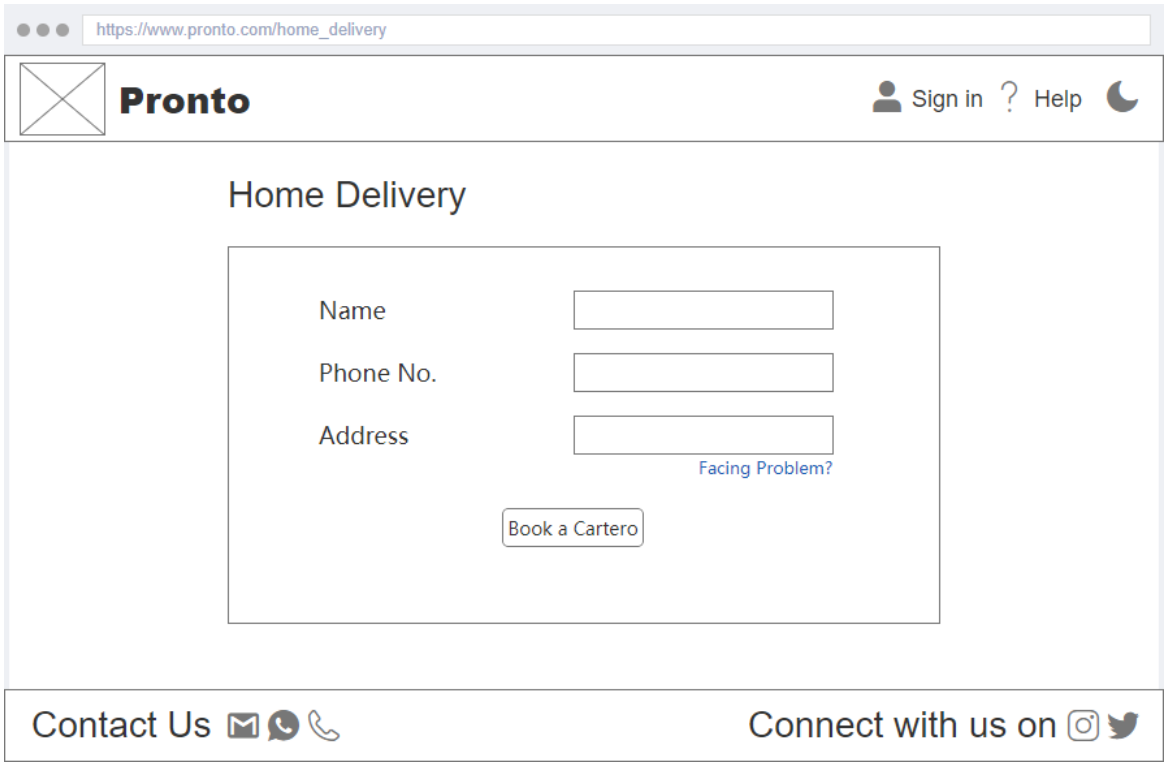
4.3.3 Login



4.3.4 Courier



4.3.5 Bike Taxi



4.3.6 Home Delivery

4.4 Security Issues

1. Security Misconfiguration

Security misconfiguration includes a variety of vulnerabilities that are all related to a lack of maintenance or attention to the web application configuration. For the application, frameworks, application server, web server, database server, and platform, a secure configuration must be created and deployed. Misconfigured security offers hackers access to private data or features and can lead to a complete system compromise.

2. Broken authentication

Broken authentication flaws also target user access. Hackers can breach the information that certifies a user's identity in this situation, such as by stealing passwords, keys, or session tokens

3. Components with known vulnerabilities

Every web application is dependent on other components to function. The list of Common Vulnerabilities and Exposures (CVE) comprises all known security flaws and hostile actors are aware of the list and use it for their evil purpose.

4. Failure to restrict URL access

This vulnerability, like many others in web application security, is related to access control privileges. URL restrictions are used by applications to prevent non-privileged users from gaining access to privileged data and resources. In case this is not properly enforced, malicious actors can access internal systems and can compromise sensitive data.

5. Cryptographic Failures

Failures related to cryptography often leads to sensitive data exposure or system compromise One relies on cryptographic algorithms and protocols every day for secure communication over the Internet. But this tool that we all rely upon so heavily is also quite brittle. Bad actors use numerous other weak links to break cryptographic software such as bugs in crypto libraries, operating systems and apps, bad design, misconfigurations, or insecure default configurations.

References

https://reactjs.org/tutorial/tutorial.html

https://www.tutorialspoint.com/mongodb/index.htm

https://nodejs.dev/learn/introduction-to-nodejs

https://www.w3schools.com/html/

https://www.w3schools.com/css/

https://learnjavascript.online/

https://owasp.org/www-project-top-ten/

SIES COLLEGE OF ARTS, SCIENCE & COMMERCE(AUTONOMOUS)
(Affiliated to University of Mumbai)
SION(W), MUMBAI-400022

DEPARTMENT OF INFORMATION TECHNOLOGY

SIES College of Arts, Science & Commerce
RISE WITH EDUCATION
Sion (West), Mumbai – 400022.
(Autonomous)

CERTIFICATE

This is to certify that the project entitled, **"Pronto"**, is bonafied work of **ROSHAN JOSE** bearing Seat No: (**TIT2122077**) and the documentation of the project is submitted online in Microsoft Teams in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

28/3/22

Internal Guide
Examiner

Date: 28/03/2022

S.I.E.S. COLLEGE OF ARTS, SCIENCE & COMMERCE MUMBAI 400 022

College Seal

28/3/22

External

TABLE OF CONTENTS

List of Tables

List of Figures

Changes made in chapter 1, 2, 3, 4

Chapter 1: Introduction

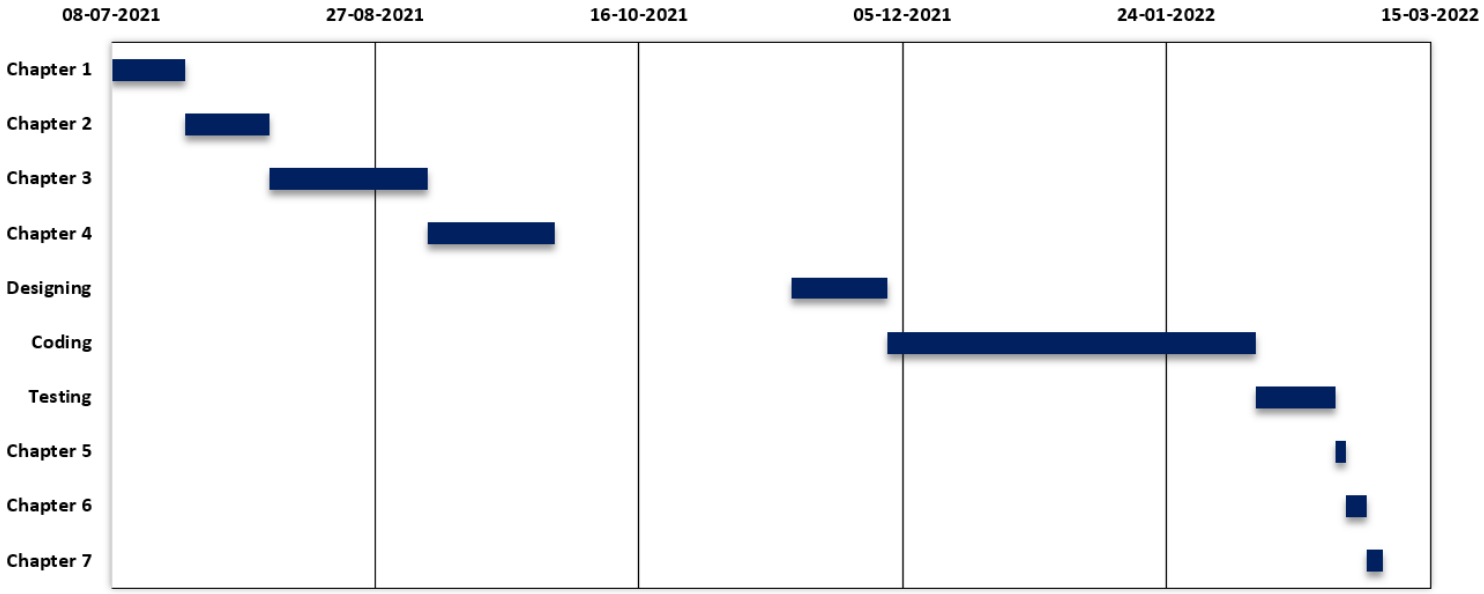
The car service provided by this site allows users to leave their car with a mechanic without having to worry about it. Pronto ensures a safe car service without any exposure to contagious surroundings. This site's car service can be used by anyone who wants to service their car without devoting their important time to it. With so many automobile services to pick from, it may be tough for a user to choose one that fulfils all of their needs.

Chapter 3: Requirements and Analysis

In order to know the status of their car, the customer had to frequently visit the service station. Users had difficulty in determining whether the required car service is available in that particular station or not. Also, there may be instances where the planned car service is not completed on time. Using car service as an example, a person had to put in a lot of effort just to find out the status of their service, which not only wastes time but also necessitates a significant amount of labor to complete the task.

With Covid-19, there is a growing need for contactless service, at-home vehicle repair, and shorter turnaround times. As more individuals use their cars to go outside of their cities on a daily basis, the desire for more dependability and professional service is growing. In large service stations, your automobile is one of a hundred that has to be inspected. As a result, the human touch has been gone and the user will have no idea which mechanic is working on your vehicle. By using this website, a user can order a car service that will be delivered to their area by simply entering their address and vehicle information on the website.

Gantt Chart



3.1 Gantt Chart

PERT Chart

Chapter 1 – A **Chapter 2 – B** **Chapter 3 – C** **Chapter 4 – D** **Designing – E**

Coding – F **Testing – G** **Chapter 5 – H** **Chapter 6 – I** **Chapter 7 – J**

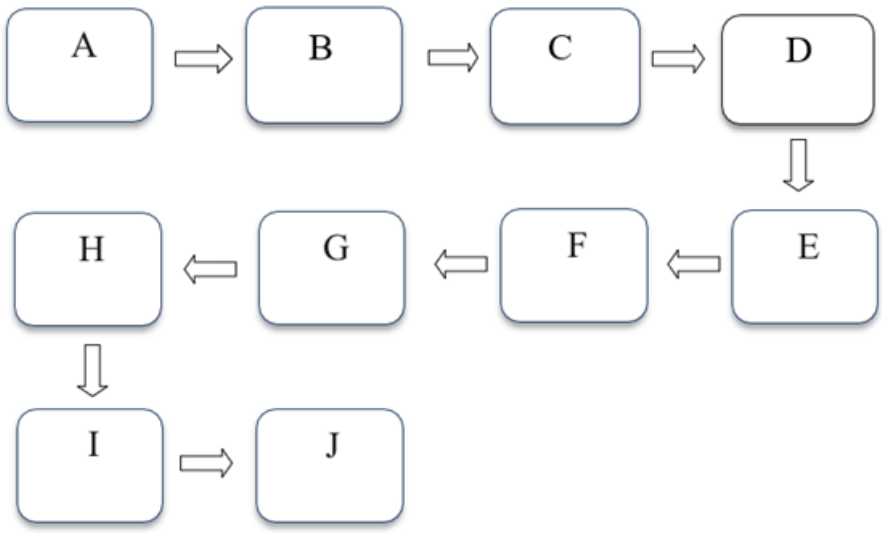


Figure 3.2 Activity

Activity	Precedence	Duration(weeks)
A	_	2
B	A	2
C	B	4
D	C	3
E	D	3
F	E	10
G	F	2
H	G	1
I	H	1
J	I	1

Table 3.1 Critical Activity

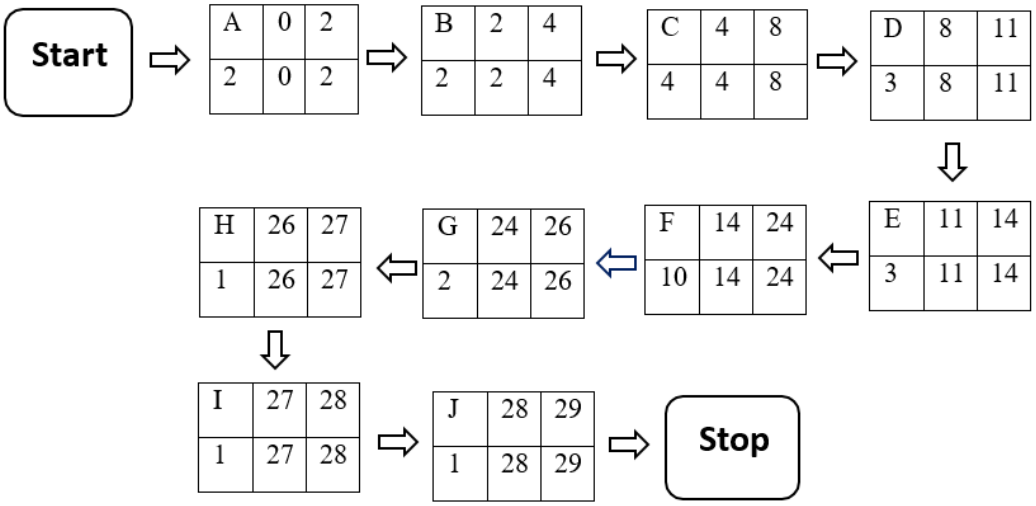
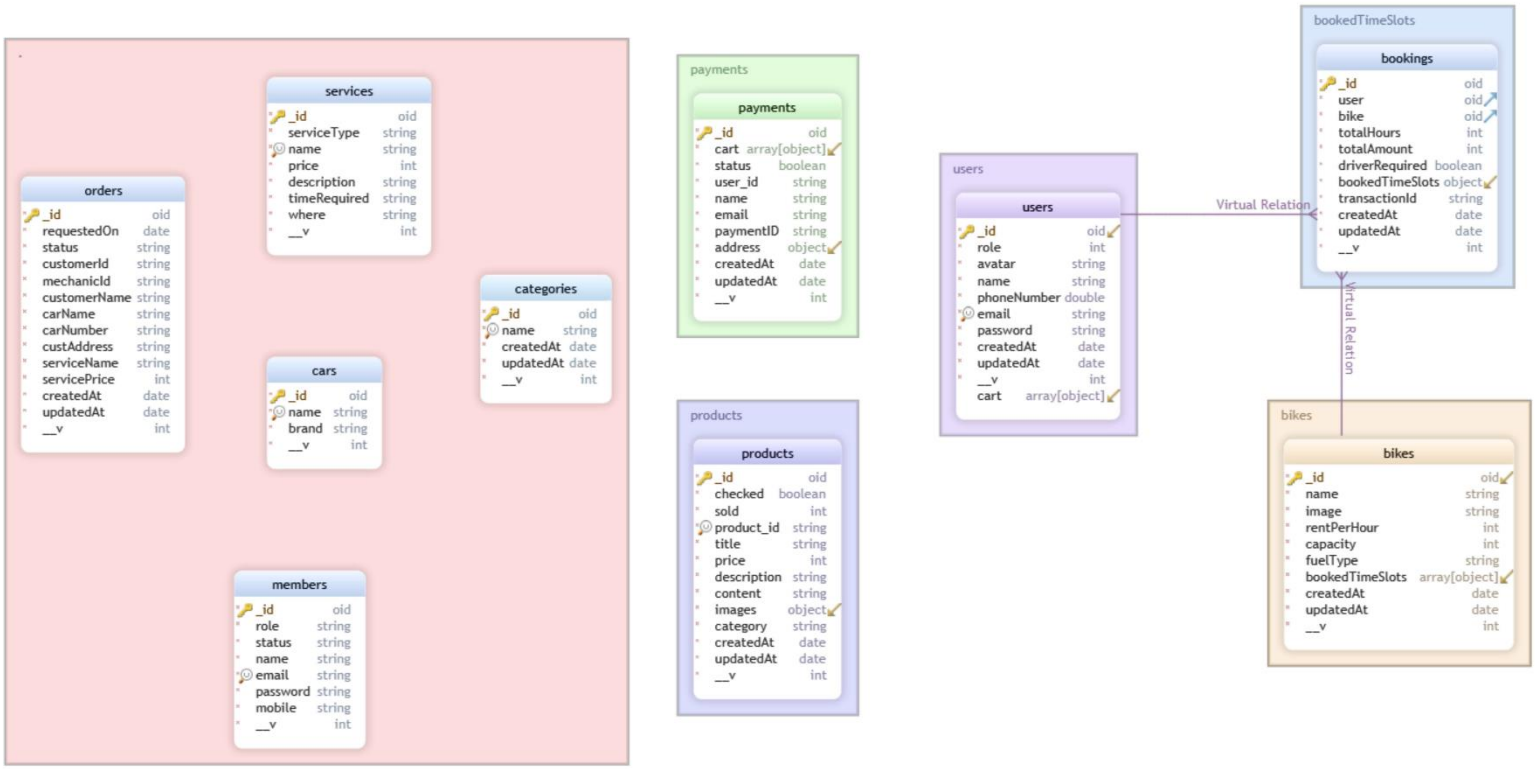


Figure 3.3 Critical Activity

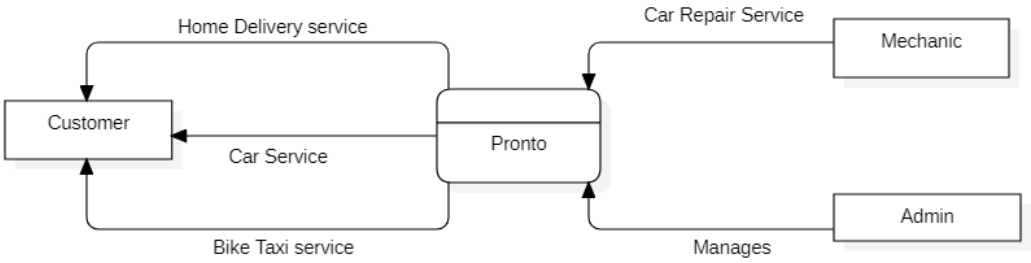
Slack = 0 for all the activities.

Therefore, Critical Path = A-B-C-D-E-F-G-H-I-J

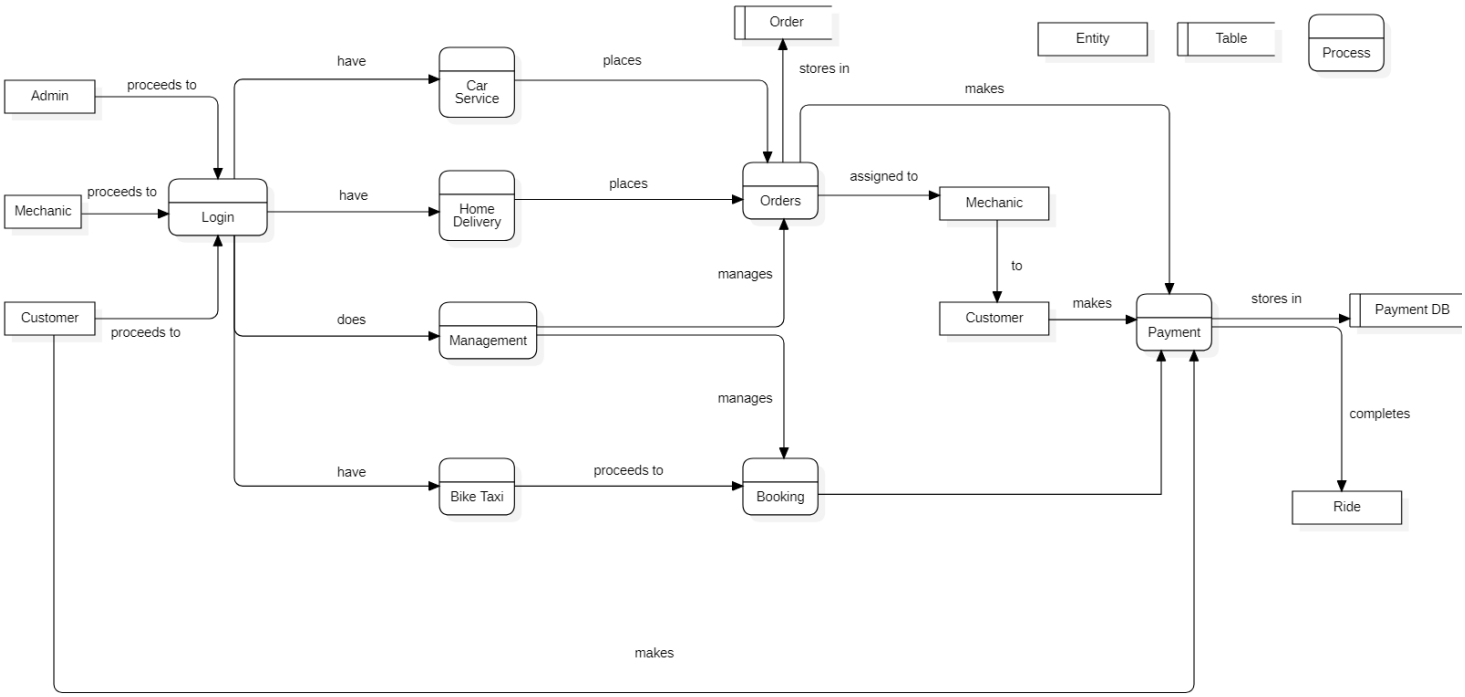


3.5.1 ER Diagram

Level 0



Level 1



3.5.2 Data Flow Diagram

Chapter 4: System Design

4.1 Basic Modules:

1. Admin module

The admin will have the right to verify and assign orders according to resources available. In case any issues are raised by the customers the admin will be able to see and solve the issues. The admin can manage the mechanics, vehicles, products, services related to it and its categories.

2. Mechanic module

The mechanic needs to login by giving username and password. The mechanics can view the assigned orders, completed orders, pending orders and details related to order**.**

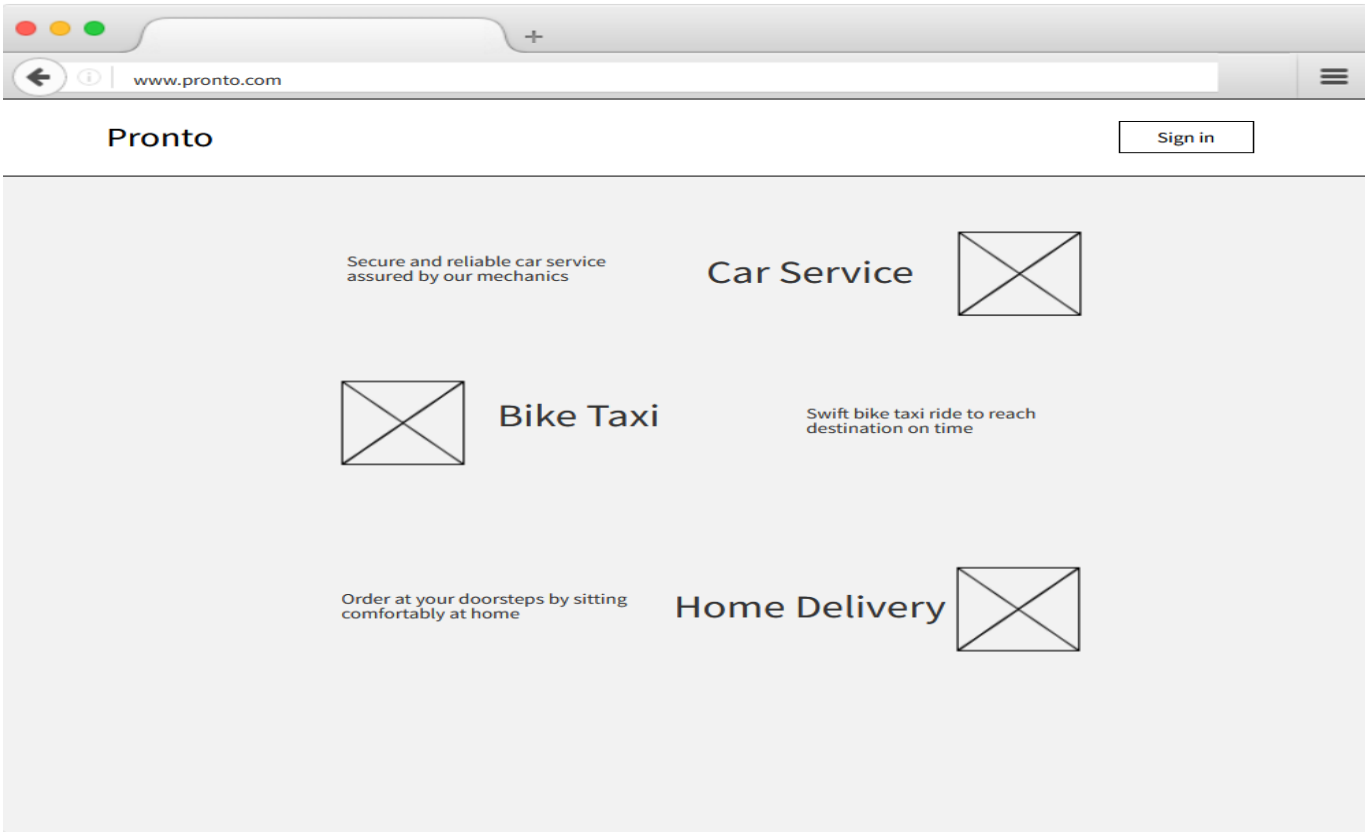
3. Delivery Module

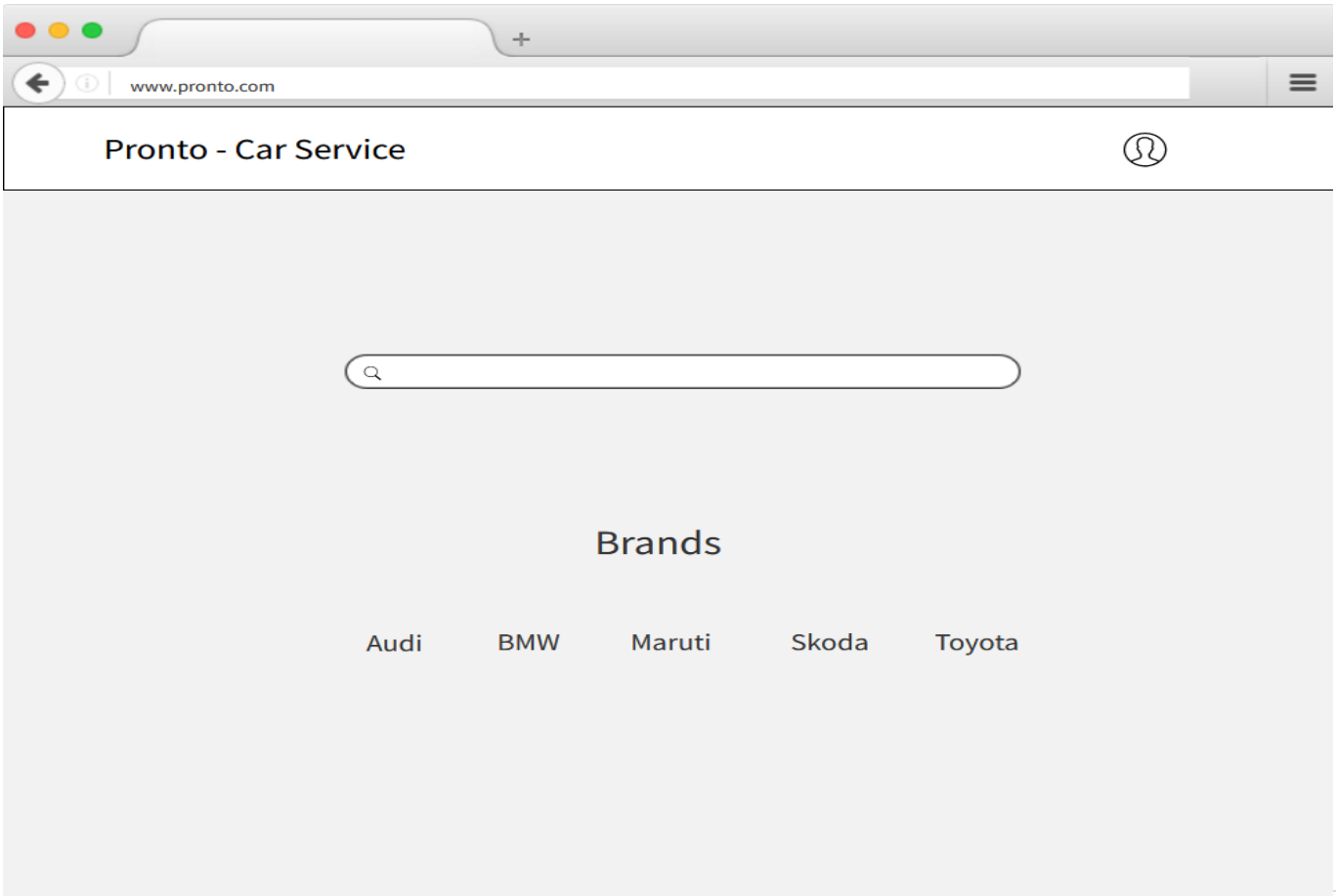
This module consists of home delivery service. The customer can choose any product as per their needs and place order and make payment.

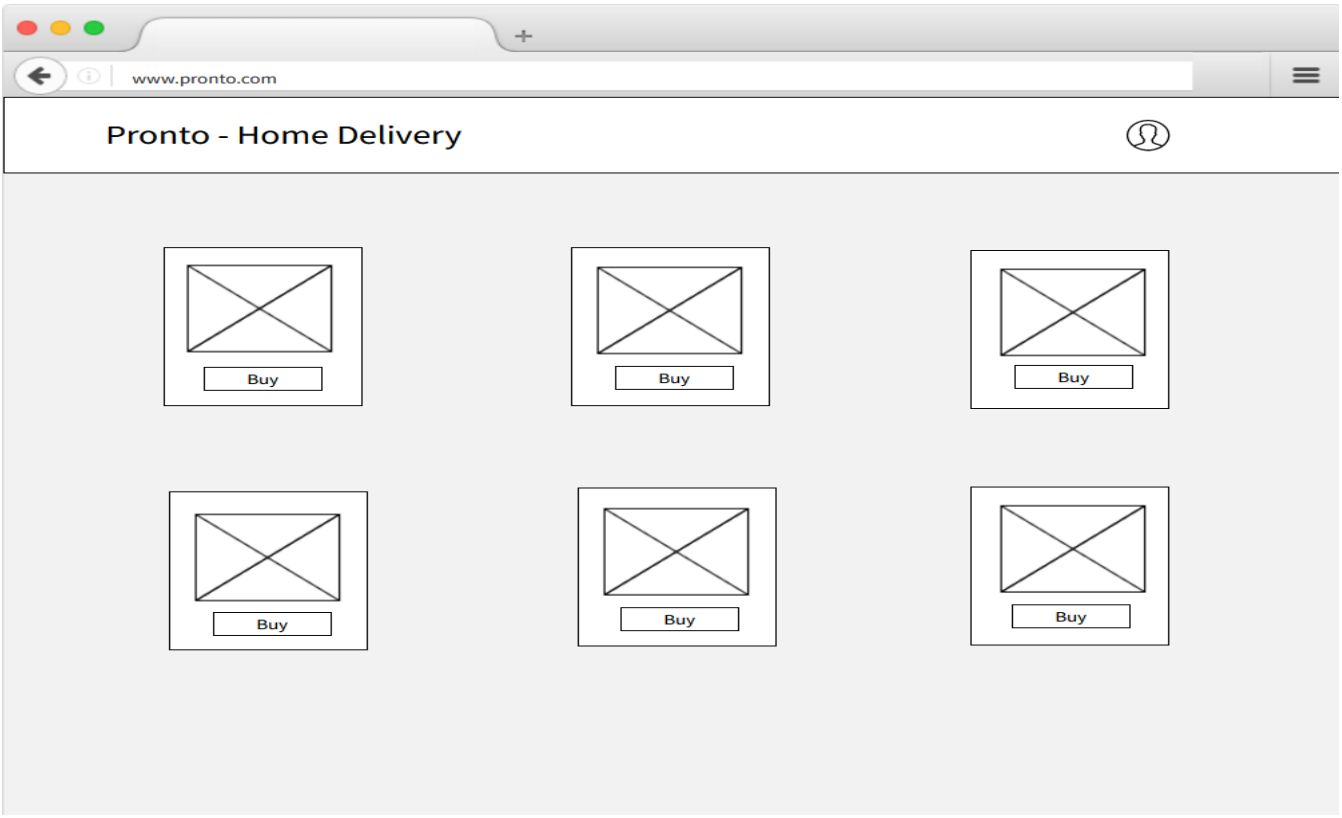
4. Car Service module

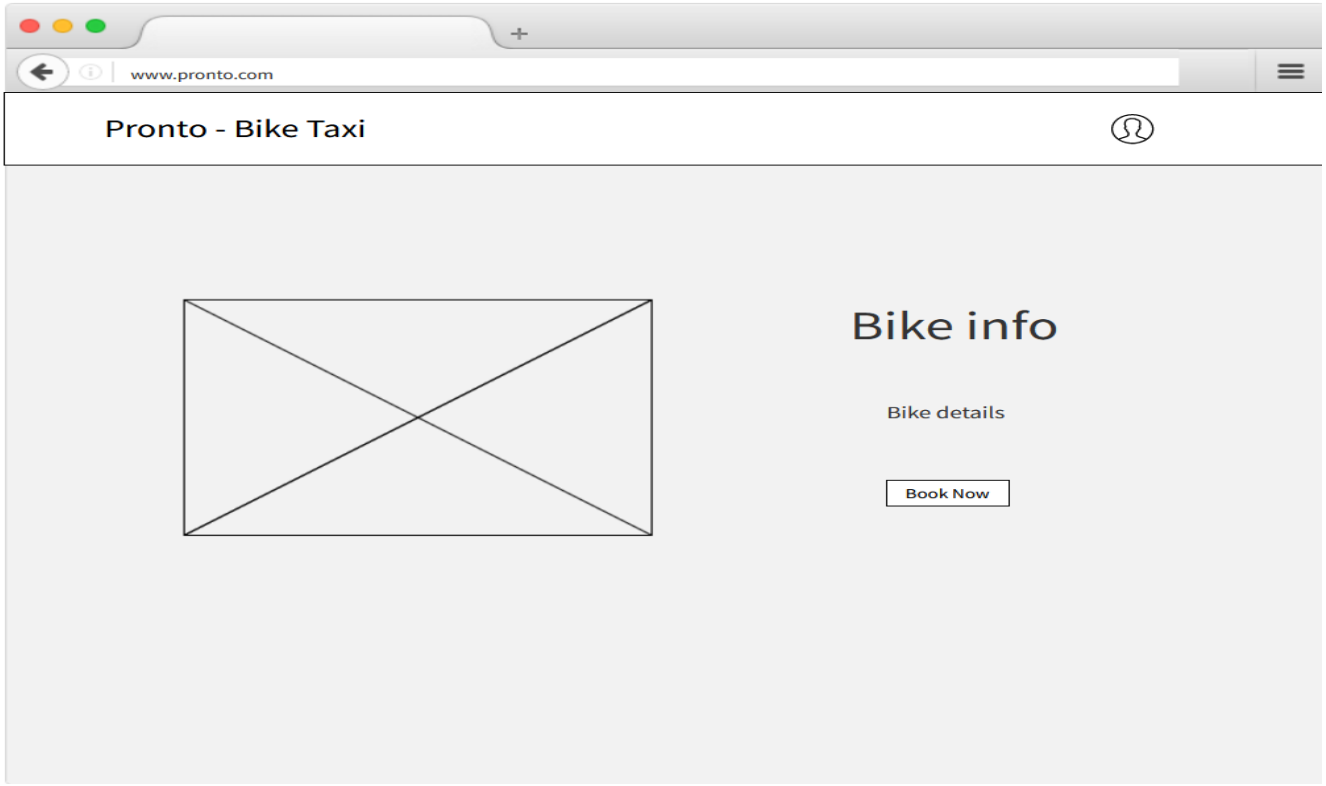
This module consists of car services. The customer can choose their vehicle models and a service based on their needs and can view their order status.

4.3 User Interface Designs






www.pronto.com
Pronto - Home Delivery
Buy
Buy
Buy
Buy
Buy
Buy


www.pronto.com
Pronto - Bike Taxi
Bike info
Bike details
Book Now

Chapter 5: Implementation and Testing

5.1 Implementation Approaches

This website is being created with three views: one for the user, one for the administrator, and one for the mechanic involved in car servicing. From the admin side: vehicles, products for delivery service, car services, and mechanics can all be added by the administrator. The administrator can also make changes to current product or vehicle details, as well as delete them if necessary. The admin can view the orders and payments made by the user.

Another view of the application is for user where the user can view the products and vehicles. If a user wants to buy something, he can put it in the cart and go through the checkout process. In addition, if a user wishes to hire a bike taxi, he can explore the available bikes and complete the booking process. The website also allows the user to select the best service for their car.

The last view is for the mechanic where he can see the orders assigned to him, update the order's status and view a list of orders that he has completed.

5.2 Coding Details and Code Efficiency

Code:

LoginForm.jsx

```
import {useHistory} from 'react-router-dom'
import React, { useContext, useState} from "react";
import {
  BoldLink,
  BoxContainer,
  Container,
  FormContainer,
  Input,
  MutedLink,
  SubmitButton,
} from "./common";
import { Marginer } from "../marginer";
import { AccountContext } from "./accountContext";
import axios from 'axios';
import { showErrMsg, showSuccessMsg } from "../utils/notification/notification";
import { dispatchLogin } from "../../redux/actions/authAction";
import {useDispatch} from 'react-redux'
```

```
const initialState = {
  email: '',
  password: '',
  err: '',
  success: ''
}
export function LoginForm(props) {
  const { switchToSignup } = useContext(AccountContext);
  const [user,setUser] = useState(initialState)
  const {email, password, err, success} = user
  const dispatch = useDispatch();
  const history = useHistory();

  const handleChangeInput = e => {
    const {name, value}= e.target
    setUser({...user, [name]:value, err: '', success:''})
  }

  const handleSubmit = async e =>{
    e.preventDefault();
    try{
      const res = await axios.post('/user/login', {email,password})
       setUser({...user, err: '', success: res.data.msg})
       localStorage.setItem('firstLogin', true)
       dispatch(dispatchLogin())
       history.push("/postLogin")

    }catch (err){
      err.response.data.msg &&
      setUser({...user, err: err.response.data.msg, success:''})
    }
  }

  const mechanic_signin = e =>{
    history.push('/mechanic_signin')
  }

  return (
    <BoxContainer>
      <Container onSubmit={handleSubmit}>
      <FormContainer >
        <Input type="email" name='email'  value = {email} onChange={handleChangeInput}
placeholder="Email*" required/>
        <Input type="password" name='password'  value = {password} onChange={handleChangeInput}
placeholder="Password*" required />
      </FormContainer>
      <Marginer direction="vertical" margin={10} />
      <MutedLink href="#">Forget your password?</MutedLink>
      <Marginer direction="vertical" margin="1.6em" />
```

```
      <SubmitButton type="submit">Signin</SubmitButton>
      <Marginer direction="vertical" margin="1em" />
      <MutedLink href="#">
        Don't have an accoun?{" "}
        <BoldLink href="#" onClick={switchToSignup}>
          Signup
        </BoldLink>
        {err && showErrMsg(err)}
        {success && showSuccessMsg(success)}
      </MutedLink>
      <br />
      <BoldLink onClick={mechanic_signin} style= {{marginLeft: "-1px"}}>
        Mechanic Signin
      </BoldLink>
      </Container>
    </BoxContainer>
  );
}
```

SignUpForm.jsx

```
import React, { useContext, useState } from "react";
import {
  BoldLink,
  BoxContainer,
  Container,
  FormContainer,
  Input,
  MutedLink,
  SubmitButton,
} from "./common";
import { Marginer } from "../marginer";
import { AccountContext } from "./accountContext";
import axios from 'axios';
import { showErrMsg, showSuccessMsg } from "../utils/notification/notification";
const initialState ={
  name: '',
  phoneNumber: '',
  email: '',
  password: '',
  cf_password:'',
  err: '',
  success: ''
}
export function SignupForm(props) {
  const { switchToSignin } = useContext(AccountContext);
  const [user,setUser] = useState(initialState)
  const {name, phoneNumber,email, password, cf_password, err, success} = user
```

```
  const handleChangeInput = e => {
    const {name, value}= e.target
    setUser({...user, [name]:value, err: '', success:''})
  }

  const handleSubmit = async e =>{
    e.preventDefault();
    try {
      const res = await axios.post('/user/register',{name,phoneNumber,email,password})
      setUser({...user, err: '' , success: res.data.msg})
    } catch (err) {
      err.response.data.msg &&
      setUser({...user, err: err.response.data.msg, success: ''})
    }
  }

  return (
    <BoxContainer>
      <Container onSubmit ={handleSubmit}>
      <FormContainer>
        <Input type="text" name ="name" value ={name} onChange={handleChangeInput}
placeholder="Full Name*" />
        <Input type="tel" name = "phoneNumber" value
={phoneNumber}  onChange={handleChangeInput} placeholder="Phone Number*" />
        <Input type="email" name = "email" value ={email} onChange={handleChangeInput}
placeholder="Email*" />
        <Input type="password" name="password" value ={password} onChange={handleChangeInput}
placeholder="Password*" />
        <Input type="password" name = "cf_password" value ={cf_password}
onChange={handleChangeInput} placeholder="Confirm Password*" />
      </FormContainer>
      <Marginer direction="vertical" margin={20} />
      <SubmitButton type="submit" >Signup</SubmitButton>
      <Marginer direction="vertical" margin="1em" />
      <MutedLink href="#">
        Already have an account?
        <BoldLink href="#" onClick={switchToSignin}>
          Signin
        </BoldLink>
        {err && showErrMsg(err)}
        {success && showSuccessMsg(success)}
      </MutedLink>
      </Container>
    </BoxContainer>
  );
}
```

Marginer.jsx

```
import React from "react";
import styled from "styled-components";
```

```
const HorizontalMargin = styled.span`
  display: flex;
  width: ${({ margin }) =>
    typeof margin === "string" ? margin : `${margin}px`};
`;

const VerticalMargin = styled.span`
  display: flex;
  height: ${({ margin }) =>
    typeof margin === "string" ? margin : `${margin}px`};
`;

function Marginer(props) {
  const { direction } = props;

  if (direction === "horizontal") return <HorizontalMargin {...props} />;
  else {
    return <VerticalMargin {...props} />;
  }
}

Marginer.defaultProps = {
  direction: "horizontal",
};

export { Marginer };
```

Index.jsx(Login)

```
import React, { useState } from "react";
import styled from "styled-components";
import { LoginForm } from "./loginForm";
import { motion } from "framer-motion";
import { AccountContext } from "./accountContext";
import { SignupForm } from "./signupForm";


const BoxContainer = styled.div`

  width: 380px;
  min-height: 640px;
  display: flex;
  flex-direction: column;
  border-radius: 19px;
  background: linear-gradient(58deg, #434343, #000000);
  box-shadow: 0 0 2px rgba(15, 15, 15, 0.28);
  position: relative;
  overflow: hidden;
```

```
`;

const TopContainer = styled.div`
  width: 100%;
  height: 230px;
  display: flex;
  flex-direction: column;
  justify-content: flex-end;
  padding: 0 1.8em;
  padding-bottom: 5em;
`;

const BackDrop = styled(motion.div)`
  width: 150%;
  height: 550px;
  position: absolute;
  display: flex;
  flex-direction: column;
  border-radius: 50%;
  transform: rotate(60deg);
  top: -340px;
  left: -140px;


  background: -webkit-linear-gradient(left, #01bf71,#4cc191);


`;

const HeaderContainer = styled.div`
  width: 100%;
  display: flex;
  flex-direction: column;
`;

const HeaderText = styled.h2`
  font-size: 30px;
  font-weight: 600;
  line-height: 1.1;
  color: #fff;
  z-index: 10;
  margin: 10;
`;

const SmallText = styled.h5`
  color: #fff;
  font-weight: 500;
  font-size: 13px;
  z-index: 10;
```

```
  margin: 0;
  margin-top: 12px;
  margin-bottom: -10px;
`;

const InnerContainer = styled.div`
  width: 100%;
  display: flex;
  flex-direction: column;
  padding: 0 1.8em;
`;



const backdropVariants = {
  expanded: {
    width: "233%",
    height: "1250px",
    borderRadius: "20%",
    transform: "rotate(60deg)",
  },
  collapsed: {
    width: "160%",
    height: "550px",
    borderRadius: "50%",
    transform: "rotate(150deg)",
  },
};

const expandingTransition = {
  type: "spring",
  duration: 2.3,
  stiffness: 30,
};

export function AccountBox(props) {
  const [isExpanded, setExpanded] = useState(false);
  const [active, setActive] = useState("signin");

  const playExpandingAnimation = () => {
    setExpanded(true);
    setTimeout(() => {
      setExpanded(false);
    }, expandingTransition.duration * 1000 - 1500);
  };

  const switchToSignup = () => {
    playExpandingAnimation();
    setTimeout(() => {
```

```
      setActive("signup");
    }, 400);
  };

  const switchToSignin = () => {
    playExpandingAnimation();
    setTimeout(() => {
      setActive("signin");
    }, 400);
  };

  const contextValue = { switchToSignup, switchToSignin };

  return (
    <AccountContext.Provider value={contextValue}>

      <BoxContainer>
        <TopContainer>
          <BackDrop
            initial={false}
            animate={isExpanded ? "expanded" : "collapsed"}
            variants={backdropVariants}
            transition={expandingTransition}
          />
          {active === "signin" && (
            <HeaderContainer>
              <HeaderText>Welcome</HeaderText>
              <HeaderText>Back</HeaderText>
              <SmallText>Please sign-in to continue!</SmallText>
            </HeaderContainer>
          )}
          {active === "signup" && (
            <HeaderContainer>
              <HeaderText>Create</HeaderText>
              <HeaderText>Account</HeaderText>
              <SmallText>Please sign-up to continue!</SmallText>
            </HeaderContainer>
          )}
        </TopContainer>
        <InnerContainer>
          {active === "signin" && <LoginForm />}
          {active === "signup" && <SignupForm />}
        </InnerContainer>
      </BoxContainer>
    </AccountContext.Provider>
  );
}
```

Bike Taxi

DefaultLayout.js

```
import React, {useContext,useEffect,useState} from "react";
import Navbar from "../navbar";
import axios from "axios";
import Sidebar from "../sidebar";
import {GlobalState} from '../../GlobalState'

function DefaultLayout(props) {
  const[isOpen, setIsOpen] = useState(false);
    const toggle = () => {
        setIsOpen(!isOpen)
    }

  const state = useContext(GlobalState)
  const [token] = state.token
  useEffect(() => {
    const getUser = async () => {
      try {
         const res = await axios.get('http://localhost:5000/user/infor/', {
             headers: {Authorization: token}
         })
         localStorage.setItem('user',JSON.stringify(res.data))
        }catch(err){
          console.log(err.response.data.msg)
        }
      }
    getUser()
  }, [token])

  return (
    <div>
      <Sidebar isOpen ={isOpen} toggle={toggle} />
      <Navbar toggle = {toggle}/>
      <div className="content">{props.children}</div>

      <div className="footer text-center">
      <hr />


      </div>
    </div>
  );
}

export default DefaultLayout;
```

Navbar.js

```
import React ,{useState,useEffect, useContext}from 'react'
import logo from '../../images/logo192.png';
import { Nav, NavbarContainer, NavLogo, MobileIcon, NavMenu, NavItem,NavLinks} from
'./NavbarElements';
import {FaBars} from 'react-icons/fa'
import {IconContext} from 'react-icons/lib'
import axios from 'axios';
import { GlobalState } from '../GlobalState';
const Navbar = ({toggle}) => {

    const [scrollNav, setScrollNav] = useState(false)
    const changeNav= () => {
        if(window.scrollY >= 80){
            setScrollNav(true)
        }
        else{
            setScrollNav(false)
        }
    };

    useEffect(() => {
        window.addEventListener('scroll', changeNav)
  },[]);

  function reloadPage(){
    window.location.href = "/postLogin"

}

    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    const [isAdmin] = state.userAPI.isAdmin
    const logoutUser = async() =>{
        await axios.get('/user/logout')
        localStorage.clear()
        window.location.href = "/";

    }
    const adminRouter = () => {
        return(
            <>
                <NavItem><NavLinks to
="/biketaxi/admin"  activeClass='active'>Admin</NavLinks></NavItem>
            </>
        )
    }
    const loggedRouter = () => {
        return(
            <>
```

```
                <NavItem><NavLinks to
="/biketaxi/userbookings"  activeClass='active'>Bookings</NavLinks></NavItem>
                <NavItem><NavLinks to ="/"
onClick={logoutUser}  activeClass='active'>Logout</NavLinks></NavItem>
            </>
        )
    }


    return (
       <>
       <IconContext.Provider value={{color: '#fff'}}>
        <Nav scrollNav ={scrollNav}>
            <NavbarContainer>
                <NavLogo to='/postLogin'  onClick={reloadPage} ><img src = {logo} alt='p'
height={40} width={40} ></img>{isAdmin ? 'Admin' : 'Pronto - BikeTaxi' }</NavLogo>
                <MobileIcon onClick={toggle}>
                    <FaBars style={{color: "white"}}/>
                </MobileIcon>
                <NavMenu>
                    <NavItem>
                        <NavLinks to="/biketaxi" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80} activeClass='active'>{isAdmin ? null : 'Home'}</NavLinks>
                    </NavItem>
                    {isAdmin && adminRouter()}
                    {
                    isLogged ? loggedRouter() : ""
                    }
                </NavMenu>

            </NavbarContainer>
        </Nav>
        </IconContext.Provider>
       </>
    );
}

export default Navbar
```

Pages.js

```
import React, {useContext} from 'react'
import { Route, BrowserRouter} from 'react-router-dom'
import Home from './pages/Home'
import BookingBike from './pages/BookingBike'
import 'antd/dist/antd.css'
import UserBookings from './pages/UserBookings'
import AddBike from './pages/AddBike'
import AdminHome from './pages/AdminHome'
```

```
import EditBike from './pages/EditBike'
import bike_store from '../../redux/bike_store'
import { Provider } from 'react-redux'
import ScrollToTop from '../eCommerce/ScrollToTop'
import { GlobalState } from '../GlobalState'


export default function Pages() {
  const state = useContext(GlobalState)
  const[isAdmin] = state.userAPI.isAdmin
  return (
    <Provider store={bike_store}>
        <div className='BikeTaxi'>
            <BrowserRouter>
                <ScrollToTop />
                <Route path='/biketaxi' component={isAdmin? AdminHome : Home} exact/>
                <Route path='/biketaxi/booking/:bikeid' exact component={BookingBike} />
                <Route path='/biketaxi/userbookings' exact component={UserBookings} />
                <Route path='/biketaxi/addbike' exact component={AddBike} />
                <Route path='/biketaxi/editbike/:bikeid' exact component={EditBike} />
                <Route path='/biketaxi/admin' component={AdminHome} exact/>
           </BrowserRouter>
        </div>
        </Provider>
  )
}
```

Sidebar.js

```
import React ,{useContext}from 'react'
import { SidebarContainer, Icon, CloseIcon, SidebarWrapper,SidebarRoute,
SidebarMenu,SidebarLink, SideBtnWrap} from './SidebarElements'
import {GlobalState} from '../GlobalState'
import axios from 'axios';

const Sidebar = ({isOpen, toggle}) => {
    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    const [isAdmin] = state.userAPI.isAdmin
    function reloadPage(){
      window.location.href = "/postLogin"

  }

  const logoutUser = async() =>{
      await axios.get('/user/logout')
      localStorage.clear()
      window.location.href = "/";
```

```
  }

  const adminRouter = () => {
      return(
          <>
              <SidebarMenu><SidebarLink to ="/biketaxi/admin">Admin</SidebarLink></SidebarMenu>
          </>
      )
  }
  const loggedRouter = () => {
      return(
          <>
              <SidebarMenu><SidebarLink to
="/biketaxi/userbookings">Bookings</SidebarLink></SidebarMenu>
              <SidebarMenu><SidebarLink to ="/"
onClick={logoutUser}>Logout</SidebarLink></SidebarMenu>
          </>
      )
  }

    return (
      <SidebarContainer isOpen ={isOpen} onClick={toggle}>
          <Icon onClick={toggle}>
              <CloseIcon />
          </Icon>
          <SidebarWrapper>
              <SidebarMenu>
                  <SidebarLink to="/biketaxi" onClick={toggle}>{isAdmin ? '' :
'Home'}</SidebarLink>
                  {isAdmin && adminRouter()}
                  {
                    isLogged ? loggedRouter() : ""
                  }
              </SidebarMenu>
              <SideBtnWrap>
                  <SidebarRoute to='/postLogin' onClick={reloadPage}>Home</SidebarRoute>
              </SideBtnWrap>
          </SidebarWrapper>
      </SidebarContainer>
    )
}

export default Sidebar
```

Home.js

```
import React , {useState,useEffect} from 'react'
import { useSelector , useDispatch } from 'react-redux'
```

```
import DefaultLayout from '../Components/DefaultLayout'
import { getAllBikes } from '../../../redux/actions/bikesActions'
import { Col, Row  , DatePicker} from 'antd'
import {Link} from 'react-router-dom'
import Spinner from '../Components/Spinner'
import moment from 'moment'
const {RangePicker} = DatePicker
export default function Home() {
    const {bikes} = useSelector(state=>state.bikesReducer)
    const {loading} = useSelector(state=>state.alertsReducer)
    const [totalBikes , setTotalbikes] = useState([])
    const dispatch = useDispatch()


    useEffect(() => {
        dispatch(getAllBikes())
    }, [dispatch])

    useEffect(() => {

        setTotalbikes(bikes)

    }, [bikes])


    function setFilter(values){

        var selectedFrom = moment(values[0] , 'MMM DD yyyy HH:mm')
        var selectedTo = moment(values[1] , 'MMM DD yyyy HH:mm')

        var temp=[]

        for(var bike of bikes){
            //eslint-disable-next-line
              if(bike.bookedTimeSlots.length == 0){
                  temp.push(bike)
              }
              else{

                   for(var booking of bike.bookedTimeSlots) {

                       if(selectedFrom.isBetween(booking.from , booking.to) ||
                       selectedTo.isBetween(booking.from , booking.to) ||
                       moment(booking.from).isBetween(selectedFrom , selectedTo) ||
                       moment(booking.to).isBetween(selectedFrom , selectedTo)
                       )
                       {

                       }
```

```
                    else{
                        temp.push(bike)
                    }

                }

            }

        }


        setTotalbikes(temp)


    }

    return (
        <DefaultLayout>

            <Row className='mt-3' justify='center'>

                <Col lg={20} sm={24} className='d-flex justify-content-left'>

                    <RangePicker showTime={{format: 'HH:mm'}} format='MMM DD yyyy HH:mm'
onChange={setFilter}/>

                </Col>

            </Row>

             {loading === true && (<Spinner/>)}



             <Row justify='center' gutter={16}>

                {totalBikes.map(bike=>{
                    return <Col lg={5} sm={24} xs={24}>
                        <div className="bike p-2 bs1">
                            <img src={bike.image} alt= '' className="bikeimg"/>

                            <div className="bike-content d-flex align-items-center justify-
content-between">

                                <div className='text-left pl-2'>
                                    <p>{bike.name}</p>
                                    <p> Rent Per Hour {bike.rentPerHour} /-</p>
                                </div>
```

```
                                    <div>
                                        <button className="btn1 mr-2"><Link
to={`/biketaxi/booking/${bike._id}`}>Book Now</Link></button>
                                    </div>

                                </div>
                            </div>
                        </Col>
                    })}

                </Row>

            </DefaultLayout>
        )
}
```

AdminHome.js

```
import React, { useState, useEffect } from "react";
import { useSelector, useDispatch } from "react-redux";
import DefaultLayout from '../Components/DefaultLayout'
import { deleteBike, getAllBikes } from "../../../redux/actions/bikesActions";
import { Col, Row } from "antd";
import { Link } from "react-router-dom";
import Spinner from '../Components/Spinner'
import { DeleteOutlined, EditOutlined } from "@ant-design/icons";
import { Popconfirm } from "antd";
export default function AdminHome() {
  const { bikes } = useSelector((state) => state.bikesReducer);
  const { loading } = useSelector((state) => state.alertsReducer);
  const [totalBikes, setTotalbikes] = useState([]);
  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllBikes());
  }, [dispatch]);

  useEffect(() => {
    setTotalbikes(bikes);
  }, [bikes]);

  return (
    <DefaultLayout>
      <Row justify="center" gutter={16} className="mt-2">
        <Col lg={20} sm={24}>
          <div className="d-flex justify-content-between align-items-center">
            <h3 className="mt-1 mr-2">Admin Panel</h3>
            <button className="btn1">
```

```
            <a href="/biketaxi/addbike">Add Bike</a>
          </button>
        </div>
      </Col>
    </Row>

    {loading === true && <Spinner />}

    <Row justify="center" gutter={16}>
      {totalBikes.map((bike) => {
        return (
          <Col lg={5} sm={24} xs={24}>
            <div className="bike p-2 bs1">
              <img alt='' src={bike.image} className="bikeimg" />

              <div className="bike-content d-flex align-items-center justify-content-between">
                <div className="text-left pl-2">
                  <p>{bike.name}</p>
                  <p> Rent Per Hour {bike.rentPerHour} /-</p>
                </div>

                <div className="mr-4">
                  <Link to={`/biketaxi/editbike/${bike._id}`}>
                    <EditOutlined
                      className="mr-3"
                      style={{ color: "green", cursor: "pointer" }}
                    />
                  </Link>

                  <Popconfirm
                    title="Are you sure to delete this bike?"
                    onConfirm={()=>{dispatch(deleteBike({bikeid : bike._id}))}}

                    okText="Yes"
                    cancelText="No"
                  >
                    <DeleteOutlined
                      style={{ color: "red", cursor: "pointer" }}
                    />
                  </Popconfirm>
                </div>
              </div>
            </div>
          </Col>
        );
      })}
    </Row>
  </DefaultLayout>
);
```

```
}
```

AddBike.js

```
import { Col , Row , Form , Input} from 'antd'
import React from 'react'
import { useDispatch , useSelector } from 'react-redux'
import DefaultLayout from '../Components/DefaultLayout'
import Spinner from '../Components/Spinner'
import { addBike } from '../../../redux/actions/bikesActions'
export default function AddBike() {

    const dispatch = useDispatch()
    const {loading} = useSelector(state=>state.alertsReducer)

    function onFinish(values){

         values.bookedTimeSlots=[]

         dispatch(addBike(values))
         console.log(values)
    }

    return (
        <DefaultLayout>
               {loading && (<Spinner />)}
               <Row justify='center mt-5'>
                   <Col lg={12} sm={24} xs={24} className='p-2'>
                       <Form className='bs1 p-2' layout='vertical' onFinish={onFinish}
style={{marginTop:'100px', background:'#fff'}}>
                           <h3>Add New Bike</h3>
                           <hr />
                           <Form.Item name='name' label='Bike name' rules={[{required: true}]}
style={{margin:'0 24px 0 8px'}}>
                               <Input style={{margin:'0 8px 15px 8px'}}/>
                           </Form.Item>
                           <Form.Item name='image' label='Image url' rules={[{required: true}]}
style={{margin:'0 24px 0 8px'}}>
                               <Input style={{margin:'0 8px 15px 8px'}}/>
                           </Form.Item>
                           <Form.Item name='rentPerHour' label='Rent per hour'
rules={[{required: true}]} style={{margin:'0 24px 0 8px'}}>
                               <Input style={{margin:'0 8px 15px 8px'}}/>
                           </Form.Item>
                           <Form.Item name='capacity' label='Capacity' rules={[{required:
true}]} style={{margin:'0 24px 0 8px'}}>
                               <Input style={{margin:'0 8px 15px 8px'}}/>
```

```
                            </Form.Item>
                            <Form.Item name='fuelType' label='Fuel Type' rules={[{required:
true}]} style={{margin:'0 24px 0 8px'}}>
                                <Input style={{margin:'0 8px 40px 8px'}}/>
                            </Form.Item>

                            <div className='text-right' >
                            <button className='btn1' style={{marginBottom: '10px'}}>Add
Bike</button>
                            </div>

                        </Form>
                    </Col>
                </Row>

        </DefaultLayout>
    )
}
```

BookingBike.js

```
import { Col, Row, Divider, DatePicker, Checkbox, Modal } from "antd";
import React, { useContext, useState, useEffect } from "react";
import { useSelector, useDispatch } from "react-redux";
import DefaultLayout from '../Components/DefaultLayout'
import Spinner from '../Components/Spinner'
import { getAllBikes } from "../../../redux/actions/bikesActions";
import moment from "moment";
import { bookBike } from "../../../redux/actions/bookingActions";
import StripeCheckout from "react-stripe-checkout";
import { GlobalState } from "../../GlobalState";
import 'aos/dist/aos.css';


const { RangePicker } = DatePicker;
export default function BookingBike({ match }) {
  const { bikes } = useSelector((state) => state.bikesReducer);
  const { loading } = useSelector((state) => state.alertsReducer);
  const [bike, setbike] = useState({});
  const dispatch = useDispatch();
  const [from, setFrom] = useState();
  const [to, setTo] = useState();
  const [totalHours, setTotalHours] = useState(0);
  const [driver, setdriver] = useState(false);
  const [totalAmount, setTotalAmount] = useState(0);
  const [showModal, setShowModal] = useState(false);
  const state = useContext(GlobalState)
  //eslint-disable-next-line
  const token = state.token
```

```
useEffect(() => {

  if (bikes.length === 0) {
    dispatch(getAllBikes());
  } else {
    setbike(bikes.find((o) => o._id === match.params.bikeid));
  }
  //eslint-disable-next-line
}, [dispatch,bikes]);

useEffect(() => {
  setTotalAmount(totalHours * bike.rentPerHour);
  if (driver) {
    setTotalAmount(totalAmount + 30 * totalHours);
  }
  //eslint-disable-next-line
}, [driver, totalHours]);

function selectTimeSlots(values) {
  setFrom(moment(values[0]).format("MMM DD yyyy HH:mm"));
  setTo(moment(values[1]).format("MMM DD yyyy HH:mm"));

  setTotalHours(values[1].diff(values[0], "hours"));
}



function onToken(token){
  const reqObj = {
      token,
      user: JSON.parse(localStorage.getItem("user"))._id,
      bike: bike._id,
      totalHours,
      totalAmount,
      driverRequired: driver,
      bookedTimeSlots: {
        from,
        to,
      },
    };

    dispatch(bookBike(reqObj));
}

return (
  <DefaultLayout>
    {loading && <Spinner />}
    <Row
      justify="center"
```

```
          className="d-flex align-items-center"
          style={{ minHeight: "90vh" , marginTop: '150px'}}
        >
          <Col lg={12} sm={24} xs={24} className='p-3'>
            <img src={bike.image} alt='' className="bikeimg2 bs1 w-100" />
          </Col>

          <Col lg={10} sm={24} xs={24} className="text-right">
            <Divider type="horizontal" dashed style={{borderBottom: '1px solid #010101', fontSize:
'30px', fontWeight: '700'}}>
              Bike Info
            </Divider>
            <div style={{ textAlign: "right", fontSize:'18px' }}>
              <p>{bike.name}</p>
              <p>{bike.rentPerHour} /- Per hour</p>
              <p>Fuel Type : {bike.fuelType}</p>
              <p>Max Persons : {bike.capacity}</p>
            </div>

            <Divider type="horizontal" dashed style={{marginBottom:'5px', fontSize:'18px'}}>
              Select Time Slots
            </Divider>
            <RangePicker
              showTime={{ format: "HH:mm" }}
              format="MMM DD yyyy HH:mm"
              onChange={selectTimeSlots}
              style={{marginTop:'10px'}}
            />
            <br />
            <button
              className="btn1 mr-2"
              style={{marginTop: '15px'}}
              onClick={() => {
                setShowModal(true);
              }}
            >
              See Booked Slots
            </button>
            {from && to && (
              <div>
                <p>
                  Total Hours : <b>{totalHours}</b>
                </p>
                <p>
                  Rent Per Hour : <b>{bike.rentPerHour}</b>
                </p>
                <Checkbox
                  onChange={(e) => {
                    if (e.target.checked) {
```

```
                    setdriver(true);
                  } else {
                    setdriver(false);
                  }
                }}
              >
                Driver Required
              </Checkbox>

              <h3>Total Amount : {totalAmount}</h3>

              <StripeCheckout
                shippingAddress
                token={onToken}
                currency='inr'
                amount={totalAmount * 100}
                stripeKey="pk_test_51K6uf8SDiEKbHKk2UPE0yDGuEIT7tGrUqJvDd18MFvNrkEw3llLqvzDaUfg9
IjZpeQKOZ523j5BV8n1SEpD2vWdX00Fv0cnB8h"
              >
                  <button className="btn1">
                Book Now
              </button>
              </StripeCheckout>


            </div>
          )}
        </Col>

        {bike.name && (
          <Modal
            visible={showModal}
            closable={false}
            footer={false}
            title="Booked time slots"
          >
            <div className="p-2">
              {bike.bookedTimeSlots.map((slot) => {
                return (
                  <button className="btn1 mt-2">
                    {slot.from} - {slot.to}
                  </button>
                );
              })}

              <div className="text-right mt-5">
                <button
                  className="btn1"
                  onClick={() => {
```

```
                  setShowModal(false);
                }}
              >
                CLOSE
              </button>
            </div>
          </div>
        </Modal>
      )}
    </Row>
  </DefaultLayout>
 );
}
```

EditBike.js

```
import { Col, Row, Form, Input } from "antd";
import React, { useEffect, useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import DefaultLayout from "../Components/DefaultLayout";
import Spinner from '../Components/Spinner'
import { editBike, getAllBikes } from "../../../redux/actions/bikesActions";
export default function EditBike({ match }) {
  const { bikes } = useSelector((state) => state.bikesReducer);
  const dispatch = useDispatch();
  const { loading } = useSelector((state) => state.alertsReducer);
  const [bike, setbike] = useState();
  const [totalbikes, settotalbikes] = useState([]);
  useEffect(() => {
    if (bikes.length === 0) {
      dispatch(getAllBikes());
    } else {
      settotalbikes(bikes);
      setbike(bikes.find((o) => o._id === match.params.bikeid));
      console.log(bike);
    }
    //eslint-disable-next-line
  }, [dispatch,bikes,bike]);

  function onFinish(values) {
    values._id = bike._id;

    dispatch(editBike(values));
    console.log(values);
  }

  return (
    <DefaultLayout>
```

```
{loading && <Spinner />}
<Row justify="center mt-5">
  <Col lg={12} sm={24} xs={24} className='p-2'>
    {totalbikes.length > 0 && (
      <Form
        initialValues={bike}
        className="bs1 p-2"
        layout="vertical"
        onFinish={onFinish}
        style={{marginTop:'100px', background:'#fff'}}
      >
        <h3>Edit Bike</h3>

        <hr />
        <Form.Item
          name="name"
          label="Bike name"
          rules={[{ required: true }]}
          style={{margin:'0 24px 0 8px'}}
        >
          <Input style={{margin:'0 8px 15px 8px'}} />
        </Form.Item>
        <Form.Item
          name="image"
          label="Image url"
          rules={[{ required: true }]}
          style={{margin:'0 24px 0 8px'}}
        >
          <Input style={{margin:'0 8px 15px 8px'}} />
        </Form.Item>
        <Form.Item
          name="rentPerHour"
          label="Rent per hour"
          rules={[{ required: true }]}
          style={{margin:'0 24px 0 8px'}}
        >
          <Input style={{margin:'0 8px 15px 8px'}} />
        </Form.Item>
        <Form.Item
          name="capacity"
          label="Capacity"
          rules={[{ required: true }]}
          style={{margin:'0 24px 0 8px'}}
        >
          <Input style={{margin:'0 8px 15px 8px'}} />
        </Form.Item>
        <Form.Item
          name="fuelType"
          label="Fuel Type"
```

```
                rules={[{ required: true }]}
                style={{margin:'0 24px 0 8px'}}
              >
                <Input style={{margin:'0 8px 45px 8px'}}/>
              </Form.Item>

              <div className="text-right">
                <button className="btn1" style={{marginBottom: '10px'}} >Edit Bike</button>
              </div>
            </Form>
          )}
        </Col>
      </Row>
    </DefaultLayout>
  );
}
```

UserBookings.js

```
import React, {useEffect,useContext } from "react";
import DefaultLayout from '../Components/DefaultLayout'
import { useDispatch, useSelector } from "react-redux";
import { getAllBookings } from "../../../redux/actions/bookingActions";
import { Col, Row } from "antd";
import Spinner from '../Components/Spinner'
import moment from "moment";
import { GlobalState } from "../../GlobalState";

export default function UserBookings() {
  const dispatch = useDispatch();
  const { bookings } = useSelector((state) => state.bookingsReducer);
  //console.log(bookings)
  const {loading} = useSelector((state) => state.alertsReducer);
  const user = JSON.parse(localStorage.getItem("user"));
  console.log(user._id)
  const state = useContext(GlobalState)
  const[isAdmin] = state.userAPI.isAdmin
  useEffect(() => {
    dispatch(getAllBookings());
  }, [dispatch]);

  return (
    <DefaultLayout>
        {loading && (<Spinner />)}
      <h3 className="text-center mt-2">{ isAdmin ? 'All Bookings':'My Bookings'}</h3>

      <Row justify="center" gutter={16}>
        <Col lg={16} sm={24}>
         {
           isAdmin ? <>
```

```
            {bookings.map((booking) => {
             return <Row gutter={16} className="bs1 mt-3 text-left">
                <Col lg={6} sm={24}>
                    <p><b>{booking.bike.name}</b></p>
                    <p>Total hours : <b>{booking.totalHours}</b></p>
                    <p>Rent per hour : <b>{booking.bike.rentPerHour}</b></p>
                    <p>Total amount : <b>{booking.totalAmount}</b></p>
                </Col>

                <Col lg={12} sm={24}>
                <p>Transaction Id : <b>{booking.transactionId}</b></p>
                <p>From: <b>{booking.bookedTimeSlots.from}</b></p>
                <p>To: <b>{booking.bookedTimeSlots.to}</b></p>
                <p>Date of booking: <b>{moment(booking.createdAt).format('MMM DD yyyy')}</b></p>
                </Col>

                <Col lg={6} sm={24} className='text-right'>
                    <img alt='' sstyle={{borderRadius:5}} src={booking.bike.image}  height="140"
className="p-2"/>
                </Col>
              </Row>;
            })}
            </>
            :<>
             {bookings.filter(o=>o.user===user._id).map((booking) => {
             return <Row gutter={16} className="bs1 mt-3 text-left">
                <Col lg={6} sm={24}>
                    <p><b>{booking.bike.name}</b></p>
                    <p>Total hours : <b>{booking.totalHours}</b></p>
                    <p>Rent per hour : <b>{booking.bike.rentPerHour}</b></p>
                    <p>Total amount : <b>{booking.totalAmount}</b></p>
                </Col>

                <Col lg={12} sm={24}>
                <p>Transaction Id : <b>{booking.transactionId}</b></p>
                <p>From: <b>{booking.bookedTimeSlots.from}</b></p>
                <p>To: <b>{booking.bookedTimeSlots.to}</b></p>
                <p>Date of booking: <b>{moment(booking.createdAt).format('MMM DD yyyy')}</b></p>
                </Col>

                <Col lg={6} sm={24} className='text-right'>
                    <img alt='' sstyle={{borderRadius:5}} src={booking.bike.image}  height="140"
className="p-2"/>
                </Col>
              </Row>;
            })}
            </>
          }
```

```
        </Col>
      </Row>
    </DefaultLayout>
  );
}
```

CarService
Customer
CustHome.js

```
import React from "react";
import CarouselComponent from "./CarouselComponent";
import Brands from "./Brands";

function Demo(props) {
  return (
    <div>
      <CarouselComponent />
      <Brands />
    </div>
  );
}

export default Demo;
```

Carousel.js

```
import React from "react";
import './CSS/CarouselComp.css'
import "antd/dist/antd.css";
import { Carousel } from "antd";
import {LeftOutlined, RightOutlined} from '@ant-design/icons'


import Image2 from "../../assets/images/carousel/ac-services_banner.jpg";
import Image3 from "../../assets/images/carousel/battery-services_banner.jpg";
import Image4 from "../../assets/images/carousel/clean-services_banner.png";
import Image5 from "../../assets/images/carousel/paint-services_banner.jpg";


export default function CarouselComponent() {

  return (
    <div className="carousel">
        <Carousel
          autoplay= {true}
          arrows prevArrow={<LeftOutlined />} nextArrow={<RightOutlined />}>

        <div>
```

```
          <img className="image" src={Image5} alt="Slider 1" />
        </div>
        <div>
          <img className="image" src={Image3} alt="Slider 1" />
        </div>
        <div>
          <img className="image" src={Image4} alt="Slider 1" />
        </div>
        <div>
          <img className="image" src={Image2} alt="Slider 1" />
        </div>

        </Carousel>
    </div>
  );
}
```

Car.js

```
import React, { useEffect, useState } from "react";

import CarouselComponent from "./CarouselComponent";
import CarService from "../../services/member/car/car_services";
import {
  Grid,
  Card,
  CardContent,
  Typography,
  TextField,
} from "@material-ui/core";
import "./CSS/Brands.css";
import SearchIcon from "@material-ui/icons/Search";

function Cars(props) {
  const { match, history } = props;
  const { params } = match;
  const { brand } = params;

  const [cars, setCars] = useState([]);
  const [filter, setfilter] = useState("");

  const handleSearchChange = (e) => {
    setfilter(e.target.value);
  };

  const retrieveCars = () => {
    CarService.getCarsByBrand(brand)
```

```
      .then((response) => {
        setCars(response);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  useEffect(() => {
    retrieveCars();
  });

  const getCarCards = (car) => {
    return (
      <Grid item xs={6} sm={4} md={3} lg={2} key={car._id}>
        <Card
          className="card"
          onClick={() => history.push(`/cust_home/services/${car._id}`)}
        >
          <CardContent>
            <Typography>{car.name}</Typography>
          </CardContent>
        </Card>
      </Grid>
    );
  };

  return (
    <div>
      <CarouselComponent />
      <div className="brand">
        <h1 className="title">{`Available ${brand} Cars`}</h1>

        <div className="search">
          <SearchIcon className="searchIcon" />
          <TextField
            className="searchInput"
            label="Search for Cars"
            onChange={handleSearchChange}
          />
        </div>

        <Grid container spacing={3} className="grid_container">
          {cars.map((car) => car.name.includes(filter) && getCarCards(car))}
        </Grid>
      </div>
    </div>
  );
}
```

```
export default Cars;
```

Brands.js

```
import React, { useEffect, useState } from "react";
import { withRouter } from "react-router-dom";
import {
  Grid,
  Card,
  CardContent,
  CircularProgress,
  Typography,
  TextField,
} from "@material-ui/core";
import SearchIcon from "@material-ui/icons/Search";
import "./CSS/Brands.css";
import CarService from "../../services/member/car/car_services";

function Brands(props) {
  const { history } = props;
  const [brands, setbrands] = useState([]);
  const [filter, setfilter] = useState("");

  const handleSearchChange = (e) => {
    setfilter(e.target.value);
  };

  const retrieveBrands = () => {
    CarService.getAllBrands()
      .then((response) => {
        setbrands(response);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  useEffect(() => {
    retrieveBrands();
  }, []);

  const getCarCard = (brand) => {
    return (
      <Grid item xs={6} sm={4} md={3} lg={2} key={brand}>
        <Card
          className="card"
          onClick={() => history.push(`/cust_home/cars/${brands[brand]}`)}
        >
```

```
            <CardContent>
              <Typography className="text">{brands[brand]}</Typography>
            </CardContent>
          </Card>
        </Grid>
      );
    };

    return (
      <div className="brand">
        <h1 className="title">Available Brands</h1>
        {/* {brands && brands.map((brand, index) => <li key={index}>{brand}</li>)} */}

        <div className="search">
          <SearchIcon className="searchIcon" />
          <TextField
            className="searchInput"
            label="Search for Brands"
            onChange={handleSearchChange}
          />
        </div>

        {brands ? (
          <Grid container spacing={3} item className="grid_container">
            {Object.keys(brands).map(
              (brand) => brands[brand].includes(filter) && getCarCard(brand)
            )}
          </Grid>
        ) : (
          <CircularProgress />
        )}
      </div>
    );
}

export default withRouter(Brands);
```

MyBookings.js

```
import React, { useEffect, useState } from "react";
import CustomerService from "../../services/customer/customer_service";
import AuthService from "../../services/customer/authentication/auth_service";
import "./CSS/MyBookings.css";
import { Card, Grid, CardContent } from "@material-ui/core";
import { MuiThemeProvider, createTheme } from "@material-ui/core/styles";

const muiBaseTheme = createTheme();

const theme = {
```

```
  overrides: {
    MuiCard: {
      root: {
        "&.MuiEngagementCard--02": {
          transition: "0.3s",
          width: '95%',
          maxWidth: '100%',
          boxShadow: "0 8px 40px -12px rgba(0,0,0,0.3)",
          "&:hover": {
            boxShadow: "0 16px 70px -12.125px rgba(0,0,0,0.3)"
          },
          "& .MuiCardMedia-root": {
            paddingTop: "56.25%"

          },
          "& .MuiCardContent-root": {
            textAlign: "left",
            padding: muiBaseTheme.spacing(3)
          },
          "& .MuiDivider-root": {
            margin: `${muiBaseTheme.spacing(3)}px 0`
          },
          "& .MuiTypography--heading": {
            fontWeight: "bold"
          },
          "& .MuiTypography--subheading": {
            lineHeight: 1.8

          },
          "& .MuiAvatar-root": {
            display: "inline-block",
            border: "2px solid white",
            "&:not(:first-of-type)": {
              marginLeft: -muiBaseTheme.spacing.unit
            }
          }
        }
      }
    }
  }
};

function MyBookings() {
  const [orders, setorders] = useState([]);

  useEffect(() => {
    const user = AuthService.getCurrentCustomer();
    CustomerService.findMyOrders(user._id)
      .then((res) => {
```

```
        setorders(res);
      })
      .catch((err) => {
        console.log(err);
      });
  }, []);

  const getOrderCards = (order) => {
    return (
      <Grid item xs={12} sm={12} md={12} lg={12} key={order._id}>
        <MuiThemeProvider theme={createTheme(theme)}>
        <Card  className="MuiEngagementCard--02">
          <CardContent>
            <h1>Your Order Request is {order.status}</h1>
            <hr />
            <h4>Car : {order.carName}</h4>
            <h4>Vehicle Number: {order.carNumber}</h4>
            <h4>Address: {order.custAddress}</h4>
            <h4>Service Name: {order.serviceName}</h4>
            <h4>Service Price: {order.servicePrice}</h4>
          </CardContent>
        </Card>
        </MuiThemeProvider>
      </Grid>
    );
  };
  return (
    <div className="container1">
      <MuiThemeProvider theme={createTheme(theme)}>
      <Card className="MuiEngagementCard--02" style={{backgroundColor: "#01bf7a"}}>
        <CardContent>
          <h1 className="booking_title">MY BOOKINGS</h1>
        </CardContent>
      </Card>
      </MuiThemeProvider>
      <br />
      {orders ? (
        <Grid container spacing={4} className="">
          {orders.map((order) => getOrderCards(order))}
        </Grid>
      ) : (
        <div>
          <br />
          <h1>NO BOOKINGS</h1>
        </div>
      )}
    </div>
  );
}
```

```
export default MyBookings;
```

Order.js

```
import React, { useEffect, useState } from "react";
import AuthService from "../../services/customer/authentication/auth_service";
import CustomerService from "../../services/customer/customer_service";
import CarService from "../../services/member/car/car_services";
import PackageService from "../../services/member/package/package_services";
import "./CSS/Order.css";
import { Card, Grid, TextField, Button } from "@material-ui/core";
import { useForm } from "react-hook-form";
import {message} from 'antd';

function Order(props) {
  //eslint-disable-next-line
  const { match, history } = props;
  const { params } = match;
  const { carId, serviceId } = params;
  const [user, setUser] = useState("");
  const [service, setService] = useState([]);
  const [car, setCar] = useState([]);
  const getPackage = () => {
    PackageService.findServiceById(serviceId)
      .then((res) => {
        setService(res);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  const getCar = () => {
    CarService.findCarById(carId)
      .then((res) => {
        setCar(res);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  useEffect(() => {
    const user = AuthService.getCurrentCustomer();
    console.log(user);
    setUser(user);

    getCar();
```

```
  getPackage();
  //eslint-disable-next-line
},[]);

const { handleSubmit, register, errors } = useForm({
  mode: "onBlur",
});

const onSubmit = (values) => {
  CustomerService.placeOrder(
    user._id,
    user.name,
    car.name,
    values.carNumber,
    values.custAddress,
    service.name,
    service.price
  )
    .then((response) => {
      message.success("Order Placed Successfully")
    })
    .catch((err) => {
      console.log(err);
      message.error(err)
    });
};

return (
  <div className="container">
    <h1 className="summary_title">ORDER SUMMARY</h1>
    <Card className="booking_card">
      <Grid container spacing={3}>
        <Grid item xs={12} sm={12} md={6} lg={6}>
          <p className="title_subHeading">PERSONAL DETAILS</p>
          <h4>Email Id: {user.email}</h4>
          <h4>Name: {user.name}</h4>
          <form onSubmit={handleSubmit(onSubmit)}>
            <Grid container spacing={4}>
              <Grid item xs={6} sm={6} md={6} lg={6}>
                <TextField
                  color="primary"
                  variant="outlined"
                  label="Vehicle Number"
                  name="carNumber"
                  margin="normal"
                  inputRef={register({
                    required: "Number is Required",
                  })}
                />
```

```
                {errors.carNumber && (
                  <span className="span">{errors.carNumber.message}</span>
                )}
              </Grid>
              <Grid item xs={6} sm={6} md={6} lg={6}>
                <TextField
                  color="primary"
                  variant="outlined"
                  label="Address"
                  multiline
                  name="custAddress"
                  margin="normal"
                  inputRef={register({
                    required: "Address is Required",
                  })}
                />
                {errors.custAddress && (
                   <span className="span">{errors.custAddress.message}</span>
                )}
              </Grid>
            </Grid>
            <Button
              type="submit"
              fullWidth
              variant="contained"
              className="place_order_bttn"
              style={{background: "black", color: "white"}}
            >
              PLACE ORDER
            </Button>
          </form>
        </Grid>
        <Grid item xs={12} sm={12} md={6} lg={6}>
          <p className="title_subHeading">SERVICE DETAILS</p>
          <h3>Service Name: {service.name}</h3>
          <h3>Total Price: {service.price}</h3>
          <h3>Time Required: {service.timeRequired}</h3>
          <h3>Selected Car: {car.name}</h3>
        </Grid>
      </Grid>
    </Card>
  </div>
 );
}

export default Order;
```

Services.js

```
import React, { useEffect, useState } from "react";
import { makeStyles } from "@material-ui/core/styles";
import Card from "@material-ui/core/Card";
import CardContent from "@material-ui/core/CardContent";
import Typography from "@material-ui/core/Typography";
import "./CSS/Services.css";
import Package from "../../services/member/package/package_services";
import { Grid } from "@material-ui/core";
import { MuiThemeProvider, createTheme } from "@material-ui/core/styles";

const muiBaseTheme = createTheme();

const theme = {
  overrides: {
    MuiCard: {
      root: {
        "&.MuiEngagementCard--01": {
          transition: "0.3s",
          maxWidth: '90%',
          margin: "auto",
          boxShadow: "0 8px 40px -12px rgba(0,0,0,0.3)",
          "&:hover": {
            boxShadow: "0 16px 70px -12.125px rgba(0,0,0,0.3)"
          },
          "& .MuiCardMedia-root": {
            paddingTop: "56.25%"
          },
          "& .MuiCardContent-root": {
            textAlign: "left",
            padding: muiBaseTheme.spacing(3)
          },
          "& .MuiDivider-root": {
            margin: `${muiBaseTheme.spacing(3)}px 0`
          },
          "& .MuiTypography--heading": {
            fontWeight: "bold"
          },
          "& .MuiTypography--subheading": {
            lineHeight: 1.8
          },
          "& .MuiAvatar-root": {
            display: "inline-block",
            border: "2px solid white",
            "&:not(:first-of-type)": {
              marginLeft: -muiBaseTheme.spacing.unit
            }
          }
        }
      }
```

```
    }
  }
};


const useStyles = makeStyles({
  title: {
    fontSize: 14,
  },
  pos: {
    marginBottom: 12,
  },
});

export default function Services(props) {
  const { match, history } = props;
  const { params } = match;
  const { car } = params;
  const classes = useStyles();

  const [services, setServices] = useState([]);

  useEffect(() => {
    Package.getAllServices()
      .then((response) => {
        setServices(response);
      })
      .catch((err) => {
        console.log(err);
      });
  }, []);

  const getServiceCards = (res) => {
    var type,
      where = "";
    //eslint-disable-next-line
    if (res.serviceType == 1) {
      type = "Car Care Services";
    } else {
      type = "Periodic Car Service";
    }
    //eslint-disable-next-line
    if (res.where == 1) {
      where = "Free Pickup & Drop";
    } else {
      where = "Service @ Doorstep";
    }
    return (
      <Grid item xs={12} sm={12} md={6} lg={6} key={res._id}>
```

```
        <MuiThemeProvider theme={createTheme(theme)}>
        <Card
          className="MuiEngagementCard--01"
          variant="outlined"
          onClick={() =>
            history.push(`/cust_home/order/car/${car}/service/${res._id}`)
          }
        >
          <CardContent>
            <Typography
              className={classes.title}
              color="textSecondary"
              gutterBottom
            >
              {type}
            </Typography>
            <Typography variant="h5" component="h2">
              {res.name}
            </Typography>
            <Typography component="h6">{res.price}</Typography>
            <Typography variant="body2" component="p">
              {res.description}
            </Typography>
            <Typography
              className={classes.title}
              color="textSecondary"
              gutterBottom
            >
              {where}
            </Typography>
            <hr></hr>
            <div className="action_buttons">
              <span className="timeline">
                {`service done in ${res.timeRequired}`}
              </span>
              <button className="buy_button">Buy</button>
            </div>
          </CardContent>
        </Card>
        </MuiThemeProvider>
      </Grid>
    );
  };

  return (
    <div className="container2">
      <button className="change_button" onClick={() => history.push(`/carservice`)}>Change
Car</button>
      <br />
```

```
      <br />
      <hr />
      <br />
      <Grid container spacing={5} className="grid_container">
        {services.map((res) => getServiceCards(res))}
      </Grid>
    </div>
  );
}
```

Home

Navbar.js

```
import React ,{useState,useEffect, useContext}from 'react'
import logo from '../../../../images/logo192.png'
import { Nav, NavbarContainer, NavLogo, MobileIcon, NavMenu, NavItem,NavLinks} from
'./NavbarElements';
import {FaBars} from 'react-icons/fa'
import {IconContext} from 'react-icons/lib'
import axios from 'axios';
import {GlobalState} from '../../../GlobalState'

const Navbar = ({toggle}) => {

    const [scrollNav, setScrollNav] = useState(false)
    const changeNav= () => {
        if(window.scrollY >= 80){
            setScrollNav(true)
        }
        else{
            setScrollNav(false)
        }
    };

    useEffect(() => {
        window.addEventListener('scroll', changeNav)
  },[]);

  function reloadPage(){
    window.location.href = "/postLogin"

}

    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    //eslint-disable-next-line
    const [userData,setUserData] = state.userAPI.userInfor
    localStorage.setItem("customer", JSON.stringify(userData));
    const [isAdmin] = state.userAPI.isAdmin
    const logoutUser = async() =>{
        await axios.get('/user/logout')
```

```
        localStorage.clear()
        window.location.href = "/";

    }
    const adminRouter = () => {
        return(
            <>
                <NavItem><NavLinks to ="/carservice">Admin</NavLinks></NavItem>
                <NavItem><NavLinks to ="/admin_home/cars">Cars</NavLinks></NavItem>
                <NavItem><NavLinks to ="/admin_home/packages">Services</NavLinks></NavItem>
                <NavItem><NavLinks to ="/admin_home/mechanics">Mechanics</NavLinks></NavItem>
                <NavItem><NavLinks to ="/admin_home/orders">Orders</NavLinks></NavItem>

            </>
        )
    }
    const loggedRouter = () => {
        return(
            <>
                <NavItem><NavLinks to ="/" onClick={logoutUser}>Logout</NavLinks></NavItem>
            </>
        )
    }


    return (
       <>
       <IconContext.Provider value={{color: '#fff'}}>
        <Nav scrollNav ={scrollNav}>
            <NavbarContainer>
                <NavLogo to='/postLogin'  onClick={reloadPage} ><img src = {logo} alt='p'
height={40} width={40} ></img>{isAdmin ? 'Admin' : 'Pronto - CarService' }</NavLogo>
                <MobileIcon onClick={toggle}>
                    <FaBars style={{color: "white"}}/>
                </MobileIcon>
                <NavMenu>
                    <NavItem>
                        <NavLinks to="/carservice" smooth ={true} duration={500} spy={true}
exact ='true' offset ={-80}>{isAdmin ? '' : 'Home'}</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks to="/working" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80}>{isAdmin ? '' : 'Working'}</NavLinks>
                    </NavItem>
                    <NavItem><NavLinks to ="/cust_home/mybookings">{isAdmin ? '' :
'Bookings'}</NavLinks></NavItem>

                    {isAdmin && adminRouter()}
                    {
```

```
                    isLogged ? loggedRouter() : ""
                    }
                </NavMenu>

            </NavbarContainer>
        </Nav>
        </IconContext.Provider>
       </>
    );
}

export default Navbar
```

Sidebar.js

```
import React ,{useContext}from 'react'
import { SidebarContainer, Icon, CloseIcon, SidebarWrapper,SidebarRoute,
SidebarMenu,SidebarLink, SideBtnWrap} from './SidebarElements'
import {GlobalState} from '../../../GlobalState'
import axios from 'axios';

const Sidebar = ({isOpen, toggle}) => {
    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    const [isAdmin] = state.userAPI.isAdmin
    function reloadPage(){
      window.location.href = "/postLogin"

  }

  const logoutUser = async() =>{
      await axios.get('/user/logout')
      localStorage.clear()
      window.location.href = "/";

  }

  const adminRouter = () => {
      return(
          <>
              <SidebarMenu><SidebarLink to ="/carservice">Admin</SidebarLink></SidebarMenu>
              <SidebarMenu><SidebarLink to ="/admin_home/cars">Cars</SidebarLink></SidebarMenu>
              <SidebarMenu><SidebarLink to
="/admin_home/packages">Services</SidebarLink></SidebarMenu>
              <SidebarMenu><SidebarLink to
="/admin_home/mechanics">Mechanics</SidebarLink></SidebarMenu>
              <SidebarMenu><SidebarLink to
="/admin_home/orders">Orders</SidebarLink></SidebarMenu>
          </>
      )
```

```
  }
  const loggedRouter = () => {
      return(
          <>
              <SidebarMenu><SidebarLink to ="/"
onClick={logoutUser}>Logout</SidebarLink></SidebarMenu>
          </>
      )
  }

    return (
      <SidebarContainer isOpen ={isOpen} onClick={toggle}>
          <Icon onClick={toggle}>
              <CloseIcon />
          </Icon>
          <SidebarWrapper>
              <SidebarMenu>
                {isAdmin ?
                 adminRouter()
                    :
                <>
                  <SidebarLink to="/carservice" onClick={toggle}>Home</SidebarLink>
                  <SidebarLink to="/working" onClick={toggle}>Working</SidebarLink>
                  <SidebarLink to="/cust_home/mybookings"
onClick={toggle}>Bookings</SidebarLink>
                </>
                }
                  {
                    isLogged ? loggedRouter() : ""
                  }
              </SidebarMenu>
              <SideBtnWrap>
                  <SidebarRoute to='/postLogin' onClick={reloadPage}>Home</SidebarRoute>
              </SideBtnWrap>
          </SidebarWrapper>
      </SidebarContainer>
    )
}

export default Sidebar
```

Working.js

```
import React from "react";
import "./Working.css";
import { makeStyles } from "@material-ui/core/styles";
import Timeline from "@material-ui/lab/Timeline";
import TimelineItem from "@material-ui/lab/TimelineItem";
import TimelineSeparator from "@material-ui/lab/TimelineSeparator";
import TimelineConnector from "@material-ui/lab/TimelineConnector";
```

```
import TimelineContent from "@material-ui/lab/TimelineContent";
import TimelineDot from "@material-ui/lab/TimelineDot";
import DriveEtaIcon from "@material-ui/icons/DriveEta";
import SettingsIcon from "@material-ui/icons/Settings";
import AttachMoneyIcon from "@material-ui/icons/AttachMoney";
import BookmarksIcon from "@material-ui/icons/Bookmarks";
import WeekendIcon from "@material-ui/icons/Weekend";
import Paper from "@material-ui/core/Paper";
import Typography from "@material-ui/core/Typography";

const useStyles = makeStyles((theme) => ({
  paper: {
    padding: "6px 16px",
  },
  secondaryTail: {
    backgroundColor: theme.palette.secondary.main,
  },
}));
function Working() {
  const classes = useStyles();
  return (
    <div className="container" style={{textAlign: "center"}}>
      <h1>How PRONTO Works?</h1>

      <Timeline align="alternate">
        <TimelineItem>
          <TimelineSeparator>
            <TimelineDot color="primary" variant="outlined">
              <DriveEtaIcon />
            </TimelineDot>
            <TimelineConnector />
          </TimelineSeparator>
          <TimelineContent>
            <Paper elevation={3} className={classes.paper}>
              <Typography variant="h6" component="h1">
                Select Your Car
              </Typography>
              <br />
              <Typography style={{textAlign: "left"}}>We Service most makes and
models</Typography>
            </Paper>
          </TimelineContent>
        </TimelineItem>
        <TimelineItem>
          <TimelineSeparator>
            <TimelineDot color="primary">
              <SettingsIcon />
            </TimelineDot>
            <TimelineConnector className={classes.secondaryTail} />
```

```
        </TimelineSeparator>
        <TimelineContent>
          <Paper elevation={3} className={classes.paper}>
            <Typography variant="h6" component="h1">
              Select The Perfect Car Service
            </Typography>
            <br />
            <Typography>From Pronto's broad portfolio of Services</Typography>
          </Paper>
        </TimelineContent>
      </TimelineItem>
      <TimelineItem>
        <TimelineSeparator>
          <TimelineDot color="primary" variant="outlined">
            <AttachMoneyIcon />
          </TimelineDot>
          <TimelineConnector />
        </TimelineSeparator>
        <TimelineContent>
          <Paper elevation={3} className={classes.paper}>
            <Typography variant="h6" component="h1">
              Get A Reasonable Quote
            </Typography>
            <br />
            <Typography style={{textAlign: "left"}}>
              Get a fair and reasonable quote from our website
            </Typography>
          </Paper>
        </TimelineContent>
      </TimelineItem>
      <TimelineItem>
        <TimelineSeparator>
          <TimelineDot color="primary">
            <BookmarksIcon />
          </TimelineDot>
          <TimelineConnector className={classes.secondaryTail} />
        </TimelineSeparator>
        <TimelineContent>
          <Paper elevation={3} className={classes.paper}>
            <Typography variant="h6" component="h1">
              Book An Appointment
            </Typography>
            <br />
            <Typography>
              We offer Free pickup and drop for all services booked
            </Typography>
          </Paper>
        </TimelineContent>
      </TimelineItem>
```

```
        <TimelineItem>
          <TimelineSeparator>
            <TimelineDot color="primary" variant="outlined">
              <WeekendIcon />
            </TimelineDot>
            <TimelineConnector />
          </TimelineSeparator>
          <TimelineContent>
            <Paper elevation={3} className={classes.paper}>
              <Typography variant="h6" component="h1">
                Relax
              </Typography>
              <br />
              <Typography style={{textAlign: "left"}}>
                Relax and spent time on other things that matter
              </Typography>
            </Paper>
          </TimelineContent>
        </TimelineItem>
      </Timeline>
    </div>
  );
}

export default Working;
```

Admin

AdminHome.js

```
import React, { useEffect, useState } from "react";
import "./CSS/AdminHome.css";
import AdminOrders from "../../../services/member/orders.js/admin_orders";
import { withRouter } from "react-router-dom";
import { Card,CardContent } from "@material-ui/core";
import { MuiThemeProvider, createTheme } from "@material-ui/core/styles";

const muiBaseTheme = createTheme();

const theme = {
  overrides: {
    MuiCard: {
      root: {
        "&.MuiEngagementCard--01": {
          transition: "0.3s",
          maxWidth: '50%',
          margin: "auto",
          boxShadow: "0 8px 40px -12px rgba(0,0,0,0.3)",
          "&:hover": {
            boxShadow: "0 16px 70px -12.125px rgba(0,0,0,0.3)"
          },
```

```
          "& .MuiCardMedia-root": {
            paddingTop: "56.25%"
          },
          "& .MuiCardContent-root": {
            textAlign: "left",
            padding: muiBaseTheme.spacing.unit * 3
          },
          "& .MuiDivider-root": {
            margin: `${muiBaseTheme.spacing.unit * 3}px 0`
          },
          "& .MuiTypography--heading": {
            fontWeight: "bold"
          },
          "& .MuiTypography--subheading": {
            lineHeight: 1.8
          },
          "& .MuiAvatar-root": {
            display: "inline-block",
            border: "2px solid white",
            "&:not(:first-of-type)": {
              marginLeft: -muiBaseTheme.spacing.unit
            }
          }
        }
      }
    }
  }
};

function AdminHome(props) {
  const [orders, setOrders] = useState([]);
  const getCompletedOrders = () => {
    AdminOrders.findCompletedOrders()
      .then((res) => {
        setOrders(res);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  useEffect(() => {
    getCompletedOrders();
  }, []);

  return (
    <div className="admin_home">

      <hr />
```

```
      <br />
      <MuiThemeProvider theme={createTheme(theme)} >
        <Card className="MuiEngagementCard--01" style={{backgroundColor:"#01bf7a"}}>
          <CardContent>
            <h1 className="heading_text">WELCOME ADMIN</h1>
          </CardContent>
        </Card>
      </MuiThemeProvider>
      <br />

      <MuiThemeProvider theme={createTheme(theme)} >
        <Card className="MuiEngagementCard--01">
          <CardContent>
          <h1>
              Your Total Earnings: &#8377;
              {orders
              .map((order) => order.servicePrice)
              .reduce((prev, next) => prev + next, 0)}
          </h1>
          </CardContent>
        </Card>
      </MuiThemeProvider>
      <br />
      <hr />
    </div>
  );
}

export default withRouter(AdminHome);
```

Cars.js

```
import React, { useState, useEffect } from "react";
import CarServices from "../../../services/member/car/car_services";
import "./CSS/Cars.css";
import MaterialTable from "material-table";
import {message} from 'antd';

function Cars() {
  const [cars, setCars] = useState([]);

  //for error handling
  //eslint-disable-next-line
  const [iserror, setIserror] = useState(false);
  //eslint-disable-next-line
  const [errorMessages, setErrorMessages] = useState([]);

  const getAllCars = () => {
    CarServices.getAllCars()
      .then((response) => {
```

```
      setCars(response);
    })
    .catch((err) => {
      console.log(err);
    });
};
useEffect(() => {
  getAllCars();
}, []);
//eslint-disable-next-line
const [columns, setColumns] = useState([
  { title: "Name", field: "name" },
  { title: "Brand", field: "brand" },
]);

const handleRowAdd = (newData, resolve) => {
  //validation
  let errorList = [];
  if (newData.name === undefined) {
    errorList.push("Please enter car name");
  }

  if (newData.brand === undefined) {
    errorList.push("Please enter Brand name");
  }

  if (errorList.length < 1) {
    CarServices.addCar(newData.name, newData.brand)
      .then((res) => {
        let dataToAdd = [...cars];
        dataToAdd.push(newData);
        setCars(dataToAdd);
        resolve();
        setErrorMessages([]);
        setIserror(false);
        message.success("New Car Added Successfully")

      })
      .catch((err) => {
        setErrorMessages(["Cannot add data. Server error!"]);
        setIserror(true);
        resolve();
      });
  } else {
    setErrorMessages(errorList);
    setIserror(true);
    resolve();
  }
};
```

```
const handleRowDelete = (oldData, resolve) => {
  CarServices.deleteCar(oldData._id)
    .then((res) => {
      const dataDelete = [...cars];
      const index = oldData.tableData.id;
      dataDelete.splice(index, 1);
      setCars([...dataDelete]);
      resolve();
      message.success("Car Deleted Successfully")

    })
    .catch((error) => {
      setErrorMessages(["Delete failed! Server error"]);
      setIserror(true);
      resolve();
      message.error(errorMessages[0])
    });
};

const handleRowUpdate = (newData, oldData, resolve) => {
  let errorList = [];
  if (newData.brand === undefined) {
    errorList.push("Please enter Brand name");
  }
  if (errorList.length < 1) {
    CarServices.updateCar(newData._id, newData.brand)
      .then((res) => {
        const dataUpdate = [...cars];
        const index = oldData.tableData.id;
        dataUpdate[index] = newData;
        setCars([...dataUpdate]);
        resolve();
        setIserror(false);
        setErrorMessages([]);
        console.log(res);
        message.success("Car Updated Successfully")
      })
      .catch((error) => {
        setErrorMessages(["Update failed! Server error"]);
        setIserror(true);
        resolve();
      });
  } else {
    setErrorMessages(errorList);
    setIserror(true);
    resolve();
    message.error(errorList[0])
  }
```

```
  };

  return (
    <div className="cars_container">
      <MaterialTable
        title="CARS DATA"
        columns={columns}
        data={cars}
        editable={{
          onRowAdd: (newData) =>
            new Promise((resolve, reject) => {
              handleRowAdd(newData, resolve);
            }),
          onRowUpdate: (newData, oldData) =>
            new Promise((resolve, reject) => {
              handleRowUpdate(newData, oldData, resolve);
            }),
          onRowDelete: (oldData) =>
            new Promise((resolve, reject) => {
              handleRowDelete(oldData, resolve);
            }),
        }}
        options={{
          headerStyle: {
            backgroundColor: "#01bf7a",
            color: "black",
            fontWeight: "bold",
            zIndex:'1',
          },
        }}
      />
    </div>
  );
}

export default Cars;
```

Mechanic.js

```
import React, { useState, useEffect } from "react";
import AuthServices from "../../../services/member/auth_service";
import MechanicServices from "../../../services/member/Mechanic/Mechanic_Services";
import "./CSS/Cars.css";
import MaterialTable from "material-table";
import TextField from "@material-ui/core/TextField";
import Button from "@material-ui/core/Button";
import Grid from "@material-ui/core/Grid";
import Container from "@material-ui/core/Container";
import { useForm } from "react-hook-form";
import {message} from 'antd';
```

```
import {Dialog, DialogTitle, DialogContent } from '@material-ui/core';
import CloseIcon from '@mui/icons-material/Close';
import Slide from '@mui/material/Slide';

const Transition = React.forwardRef(function Transition(props, ref) {
    return <Slide direction="up" ref={ref} {...props} />;
  });


function Mechanic() {
  const [mechanic, setMechanic] = useState([]);

  const getAllMecahnic = () => {
    MechanicServices.findAll()
      .then((response) => {
        setMechanic(response);
      })
      .catch((err) => {
        console.log(err);
      });
  };
  useEffect(() => {
    getAllMecahnic();
  }, []);
  //eslint-disable-next-line
  const [columns, setColumns] = useState([
    { title: "ID", field: "_id" },
    { title: "Name", field: "name" },
    { title: "Email", field: "email" },
    { title: "Mobile", field: "mobile" },
    { title: "Status", field: "status" },
  ]);

  const { handleSubmit, register, errors } = useForm({
    mode: "onBlur",
  });

  const [display, setdisplay] = useState(false);
  const openForm = () => {
    setdisplay(true);
  };

  const closeForm = () => {
    setdisplay(false);
  };

  const onSubmit = (values) => {
    AuthServices.registerMechanic(
      values.name,
```

```
      values.email,
      values.password,
      values.mobile
    )
      .then((res) => {
        message.success("Mechanic Added Successfully")
        setdisplay(false)
      })
      .catch((err) => {
        console.log(err);
      });
  };

  const handleRowDelete = (oldData, resolve) => {
    MechanicServices.deleteAccount(oldData._id)
      .then((res) => {
        const dataDelete = [...mechanic];
        const index = oldData.tableData.id;
        dataDelete.splice(index, 1);
        setMechanic([...dataDelete]);
        resolve();
        message.success("Mechanic Removed Successfully")
      })
      .catch((error) => {
        message.error("Delete failed! Server Error")
        resolve();
      });
  };

  return (
    <div className="cars_container">
      <h3>Mechanic Operations</h3>
      <br />

      <button onClick={openForm}>Add Mechanic</button>
      <br />
      <MaterialTable
        title="MECHANIC DATA"
        columns={columns}
        data={mechanic}
        editable={{
          onRowDelete: (oldData) =>
            new Promise((resolve, reject) => {
              handleRowDelete(oldData, resolve);
            }),
        }}
        options={{
          headerStyle: {
            backgroundColor: "#01bf7a",
```

```
            color: "black",
            fontWeight: 'bold',
            zIndex:'1',
          },
        }}
      />

        <Container maxWidth="sm">
        <div className="login__form">
          <Dialog open={display} maxWidth="sm" TransitionComponent={Transition} keepMounted>
            <DialogTitle sx={{ m: 0, p: 2 }}><div><h4>Create Mechanic
Account</h4></div></DialogTitle>
            <CloseIcon onClick={closeForm} sx={{
            position: 'absolute',
            right: 8,
            top: 8,
            cursor: "pointer",
            }}/>
            <DialogContent dividers style={{overflow: "hidden"}}>
            <form onSubmit={handleSubmit(onSubmit)}>
              <Grid container spacing={2}>
                <Grid item xs={12} sm={6}>
                  <TextField
                    autoComplete="name"
                    name="name"
                    variant="outlined"
                    fullWidth
                    label="Name"
                    inputRef={register({
                      required: "Name is Required",
                    })}
                  />
                  {errors.name && <span>{errors.name.message}</span>}
                </Grid>
                <Grid item xs={12} sm={6}>
                  <TextField
                    variant="outlined"
                    fullWidth
                    label="Mobile"
                    name="mobile"
                    inputRef={register({
                      required: "Mobile is Required",
                    })}
                  />
                  {errors.mobile && <span>{errors.mobile.message}</span>}
                </Grid>
                <Grid item xs={12}>
                  <TextField
                    variant="outlined"
```

```
                fullWidth
                label="Email Address"
                name="email"
                autoComplete="email"
                inputRef={register({
                  required: "Email is Required",
                  pattern: {
                    value: /^[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,}$/i,
                    message: "Invalid email address",
                  },
                })}
              />
              {errors.email && <span>{errors.email.message}</span>}
            </Grid>
            <Grid item xs={12}>
              <TextField
                variant="outlined"
                fullWidth
                name="password"
                label="Password"
                type="password"
                inputRef={register({
                  required: "Password is Required",
                })}
              />
              {errors.password && <span>{errors.password.message}</span>}
            </Grid>
          </Grid>
          <br />
          <br />
          <Button
            type="submit"
            fullWidth
            variant="contained"
            className="login_button"
            style={{backgroundColor:"#01bf7a", fontWeight:'bold'}}
          >
            CREATE ACCOUNT
          </Button>
        </form>
        </DialogContent>
      </Dialog>
      </div>
      <br />
      <br />
      <br />
      <br />
    </Container>
  </div>
```

```
  );
}

export default Mechanic;
```

Order.js

```
import React, { useState, useEffect } from "react";
import AdminOrders from "../../../services/member/orders.js/admin_orders";
import "./CSS/Cars.css";
import MaterialTable from "material-table";

function Orders() {
  const [orders, setOrders] = useState([]);
  const [completedOrders, setCompletedOrders] = useState([]);

  //for error handling
  //eslint-disable-next-line
  const [iserror, setIserror] = useState(false);
  //eslint-disable-next-line
  const [errorMessages, setErrorMessages] = useState([]);

  const getPlacedOrders = () => {
    AdminOrders.findPlacedOrders()
      .then((response) => {
        setOrders(response);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  const getCompletedOrders = () => {
    AdminOrders.findCompletedOrders()
      .then((res) => {
        setCompletedOrders(res);
      })
      .catch((err) => {
        console.log(err);
      });
  };

  useEffect(() => {
    getPlacedOrders();
    getCompletedOrders();
  }, []);

  const dynamicMechanicsLookUp = {
    "620375eb03ae092e3430d47e": "Mechanic 1",
    "6206066e3f529a3f2492c152": "Mechanic 2",
```

```
  "6206afc4e365392c94d1d0cb": "Mechanic 3",
  "6206b003e365392c94d1d0d0": "Mechanic 4",
};
//eslint-disable-next-line
const [columns, setColumns] = useState([
  { title: "OrderId", field: "_id", editable: "never" },
  { title: "Customer Name", field: "customerName", editable: "never" },
  { title: "Car Name", field: "carName", editable: "never" },
  { title: "Car Number", field: "carNumber", editable: "never" },
  { title: "Address", field: "custAddress", editable: "never" },
  { title: "Service Name", field: "serviceName", editable: "never" },
  { title: "Price", field: "servicePrice", editable: "never" },
  {
    title: "Assign Mechanic",
    field: "mechanicId",
    lookup: dynamicMechanicsLookUp,
  },
]);
//eslint-disable-next-line
const [column, setColumn] = useState([
  { title: "OrderId", field: "_id" },
  { title: "Customer Name", field: "customerName" },
  { title: "Car Name", field: "carName" },
  { title: "Car Number", field: "carNumber" },
  { title: "Address", field: "custAddress" },
  { title: "Service Name", field: "serviceName" },
  { title: "Price", field: "servicePrice" },
  { title: "Assigned Mechanic", field: "mechanicId" },
]);

const handleRowUpdate = (newData, oldData, resolve) => {
  let errorList = [];
  if (errorList.length < 1) {
    AdminOrders.assignOrder(newData._id, newData.mechanicId)
      .then((res) => {
        const dataUpdate = [...orders];
        const index = oldData.tableData.id;
        dataUpdate[index] = newData;
        setOrders([...dataUpdate]);
        resolve();
        setIserror(false);
        setErrorMessages([]);
        // enqueueSnackbar(res, {
        //   variant: "success",
        // });
      })
      .catch((error) => {
        setErrorMessages(["Update failed! Server error"]);
        setIserror(true);
```

```
        resolve();
      });
  } else {
    setErrorMessages(errorList);
    setIserror(true);
    resolve();
  }
};

const [display, setdisplay] = useState(false);
const openTable = () => {
  setdisplay(true);
};

const closeTable = () => {
  setdisplay(false);
};
return (
  <div className="cars_container">
    <br />

    <button onClick={openTable}>See Completed Orders</button>
    <br />
    {orders ? (
      <MaterialTable
        title="CURRENT ORDERS DATA"
        columns={columns}
        data={orders}
        editable={{
          onRowUpdate: (newData, oldData) =>
            new Promise((resolve, reject) => {
              handleRowUpdate(newData, oldData, resolve);
            }),
        }}
        options={{
          headerStyle: {
            backgroundColor: "#01bf7a",
            color: "black",
            fontWeight: "bold",
          },
          exportButton: true,
        }}
      />
    ) : (
      <div>
        <br />
        <h2>NO CURRENT ORDERS RIGHT NOW</h2>
      </div>
    )}
```

```
        <br />
        <br />
        <br />

        {display ? (
          <div>
            <h1>COMPLETED ORDERS</h1>
            <MaterialTable
              title="CURRENT ORDERS DATA"
              columns={column}
              data={completedOrders}
              options={{
                headerStyle: {
                  backgroundColor: "#01bf7a",
                  color: "black",
                  fontWeight: "bold",
                  zIndex:'1',
                },
                exportButton: true,
              }}
            />
            <br />
            <button onClick={closeTable}>Close Table</button>
            <br />
            <br />
            <br />
          </div>
        ) : null}
      </div>
    );
}

export default Orders;
```

Services.js

```
import React, { useState, useEffect } from "react";
import PackageServices from "../../../services/member/package/package_services";
import "./CSS/Cars.css";
import MaterialTable from "material-table";
import {message} from 'antd';

function Services() {
  const [services, setServices] = useState([]);

  //for error handling
  //eslint-disable-next-line
  const [iserror, setIserror] = useState(false);
  //eslint-disable-next-line
```

```
const [errorMessages, setErrorMessages] = useState([]);

const getAllServices = () => {
  PackageServices.getAllServices()
    .then((response) => {
      setServices(response);
    })
    .catch((err) => {
      console.log(err);
    });
};
useEffect(() => {
  getAllServices();
}, []);

const dynamicTypeLookUp = {
  1: "Car Care Services",
  2: "Periodic Car Service",
};
const dynamicWhereLookUp = {
  1: "Free Pickup & Drop",
  2: "Service @ Doorstep",
};
//eslint-disable-next-line
const [columns, setColumns] = useState([
  {
    title: "TYPE",
    field: "serviceType",
    lookup: dynamicTypeLookUp,
  },
  { title: "NAME", field: "name" },
  { title: "PRICE", field: "price" },
  { title: "DESCRIPTION", field: "description" },
  { title: "TIME", field: "timeRequired" },
  { title: "WHERE", field: "where", lookup: dynamicWhereLookUp },
]);

const handleRowAdd = (newData, resolve) => {
  //validation
  let errorList = [];
  if (newData === undefined) {
    errorList.push("All fields are Required");
  }
  if (errorList.length < 1) {
    PackageServices.addService(
      newData.serviceType,
      newData.name,
      newData.price,
      newData.description,
```

```
      newData.timeRequired,
      newData.where
    )
      .then((res) => {
        let dataToAdd = [...services];
        dataToAdd.push(newData);
        setServices(dataToAdd);
        resolve();
        setErrorMessages([]);
        setIserror(false);
        message.success("Service Added Successfully")
      })
      .catch((err) => {
        setErrorMessages(["Cannot add data. Server error!"]);
        setIserror(true);
        resolve();
      });
  } else {
    setErrorMessages(errorList);
    setIserror(true);
    resolve();
    message.error(errorList[0])
  }
};

const handleRowDelete = (oldData, resolve) => {
  PackageServices.deleteService(oldData._id)
    .then((res) => {
      const dataDelete = [...services];
      const index = oldData.tableData.id;
      dataDelete.splice(index, 1);
      setServices([...dataDelete]);
      resolve();
      message.success("Service Deleted Successfully")
    })
    .catch((error) => {
      setErrorMessages(["Delete failed! Server error"]);
      setIserror(true);
      resolve();
      message.error(errorMessages[0])
    });
};

const handleRowUpdate = (newData, oldData, resolve) => {
  let errorList = [];
  if (errorList.length < 1) {
    PackageServices.updateService(
      newData._id,
      newData.serviceType,
```

```
        newData.name,
        newData.price,
        newData.description,
        newData.timeRequired,
        newData.where
      )
        .then((res) => {
          const dataUpdate = [...services];
          const index = oldData.tableData.id;
          dataUpdate[index] = newData;
          setServices([...dataUpdate]);
          resolve();
          setIserror(false);
          setErrorMessages([]);
          message.success("Service Updated Successfully")
        })
        .catch((error) => {
          setErrorMessages(["Update failed! Server error"]);
          setIserror(true);
          resolve();
        });
    } else {
      setErrorMessages(errorList);
      setIserror(true);
      resolve();
    }
  };

  return (
    <div className="cars_container">
      <MaterialTable
        title="SERVICES DATA"
        columns={columns}
        data={services}
        editable={{
          onRowAdd: (newData) =>
            new Promise((resolve, reject) => {
              handleRowAdd(newData, resolve);
            }),
          onRowUpdate: (newData, oldData) =>
            new Promise((resolve, reject) => {
              handleRowUpdate(newData, oldData, resolve);
            }),
          onRowDelete: (oldData) =>
            new Promise((resolve, reject) => {
              handleRowDelete(oldData, resolve);
            }),
        }}
        options={{
```

```
          headerStyle: {
            backgroundColor: "#01bf7a",
            color: "black",
            fontWeight: "bold",
            zIndex:'1',
          },
        }}
      />
    </div>
  );
}

export default Services;
```

Mechanic

FindOrder.js

```
import React, { useState, useEffect } from "react";

import MechanicOrders from "../../../services/member/Mechanic/Mechanic_Orders";
import AuthService from "../../../services/member/auth_service";
import Navbar from "./navbar";
import "../Admin/CSS/Cars.css";
import "../Admin/CSS/AdminHome.css";
import MaterialTable from "material-table";
import {message} from 'antd'
import { Card, CardContent } from "@material-ui/core";
import { MuiThemeProvider, createTheme } from "@material-ui/core/styles";
import Sidebar from "./sidebar";

const muiBaseTheme = createTheme();

const theme = {
  overrides: {
    MuiCard: {
      root: {
        "&.MuiEngagementCard--01": {
          transition: "0.3s",
          maxWidth: '90%',
          margin: "auto",
          boxShadow: "0 8px 40px -12px rgba(0,0,0,0.3)",
          "&:hover": {
            boxShadow: "0 16px 70px -12.125px rgba(0,0,0,0.3)"
          },
          "& .MuiCardMedia-root": {
            paddingTop: "56.25%"
          },
```

```
          "& .MuiCardContent-root": {
            textAlign: "left",
            padding: muiBaseTheme.spacing(3)
          },
          "& .MuiDivider-root": {
            margin: `${muiBaseTheme.spacing(3)}px 0`
          },
          "& .MuiTypography--heading": {
            fontWeight: "bold"
          },
          "& .MuiTypography--subheading": {
            lineHeight: 1.8
          },
          "& .MuiAvatar-root": {
            display: "inline-block",
            border: "2px solid white",
            "&:not(:first-of-type)": {
              marginLeft: -muiBaseTheme.spacing.unit
            }
          }
        }
      }
    }
  }
};

function FindOrders() {

  const[isOpen, setIsOpen] = useState(false);
    const toggle = () => {
        setIsOpen(!isOpen)
  }
  const [orders, setOrders] = useState([]);

  //for error handling
  //eslint-disable-next-line
  const [iserror, setIserror] = useState(false);
  //eslint-disable-next-line
  const [errorMessages, setErrorMessages] = useState([]);

  useEffect(() => {
    const mechanic = AuthService.getCurrentMechanic();
    MechanicOrders.getInProcessOrders(mechanic.userId)
      .then((response) => {
        setOrders(response);
      })
      .catch((err) => {
        console.log(err);
      });
```

```
}, []);

const dynamicMechanicsLookUp = {
  ACCEPTED: "ACCEPTED",
  REJECT: "REJECTED",
  COMPLETED: "COMPLETED",
};
//eslint-disable-next-line
const [columns, setColumns] = useState([
  { title: "OrderId", field: "_id", editable: "never" },
  { title: "Customer Name", field: "customerName", editable: "never" },
  { title: "Car Name", field: "carName", editable: "never" },
  { title: "Car Number", field: "carNumber", editable: "never" },
  { title: "Address", field: "custAddress", editable: "never" },
  { title: "Service Name", field: "serviceName", editable: "never" },
  { title: "Price", field: "servicePrice", editable: "never" },
  {
    title: "Status",
    field: "status",
    lookup: dynamicMechanicsLookUp,
  },
]);

const handleRowUpdate = (newData, oldData, resolve) => {
  let errorList = [];
  if (errorList.length < 1) {
    MechanicOrders.updateOrder(newData._id, newData.status)
      .then((res) => {
        const dataUpdate = [...orders];
        const index = oldData.tableData.id;
        dataUpdate[index] = newData;
        setOrders([...dataUpdate]);
        resolve();
        setIserror(false);
        setErrorMessages([]);
        message.success("Order Updated Successfully")
      })
      .catch((error) => {
        setErrorMessages(["Update failed! Server error"]);
        setIserror(true);
        resolve();
      });
  } else {
    setErrorMessages(errorList);
    setIserror(true);
    resolve();
  }
};
```

```
return (
  <>
  <Sidebar isOpen ={isOpen} toggle={toggle} />
  <Navbar toggle = {toggle}/>

  <div className="admin_home">
    <hr />
    <br />
    <MuiThemeProvider theme={createTheme(theme)} >
      <Card className="MuiEngagementCard--01" style={{backgroundColor:"black"}}>
        <CardContent>
          <h1 className="heading_text" style={{color: "white"}}>WELCOME MECHANIC</h1>
        </CardContent>
      </Card>
    </MuiThemeProvider>
    <br />
    <hr />
  </div>

  <div className="cars_container">
    {orders ? (
      <MaterialTable
        title="IN PROCESS ORDERS DATA"
        columns={columns}
        data={orders}
        editable={{
          onRowUpdate: (newData, oldData) =>
            new Promise((resolve, reject) => {
              handleRowUpdate(newData, oldData, resolve);
            }),
        }}
        options={{
          headerStyle: {
            backgroundColor: "#01bf7a",
            color: "black",
            fontWeight: "bold",
            zIndex:'1',
          },
          exportButton: true,
        }}
      />
    ) : (
      <div>
        <br />
        <h2>  NO ASSIGNED ORDERS RIGHT NOW</h2>
      </div>
    )}
  </div>
  </>
```

```
  );

}

export default FindOrders;
```

MyOrder.js

```
import React, { useState, useEffect } from "react";
import MechanicOrders from "../../../services/member/Mechanic/Mechanic_Orders";
import AuthService from "../../../services/member/auth_service";
import Navbar from './navbar'
import "../Admin/CSS/Cars.css";
import MaterialTable from "material-table";
import Sidebar from "./sidebar";

function MyOrders() {
  const[isOpen, setIsOpen] = useState(false);
    const toggle = () => {
        setIsOpen(!isOpen)
  }
  const [orders, setOrders] = useState([]);
  // const { enqueueSnackbar, closeSnackbar } = useSnackbar();

  //for error handling
  //eslint-disable-next-line
  const [iserror, setIserror] = useState(false);
  //eslint-disable-next-line
  const [errorMessages, setErrorMessages] = useState([]);

  useEffect(() => {
    const mechanic = AuthService.getCurrentMechanic();
    MechanicOrders.getAllOrders(mechanic.userId)
      .then((response) => {
        setOrders(response);
      })
      .catch((err) => {
        console.log(err);
      });
  }, []);
  //eslint-disable-next-line
  const [columns, setColumns] = useState([
    { title: "OrderId", field: "_id" },
    { title: "Customer Name", field: "customerName" },
    { title: "Car Name", field: "carName" },
    { title: "Car Number", field: "carNumber" },
    { title: "Address", field: "custAddress" },
    { title: "Service Name", field: "serviceName" },
    { title: "Price", field: "servicePrice" },
    { title: "Status", field: "status" },
```

```
  ]);

  return (
    <>
    <Sidebar isOpen ={isOpen} toggle={toggle} />
    <Navbar toggle = {toggle}/>
    <div className="cars_container">
      <MaterialTable
        title="MY ORDERS DATA"
        columns={columns}
        data={orders}
        options={{
          headerStyle: {
            backgroundColor: "#01bf7a",
            color: "black",
            fontWeight: "bold",
            zIndex:'1',
          },
          exportButton: true,
        }}
      />
    </div>
    </>
  );
}

export default MyOrders;
```

Navbar.js

```
import React ,{useState,useEffect}from 'react'
import logo from '../../../../../images/logo192.png'
import { Nav, NavbarContainer, NavLogo, MobileIcon, NavMenu, NavItem,NavLinks} from
'./NavbarElements';
import {FaBars} from 'react-icons/fa'
import {IconContext} from 'react-icons/lib'

const Navbar = ({toggle}) => {

    const [scrollNav, setScrollNav] = useState(false)
    const changeNav= () => {
        if(window.scrollY >= 80){
            setScrollNav(true)
        }
        else{
            setScrollNav(false)
        }
    };

    useEffect(() => {
```

```
        window.addEventListener('scroll', changeNav)
  },[]);

    const logoutUser = async() =>{
        localStorage.clear('mechanic')
        window.location.href = "/";

    }
    return (
       <>
       <IconContext.Provider value={{color: '#fff'}}>
        <Nav scrollNav ={scrollNav}>
            <NavbarContainer>
                <NavLogo><img src = {logo} alt='p' height={40} width={40} ></img>Pronto-
Mechanic</NavLogo>
                <MobileIcon onClick={toggle}>
                    <FaBars style={{color: "white"}}/>
                </MobileIcon>
                <NavMenu>
                    <NavItem>
                        <NavLinks to="/mechanic_home/findOrders" smooth ={true} duration={500}
spy={true} exact ='true' offset ={-80}>Find Orders</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks to="/mechanic_home/myorders" smooth ={true} duration={500}
spy={true} exact ='true' offset ={-80}>My Orders</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks onClick={logoutUser}>Logout</NavLinks>
                    </NavItem>
                </NavMenu>
            </NavbarContainer>
        </Nav>
        </IconContext.Provider>
       </>
    );
}

export default Navbar
```

Sidebar.js

```
import React from 'react'
import { SidebarContainer, Icon, CloseIcon, SidebarWrapper,SidebarRoute,
SidebarMenu,SidebarLink, SideBtnWrap} from './SidebarElements'

const Sidebar = ({isOpen, toggle}) => {
  const logoutUser = async() =>{
    localStorage.clear('mechanic')
    window.location.href = "/";
```

```
  }

    return (
      <SidebarContainer isOpen ={isOpen} onClick={toggle}>
          <Icon onClick={toggle}>
              <CloseIcon />
          </Icon>
          <SidebarWrapper>
              <SidebarMenu>
                  <SidebarLink to ="/mechanic_home/findOrders">Find Orders</SidebarLink>
                  <SidebarLink to ="/mechanic_home/myorders">My Orders</SidebarLink>
              </SidebarMenu>
              <SideBtnWrap>
                  <SidebarRoute onClick={logoutUser}>Logout</SidebarRoute>
              </SideBtnWrap>
          </SidebarWrapper>
      </SidebarContainer>
    )
}

export default Sidebar
```

Pages.js

```
import React, {useContext} from 'react';
import {Switch, Route} from 'react-router-dom'
import {GlobalState} from '../../GlobalState'
import NotFound from '../../utils/NotFound/NotFound'
import CustHome from '../components/Customer/CustHome'
import Cars from '../components/Customer/Cars'
import Services from '../components/Customer/Services'
import Order from '../components/Customer/Order'
import MyBookings from '../components/Customer/MyBookings'
import AdminHome from '../components/Member/Admin/AdminHome'
import ACars from '../components/Member/Admin/Cars'
import AServices from '../components/Member/Admin/Services'
import Mechanic from '../components/Member/Admin/Mechanic'
import AOrders from '../components/Member/Admin/Order'
import whyus from '../components/Home/WhyUs'
import working from '../components/Home/Working'
import MechanicHome from '../components/Member/Mechanic/MechanicHome'
import FindOrders from '../components/Member/Mechanic/FindOrders';
import MyOrders from './Member/Mechanic/MyOrders';

export default function Pages() {
    const state = useContext(GlobalState)
    const[isAdmin] = state.userAPI.isAdmin

    return(
```

```
        <Switch>
            <Route path = "/carservice" component ={isAdmin ? AdminHome : CustHome} exact />
            <Route path = "/cust_home/cars/:brand" render ={(props) => <Cars {...props} />}
exact/>
            <Route path = "/cust_home/services/:car" render ={(props) => <Services {...props}
/>} exact/>
            <Route path = "/cust_home/order/car/:carId/service/:serviceId" render ={(props) =>
<Order {...props} />} exact/>
            <Route path = "/cust_home/mybookings" render ={(props) => <MyBookings {...props} />}
exact/>
            <Route path = "/admin_home/cars" component={isAdmin ? ACars : NotFound} exact />
            <Route path = "/admin_home/packages" component={AServices} exact />
            <Route path = "/admin_home/mechanics" component={Mechanic} exact />
            <Route path ="/admin_home/orders" component={AOrders} exact />
            <Route path = "/working" component = {working} exact />
            <Route path ="/mechanic_home/findOrders" component = {FindOrders} />
            <Route path ="/mechanic_home/myorders" component = {MyOrders} />
        </Switch>
    );
}
```

HomeDelivery

Cart.js

```
import React, {useContext, useState, useEffect} from 'react'
import {GlobalState} from '../../../GlobalState'
import axios from 'axios'
import PaypalButton from './PaypalButton'

export default function Cart() {
    const state = useContext(GlobalState)
    const [cart, setCart] = state.userAPI.cart
    const [total, setTotal] = useState(0)
    const [callback, setCallback]= state.userAPI.callback
    const [token] = state.tokens


    useEffect(() => {
        const getTotal = () => {
            const total = cart.reduce((prev,item) => {
                return prev +(item.price*item.quantity)
            },0)
            setTotal(total)
        }
        getTotal()


    }, [cart])
```

```
const addToCart = async (cart) => {
    await axios.patch('/user/addcart', {cart}, {
         headers: {Authorization: token}
     })
 }

const increment = (id) =>{
    cart.forEach(item => {
        if(item._id === id){
            item.quantity += 1
        }
    })
    setCart([...cart])
    addToCart(cart)
}

const decrement = (id) =>{
    cart.forEach(item => {
        if(item._id === id){
            item.quantity === 1 ? item.quantity = 1 : item.quantity -= 1
        }
    })
    setCart([...cart])
    addToCart(cart)
}

const removeProduct = id =>{
    if(window.confirm("Do You want to delete the Item?")){
        cart.forEach((item, index) => {
            if(item._id === id){
                cart.splice(index, 1)
            }
        })

        setCart([...cart])
        addToCart(cart)
    }
}

const tranSuccess = async(payment) =>{
    const {paymentID, address} = payment;

  await axios.post('/api/payment', {cart, paymentID, address}, {
        headers: {Authorization: token}
    })

    setCart([])
    addToCart([])
```

```
        alert("You have successfully placed the order.")
        setCallback(!callback)
    }

    if(cart.length === 0)
    {
        return <h2 style={{textAlign: "center", fontSize: "5rem"}}>Cart Empty</h2>
    }
    return (
        <div>
            <br />
           {
               cart.map(product =>(
                    <div className="detail cart" key={product._id}>
                        <img src={product.images.url} alt=""/>
                        <div className="box-detail">
                            <h2>{product.title}</h2>


                            <span>₹{product.price * product.quantity}</span>
                            <p>{product.description}</p>
                            <p>{product.content}</p>
                            <div className="amount">
                                <button onClick={() => decrement(product._id)}> - </button>
                                <span> {product.quantity} </span>
                                <button onClick={() =>increment(product._id)}> + </button>
                            </div>

                            <div className="delete" onClick={() =>
removeProduct(product._id)}>X</div>
                        </div>
                    </div>
               ) )
           }

           <div className="total">
               <h3>Total: ₹ {total}</h3>
               <PaypalButton  total ={ total} tranSuccess = {tranSuccess}/>
           </div>
        </div>
    )
}
```

PaypalButton.js

```
import React from 'react';
import PaypalExpressBtn from 'react-paypal-express-checkout';

export default class PaypalButton extends React.Component {
    render() {
```

```
        const onSuccess = (payment) => {
            // Congratulation, it came here means everything's fine!
                    console.log("The payment was succeeded!");
                    this.props.tranSuccess(payment)
                    // You can bind the "payment" object's value to your state or props or
whatever here, please see below for sample returned data
        }

        const onCancel = (data) => {
            // User pressed "cancel" or close Paypal's popup!
            console.log('The payment was cancelled!', data);
            // You can bind the "data" object's value to your state or props or whatever here,
please see below for sample returned data
        }

        const onError = (err) => {
            // The main Paypal's script cannot be loaded or somethings block the loading of that
script!
            console.log("Error!", err);
            // Because the Paypal's main script is loaded asynchronously from
"https://www.paypalobjects.com/api/checkout.js"
            // => sometimes it may take about 0.5 second for everything to get set, or for the
button to appear
        }

        let env = 'sandbox'; // you can set here to 'production' for production
        let currency = 'USD'; // or you can set this value from your props or state
        let total = this.props.total; // same as above, this is the total amount (based on
currency) to be paid by using Paypal express checkout
        // Document on Paypal's currency code:
https://developer.paypal.com/docs/classic/api/currency_codes/

        const client = {
            sandbox:    'ARoElwYoAA-fGgR6V2XSlv1Yg09kX4JFA67MFAWNLnxzhPMEYRCx-FfKCwQ9ZuhXrLY-
_Yl-uaoOwOnF',
            production: '',
        }
        // In order to get production's app-ID, you will have to send your app to Paypal for
approval first
        // For sandbox app-ID (after logging into your developer account, please locate the
"REST API apps" section, click "Create App"):
        //   => https://developer.paypal.com/docs/classic/lifecycle/sb_credentials/
        // For production app-ID:
        //   => https://developer.paypal.com/docs/classic/lifecycle/goingLive/

        // NB. You can also have many Paypal express checkout buttons on page, just pass in the
correct amount and they will work!

        let style = {
```

```
            size: 'medium',
            color: 'blue',
            shape: 'rect',
            label: 'checkout',
            tagline: false
        }
        return (
            <PaypalExpressBtn
             env={env}
             client={client}
             currency={currency}
             total={total}
             onError={onError}
             onSuccess={onSuccess}
             onCancel={onCancel}
             style={style} />
        );
    }
}
```

Categories.js

```
import React, {useState, useContext} from 'react'
import {GlobalState} from '../../../GlobalState'
import axios from 'axios'

export default function Categories() {
    const state = useContext(GlobalState)
    const [categories] = state.categoriesAPI.categories
    const [category, setCategory] = useState('')
    const [token] = state.token
    const [callback, setCallback] = state.categoriesAPI.callback
    const [onEdit, setOnEdit] = useState(false)
    const [id, setID] = useState('')

    const createCategory = async e =>{
        e.preventDefault()
        try {
            if(onEdit){
                const res = await axios.put(`/api/category/${id}`, {name: category}, {
                    headers: {Authorization: token}
                })
                alert(res.data.msg)
            }else{
                const res = await axios.post('/api/category', {name: category}, {
                    headers: {Authorization: token}
                })
                alert(res.data)
```

```
            }
            setOnEdit(false)
            setCategory('')
            setCallback(!callback)

        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    const editCategory = async (id, name) =>{
        setID(id)
        setCategory(name)
        setOnEdit(true)
    }

    const deleteCategory = async id =>{
        try {
            const res = await axios.delete(`/api/category/${id}`, {
                headers: {Authorization: token}
            })
            alert(res.data.msg)
            setCallback(!callback)
        } catch (err) {
            alert(err.response.data.msg)
        }
    }



    return (
        <div>
            <div className="categories">
            <form onSubmit={createCategory}>
                <label htmlFor="category">Category</label>
                <input type="text" name="category" value={category} required
                onChange={e => setCategory(e.target.value)} />

                <button type="submit">{onEdit? "Update" : "Create"}</button>
            </form>

            <div className="col">
                {
                    categories.map(category => (
                        <div className="row" key={category._id}>
                            <p>{category.name}</p>
                            <div>
                                <button onClick={() => editCategory(category._id,
category.name)}>Edit</button>
```

```
                                <button onClick={() =>
deleteCategory(category._id)}>Delete</button>
                            </div>
                        </div>
                    ))
                }
            </div>
        </div>
        </div>
    )
}
```

CreateProducts.js

```
import React, {useState, useContext, useEffect} from 'react'
import axios from 'axios'
import { GlobalState } from '../../../GlobalState'
import Loading from '../../../utils/loading/Loading'
import { useParams, useHistory } from 'react-router-dom'

const initialState = {
    product_id: '',
    title: '',
    price: 0,
    description: '',
    content: '',
    category: '',
    _id: ''
}
export default function CreateProduct() {
    const state = useContext(GlobalState)
    const[product,setProduct] = useState(initialState)
    const [categories] = state.categoriesAPI.categories
    const[images, setImages]=useState(false)
    const[loading,setLoading]= useState(false)

    const [isAdmin] = state.userAPI.isAdmin
    const [token] = state.token

    const history = useHistory()
    const param = useParams()

    const [products] = state.productsAPI.products
    const [onEdit, setOnEdit] = useState(false)
    const [callback, setCallback] = state.productsAPI.callback

    useEffect(() => {
        if(param.id){
            setOnEdit(true)
            products.forEach(product => {
```

```
            if(product._id === param.id) {
                setProduct(product)
                setImages(product.images)
            }
        })
    }else{
        setOnEdit(false)
        setProduct(initialState)
        setImages(false)
    }
}, [param.id, products])

const handleUpload = async e =>{
    e.preventDefault()
    try {
        if(!isAdmin) return alert("You're not an admin")
        const file = e.target.files[0]

        if(!file) return alert("File not exist.")

        if(file.size > 1024 * 1024) // 1mb
            return alert("Size too large!")

        if(file.type !== 'image/jpeg' && file.type !== 'image/png') // 1mb
            return alert("File format is incorrect.")

        let formData = new FormData()
        formData.append('file', file)

        setLoading(true)
        const res = await axios.post('/api/upload', formData, {
            headers: {'content-type': 'multipart/form-data', Authorization: token}
        })
        setLoading(false)
        setImages(res.data)

    } catch (err) {
        alert(err.response.data.msg)
    }
}

const handleDestroy = async () => {
    try {
        if(!isAdmin) return alert("You're not an admin")
        setLoading(true)
        await axios.post('/api/destroy', {public_id: images.public_id}, {
            headers: {Authorization: token}
        })
        setLoading(false)
```

```
            setImages(false)
        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    const handleChangeInput = e =>{
        const {name, value} = e.target
        setProduct({...product, [name]:value})
    }

    const handleSubmit = async e =>{
        e.preventDefault()
        try {
            if(!isAdmin) return alert("You're not an admin")
            if(!images) return alert("No Image Upload")

            if(onEdit){
                await axios.put(`/api/products/${product._id}`, {...product, images}, {
                    headers: {Authorization: token}
                })
            }else{
                await axios.post('/api/products', {...product, images}, {
                    headers: {Authorization: token}
                })
            }
            setCallback(!callback)
            history.push("/homedel")
        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    const styleUpload = {
        display: images ? "block" : "none"
    }
    return (
        <div className="create_product">
        <div className="upload">
            <input type="file" name="file" id="file_up" onChange={handleUpload}/>
            {
                loading ? <div id="file_img"><Loading /></div>

                :<div id="file_img" style={styleUpload}>
                    <img src={images ? images.url : ''} alt=""/>
                    <span onClick={handleDestroy}>X</span>
                </div>
            }
```

```
</div>

<form onSubmit={handleSubmit}>
    <div className="row">
        <label htmlFor="product_id">Product ID</label>
        <input type="text" name="product_id" id="product_id" required
        value={product.product_id} onChange={handleChangeInput} disabled={onEdit} />
    </div>

    <div className="row">
        <label htmlFor="title">Title</label>
        <input type="text" name="title" id="title" required
        value={product.title} onChange={handleChangeInput} />
    </div>

    <div className="row">
        <label htmlFor="price">Price</label>
        <input type="number" name="price" id="price" required
        value={product.price} onChange={handleChangeInput} />
    </div>

    <div className="row">
        <label htmlFor="description">Description</label>
        <textarea type="text" name="description" id="description" required
        value={product.description} rows="5" onChange={handleChangeInput} />
    </div>

    <div className="row">
        <label htmlFor="content">Content</label>
        <textarea type="text" name="content" id="content" required
        value={product.content} rows="7" onChange={handleChangeInput} />
    </div>

    <div className="row">
        <label htmlFor="categories">Categories: </label>
        <select name="category" value={product.category} onChange={handleChangeInput} >
            <option value="">Please select a category</option>
            {
                categories.map(category => (
                    <option value={category._id} key={category._id}>
                        {category.name}
                    </option>
                ))
            }
        </select>
    </div>

    <button type="submit">{onEdit? "Update" : "Create"}</button>
</form>
```

```
    </div>
    )
}

DetailProduct.js
import React,{useContext, useEffect, useState}from "react";
import { Link } from "react-router-dom";
import { useParams } from "react-router-dom";
import { GlobalState } from "../../../GlobalState";
import ProductItem from "../../../utils/productItem/ProductItem";

export default function DetailProduct(){
    const params = useParams()
    const state = useContext(GlobalState)
    const[products]=state.productsAPI.products
    const addCart = state.userAPI.addCart
    const [detailProduct, setDetailProduct] = useState([])

    useEffect(() => {
        if(params.id){
            products.forEach(product =>{
                if(product._id === params.id) setDetailProduct(product)

            })
        }
    }, [params.id, products])

    if(detailProduct.length === 0) return null;
    return(
        <>
            <div className="detail">
                <img src={detailProduct.images.url} alt="" />
                <div className="box-detail">
                    <div className="row">
                        <h2>{detailProduct.title}</h2>
                        <h6>#id: {detailProduct.product_id}</h6>
                    </div>
                    <span>₹{detailProduct.price}</span>
                    <p>{detailProduct.description}</p>
                    <p>{detailProduct.content}</p>
                    <p>Sold: {detailProduct.sold}</p>
                    <Link to='/homedel/cart' className="cart" onClick= {() =>
addCart(detailProduct)}>Buy Now</Link>
                </div>
            </div>
            <div>
                <h2>Related Products</h2>
                <div className="products">
                    {
```

```
                        products.map(product =>{
                            return product.category === detailProduct.category
                                ?<ProductItem key ={product._id} product={product}/> : null
                        })
                    }
                </div>
            </div>
        </>

    )
}
```

OrderHistory.js

```
import React , {useContext} from 'react'
import {GlobalState} from '../../../GlobalState'
import {Link} from 'react-router-dom'

function OrderHistory() {
    const state = useContext(GlobalState)
    const [history] = state.userAPI.history

    return (
        <div className="history-page">
            <h2>History</h2>
            <h4>You have {history.length} orders</h4>
            <table>
                <thead>
                    <tr>
                        <th>Payment ID</th>
                        <th>Date of Purchase</th>
                        <th></th>
                    </tr>
                </thead>
                <tbody>
                    {
                        history.map(items => (
                            <tr key={items._id}>
                                <td>{items.paymentID}</td>
                                <td>{new Date(items.createdAt).toLocaleDateString()}</td>
                                <td><Link to={`/history/${items._id}`}>View</Link></td>
                            </tr>
                        ))
                    }
                </tbody>
            </table>
        </div>
    )
}
```

```
export default OrderHistory
```

OrderDetail.js

```
import React, {useState, useEffect, useContext} from 'react'
import {useParams} from 'react-router-dom'
import {GlobalState} from '../../../GlobalState'
import easyinvoice from 'easyinvoice';

function OrderDetails(){
    const state = useContext(GlobalState)
    const[isAdmin] = state.userAPI.isAdmin
    const [history] = state.userAPI.history
    const [orderDetails, setOrderDetails] = useState([])

    const params = useParams()

    useEffect(() => {
        if(params.id){
            history.forEach(item =>{
                if(item._id === params.id) {
                setOrderDetails(item)
                //For billing purpose
                localStorage.setItem('Name',JSON.stringify(item.address.recipient_name))
                localStorage.setItem('Address', JSON.stringify(item.address.line1))
                localStorage.setItem('Zip', JSON.stringify(item.address.postal_code))
                localStorage.setItem('City', JSON.stringify(item.address.city))
                localStorage.setItem('Country', JSON.stringify(item.address.country_code))
                localStorage.setItem('InvNo', JSON.stringify(item.paymentID.slice(6,20)))
                localStorage.setItem('ID', JSON.stringify(item.paymentID))
                localStorage.setItem('Date', JSON.stringify(item.createdAt.slice(0,10)))
                localStorage.setItem('Quantity', JSON.stringify(item.cart[0].quantity))
                localStorage.setItem('Descrip',  JSON.stringify(item.cart[0].description))
                localStorage.setItem('Price', JSON.stringify(item.cart[0].price))
                localStorage.setItem('Title', JSON.stringify(item.cart[0].title.toUpperCase()))
                }
            })
        }
    },[params.id, history])

    /* Invoice Section */
    function Invoice(){

        var data = {
            //"documentTitle": "RECEIPT", //Defaults to INVOICE
        "images": {
        //     // The logo on top of your invoice
```

```
        "logo":
"https://cdn.discordapp.com/attachments/902188275257704462/943553093428183131/logo192.png",},
//or base64
        //"logoExtension": "png", //only when logo is base64
        "sender": {
            "company": "Pronto",
            "address": "Sample Street",
            "zip": "421306",
            "city": "Mumbai",
            "country": "India"
            //"custom1": "custom value 1",
            //"custom2": "custom value 2",
            //"custom3": "custom value 3"
        },
        "client": {
            "company": JSON.parse(localStorage.getItem('Name')),
            "address": JSON.parse(localStorage.getItem('Address')),
            "zip": JSON.parse(localStorage.getItem('Zip')),
            "city": JSON.parse(localStorage.getItem('City')),
            "country": JSON.parse(localStorage.getItem('Country')),
            "custom1": JSON.parse(localStorage.getItem('ID'))
            //"custom1": "custom value 1",
            //"custom2": "custom value 2",
            //"custom3": "custom value 3"
        },
        "information": {
        "number": JSON.parse(localStorage.getItem('InvNo')),
        "date": JSON.parse(localStorage.getItem('Date')),
        "due-date": "NIL"
        },
        "products": [
            {
                "quantity": JSON.parse(localStorage.getItem('Quantity')),
                "description": JSON.parse(localStorage.getItem('Title')),
                "tax-rate": 5,
                "price": JSON.parse(localStorage.getItem('Price'))
            },
        ],
        "bottom-notice": "Thank You For Shopping With Us!!",
        "settings": {
            "currency": "INR",
            "tax-notation": "GST", //or gst
            "marginTop": 25,
            "marginRight": 25,
            "marginLeft": 25,
            "marginBottom": 25,
        },
        "translate": {
            "invoice": "Bill",
```

```
            "number": "Bill No.",
        }


        };
        const downloadInvoice = () => {
        easyinvoice.createInvoice(data, async function(result){
            // console.log(result.pdf)
            easyinvoice.download("invoice.pdf")
            localStorage.clear()
        })
        }
        downloadInvoice()
        }
/*-------------------------------------------------------------------------------------------
-------- */

    if(orderDetails.length === 0) return null;

    return (
        <div className="history-page" style={{paddingTop:'20px'}}>
            <table>
                <thead>
                    <tr>
                        <th>Name</th>
                        <th>Address</th>
                        <th>Postal Code</th>
                        <th>Country Code</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>{orderDetails.address.recipient_name}</td>
                        <td>{orderDetails.address.line1 + " - " +
orderDetails.address.city}</td>
                        <td>{orderDetails.address.postal_code}</td>
                        <td>{orderDetails.address.country_code}</td>
                    </tr>
                </tbody>
            </table>

            <table style={{margin: "30px 0px"}}>
                <thead>
                    <tr>
                        <th></th>
                        <th>Products</th>
                        <th>Quantity</th>
                        <th>Price</th>
                        {isAdmin? null : <th>Bill</th>}
```

```
                    </tr>
                </thead>
                <tbody>
                    {
                        orderDetails.cart.map(item =>(
                        <tr key={item._id}>
                            <td><img src={item.images.url} alt="" /></td>
                            <td>{item.title}</td>
                            <td>{item.quantity}</td>
                            <td>₹ {item.price * item.quantity}</td>
                            {isAdmin? null : <td><button className = 'btn_invoice'
onClick={Invoice}>Download Bill</button> </td>}
                        </tr>
                        ))
                    }

                </tbody>
            </table>
        </div>
    )
}

export default OrderDetails
```

Filter.js

```
import React, {useContext} from "react";
import { GlobalState } from "../../../GlobalState";

export default function Filters() {
    const state = useContext(GlobalState)
    const [categories] = state.categoriesAPI.categories

    const [category, setCategory] = state.productsAPI.category
    const [sort, setSort] = state.productsAPI.sort
    const [search, setSearch] = state.productsAPI.search


    const handleCategory = e => {
        setCategory(e.target.value)
        setSearch('')
    }

    return (
        <div className="filter_menu">
            <div className="row">
                <span>Filters: </span>
                <select name="category" value={category} onChange={handleCategory} >
                    <option value=''>All Products</option>
```

```
                    {
                        categories.map(category => (
                            <option value={"category=" + category._id} key={category._id}>
                                {category.name}
                            </option>
                        ))
                    }
                </select>
            </div>

            <input type="text" value={search} placeholder="Enter your search!"
            onChange={e => setSearch(e.target.value.toLowerCase())} />

            <div className="row sort">
                <span>Sort By: </span>
                <select value={sort} onChange={e => setSort(e.target.value)} >
                    <option value=''>Newest</option>
                    <option value='sort=oldest'>Oldest</option>
                    <option value='sort=-sold'>Best sales</option>
                    <option value='sort=-price'>Price: High-Low</option>
                    <option value='sort=price'>Price: Low-High</option>
                </select>
            </div>
        </div>
    )
}
```

LoadMore.js

```
import React, {useContext} from 'react'
import { GlobalState } from '../../../GlobalState'


export default function LoadMore() {
        const state = useContext(GlobalState)
        const [page, setPage] = state.productsAPI.page
        const [result] = state.productsAPI.result

        return (
            <div className="load_more">
                {
                    result < page * 9 ? ""
                    : <button onClick={() => setPage(page+1)}>Load more</button>
                }
            </div>
        )
}
```

Product.js

```
import React,{useContext, useState} from 'react'
import {GlobalState} from '../../../GlobalState'
import ProductItem from '../../../utils/productItem/ProductItem'
import Loading from '../../../utils/loading/Loading'
import axios from 'axios'
import Filters from './Filters'
import LoadMore from './LoadMore'

function Products() {

    const state = useContext(GlobalState)
    const [products,setProducts] = state.productsAPI.products
    const [isAdmin] = state.userAPI.isAdmin
    const [token] = state.token
    const [callback, setCallback] = state.productsAPI.callback
    const [loading, setLoading] = useState(false)
    const [isCheck, setIsCheck] = useState(false)

    const handleCheck = (id) =>{
        products.forEach(product => {
            if(product._id === id) product.checked = !product.checked
        })
        setProducts([...products])
    }

    const deleteProduct = async(id, public_id) => {
        try {
            setLoading(true)
            const destroyImg = axios.post('/api/destroy', {public_id},{
                headers: {Authorization: token}
            })
            const deleteProduct = axios.delete(`/api/products/${id}`, {
                headers: {Authorization: token}
            })

            await destroyImg
            await deleteProduct
            setCallback(!callback)
            setLoading(false)
        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    const checkAll = () =>{
        products.forEach(product => {
            product.checked = !isCheck
        })
        setProducts([...products])
```

```
        setIsCheck(!isCheck)
    }

    const deleteAll = () =>{
        products.forEach(product => {
            if(product.checked) deleteProduct(product._id, product.images.public_id)
        })
    }

    if(loading) return <div><Loading /></div>

    return (
        <>
        <Filters />
        {
            isAdmin &&
            <div className='delete-all'>
                <span>Select all</span>
                <input type="checkbox" checked={isCheck} onChange={checkAll} />
                <button onClick={deleteAll}>Delete ALL</button>
            </div>
        }
        <div className="products">
            {
                products.map(product =>{
                    return <ProductItem key={product._id} product={product}
                    isAdmin={isAdmin} deleteProduct={deleteProduct} handleCheck={handleCheck} />
                })
            }
        </div>
        <LoadMore />
        {products.length === 0 &&<Loading />}
        </>
    )
}

export default Products
```

Sidebar.js

```
import React, {useContext} from 'react'
import { SidebarContainer, Icon, CloseIcon, SidebarWrapper,
SidebarMenu,SidebarLink,SidebarRoute, SideBtnWrap} from './SidebarElements'
import {GlobalState} from '../../../GlobalState'
import axios from 'axios'

export default function Sidebar({isOpen, toggle}){
    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
```

```
    const [isAdmin] = state.userAPI.isAdmin

    function reloadPage(){
        window.location.href = "/postLogin"

    }

    const logoutUser = async() =>{
        await axios.get('/user/logout')
        localStorage.clear()
        window.location.href = "/";

    }

    const adminRouter = () => {
        return(
            <>
                <SidebarMenu><SidebarLink to ="/create_product">Create
Products</SidebarLink></SidebarMenu>
                <SidebarMenu><SidebarLink to ="/category">Categories</SidebarLink></SidebarMenu>
            </>
        )
    }
    const loggedRouter = () => {
        return(
            <>
                <SidebarMenu><SidebarLink to ="/history">History</SidebarLink></SidebarMenu>
                <SidebarMenu><SidebarLink to ="/"
onClick={logoutUser}>Logout</SidebarLink></SidebarMenu>
            </>
        )
    }

    return (
      <SidebarContainer isOpen ={isOpen} onClick={toggle}>
          <Icon onClick={toggle}>
              <CloseIcon />
          </Icon>
          <SidebarWrapper>
              <SidebarMenu>
              <SidebarLink to="/homedel" onClick={toggle}>{isAdmin ? 'Products' :
'Shop'}</SidebarLink>
              {isAdmin && adminRouter()}
                {
                    isLogged ? loggedRouter() : ""
                }
              </SidebarMenu>
              <SideBtnWrap>
                  <SidebarRoute to='/postLogin' onClick={reloadPage}>Home</SidebarRoute>
```

```
            </SideBtnWrap>
          </SidebarWrapper>
        </SidebarContainer>
    )
}
```

Navbar.js

```
import React ,{useState,useEffect, useContext}from 'react'
import logo from '../../images/logo192.png';
import { Nav, NavbarContainer, NavLogo, MobileIcon, NavMenu, Navspans, NavItem,NavLinks,
NavCart} from './icon/NavbarElements';
import {FaBars} from 'react-icons/fa'
import {IconContext} from 'react-icons/lib'
import axios from 'axios';
import { GlobalState } from '../GlobalState';
import ShoppingCartIcon from '@mui/icons-material/ShoppingCart';

const Navbar = ({toggle}) => {

    const [scrollNav, setScrollNav] = useState(false)
    const changeNav= () => {
        if(window.scrollY >= 80){
            setScrollNav(true)
        }
        else{
            setScrollNav(false)
        }
    };

    useEffect(() => {
        window.addEventListener('scroll', changeNav)
  },[]);

  function reloadPage(){
    window.location.href = "/postLogin"

}

    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    const [isAdmin] = state.userAPI.isAdmin
    const [cart] = state.userAPI.cart
    const logoutUser = async() =>{
        await axios.get('/user/logout')
        localStorage.clear()
        window.location.href = "/";
```

```
    }
    const adminRouter = () => {
        return(
            <>
                <NavItem><NavLinks to ="/create_product">Create Products</NavLinks></NavItem>
                <NavItem><NavLinks to ="/category">Categories</NavLinks></NavItem>
            </>
        )
    }
    const loggedRouter = () => {
        return(
            <>
                <NavItem><NavLinks to ="/history">History</NavLinks></NavItem>
                <NavItem><NavLinks to ="/" onClick={logoutUser}>Logout</NavLinks></NavItem>
            </>
        )
    }


    return (
       <>
       <IconContext.Provider value={{color: '#fff'}}>
        <Nav scrollNav ={scrollNav}>
            <NavbarContainer>
                <NavLogo to='/postLogin'  onClick={reloadPage} ><img src = {logo} alt='p'
height={40} width={40} ></img>{isAdmin ? 'Admin' : 'Pronto - Delivery' }</NavLogo>
                <MobileIcon onClick={toggle}>
                    <FaBars style={{color: "black"}}/>
                </MobileIcon>
                <NavMenu>
                    <NavItem>
                        <NavLinks to="/homedel" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80}>{isAdmin ? 'Products' : 'Shop'}</NavLinks>
                    </NavItem>
                    {isAdmin && adminRouter()}
                    {
                    isLogged ? loggedRouter() : ""
                    }

{
                isAdmin ? ''
                : <NavCart>
                    <Navspans>{cart.length}</Navspans>
                    <NavLinks to='/homedel/cart'>
                        <ShoppingCartIcon htmlColor='white'/>
                    </NavLinks>
                   </NavCart>
                }
```

```
                </NavMenu>

            </NavbarContainer>
        </Nav>
        </IconContext.Provider>
       </>
    );
}

export default Navbar
```

Pages.js

```
import React, { useContext } from 'react'
import {Switch , Route} from 'react-router-dom'
import Products from './products/Products'
import cart from './cart/cart'
import NotFound from '../../utils/NotFound/NotFound'
import DetailProduct from './detailProduct/detailProduct'
import { GlobalState } from '../../GlobalState'
import OrderHistory from './history/OrderHistory'
import OrderDetails from './history/OrderDetails'
import Categories from './categories/Categories'
import CreateProduct from './createProduct/CreateProduct'
import invoice from '../../../invoice'


export default function Pages() {
    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    const [isAdmin] =state.userAPI.isAdmin

    return (
        <div style={{maxWidth: "1230px",
            minHeight: "100vh",
            margin: "0 auto",
            padding: "0 20px",
            boxShadow: "0 0 35px #eee"}} >
        <Switch>
            <Route path = "/homedel"  component={isLogged ?Products : NotFound} exact/>
            <Route path = "/homedel/detail/:id" exact component ={DetailProduct} />
            <Route path = "/homedel/cart"  component={isLogged ? cart : NotFound} exact/>
            <Route path = "/history" component= {isLogged ? OrderHistory: NotFound} exact />
            <Route path = "/history/:id" component= {isLogged ? OrderDetails: NotFound} exact
/>
            <Route path = "/category" component={isAdmin? Categories : NotFound} exact/>
            <Route path = "/create_product" component={isAdmin? CreateProduct : NotFound}
exact/>
```

```
            <Route path = "/homedel/edit_product/:id" component={isAdmin? CreateProduct :
NotFound} exact/>
            <Route path='/invoice' component={invoice} exact />
            <Route path = "*" component={NotFound} exact />
        </Switch>
        </div>
    )
}
```

Footer.js

```
import React, {useState} from 'react'
import {animateScroll as scroll} from 'react-scroll'
import {
    FooterContainer,
    FooterWrap,
    FooterLinksContainer,
    FooterLinksWrapper,
    FooterLinkItems,
    FooterLinkTitle,
    FooterLink,
    FooterLinks,
    SocialMedia,
    SocialMediaWrap,
    SocialLogo,
    WebsiteRights,
    SocialIcons,
    SocialIconLink} from './FooterElements'
import {FaFacebook, FaInstagram, FaTwitter} from 'react-icons/fa'
import {Dialog, DialogTitle, DialogContent } from '@material-ui/core';
import Mail from '@material-ui/icons/Mail'
import Feedback from '../../pages/Feedback';
import CloseIcon from '@mui/icons-material/Close';
import Slide from '@mui/material/Slide';

const Transition = React.forwardRef(function Transition(props, ref) {
    return <Slide direction="up" ref={ref} {...props} />;
  });

const Footer = () => {
    const [openPopup, setOpenPopup] = useState(false)
    const toggleHome = () =>{
        scroll.scrollToTop();
    }
    const togglePopup = () => {
        setOpenPopup(true);
    }
```

```
    return (
        <FooterContainer>
            <FooterWrap>
                <FooterLinksContainer>
                    <FooterLinksWrapper>
                        <FooterLinkItems>
                            <FooterLinkTitle>Home <br/><br/></FooterLinkTitle>
                                <FooterLinks to='about'>About</FooterLinks>
                                <FooterLinks to = "services">Services</FooterLinks>
                                <FooterLink to = "/signin">Sign In</FooterLink>
                        </FooterLinkItems>
                        <FooterLinkItems>
                            <FooterLinkTitle>Contact Us </FooterLinkTitle>
                                <FooterLink onClick={togglePopup}>Contact</FooterLink>
                                <FooterLink onClick={togglePopup}>Support</FooterLink>
                                <FooterLink onClick={togglePopup}>Address</FooterLink>
                        </FooterLinkItems>
                    </FooterLinksWrapper>
                    <FooterLinksWrapper>
                        <FooterLinkItems>
                            <FooterLinkTitle>About Us</FooterLinkTitle>
                                <FooterLink to = "/signin">How it Works</FooterLink>

                                <FooterLink to = "/signin">Terms and Conditions</FooterLink>
                        </FooterLinkItems>
                        <FooterLinkItems>
                        <FooterLinkTitle>Community <br/><br/></FooterLinkTitle>
                                <FooterLink to = "/signin">Announcements</FooterLink>
                                <FooterLink to = "/signin">Discussion board</FooterLink>
                                <FooterLink to = "/signin">Job opportunities</FooterLink>
                        </FooterLinkItems>
                    </FooterLinksWrapper>
                </FooterLinksContainer>
                <SocialMedia>
                    <SocialMediaWrap>
                        <SocialLogo to='/' onClick={toggleHome}>
                            Pronto
                        </SocialLogo>
                        <WebsiteRights>Pronto © {new Date().getFullYear()} All rights reserved.
    </WebsiteRights>
                        <SocialIcons>
                            <SocialIconLink href='/' target="_blank" aria-
    label="Facebook"><FaFacebook /></SocialIconLink>
                            <SocialIconLink href='/' target="_blank" aria-
    label="Instagram"><FaInstagram /></SocialIconLink>
                            <SocialIconLink href='/' target="_blank" aria-
    label="Twitter"><FaTwitter /></SocialIconLink>
                            <SocialIconLink href='/' target="_blank" aria-label="Gmail"><Mail
    /></SocialIconLink>
```

```
                        </SocialIcons>
                    </SocialMediaWrap>
                </SocialMedia>
            </FooterWrap>
            <Dialog open={openPopup} maxWidth="xl" TransitionComponent={Transition} keepMounted>
            <DialogTitle sx={{ m: 0, p: 2 }}><div style={{fontWeight:'bold'}}>User
Feedback</div></DialogTitle>
            <CloseIcon onClick={() => setOpenPopup(false)} sx={{
            position: 'absolute',
            right: 8,
            top: 8,
            cursor: "pointer",
             }}/>
            <DialogContent dividers style={{marginTop: '-110px', overflow:'hidden'}}>
                <Feedback />
            </DialogContent>
            </Dialog>
        </FooterContainer>
    )
}

export default Footer
```

HeroSection.js

```
import React, {useState} from 'react'
import Video from '../../videos/video1.mp4'
import {Button} from '../ButtonElement'
import { HeroContainer, HeroBg, VideoBg, HeroContent, HeroH1, HeroP, HeroBtnWrapper,
ArrowForward,ArrowRight } from './HeroElements'

const HeroSection = () => {
    const[hover, setHover] = useState(false)
    const onHover =() =>{
        setHover(!hover)
    }
    return (
        <HeroContainer id="about">
            <HeroBg>
                <VideoBg autoPlay loop muted src ={Video} type='video/mp4'/>
            </HeroBg>
            <HeroContent>
                <HeroH1>Connecting you and your wishes</HeroH1>
                <HeroP>Save time with the fastest services</HeroP>
                <HeroBtnWrapper>
                    <Button to="signup" onMouseEnter={onHover} onMouseLeave ={onHover }
primary="true" dark="true"smooth ={true} duration={500} spy={true} exact ='true' offset ={-80}>
                        Start now {hover ?<ArrowForward /> :<ArrowRight />}
                    </Button>
```

```
                </HeroBtnWrapper>
            </HeroContent>
        </HeroContainer>
    )
}

export default HeroSection
```

InfoSection.js

```
import React from 'react'
import {
     InfoContainer,
     InfoWrapper,
     InfoRow,
     Column1,
     TextWrapper,
     TopLine,
     Heading,
     Subtitle,
     BtnWrap,
     ImgWrap,
     Img,
     Column2,
     SignBtnLink} from './InfoElements'


const InfoSection = ({
    lightBg,
    id,
    imgStart,
    topLine,
    lightText,
    headLine,
    darkText,
    description,
    buttonLabel,
    img,
    alt,
    primary,
    dark,
    dark2}) => {
    return (
        <>
          <InfoContainer  lightBg ={lightBg} id={id}>
              <InfoWrapper>
                  <InfoRow imgStart = {imgStart}>
                      <Column1>
                        <TextWrapper>
```

```
                            <TopLine>{topLine}</TopLine>
                            <Heading darkText = {darkText}>{headLine}</Heading>
                            <Subtitle darkText = {darkText}>{description}</Subtitle>
                            <BtnWrap>
                                <SignBtnLink to= {'/signin'}
                                smooth ={true}
                                duration={500}
                                spy={true}
                                exact ="true"
                                offset ={-80}
                                primary={primary ? 1 : 0}
                                dark={dark ? 1 : 0}
                                dark2 ={dark2 ? 1 : 0}>{buttonLabel}</SignBtnLink>
                            </BtnWrap>
                        </TextWrapper>
                    </Column1>
                    <Column2>
                        <ImgWrap>
                            <Img  src = {img.default} alt='Signup' />
                        </ImgWrap>
                    </Column2>
                </InfoRow>
            </InfoWrapper>
        </InfoContainer>
    </>
  )
}

export default InfoSection
```

Data.js

```
export const homeobj1 = {
    id: 'signup',
    lightBg: true,
    lightText: false,
    lightTextDesc: true,
    topLine: 'Pronto',
    headLine: 'SignUp',
    description: 'Just one click away to explore a seamless experience',
    buttonLabel: 'Get Started',
    imgStart: false,
    img: require('../../images/signup1.svg'),
    alt: 'Signup',
    dark: false,
    primary: false,
    darkText: true,
}
```

MechanicLogin.js

```
import {useHistory} from 'react-router-dom'
import React, {useState} from "react";
import {
  BoldLink,
  BoxContainer,
  Container,
  FormContainer,
  Input,
  MutedLink,
  SubmitButton,
} from "./common";
import { Marginer } from "../marginer";
import { showErrMsg, showSuccessMsg } from "../utils/notification/notification";
import AuthService from '../CarService/services/member/auth_service';

const initialState = {
  email: '',
  password: '',
  err: '',
  success: ''
}
export function LoginForm(props) {
  const [user,setUser] = useState(initialState)
  const {email, password, err, success} = user
  const history = useHistory();

  const handleChangeInput = e => {
    const {name, value}= e.target
    setUser({...user, [name]:value, err: '', success:''})
  }

  const handleSubmit = async e =>{
      e.preventDefault();
      try {
        AuthService.login(email, password).then((respone) => {
        if(respone.role === "MECHANIC"){
          history.push("/mechanic_home/findOrders");
        }
        else{
          setUser({...user, err: 'Check Your Credentials', success:''})
        }
      });
      } catch (error) {
        setUser({...user, err: 'Check Your Credentials', success:''})
      }

  }
```

```
  const user_signin = e =>{
    history.push('/signin')
  }

  return (
    <BoxContainer>
      <Container onSubmit={handleSubmit}>
      <FormContainer >
        <Input type="email" name='email'  value = {email} onChange={handleChangeInput}
placeholder="Email*" required/>
        <Input type="password" name='password'  value = {password} onChange={handleChangeInput}
placeholder="Password*" required />
      </FormContainer>
      <Marginer direction="vertical" margin={10} />
      <MutedLink href="#">Forget your password?</MutedLink>
      <Marginer direction="vertical" margin="1.6em" />
      <SubmitButton type="submit">Signin</SubmitButton>
      <Marginer direction="vertical" margin="1em" />
      {err && showErrMsg(err)}
      {success && showSuccessMsg(success)}
      <BoldLink onClick={user_signin} style= {{marginLeft: "-1px"}}>
        User Signin
      </BoldLink>
      </Container>
    </BoxContainer>
  );
}
```

Navbar.js

```
import React ,{useState,useEffect}from 'react'
import logo from '../../images/logo192.png';
import { Nav, NavbarContainer, NavLogo, MobileIcon, NavMenu, NavItem,NavLinks, NavBtn,
NavBtnLink } from './NavbarElements';
import {FaBars} from 'react-icons/fa'
import {IconContext} from 'react-icons/lib'
import {animateScroll as scroll } from 'react-scroll';
const Navbar = ({toggle}) => {
    const [scrollNav, setScrollNav] = useState(false)
    const changeNav= () => {
        if(window.scrollY >= 80){
            setScrollNav(true)
        }
        else{
            setScrollNav(false)
        }
```

```
    };

    useEffect(() => {
        window.addEventListener('scroll', changeNav)
  },[]);

  const toggleHome = () =>{
      scroll.scrollToTop();
  }
    return (
       <>
       <IconContext.Provider value={{color: '#fff'}}>
        <Nav scrollNav ={scrollNav}>
            <NavbarContainer>
                <NavLogo to='/' onClick={toggleHome} ><img src = {logo} alt='p' height={40}
width={40} ></img>Pronto</NavLogo>
                <MobileIcon onClick={toggle}>
                    <FaBars />
                </MobileIcon>
                <NavMenu>
                    <NavItem>
                        <NavLinks to="about" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80}>About</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks to="services" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80} activeClass='active'>Services</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks to="signup" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80}>SignUp</NavLinks>
                    </NavItem>
                </NavMenu>
                <NavBtn>
                    <NavBtnLink to="/signin">Sign In</NavBtnLink>
                </NavBtn>
            </NavbarContainer>
        </Nav>
        </IconContext.Provider>
       </>
    );
}

export default Navbar
```

Sidebar.js

```
import React from 'react'
import { SidebarContainer, Icon, CloseIcon, SidebarWrapper,
```

```
SidebarMenu,SidebarLink,SidebarRoute, SideBtnWrap} from './SidebarElements'
const Sidebar = ({isOpen, toggle}) => {
    return (
      <SidebarContainer isOpen ={isOpen} onClick={toggle}>
          <Icon onClick={toggle}>
              <CloseIcon />
          </Icon>
          <SidebarWrapper>
              <SidebarMenu>
                  <SidebarLink to="about" onClick={toggle}>About</SidebarLink>
                  <SidebarLink to="services" onClick={toggle}>Services</SidebarLink>
                  <SidebarLink to="signup" onClick={toggle}>Signup</SidebarLink>
              </SidebarMenu>
              <SideBtnWrap>
                  <SidebarRoute to='/signin'>Sign In</SidebarRoute>
              </SideBtnWrap>
          </SidebarWrapper>
      </SidebarContainer>
    )
}

export default Sidebar
```

ButtonElements.js

```
import styled from 'styled-components'
import {Link} from 'react-scroll'

export const Button = styled(Link)`
    border-radius: 50px;
    text-decoration: none;
    background: ${({primary}) =>(primary ? '#01BF71' : '#010606')};
    white-space: nowrap;
    padding: ${({big}) =>(big ? '14px 48px' : '12px 30px')};
    color: ${({dark}) => (dark ? '#010606' : '#fff')} !important;
    font-size: ${({fontBig}) => (fontBig ? '20px': '16px')};
    outline: none;
    border: none;
    cursor: pointer;
    display: flex;
    justify-content: center;
    align-items: center;
    transition: all 0.2s ease-in-out;

    &:hover {
        transition: all 0.2s ease-in-out;
        background: ${({primary}) => (primary ? '#fff' : '#01BF71')}
    }
`
```

GlobalState.js

```
import axios from 'axios'
import React, {createContext,useState, useEffect} from 'react'
import ProductsAPI from '../api/ProductsAPI'
import UserAPI from '../api/UserAPI'
import CategoriesAPI from '../api/CategoriesAPI'

export const GlobalState = createContext()

const initialstate = {
    tokens: ""
}
export const DataProviders = ({children}) =>{

    const [token, setToken] = useState(false)
    const [tokens, setTokens] = useState(initialstate)



    useEffect(()=>{
        const refreshToken = async () =>{
            const res = await axios.post("/user/refresh_token")
            setTokens(res.data.access_token)
            setToken(res.data.access_token)
            setTimeout(()=>{
                refreshToken()
            },10*60*1000)

        }
        refreshToken()
    },[])



    const state ={
        token: [token, setToken],
        tokens: [tokens, setTokens],
        productsAPI: ProductsAPI(),
        userAPI: UserAPI(tokens, token),
        categoriesAPI: CategoriesAPI(),
    }
    return(
    <GlobalState.Provider value={state}>
        {children}
    </GlobalState.Provider>
```

```
    )
}
```

PostLogin

InfoSection.js

```
import React from 'react'
import {
     InfoContainer,
     InfoWrapper,
     InfoRow,
     Column1,
     TextWrapper,
     TopLine,
     Heading,
     Subtitle,
     BtnWrap,
     ImgWrap,
     Img,
     Column2,
     SignBtnLink} from './InfoElements'


const InfoSection = ({
    lightBg,
    id,
    imgStart,
    topLine,
    lightText,
    headLine,
    darkText,
    description,
    buttonLabel,
    img,
    alt,
    primary,
    dark,
    dark2,
    routesite}) => {
    return (
        <>
          <InfoContainer  lightBg ={lightBg} id={id}>
              <InfoWrapper>
                  <InfoRow imgStart = {imgStart}>
                      <Column1>
                        <TextWrapper>
                            <TopLine>{topLine}</TopLine>
                            <Heading lightText={lightText}>{headLine}</Heading>
```

```
                            <Subtitle darkText = {darkText}>{description}</Subtitle>
                            <BtnWrap>
                                <SignBtnLink to= {routesite}
                                smooth ={true}
                                duration={500}
                                spy={true}
                                exact ="true"
                                offset ={-80}
                                primary={primary ? 1 : 0}
                                dark={dark ? 1 : 0}
                                dark2 ={dark2 ? 1 : 0}>{buttonLabel}</SignBtnLink>
                            </BtnWrap>
                        </TextWrapper>
                    </Column1>
                    <Column2>
                        <ImgWrap>
                            <Img  src = {img.default} alt='Signup' />
                        </ImgWrap>
                    </Column2>
                </InfoRow>
            </InfoWrapper>
        </InfoContainer>
    </>
  )
}

export default InfoSection
```

Data.js

```
export const homeobj1 = {
    id: 'courier',
    lightBg: false,
    lightText: true,
    lightTextDesc: false,
    topLine: 'Car Service',
    headLine: 'Makes Your Job Easier For You',
    description: 'Just one click away to explore a seamless experience',
    buttonLabel: 'Get Started',
    imgStart: false,
    img: require('../../../images/carser2.svg'),
    alt: 'Signup',
    dark: true,
    primary: true,
    darkText:false,
    routesite: '/carService'
}

export const homeobj2 = {
```

```
    id: 'bike-taxi',
    lightBg: true,
    lightText: false,
    lightTextDesc: false,
    topLine: 'Bike-Taxi',
    headLine: '123',
    description: 'Just one click away to explore a seamless experience',
    buttonLabel: 'Get Started',
    imgStart: true,
    img: require('../../../images/newbike.png'),
    alt: 'Signup',
    dark: false,
    primary: false,
    darkText: true,
    routesite: '/biketaxi'

}

export const homeobj3 = {
    id: 'delivery',
    lightBg: false,
    lightText: true,
    lightTextDesc: false,
    topLine: 'Home Delivery',
    headLine: 'You want it we got it',
    description: 'Just one click away to explore a seamless experience',
    buttonLabel: 'Get Started',
    imgStart: false,
    img: require('../../../images/new homedel.svg'),
    alt: 'Signup',
    dark: true,
    primary: true,
    darkText: false,
    routesite: '/homedel'

}
```

Navbar.js

```
import React ,{useState,useEffect, useContext}from 'react'
import logo from '../../../images/logo192.png';
import { Nav, NavbarContainer, NavLogo, MobileIcon, NavMenu, NavItem,NavLinks, NavBtn } from
'./NavbarElements';
import {FaBars} from 'react-icons/fa'
import {IconContext} from 'react-icons/lib'
import {animateScroll as scroll } from 'react-scroll';
import {ListItemIcon, MenuItem, Dialog, DialogTitle, DialogContent } from '@material-ui/core';
import Menu from '@material-ui/core/Menu'
```

```
import { Avatar } from '@material-ui/core';
import Logout from '@mui/icons-material/Logout'
import { Link } from 'react-router-dom';
import axios from 'axios';
import { GlobalState } from '../../GlobalState';
import Feedback from '../../../pages/Feedback'
import CloseIcon from '@mui/icons-material/Close';
import RateReviewIcon from '@mui/icons-material/RateReview';
import Slide from '@mui/material/Slide';

const Transition = React.forwardRef(function Transition(props, ref) {
    return <Slide direction="up" ref={ref} {...props} />;
  });

const Navbar = ({toggle}) => {
    const state = useContext(GlobalState)
    const [isAdmin] = state.userAPI.isAdmin
    const [scrollNav, setScrollNav] = useState(false)
    const [anchorEl, setAcnchorEl] = useState(null);
    const [openPopup, setOpenPopup] = useState(false)


    const userLogout = async() =>{
        await axios.get('/user/logout')
        localStorage.clear()
    }

    const changeNav= () => {
        if(window.scrollY >= 80){
            setScrollNav(true)
        }
        else{
            setScrollNav(false)
        }
    };

    const handleClose = () =>{
        setAcnchorEl(null);
    };

    const handleClick = (event) => {
        setAcnchorEl(event.currentTarget);
    };

    useEffect(() => {
        window.addEventListener('scroll', changeNav)
        window.scrollTo(0, 0)
  },[]);
```

```
  const toggleHome = () =>{
      scroll.scrollToTop();
  }
  const togglePopup = () => {
      setOpenPopup(true);
  }
    return (
       <>
       <IconContext.Provider value={{color: '#fff'}}>
        <Nav scrollNav ={scrollNav}>
            <NavbarContainer>
                <NavLogo onClick={toggleHome} ><img src = {logo} alt='p' height={40} width={40}
></img>{isAdmin ? 'Pronto-Admin' : 'Pronto' }</NavLogo>
                <MobileIcon onClick={toggle}>
                    <FaBars />
                </MobileIcon>
                <NavMenu>
                    <NavItem>
                        <NavLinks to="courier" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80}>Car-Service</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks to="bike-taxi" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80} activeClass='active'>Bike-Taxi</NavLinks>
                    </NavItem>
                    <NavItem>
                        <NavLinks to="delivery" smooth ={true} duration={500} spy={true} exact
='true' offset ={-80}>Delivery</NavLinks>
                    </NavItem>
                </NavMenu>
                <NavBtn onClick ={handleClick}>
                    <Avatar sx={{ width: 32, height: 32}}></Avatar>
                 </NavBtn>
                <Menu keepMounter
                anchorEl= {anchorEl}
                onClose={handleClose}
                onClick={handleClose}
                open= {Boolean(anchorEl)}
                PaperProps={{
                    elevation: 0,
                    sx: {
                      overflow: 'visible',
                      filter: 'drop-shadow(0px 2px 8px rgba(0,0,0,0.32))',
                      mt: 1.5,
                      '& .MuiAvatar-root': {
                        width: 32,
                        height: 32,
                        ml: 3,
                        mr: 1,
```

```
                    },
                    '&:before': {
                      content: '""',
                      display: 'block',
                      position: 'absolute',
                      top: 0,
                      right: 14,
                      width: 10,
                      height: 10,
                      bgcolor: 'black',
                      transform: 'translateY(-50%) rotate(45deg)',
                      zIndex: 0,
                    },
                  },
                }}
                transformOrigin={{ horizontal: 'right', vertical: 'top' }}
                anchorOrigin={{ horizontal: 'right', vertical: 'bottom' }}
                >
                    {isAdmin?
                      null :
                      <MenuItem onClick={togglePopup}>
                      <ListItemIcon>
                          <RateReviewIcon fontSize='medium'/>
                      </ListItemIcon>
                      Feedback
                      </MenuItem>
                     }

                    <MenuItem component={Link} to="/" onClick={userLogout}>
                    <ListItemIcon>
                        <Logout fontSize="meduim" />
                    </ListItemIcon>
                    Logout
                    </MenuItem>
                </Menu>
            </NavbarContainer>

        </Nav>
        </IconContext.Provider>
        <Dialog open={openPopup} maxWidth="xl" TransitionComponent={Transition} keepMounted >
            <DialogTitle sx={{ m: 0, p: 2 }}><div style={{fontWeight:'bold'}}>User
Feedback</div></DialogTitle>
            <CloseIcon onClick={() => setOpenPopup(false)} sx={{
            position: 'absolute',
            right: 8,
            top: 8,
            cursor: "pointer",
          }}/>
            <DialogContent dividers style={{marginTop: '-110px', overflow:'hidden'}}>
```

```
                <Feedback />
            </DialogContent>
        </Dialog>
       </>
    );
}

export default Navbar
```

Sidebar.js

```
import React ,{useState}from 'react'
import { SidebarContainer, Icon, CloseIcon, SidebarWrapper, SidebarMenu,SidebarLink,
SideBtnWrap} from './SidebarElements'
import {ListItemIcon, MenuItem } from '@material-ui/core';
import Menu from '@material-ui/core/Menu'
import { Avatar } from '@material-ui/core';
import Logout from '@mui/icons-material/Logout'
import { Link } from 'react-router-dom';

const Sidebar = ({isOpen, toggle}) => {
    const [anchorEl, setAcnchorEl] = useState(null);
    const handleClose = () =>{
        setAcnchorEl(null);
    };

    const handleClick = (event) => {
        setAcnchorEl(event.currentTarget);
    };
    return (
      <SidebarContainer isOpen ={isOpen} onClick={toggle}>
          <Icon onClick={toggle}>
              <CloseIcon />
          </Icon>
          <SideBtnWrap onClick= {handleClick}>
              <Avatar sx={{ width: 32, height: 32 }}></Avatar>
              </SideBtnWrap>
              <Menu keepMounter
                anchorEl= {anchorEl}
                onClose={handleClose}
                onClick={handleClose}
                open= {Boolean(anchorEl)}
                PaperProps={{
                    elevation: 0,
                    sx: {
                      overflow: 'visible',
                      filter: 'drop-shadow(0px 2px 8px rgba(0,0,0,0.32))',
                      mt: 1.5,
                      '& .MuiAvatar-root': {
```

```
                    width: 5,
                    height: 5,
                    ml: -0.5,
                    mr: 1,
                  },
                  '&:before': {
                    content: '""',
                    display: 'block',
                    position: 'absolute',
                    dense: false,
                    top: 0,
                    right: 14,
                    width: 15,
                    height: 15,
                    bgcolor: 'background.paper',
                    transform: 'translateY(-50%) rotate(45deg)',
                    zIndex: 0,
                  },
                },
              }}
              transformOrigin={{ horizontal: 'right', vertical: 'top' }}
              anchorOrigin={{ horizontal: 'right', vertical: 'bottom' }}
              >
                <MenuItem component={Link} to="/">
                <ListItemIcon>
                    <Logout fontSize="large" />
                </ListItemIcon>
                Logout
                </MenuItem>
            </Menu>
        <SidebarWrapper>
            <SidebarMenu>
                <SidebarLink to="courier" onClick={toggle}>Car-Services</SidebarLink>
                <SidebarLink to="bike-taxi" onClick={toggle}>Bike-Taxi</SidebarLink>
                <SidebarLink to="delivery" onClick={toggle}>Home Delivery</SidebarLink>
            </SidebarMenu>
        </SidebarWrapper>
      </SidebarContainer>
    )
}

export default Sidebar
```

App.js

```
import React from 'react';
import './App.css';
import {BrowserRouter as Router, Switch , Route, withRouter} from 'react-router-dom';
```

```
import Home from './pages';
import SigninPage from './pages/signin';
import DataProvider from './redux/store';
import ActivationEmail from './components/accountBox/ActivationEmail';
import PostLogin from './pages/postLogin';
import UnderConstruct from './components/UnderConstruct';
import Ecommerce from './pages/eCommerce';
import BikeTaxi from './pages/bikeTaxi';
import CarService from './pages/carService'
import MechanicSignin from './pages/mechanicSignin';
import MechanicHome from './components/CarService/components/Member/Mechanic/MechanicHome';
import FindOrders from './components/CarService/components/Member/Mechanic/FindOrders';
import MyOrders from './components/CarService/components/Member/Mechanic/MyOrders';
import Feedback from './pages/Feedback';

function App() {

  return (
    <div className='App'>
    <Router>

      {/* homepage and login code */}
      <Switch>

        <Route path = "/" component={Home} exact />
        <Route path = "/Feedback" component={Feedback} exact />
        <DataProvider>
          <Route path = "/signin" component={SigninPage} exact />
          <Route path = "/mechanic_signin" component={MechanicSignin} exact />
          <Route path = "/mechanic_home" component = {MechanicHome} exact />
          <Route path="/mechanic_home/findOrders" component={FindOrders} />
          <Route path="/mechanic_home/myorders" component={MyOrders} />
          <Route path ="/user/activate/:activation_token" component={ActivationEmail} exact />
          <Route path ="/postLogin" component={withRouter(PostLogin)} exact />
          <Route path = "/UnderConstruct" component={UnderConstruct} exact />
        </DataProvider>
        </Switch>

      {/* Home Delivery */}
      <Route path='/homedel'>
        <Ecommerce />
      </Route>
      {/* Bike Taxi */}
      <Route path='/biketaxi'>
        <BikeTaxi />
      </Route>
      {/* Car Service */}
      <Route path='/carService' component={CarService} exact/>
    </Router>
```

```
    </div>
  );
}

export default App;
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';
import reportWebVitals from './reportWebVitals';
import "./index.css"

ReactDOM.render(
  <React.StrictMode>
      <App />
  </React.StrictMode>,
  document.getElementById('root')
);
reportWebVitals();
```

COCOMO:

Cocomo (Constructive Cost Model) is a regression model based on LOC i.e. number of Lines of Code. It is a procedural cost estimate model for software projects and often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time and quality. It was proposed by Barry Boehm in 1970 and is based on the study of 63 projects, which make it one of the best-documented models.

Different models of Cocomo have been proposed to predict the cost estimation at different levels, based on the amount of accuracy and correctness required. All of these models can be applied to a variety of projects, whose characteristics determine the value of constant to be used in subsequent calculations.

- **Organic** – A software project is said to be an organic type if the team size required is adequately small, the problem is well understood and has been solved in the past and also the team members have a nominal experience regarding the problem.

- **Semi-detached** – A development project can be treated with semidetached type if the development consists of a mixture of experienced and inexperienced staff. Team members may

have finite experience in related systems but may be unfamiliar with some aspects of the order being developed.

- **Embedded** – A software project with requiring the highest level of complexity, creativity, and experience requirement fall under this category. Such software requires a larger team size than the other two models and also the developers need to be sufficiently experienced and creative to develop such complex models.

The formula used for the cost estimation for the COCOMO model is given below:

Effort= a(KLOC)^b (KLOC=lines of code(thousands))

Duration = c(Effort)^d

Staffing = Effort/ Duration

Mode	**"a" variable**	**"b" variable**	**"c" variable**	**"d" variable**	**KLOC**
Semi-Detached	3	1.12	2.5	0.35	11.685
Effort (person-months)		**Duration (months)**		**Staffing**	
47.083		9.626		4.891	

5.3 Testing Approaches

Testing occurs after the coding process, when the real code is compared to the code that has been executed. It is used to locate problems and warnings, which helps in the development of a user-friendly application with the fewest flaws possible by correcting any errors and warnings discovered during debugging. If the testing is completed successfully, the software will be free of all errors. There are two main types of testing - Blackbox testing and Whitebox testing.

Blackbox testing is a software testing method used for validation in which the internal structure/design/implementation of the item being tested is not known to the tester. Whitebox testing is a software testing method used for verification in which the internal structure/ design/implementation of the item being tested is known to the tester. It is used in case of unit and integration testing.

Registration

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
1	Enter Valid email, phone number and password	1.Enter Valid email. 2.Enter Valid Password. 3.Enter valid Phone Number	email: shivamspamacc@gmail.com Password: user123 Phone Number:9882233445	Success Message should be displayed saying "Register Success! Please activate your email to start."	No message was displayed.	Fail
2	Enter invalid email	1.Enter Invalid email.	email: shivamspam.com Password: user123	Error Message should be displayed saying "Invalid Email"	No message was displayed.	Fail
3	Enter Password That is less than 6 characters	2.Enter password less than 6 character.	email: shivamspamacc@gmail.com Password: User12	Error Message should be displayed saying "Password must be at least 6 characters"	No message was displayed.	Fail
4	Enter invalid phone number	1.Enter invalid phone number	Phone Number: 988223344	Error Message should be displayed saying "Enter a Valid number"	No message was displayed.	Fail
5	Enter already existing email id	1.Enter a email id that is already registered	Email: siestyit@gmail.com	Error Message should be displayed saying "This email already exists"	No message was displayed.	Fail
6	Keep any of the fields empty	1.Keep any of the fields empty	Full Name: User Phone Number: Keep empty Email: user@gmail.com Password: User@123 Confirm Password: User@123	Error Message should be displayed saying "Please fill in all fields"	No message was displayed.	Fail

Login

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
1	Enter Valid email and valid password	1.Enter Valid email. 2.Enter Valid Password. 3.Login	email: shivamspamacc@gmail.com Password: user123	Redirected to postLogin Page	Redirected to postLogin Page.	Pass
2	Enter invalid email And valid password	1.Enter Invalid email. 2.Enter Valid password	email: shivamspamac@gmail.com Password: user123	Error Message is displayed saying “This email does not exist”	No message was displayed.	Fail
3	Enter valid email and invalid password	1.Enter Valid email. 2.Enter Invalid password.	email: shivamspamacc@gmail.com Password: user12	Error Message is displayed saying “Password is incorrect”	No message was displayed.	Fail

5.4 Modifications and Improvements:

During the testing phase, various defects and mistakes were discovered, disrupting the system's usual operation. As we encountered challenges with validation, we made a series of improvements to the signing and registration page. A suitable modification to the printable bill template was made to reflect more order details. The ability to filter bikes based on booked slots has been improved, providing us with more precise data.

The following modifications should be made to enhance the project's functionality and performance.

5.5 Test case

Register

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
1	Enter Valid email, phone number and password	1.Enter Valid email. 2.Enter Valid Password. 3.Enter valid Phone Number	email: shivamspamacc@gmail.com Password: user123 Phone Number:9882233445	Success Message should be displayed saying "Register Success! Please activate your email to start."	Success Message saying "Register Success! Please activate your email to start" displayed	Pass
2	Enter invalid email	1.Enter Invalid email.	email: shivamspam.com Password: user123	Error Message should be displayed saying "Invalid Email"	Error Message saying "Invalid Email" displayed.	Pass
3	Enter Password That is less than 6 characters	2.Enter password less than 6 character.	email: shivamspamacc@gmail.com Password: User12	Error Message should be displayed saying "Password must be at least 6 characters"	Error message saying "Password must be at least 6 characters" displayed	Pass
4	Enter invalid phone number	1.Enter invalid phone number	Phone Number: 988223344	Error Message should be displayed saying "Enter a Valid number"	Error message saying "Enter a valid number" displayed	Pass
5	Enter already existing email id	1.Enter a email id that is already registered	Email: siestyit@gmail.com	Error Message should be displayed saying "This email already exists"	Error message saying "This email already exists" displayed	Pass
6	Keep any of the fields empty	1.Keep any of the fields empty	Full Name: User Phone Number: Keep empty Email: user@gmail.com Password: User@123 Confirm Password: User@123	Error Message should be displayed saying "Please fill in all fields"	Error message saying "Please fill in all fields" displayed	Pass

Login

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
1	Enter Valid email and valid password	1.Enter Valid email. 2.Enter Valid Password. 3.Login	email: shivamspamacc@gmail.com Password: user123	Redirected to postLogin Page	Redirected to postLogin Page.	Pass
2	Enter invalid email And valid password	1.Enter Invalid email. 2.Enter Valid password	email: shivamspamac@gmail.com Password: user123	Error Message is displayed saying "This email does not exist"	Error message saying "This email does not exist" is displayed	Pass
3	Enter valid email and invalid password	1.Enter Valid email. 2.Enter Invalid password.	email: shivamspamacc@gmail.com Password: user12	Error Message is displayed saying "Password is incorrect"	Error message saying "Password is incorrect" is displayed	Pass

CarService – User View

Test Case Id	Test Case Description	Test Steps	Test Data / User Inputs	Expected Result	Actual Result	Status
1	Select a brand	1.Select a brand	Kia brand selected	Redirected to available cars of the band	Redirected to available cars of the brand	Pass
2	Select a Car	1.Select a car	Seltos car is selected	Redirected to available services	Redirected to available services	Pass
3	Click on Change car button	1.Click on change car button	Change car button clicked	Redirected to carservice homepage	Redirected to carservice homepage	Pass
4	Buy any service	1. Click on the buy button present on the service card	Buy button clicked	Redirected to order summary page	Redirected to order summary page	Pass

5	Place order without entering required values	1. Click on place order button without entering anything in the textboxes	Vehicle Number: Empty Address: Empty	Error message should be displayed saying "Number is Required or Address is Required"	Error message saying "Number is Required or Address is Required" is displayed	Pass
6	Place order	1. Enter vehicle Number 2.Enter Address 3.Click on Place order button	Vehicle Number: MH 05 CV 1900 Address : Kalyan	Success message should be displayed saying "Order Placed Successfully"	Success message saying "Order Placed Successfully" is displayed	Pass
7	Bookings Section	1. From the navbar click on Bookings	Click on Booking	Redirected to myBookings page	Redirected to myBookings page	Pass
8	Working section	1. From the navbar click on Working	Click on Working	Redirected to working Page	Redirected to working page	Pass
9	Logout	1.From navbar click on Logout	Click on Logout	User Logged out and redirected to landing page	User logged out and redirected to landing page	Pass
10	Going Back to post Login page	1.From the navbar click on the pronto icon	Click on pronto icon	Redirected to postLogin Page	Redirected to postLogin Page	Pass
11	Home	1. From the navbar click on Home	Click on Home	Redirected to carservice Home page	Redirected to carService Homepage	Pass

Home Delivery – User View

Test Case Id	Test Case Description	Test Steps	Test Data/User Inputs	Expected Result	Actual Result	Status
1	Filter Product	1.From the drop-down option of Filter select	Select Grocery from the dropdown	Items in category of Grocery should be displayed	Items in the category of Grocery is displayed	Pass

		Grocery				
2	Sorting Products	1.From the drop-down option of sort select Price: Low-High	Select Low-High from the dropdown	Items are sorted in low to high price	Items are sorted in low to high price	Pass
3	View Products	1. Click on the View button on the Items Card	Click on View Button	Redirected to Product Detail page	Redirected to Product Detail page	Pass
4	Buy Product	1.Click on the Buy button on the Item card or from the Product Detail Page	Buy Button clicked	If user click on Buy from Product Details page then Product is added to cart and redirected to user cart else the product is added to cart	If user click on Buy from Product Details page then Product is added to cart and redirected to user cart else the product is added to cart	Pass
5	Go to Cart	1.From the navbar click on the cart Icon	Click on the cart icon	Redirected to Cart page	Redirected to Cart page	Pass
6	Remove Product from cart	1. Click on the **X** icon on the card	**X** icon is clicked	Item Removed from the cart	Item Removed from the cart	Pass
7	Checkout	1.Click on the Paypal Checkout button on the Cart Page	Paypal Checkout button is clicked	Redirected to Paypal api and on completion of transaction the cart is emptied, and order is placed successfully and reflected in History Section	Redirected to Paypal api and on completion of transaction the cart is emptied, and order is placed successfully and reflected in History Section	Pass
8	History	1. From the navbar click on History	Click on History	Redirected to History Page	Redirected to History page	Pass
9	Order Detail	1. Navigate to History section using	Click on view Button	Redirected to Order Details Page	Redirected to Order Details Page	Pass

		Navbar 2.Click on the view button in for any order				
10	Download Bill	1. Navigate to History section using Navbar 2.Click on the view button in 3.Click on Download bill button	Click on download bill button	User gets the option to either download or view the bill.	User gets the option to either download or view the bill	Pass
11	Logout	1.From the navbar click on Logout	Click on Logout	User Logged out and redirected to landing page	User Logged out and redirected to landing page	Pass
12	Shop	1. From the navbar click on Shop	Click on Shop	Redirected to HomeDelivery Home page	Redirected to HomeDelivery Homepage	Pass

BikeTaxi – Admin View

Test Case Id	Test Case Description	Admin Input	Expected Result	Actual Result	Status
1	Add Bike	Admin adds a new bike Bike name: Ather 450x Image url: https://bd.gaadicdn.com/ather450x.jpg Rent per Hour: 400 Capacity: 2 Fuel Type: Electric	Success message should be displayed, and user is redirected to homepage where the added bike should be reflected	Success message is displayed, and the user is redirected to homepage where the bike added is seen	Pass
2	Edit bike	Admin changes the bike details	Updated values should be reflected	Updated values reflected	Pass
3	Delete Bike	Admin deletes a bike	Bike should be deleted and same should be seen on the	Bike is deleted and same is seen on the homepage	Pass

			homepage		
4	Booking History	Admin can see all the bookings done by the user	All booking details are shown	All booking details are shown	Pass
5	Logout	Click on Logout	User Logged out and redirected to landing page	User logged out and redirected to landing page	Pass

Chapter 6: Results and Discussions

6.1 Test Reports

Test cases were created to validate the system as well as its functioning. Different modules were examined, and each module has its own set of test cases. Each module was subjected to a different set of test scenarios. The number of test cases, on the other hand, might be raised to test relatively simple and non-essential criteria.

Some of the modules are:

1. Login/Registration Module
2. Post Login Module
3. Car Service Module
4. Bike Taxi Module
5. Delivery Module
6. Mechanic Module
7. Feedback Module

It is evident from the results that there were flaws in the project. Because the validation in the Login and Registration Modules was not operating as intended, the code was improved to produce the expected results. Due to the implementation of optimizations, it is possible to be confident that the project is now accurate and in proper working condition.

6.2 User Documentation

This website can be used to book car services, bike taxis, and to place orders for products to be delivered right to your door. Cars, bikes, services, mechanics, products, and orders can all be managed by the admins. Mechanics can see orders that have been assigned to them as well as orders that have been completed. For a better business-to-consumer interaction, a feedback system is also in place.

The functions provided by this site are:

- **Landing Page**

When a user visits the website, the first screen they see is this one. This page is divided into three sections: About, Services, and Sign Up. A visual content is visible in the about section, along with the website's tagline. In the services part, a user can get a quick overview of all three services, and in the Sign-Up section, after clicking the button, the user will be redirected to the login page. A common navigation bar, as well as the website logo and sign in button, is visible to the user.

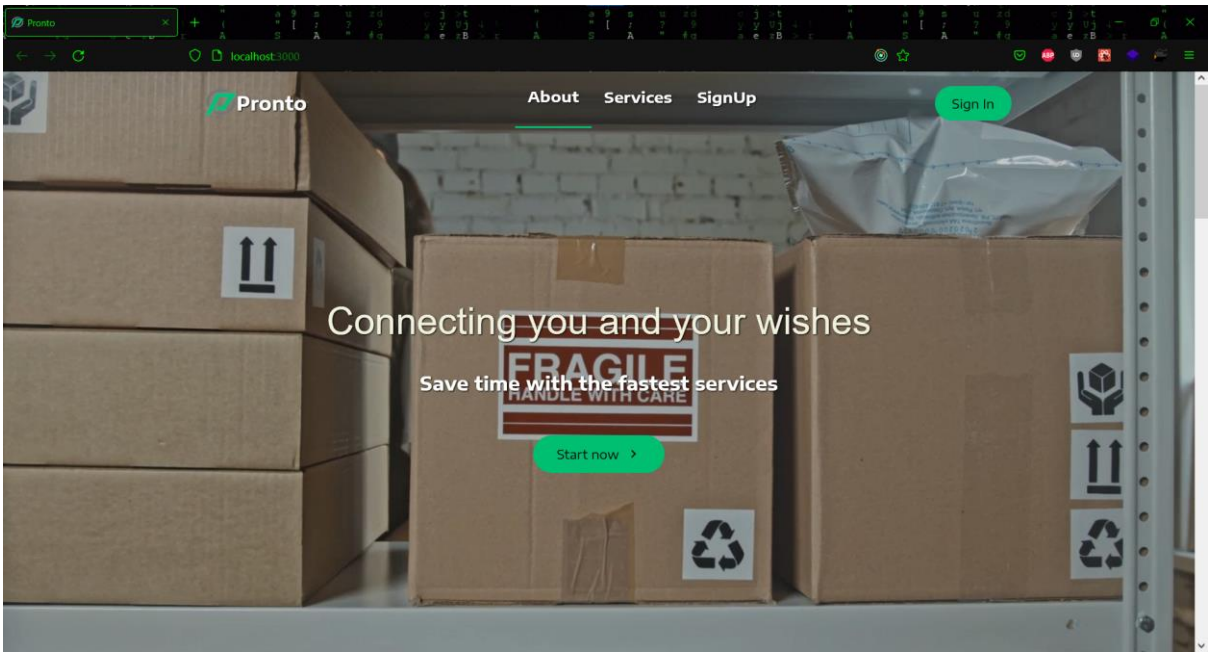


Figure 6.1 About Page

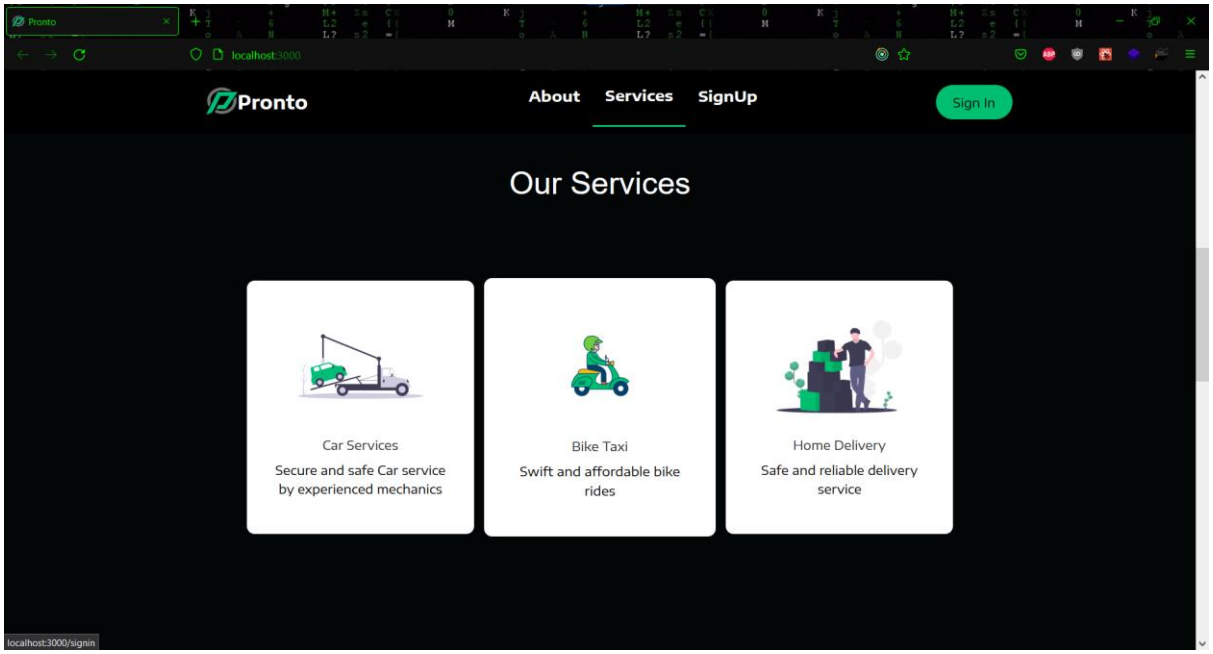


Figure 6.2 Service Section

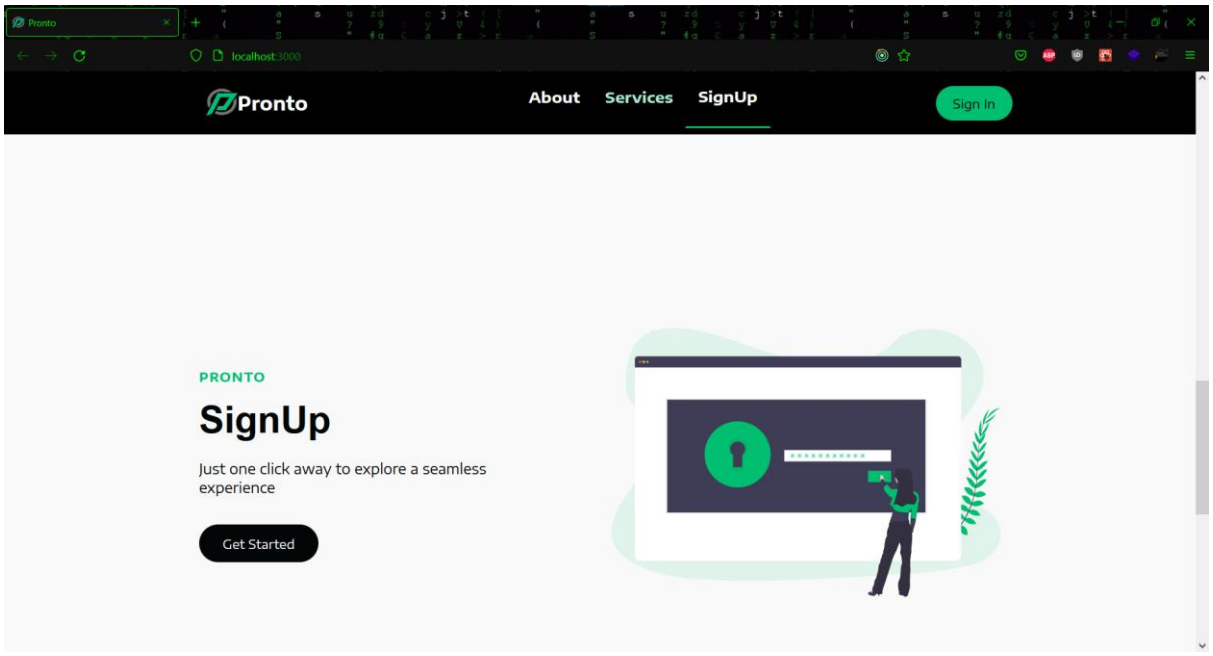


Figure 6.3 Sign Up Section

- **Login/Registration page**

This is the Sign Up and Sign In part, where a user can fill out his or her information and register for the service. Admins and users can both log in on the same page, but mechanics must use a different login page.

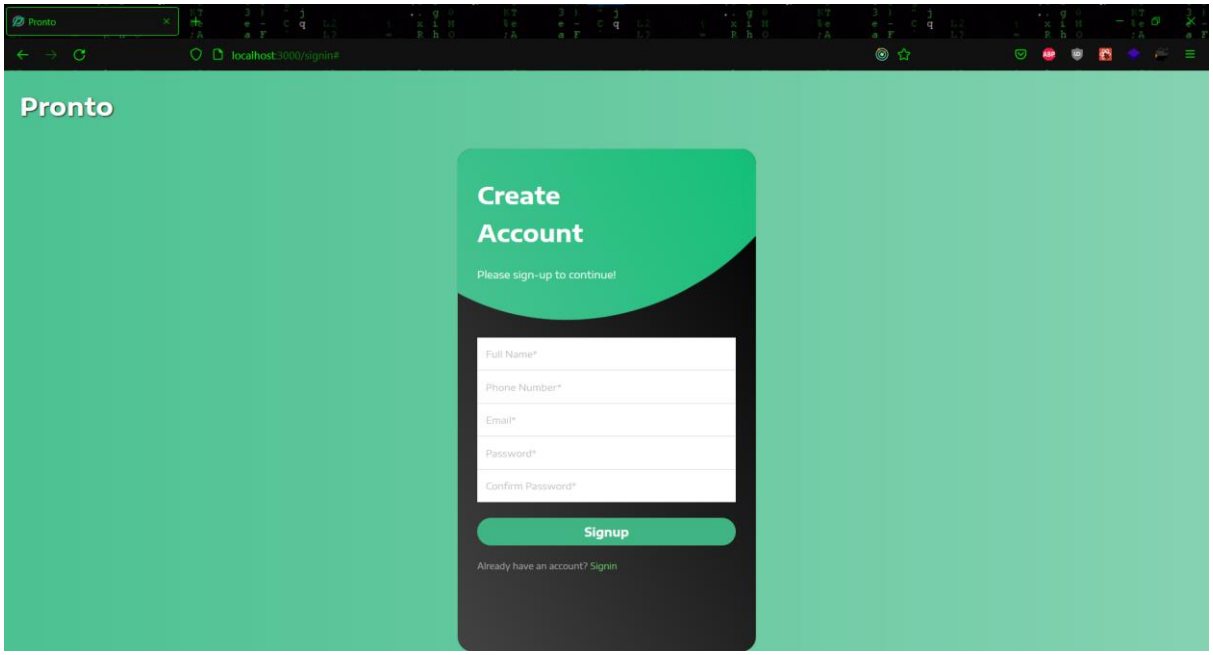


Figure 6.4 Sign up page

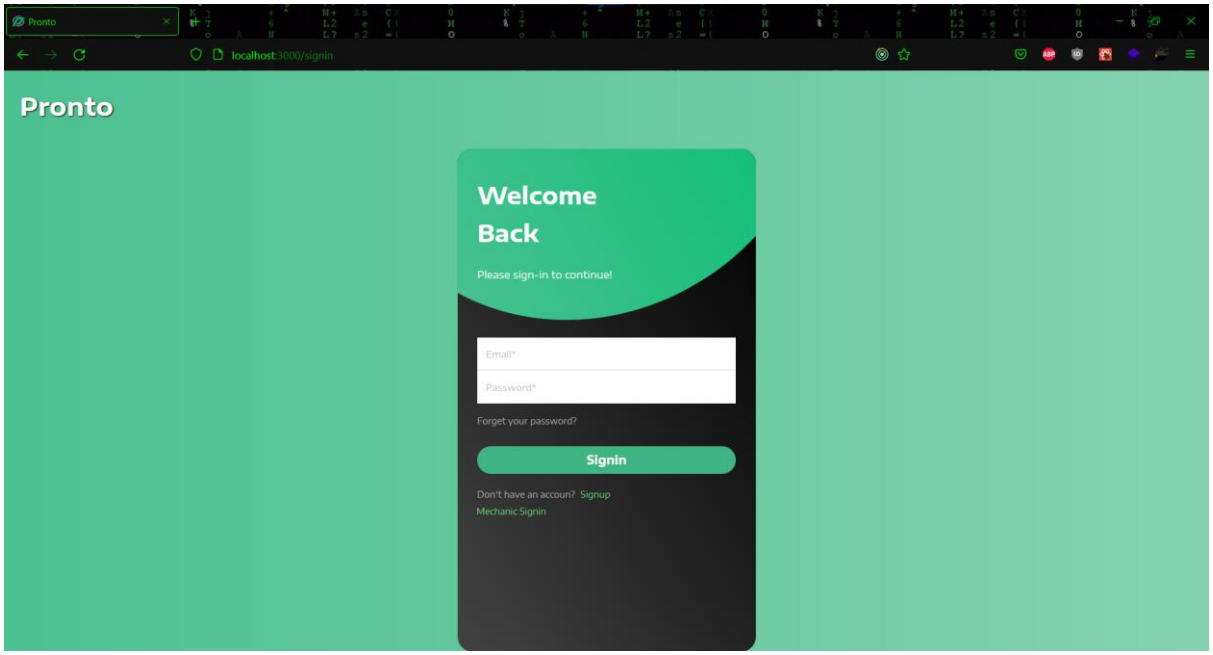


Figure 6.5 Sign in page for User and Admin

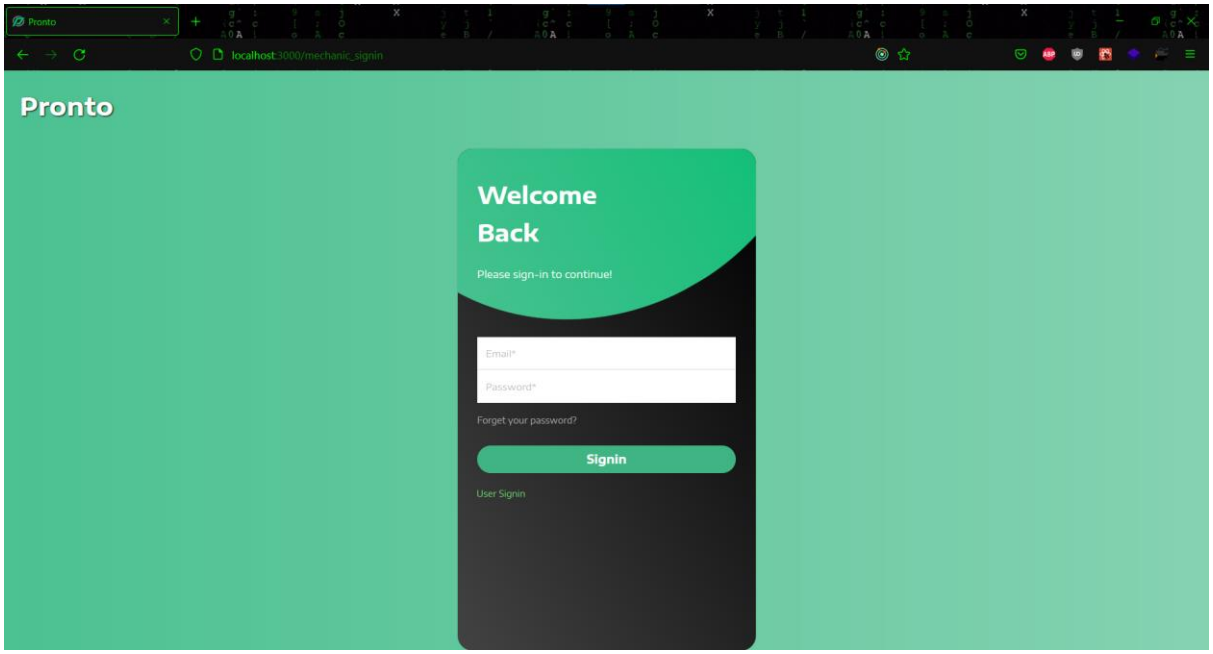


Figure 6.6 Sign in page for Mechanic

- **Post Login**

This is the website's post-login page, to which the user is forwarded after logging in. A user can get a lot more in-depth look at all three services here. By simply pressing the button, they can access any of the required services. They will be taken to the service page after clicking the button.

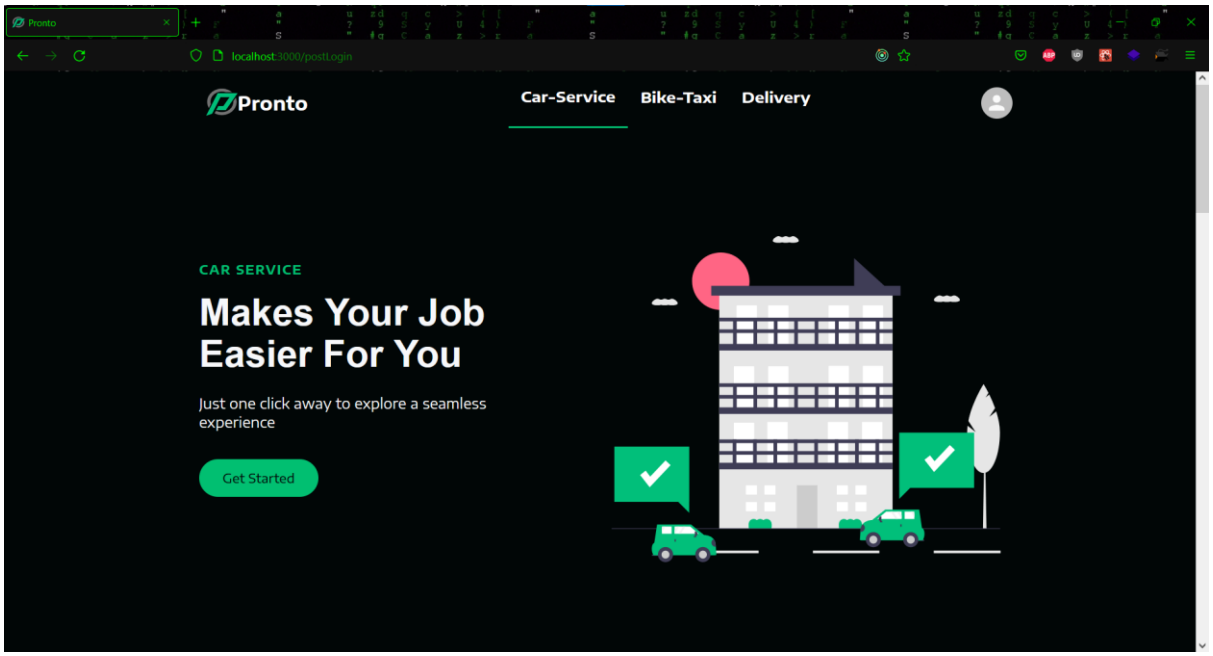


Figure 6.7 Car Service section

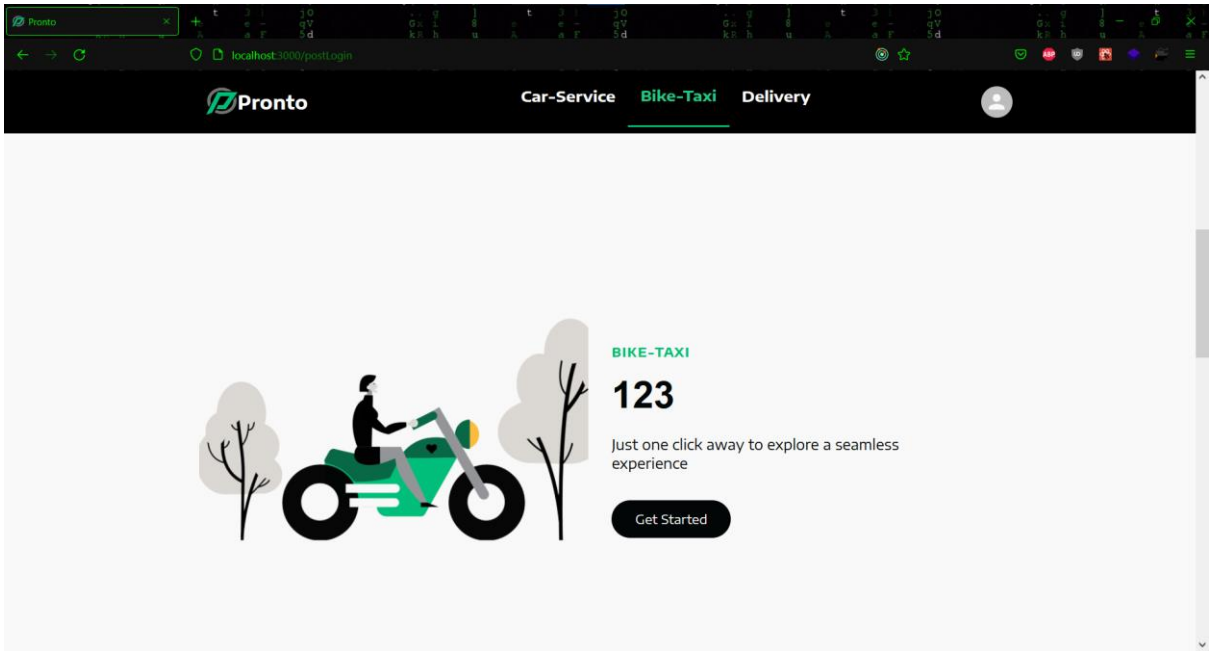


Figure 6.8 Bike Taxi Section

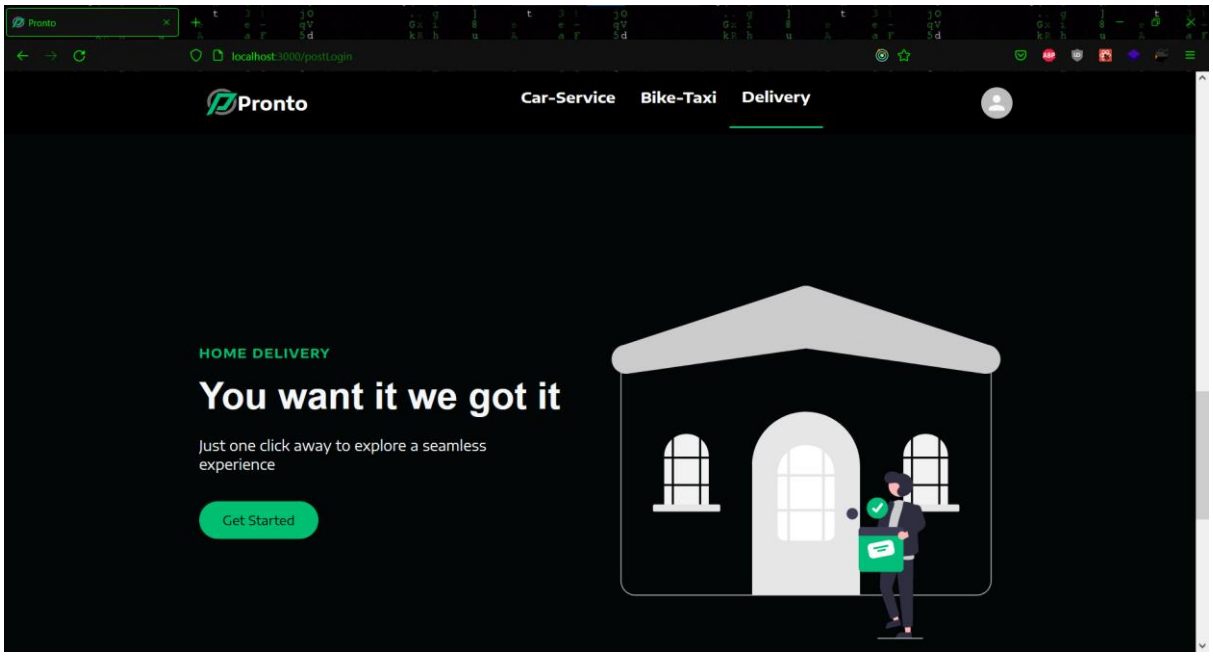


Figure 6.9 Delivery Service

- **User View**

In the user view a user can access all the services after landing at the post login page. Some user views on the services are shown below.

- **Car Service**

This is the car service's user homepage. A user can view a carousel explaining various car services here. A user can search for the brand of car he or she has and then choose a model from that list. After confirming the car model, the user can select the desired service. After you've chosen a service, you'll see an order summary containing all of the order's details. In the booking session, the user can check the status of the booking.

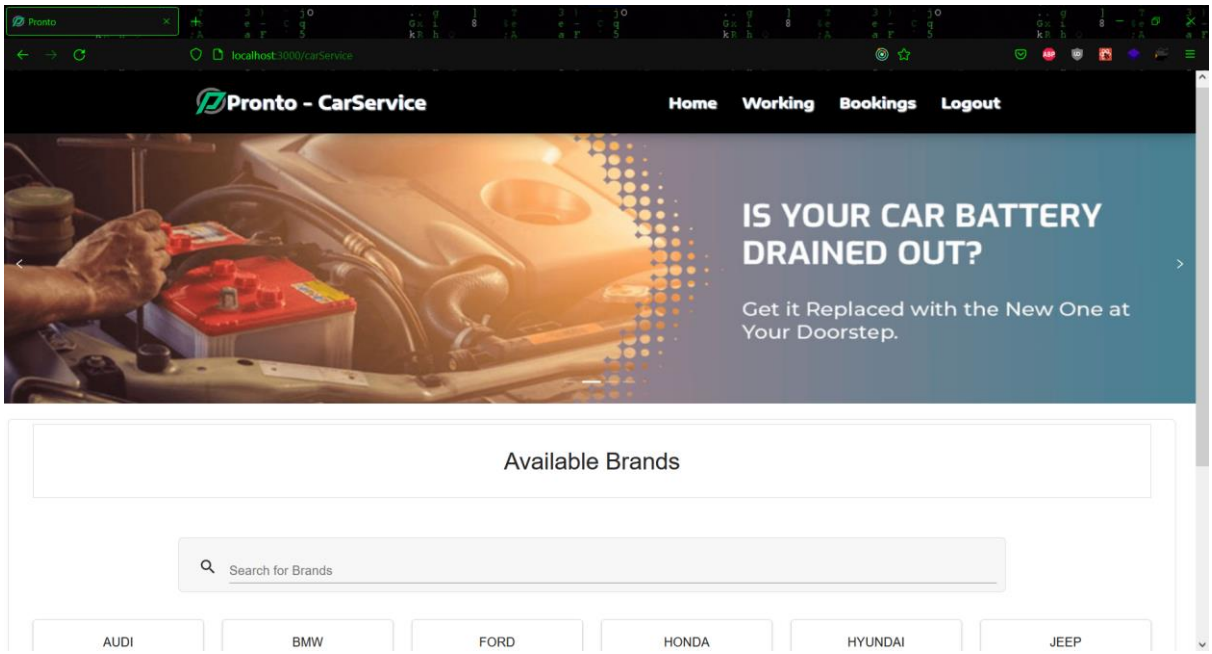


Figure 6.10 Car service home page

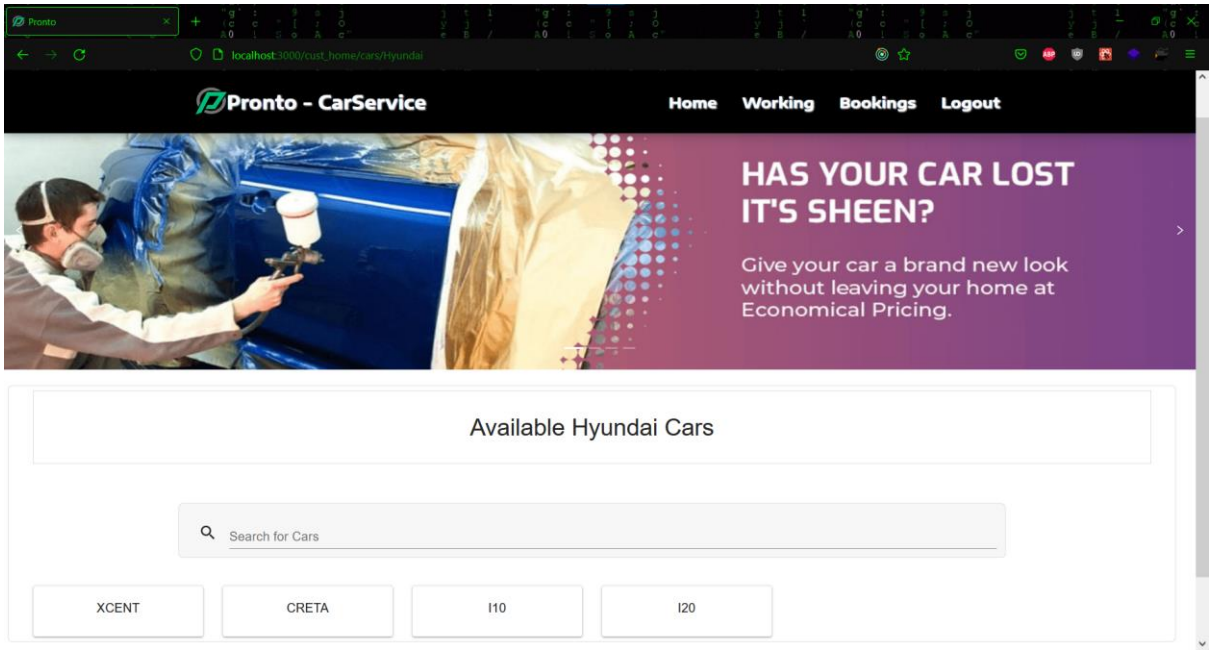


Figure 6.11 Car models

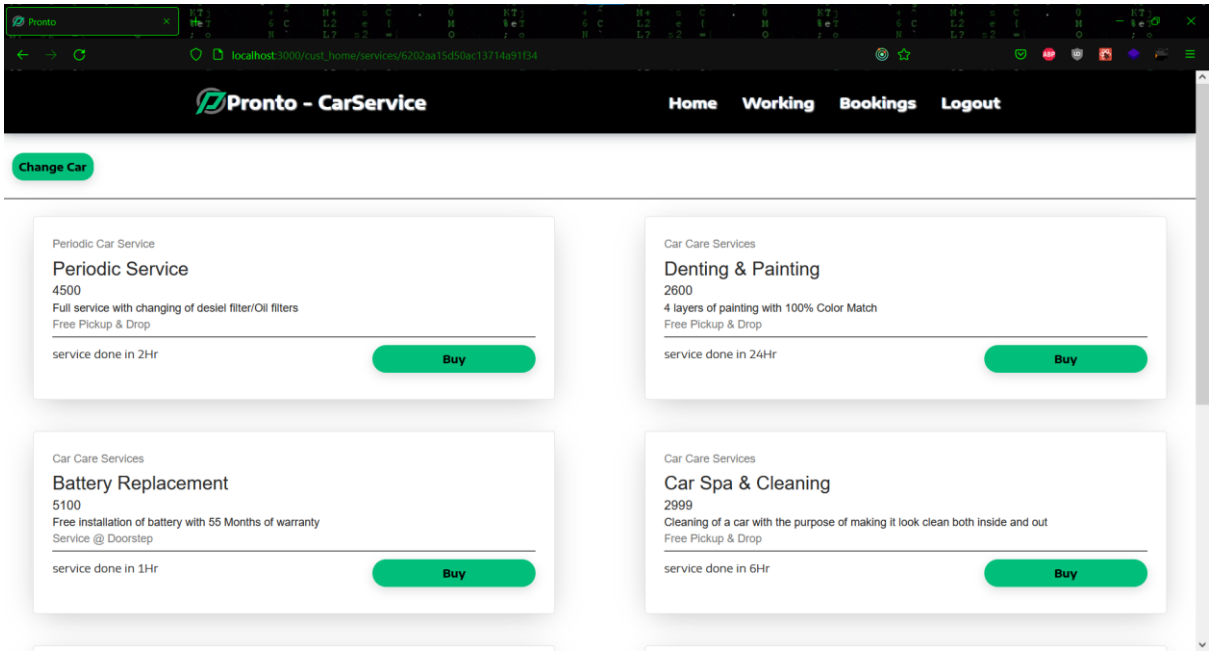


Figure 6.12 Car services

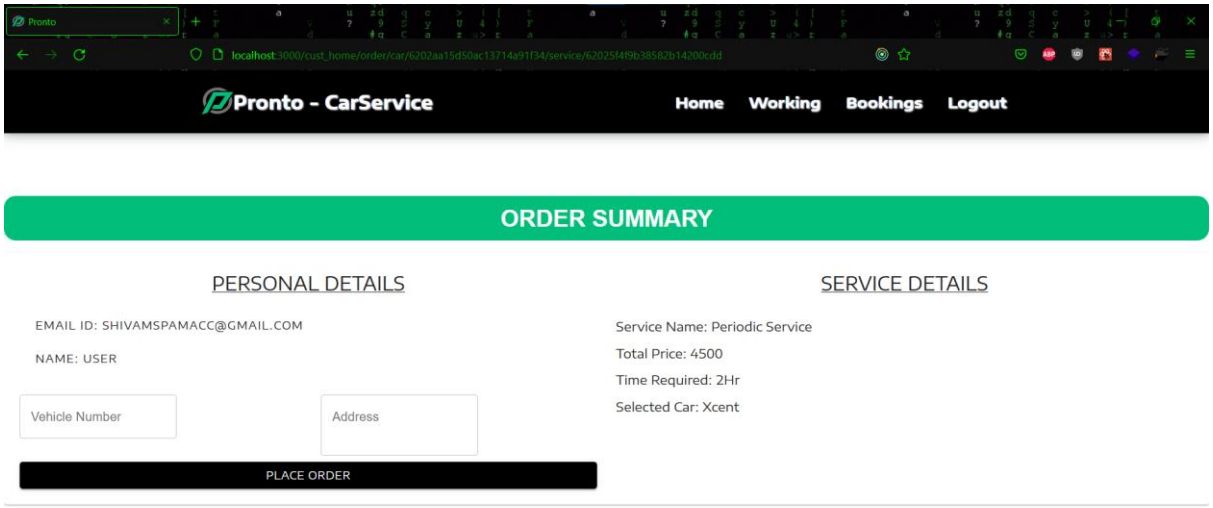


Figure 6.13 Order Summary

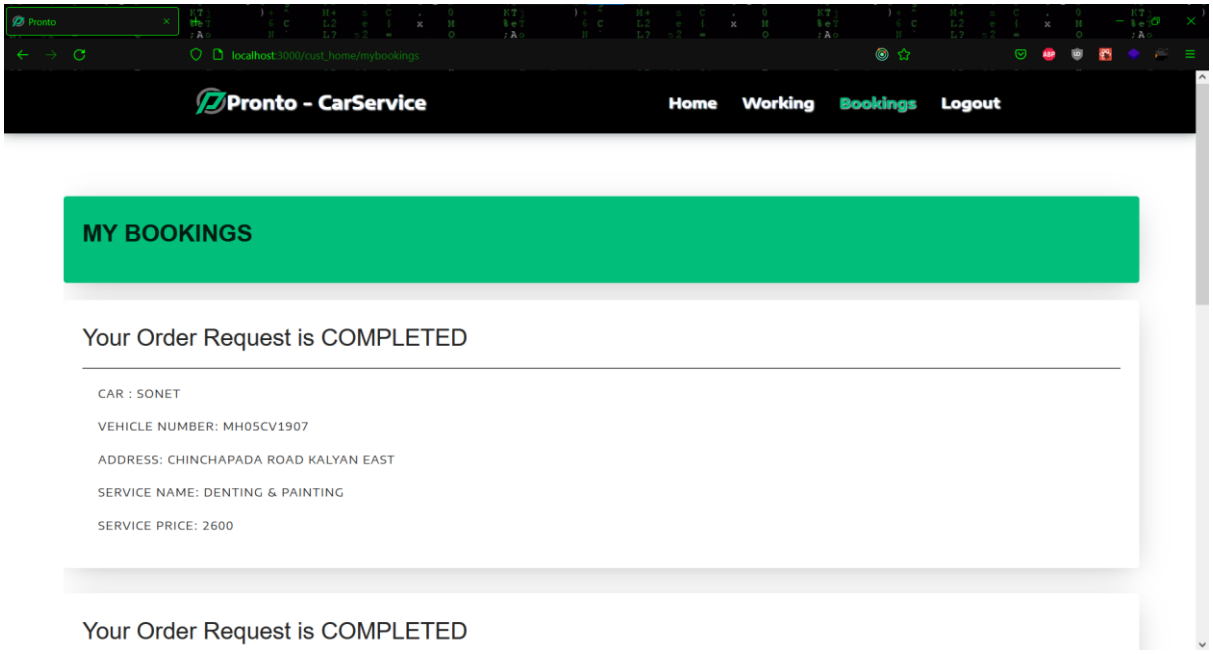


Figure 6.14 Car booking page

- **Bike Taxi**

After getting into bike taxi page, the user will be directed first to the homepage, where he or she will be able to view all of the bikes available for booking. After deciding on a bike, the user can book it by clicking the Book Now option. The user can see the bike's specifications as well as time slots. After everything has been finalized, the user can proceed to payment, and once payment has been confirmed, the user will be able to see his or her order in the booking section.

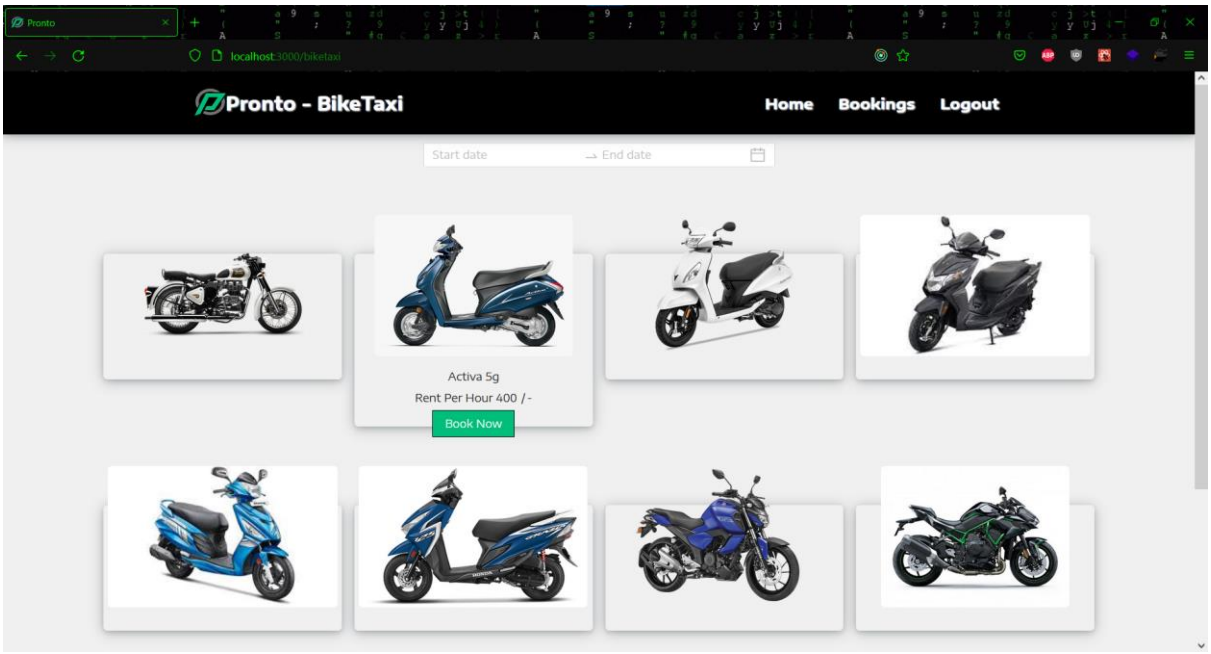


Figure 6.15 Bike taxi home page

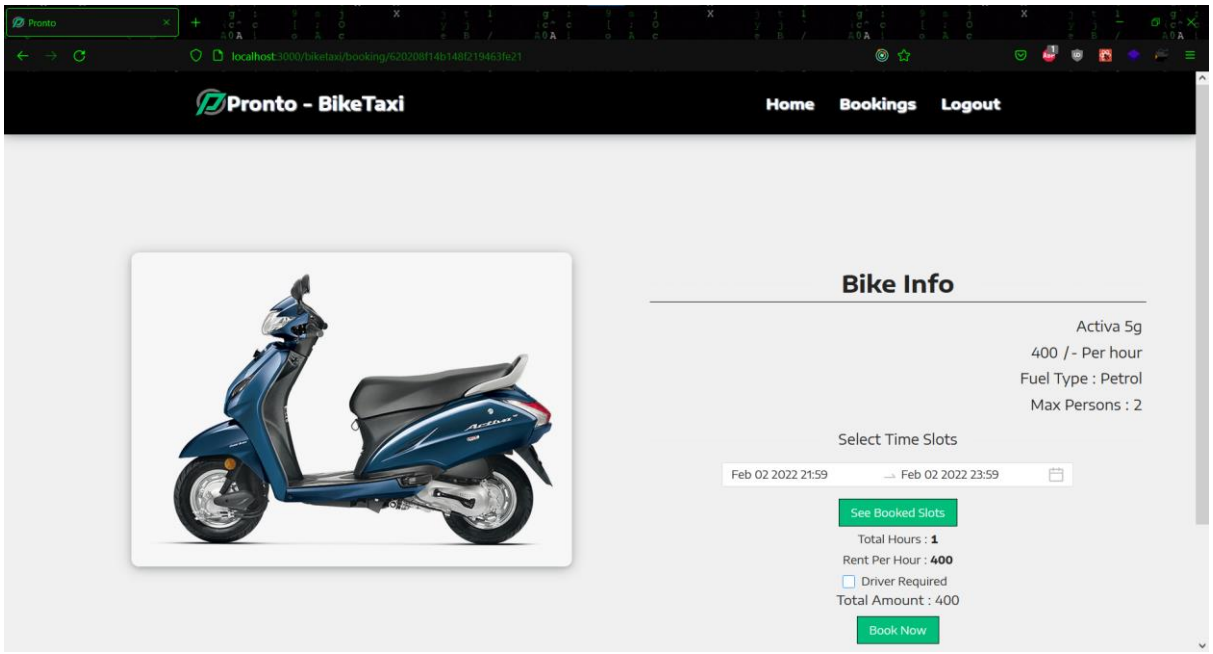


Figure 6.16 Bike Info

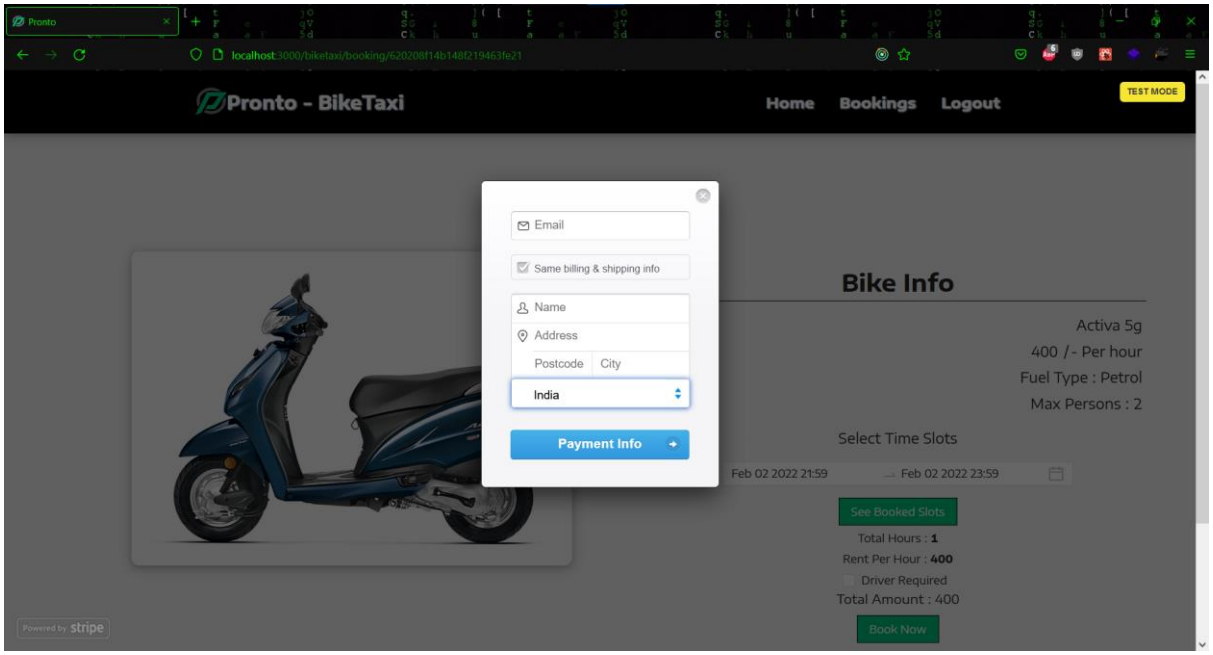


Figure 6.17 Payment Initiation

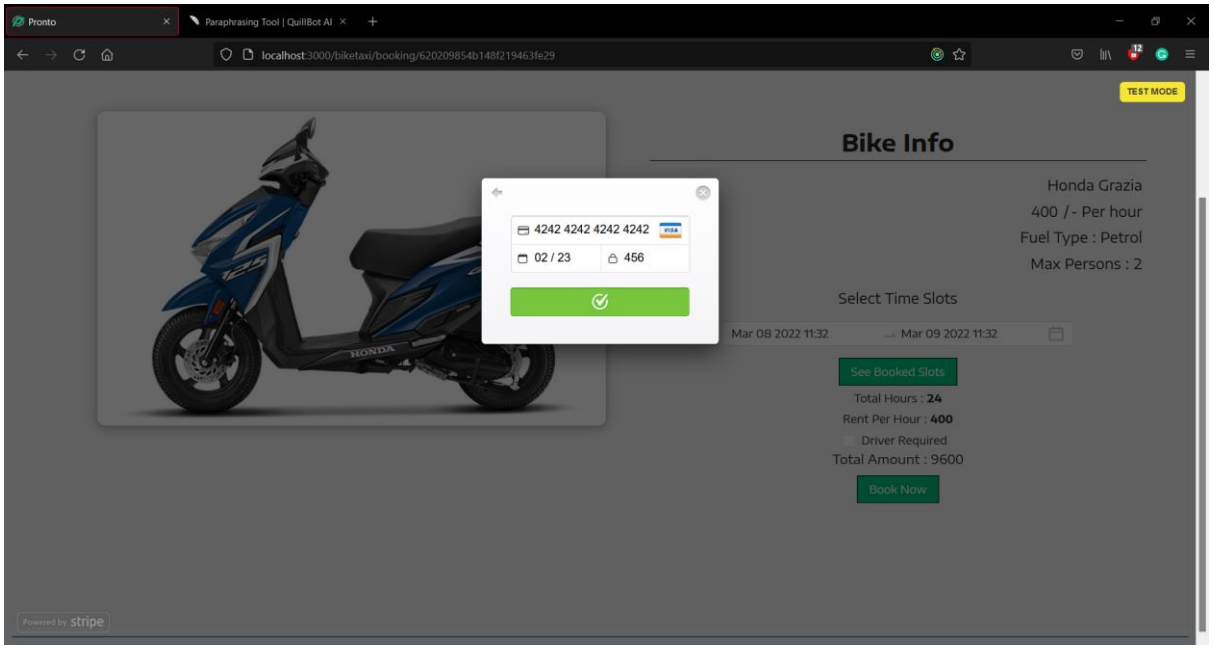
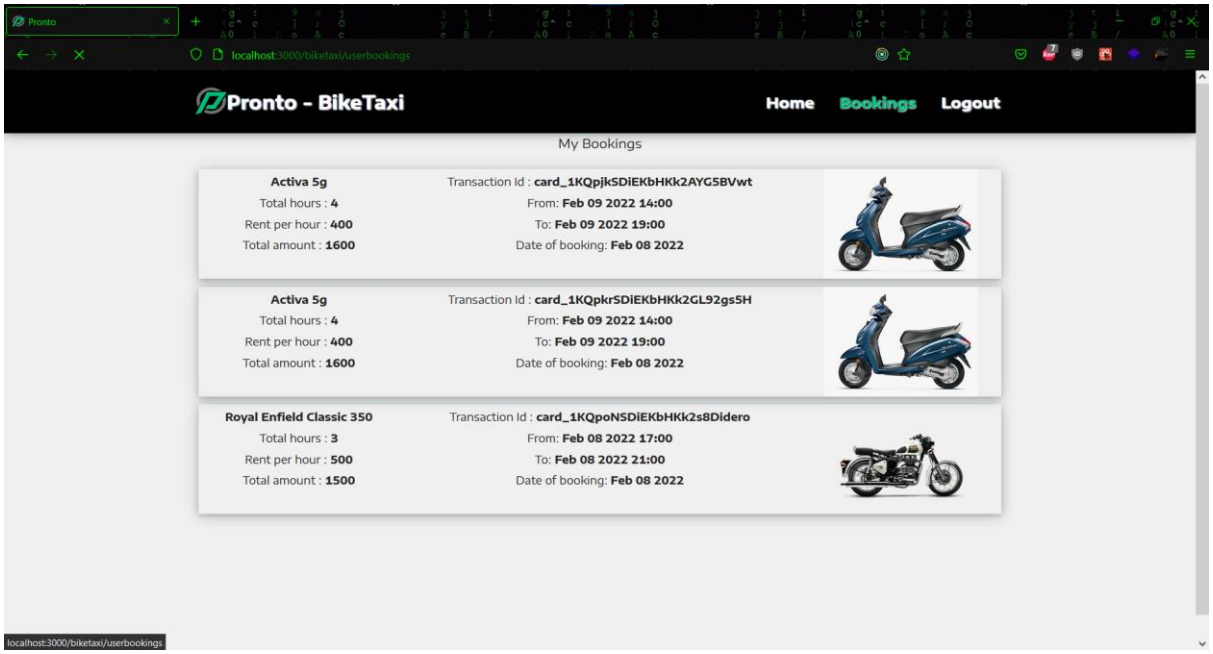


Figure 6.18 Payment confirmed



6.19 User bike bookings

- **Home Delivery**

When a user enters into home delivery service, the first page he or she sees is the home page, where the user can browse through various products. The user has the ability to filter products by categories and sort them by price. The user can access detailed information about a product by clicking the view button. Once the product is finalized the user can add it into the cart and can proceed with payment. The payment window opens out and once the payment is done then the success message is shown. In the history section, the user can review his or her previous orders and download the bill for the desired order.

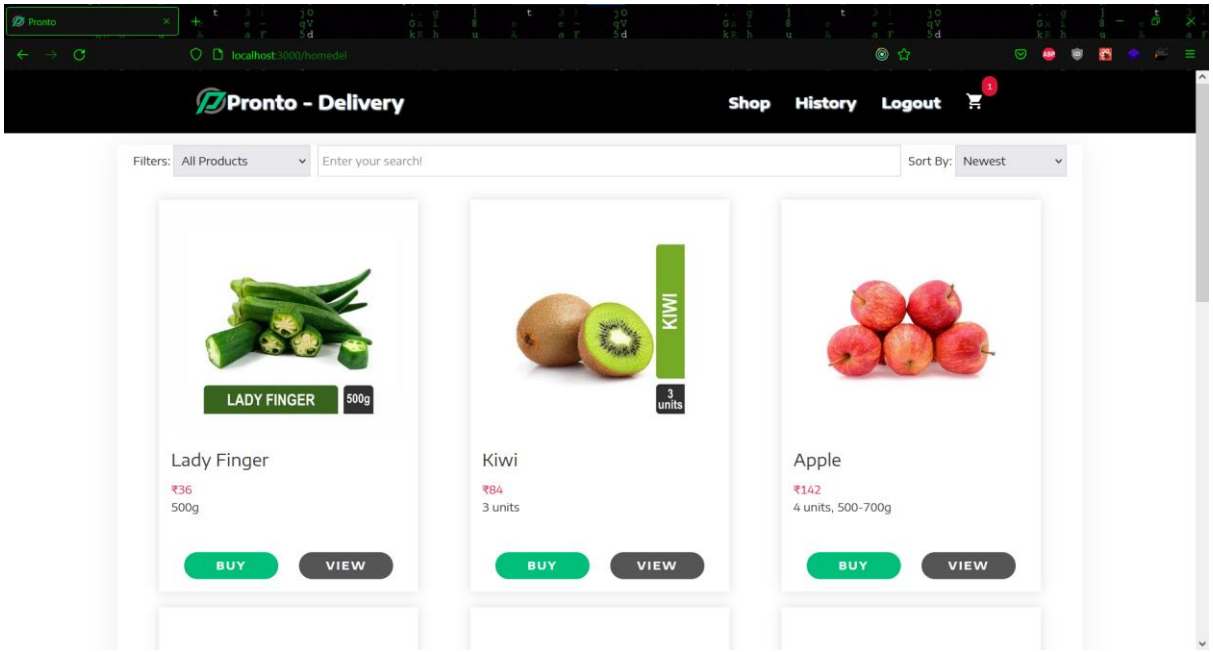


Figure 6.20 Delivery home page

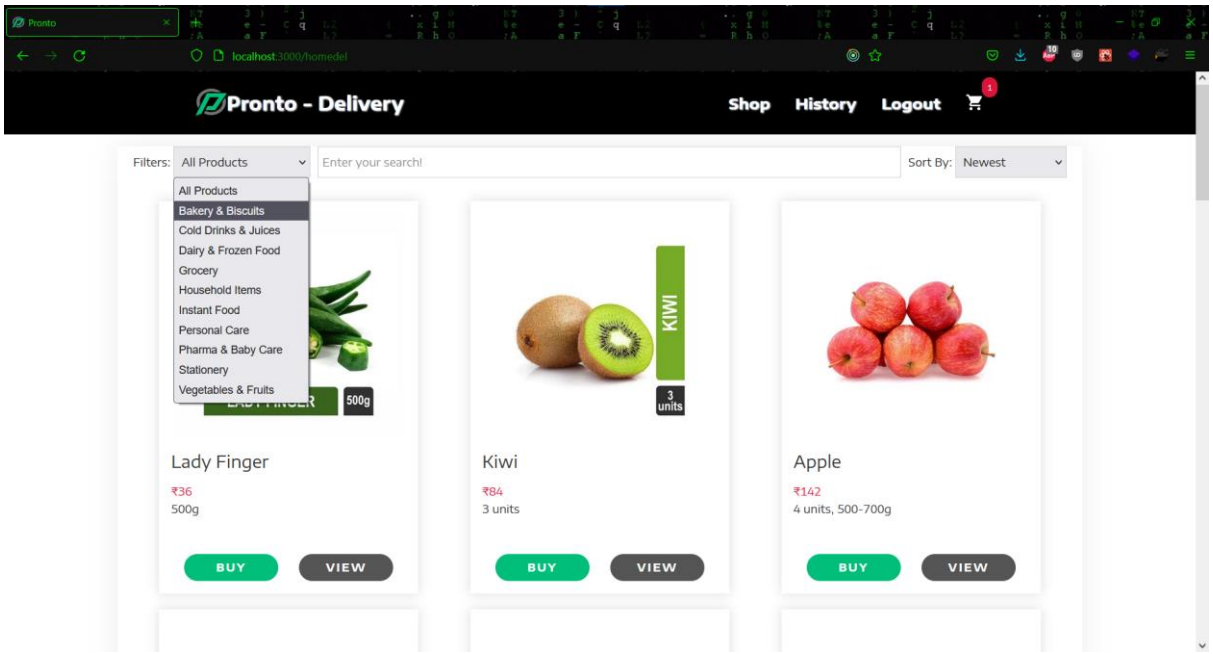


Figure 6.21 Product categories

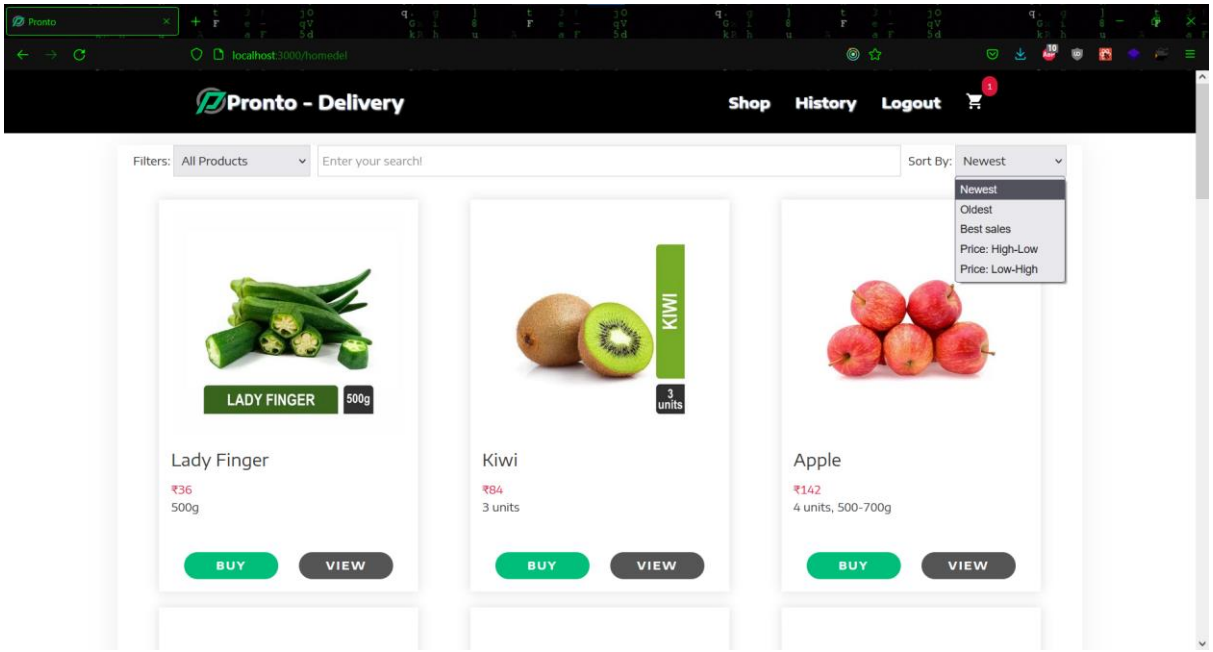


Figure 6.22 Sort Product

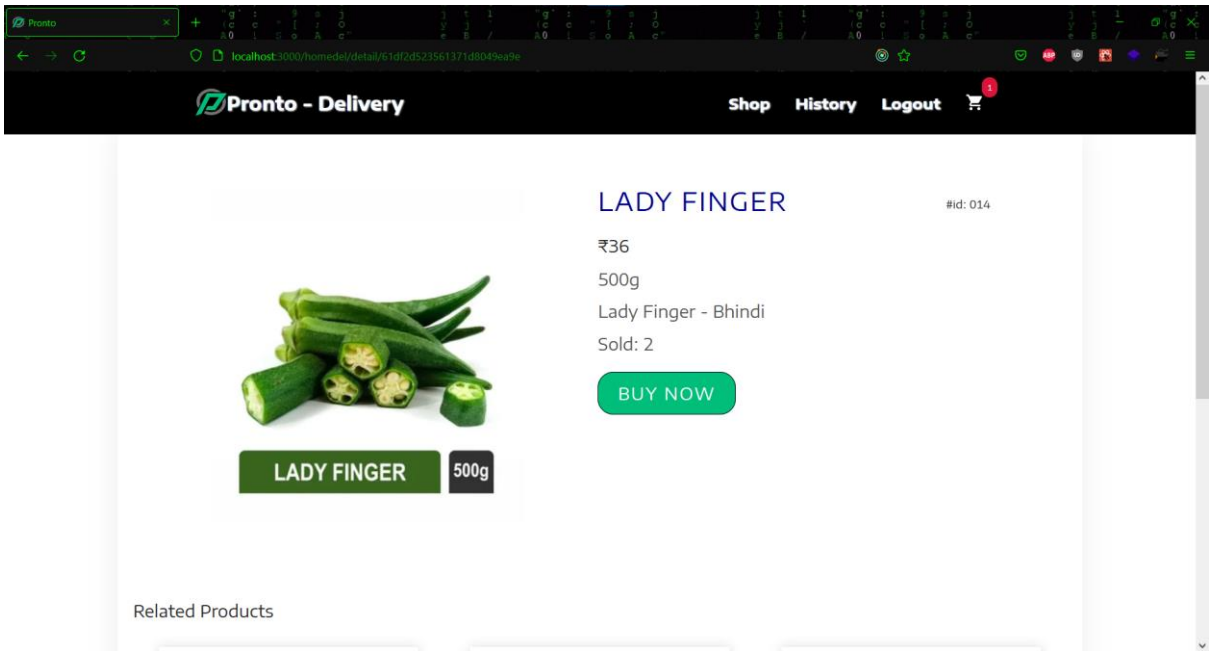


Figure 6.23 Product Description

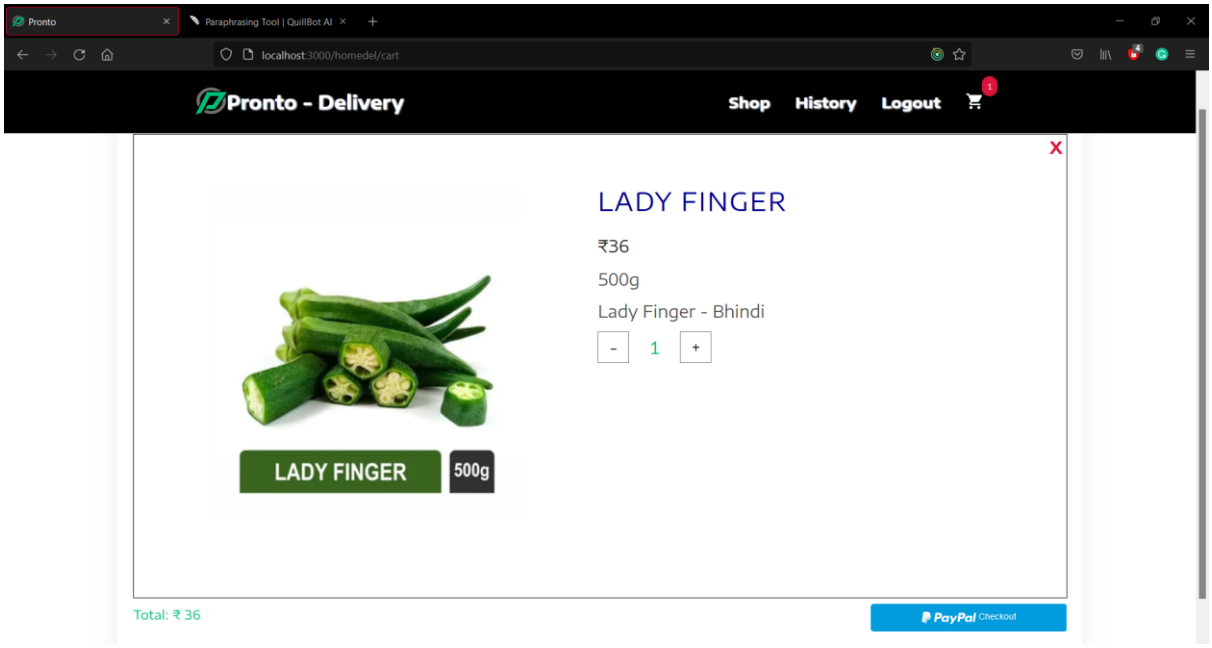


Figure 6.24 Product Cart

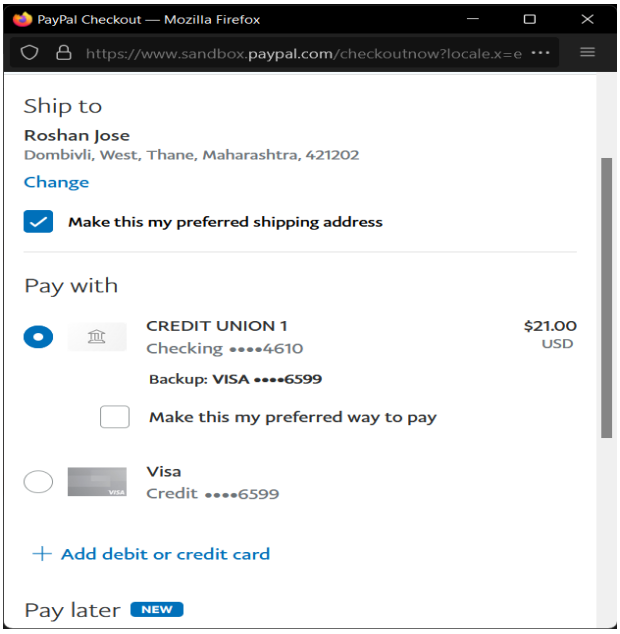


Figure 6.25 PayPal window

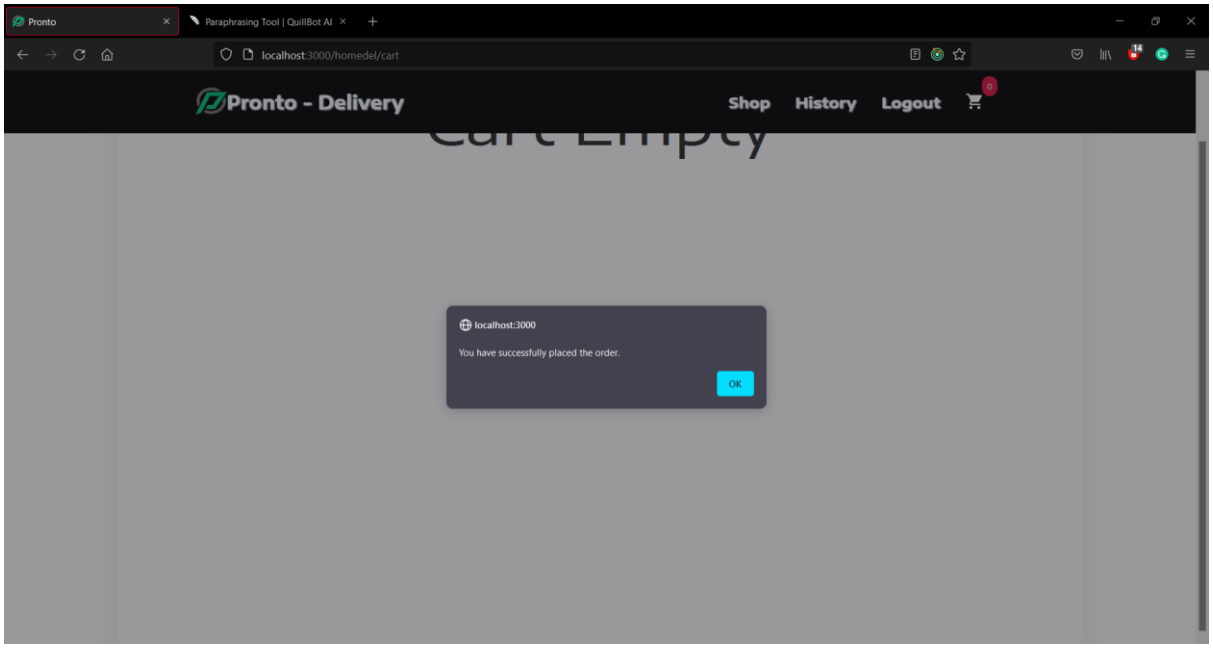


Figure 6.26 Success message

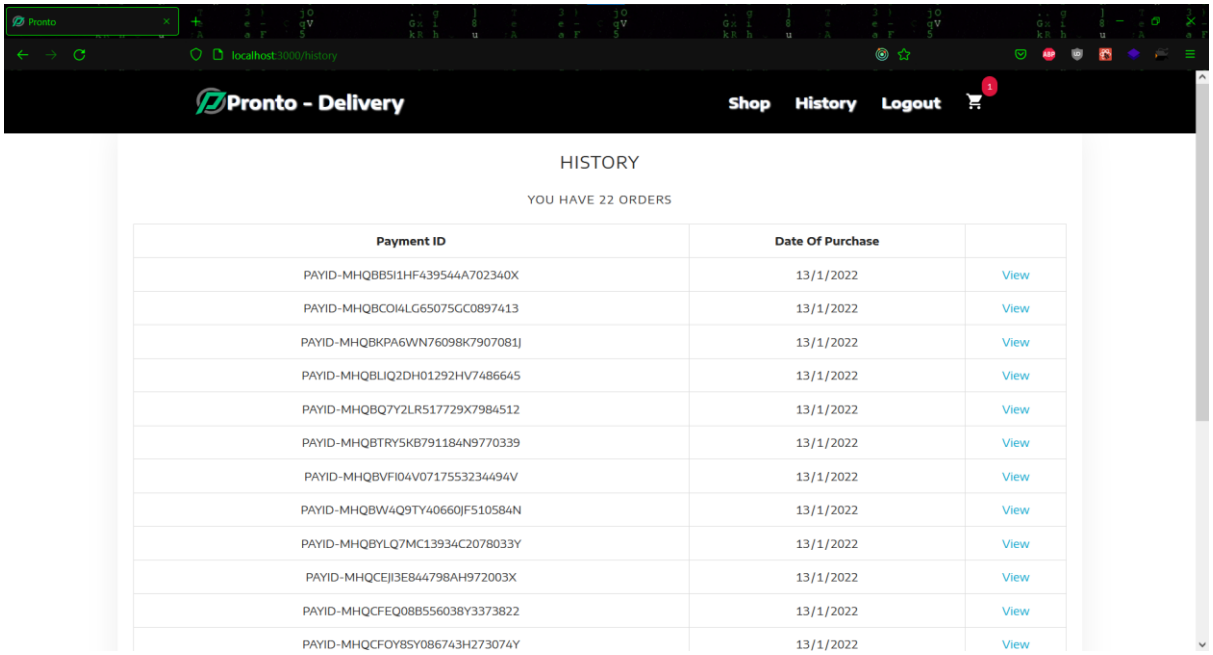


Figure 6.27 Order History

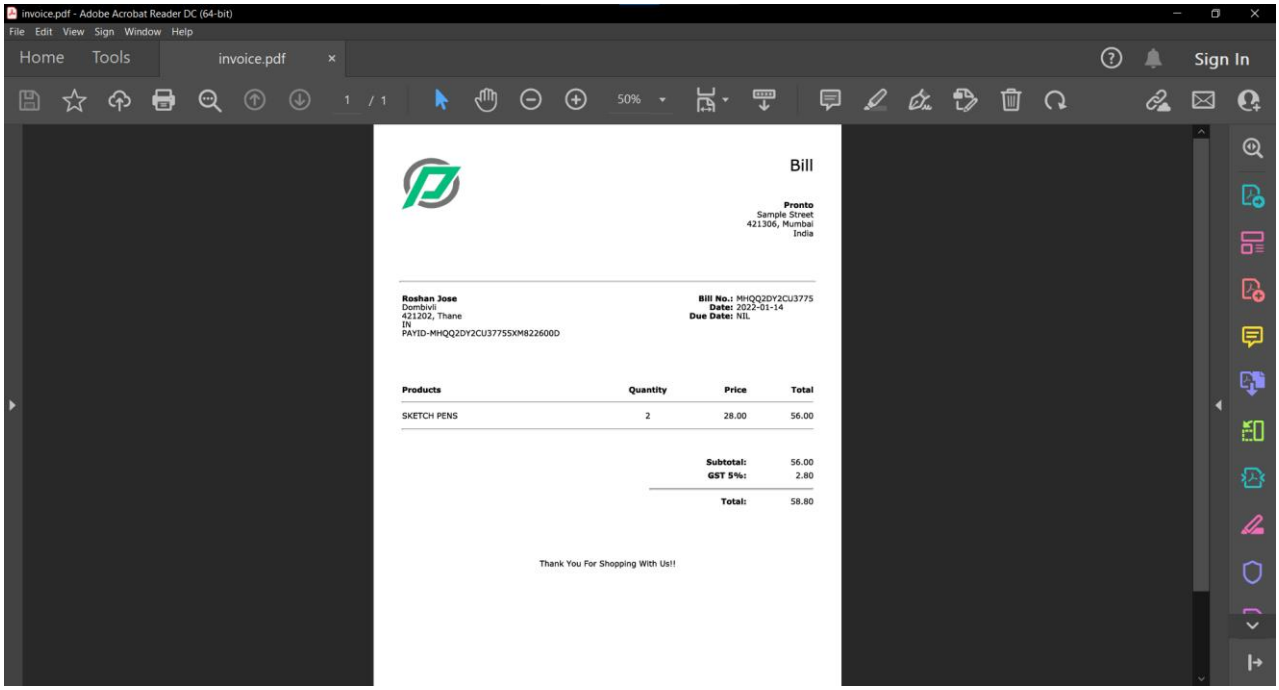


Figure 6.28 Product bill

- **Admin View**

In the admin view an admin can analyze all the services after landing at the post login page. Some admin view of the services are shown below.

- **Car Service**

The Car Service Admin view, which has a common navigation bar, is shown below. First is the admin's homepage, which displays total revenue. The admin may see all of the cars under Cars Data and change or delete them as needed, as well as add new cars. The next section is Service data, which lists all of the services and allows the administrator to delete, modify, or add services as needed. The admin can keep track of all the mechanics on the Mechanics page and add new mechanics using the Add Mechanic button. Admins can also view all of the orders that users have placed and assign mechanics to them under Current Orders Data. The data for completed orders can be retrieved by pressing the See Completed Orders button, which displays a list of all completed orders along with the assigned mechanic for each one, which can be saved as a csv or pdf file.

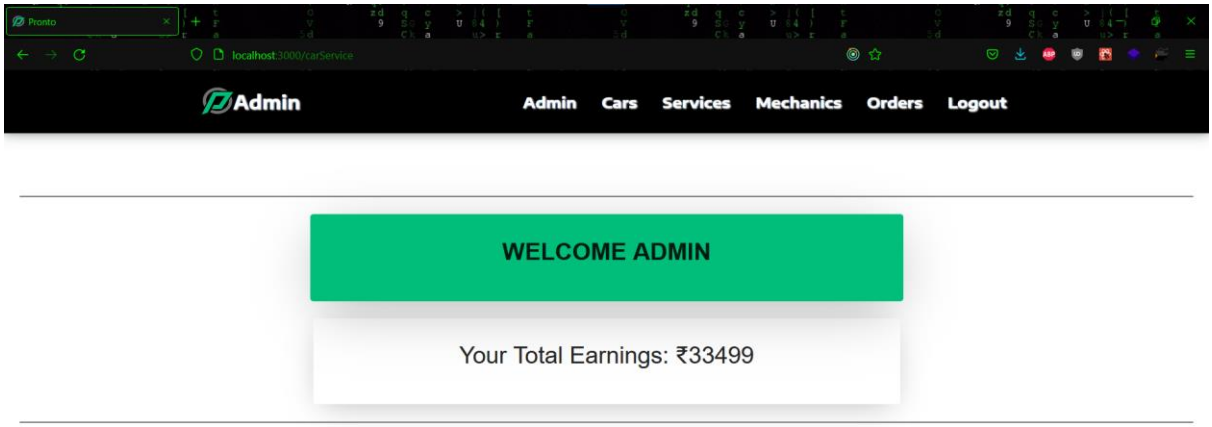


Figure 6.29 Admin home page

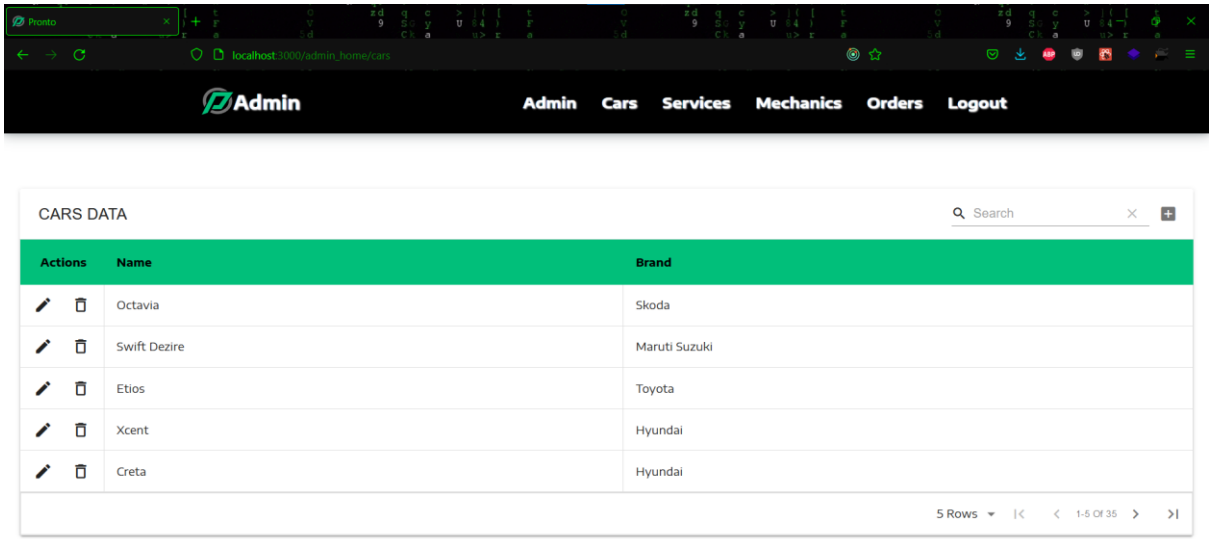


Figure 6.30 Car data

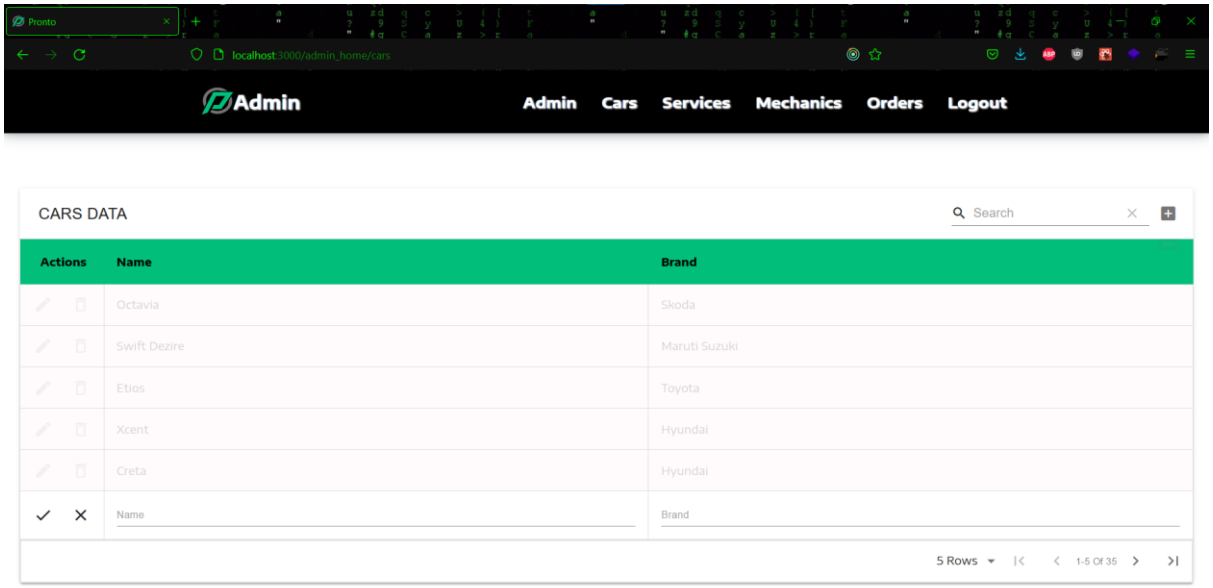


Figure 6.31 Adding new car

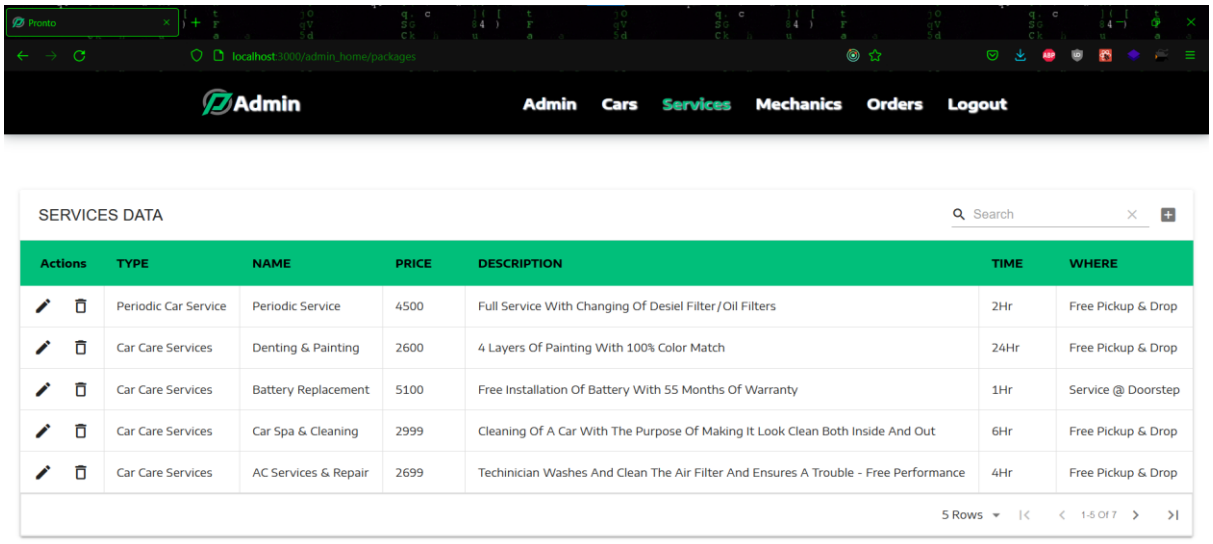


Figure 6.32 Car service data

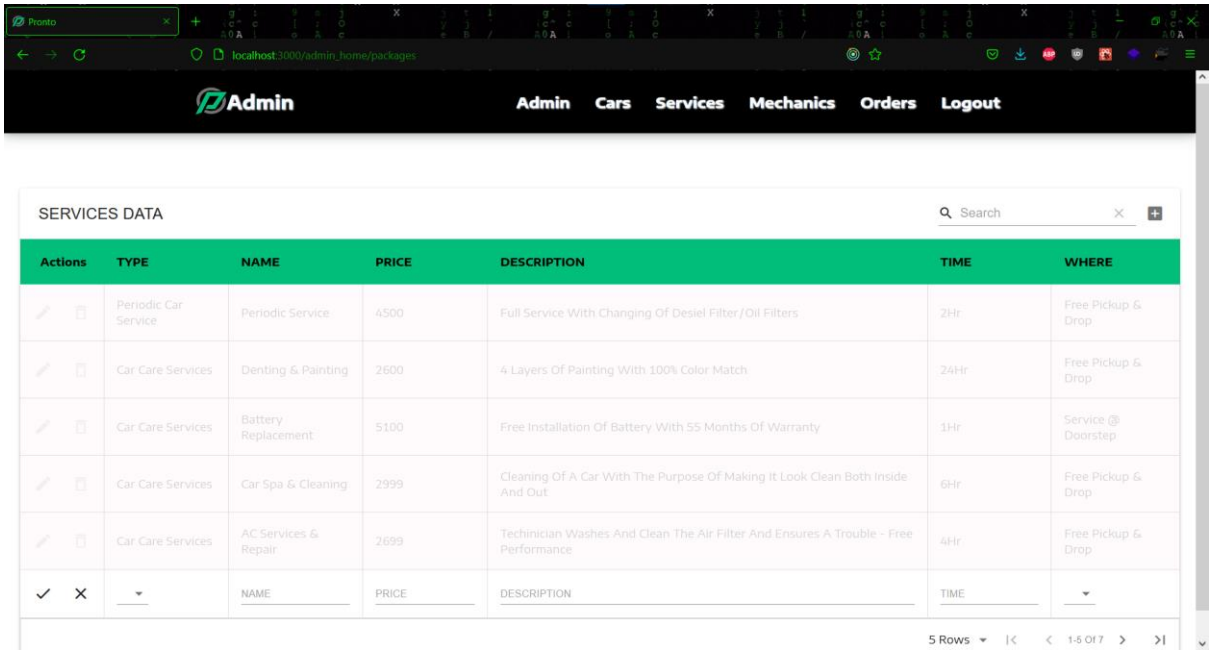


Figure 6.33 Adding new car service

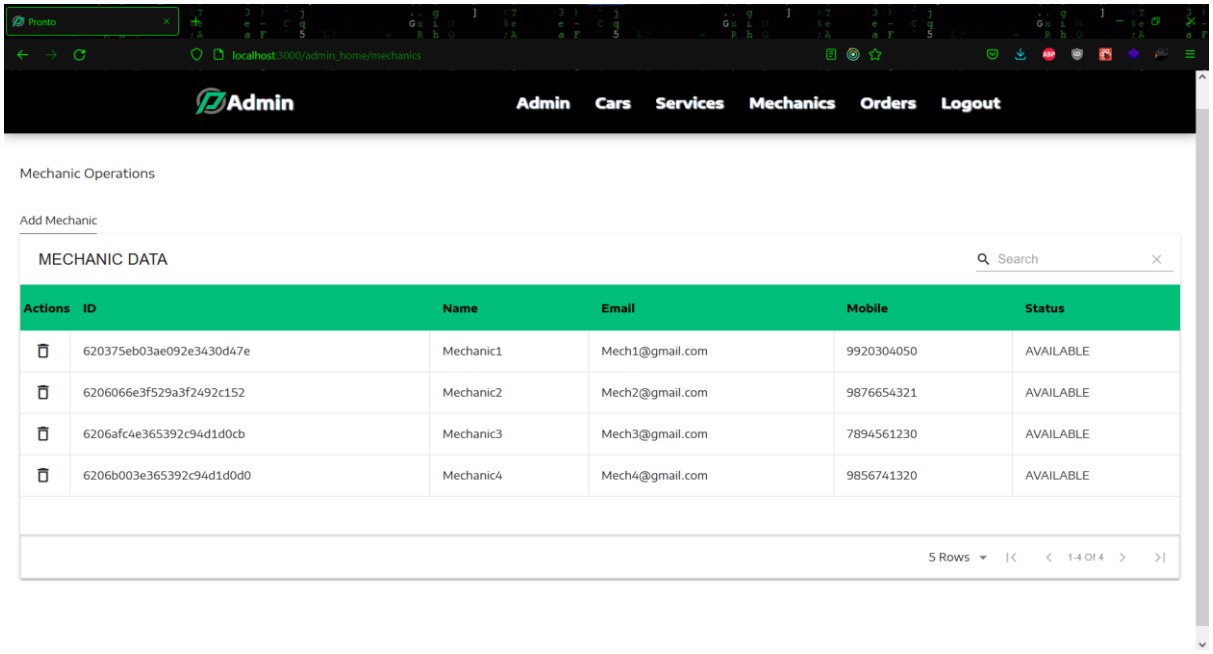


Figure 6.34 Mechanic data

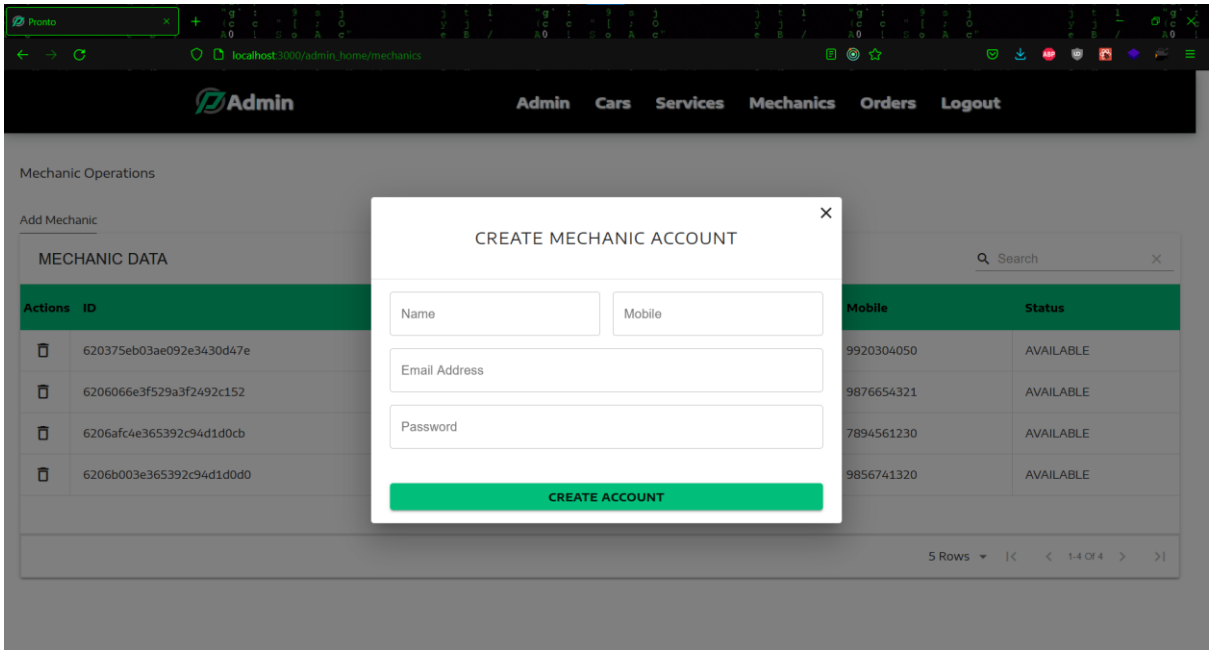


Figure 6.35 Adding new mechanic

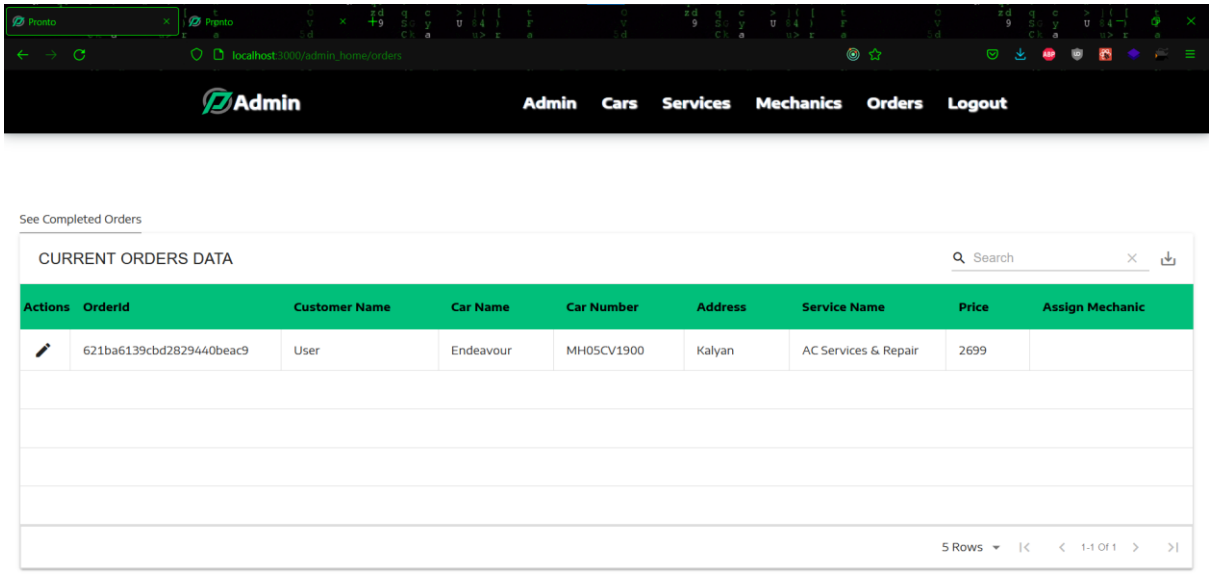


Figure 6.36 Current orders data

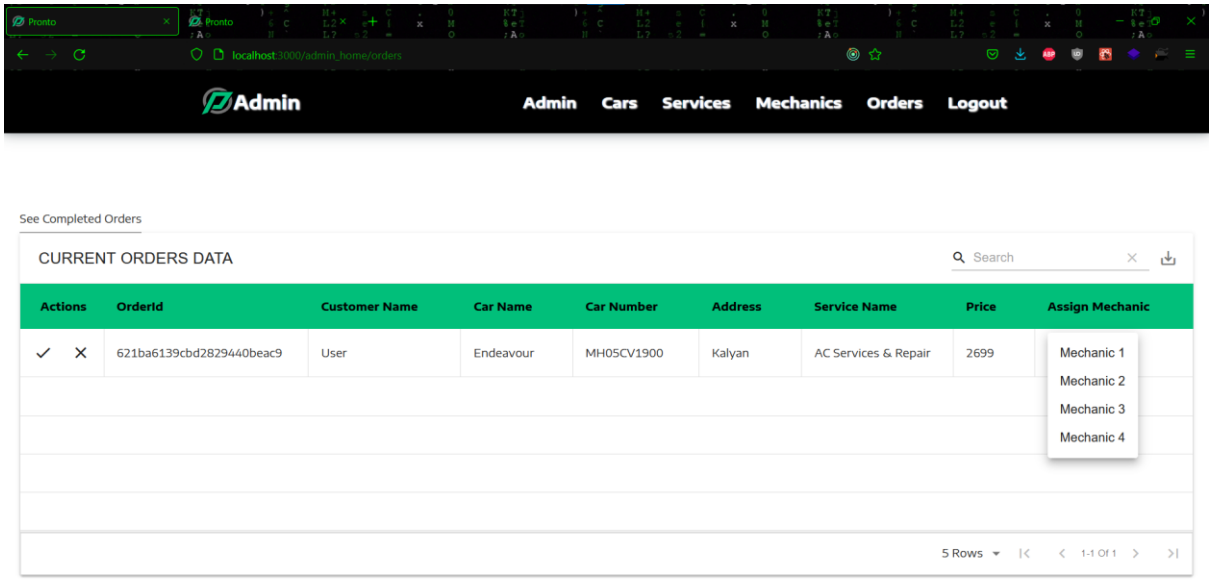


Figure 6.37 Assigning mechanic

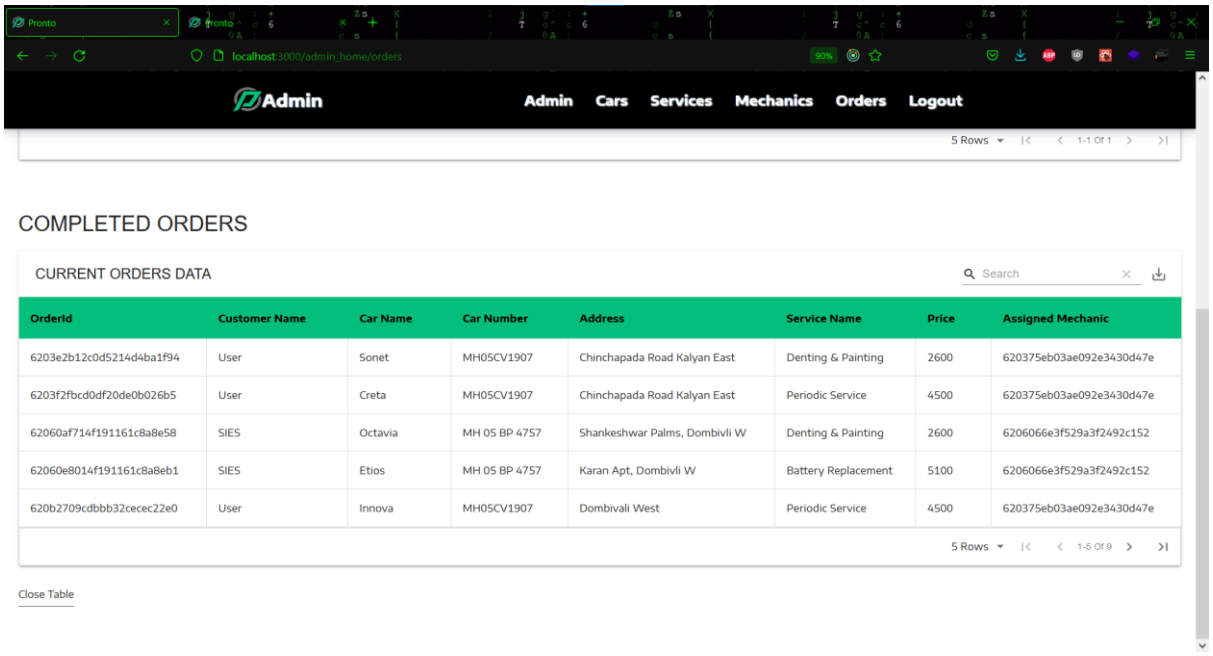


Figure 6.38 Completed order data

- **Bike Taxi**

  This is the Bike Taxi Admin view, which features a common navigation bar. The first one is the homepage, which allows the administrator to add, edit, and delete bikes. Admins can easily add bikes by clicking the add bike button and giving a few details. The edit button can be by the administrator to update any of the bike's details. The delete button can be used to remove the bike if necessary. Admin may also see and track all of the orders that have been placed by the users.

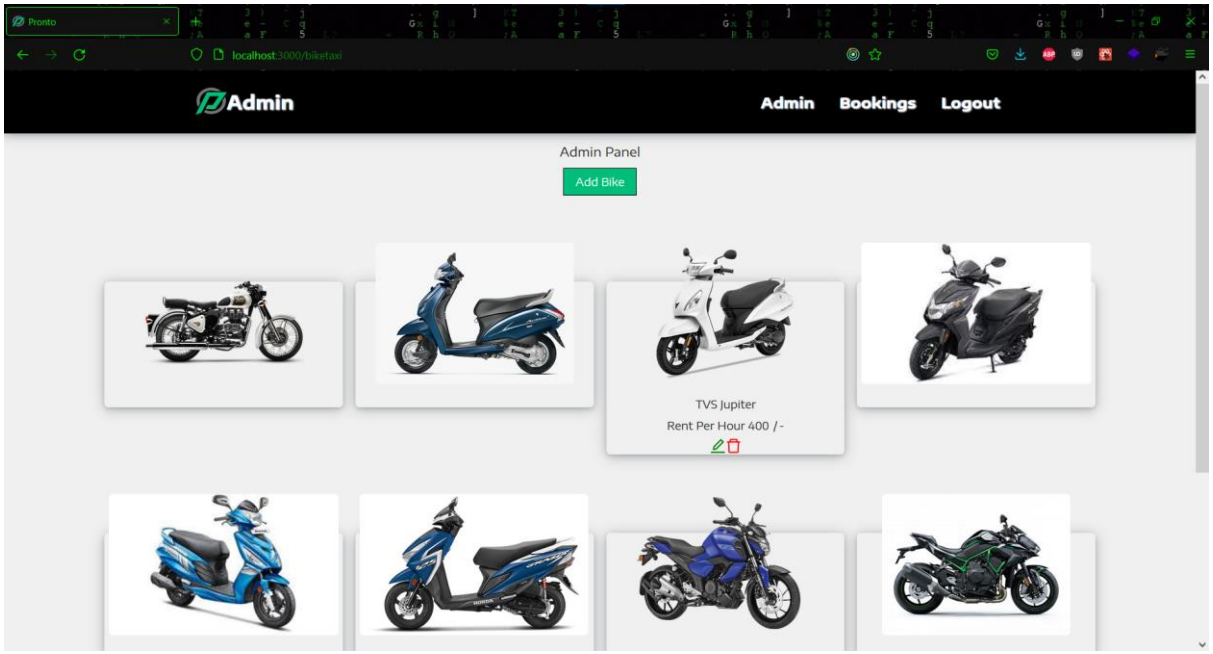


Figure 6.39 Bike admin home page

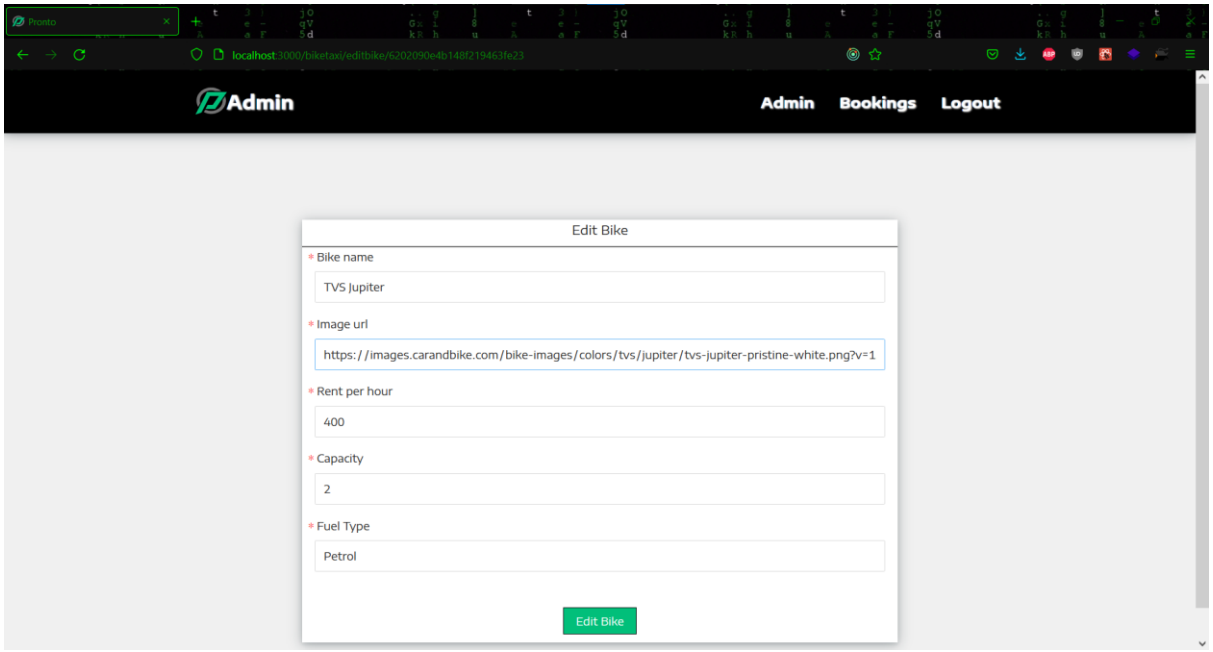


Figure 6.40 Edit bike details page

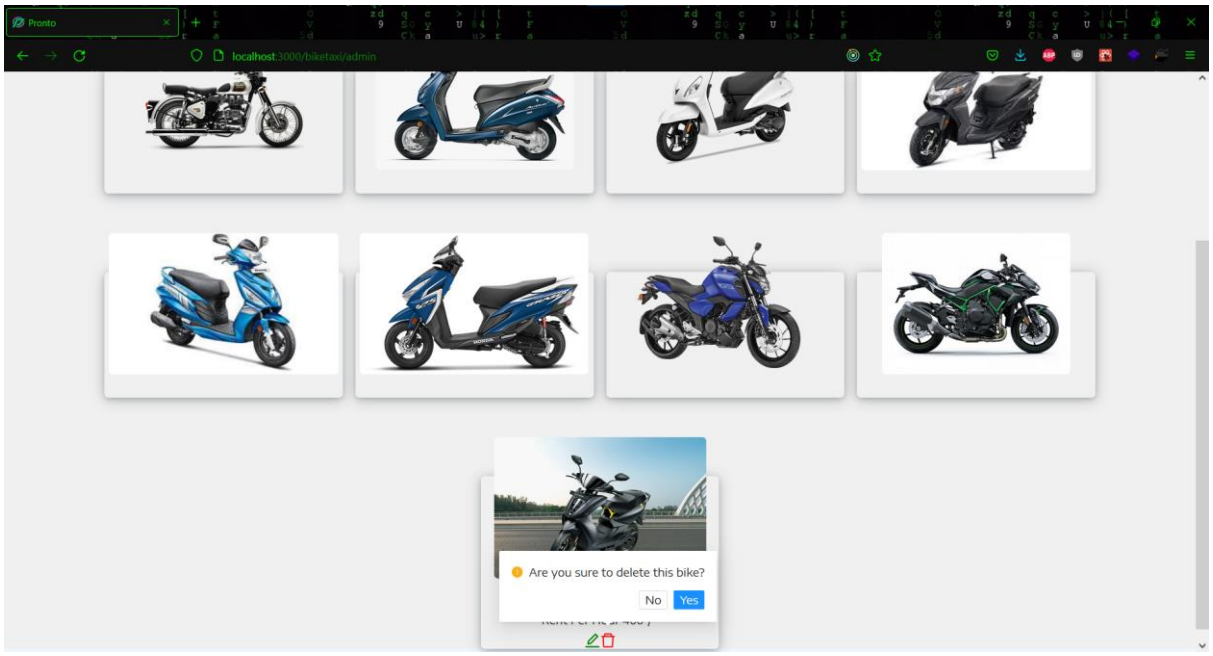


Figure 6.41 Delete bike

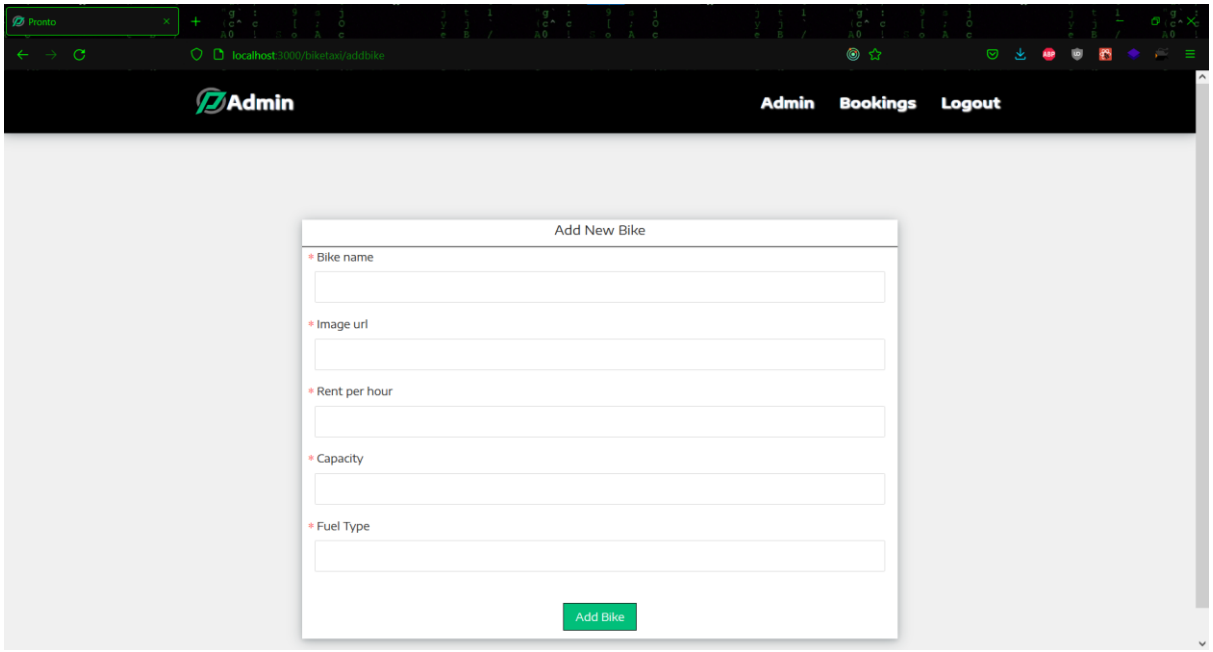


Figure 6.42 Add new bike page

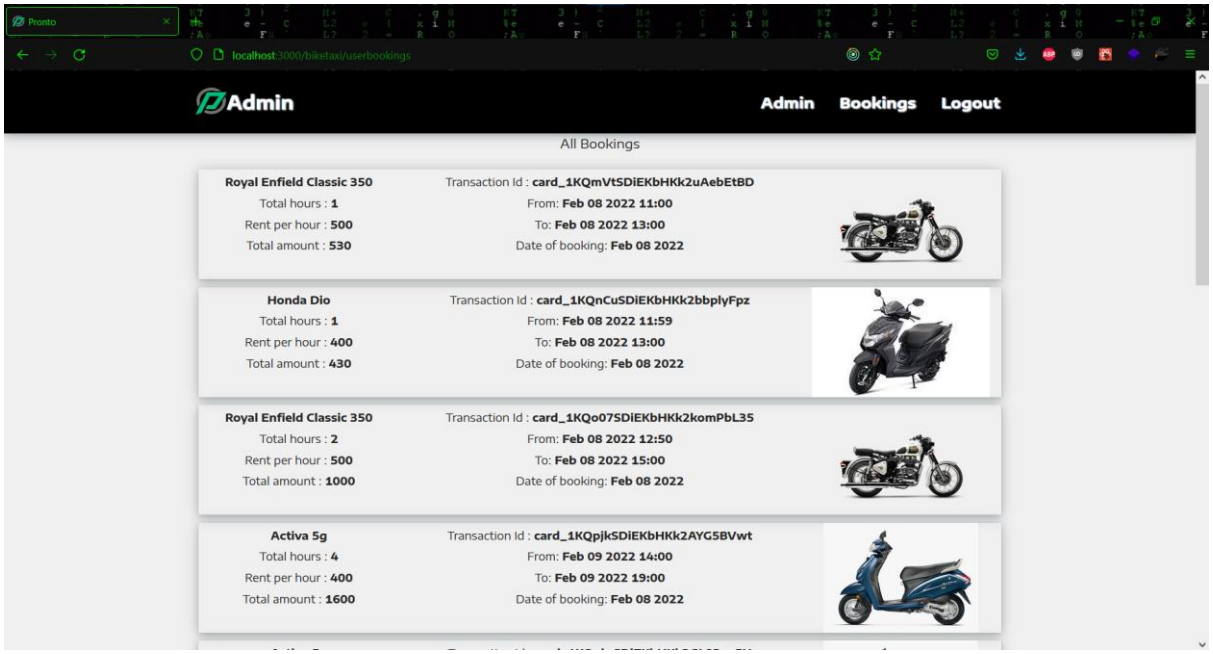


Figure 6.43 All bookings

- **Home Delivery**

  This is the admin view for Home Delivery, which has a common navigation bar. The first is the homepage, where the admin can delete or edit products. If the administrator needs to make changes to the product, they can do so using the edit button on each item card. If a product needs to be removed, it can be easily done using the delete button provided on each item card. The Delete all button allows the administrator to delete all of the products with a single click. There are search, filter, and sorting options for a better admin experience. The admin can alter i.e., edit, delete, or add categories in the categories page. Admins can also use the Create Products page to create new products. Admin can also see all of the orders that have been placed by the users and keep track of them.

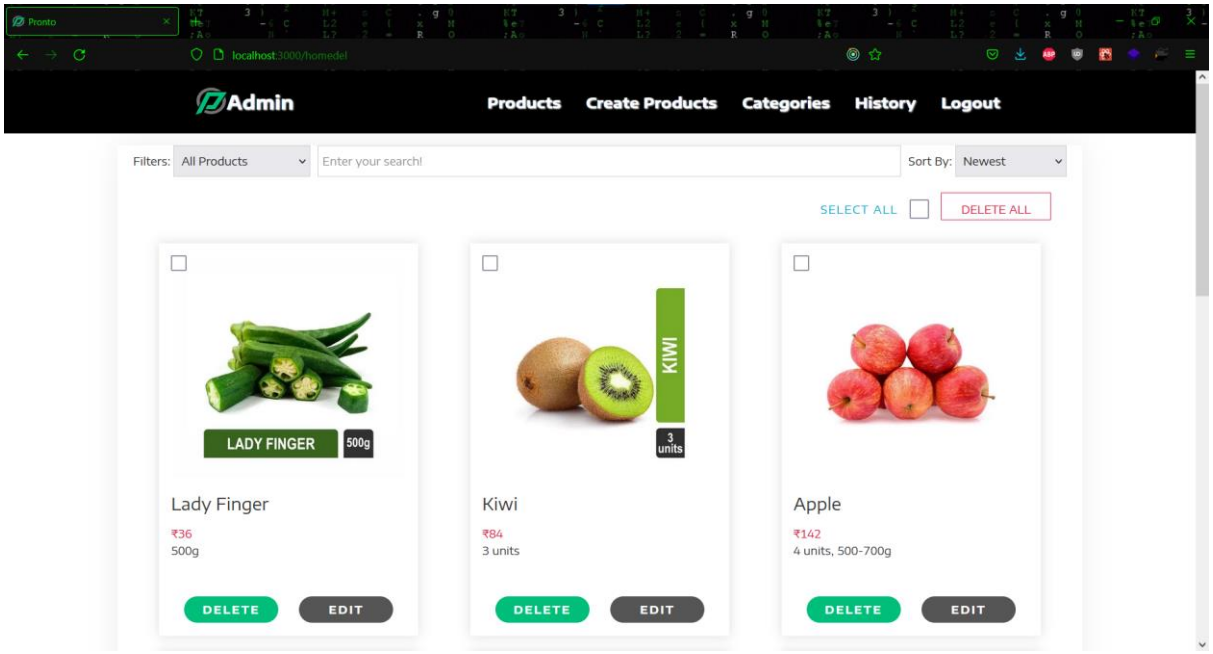


Figure 6.44 Admin delivery home page

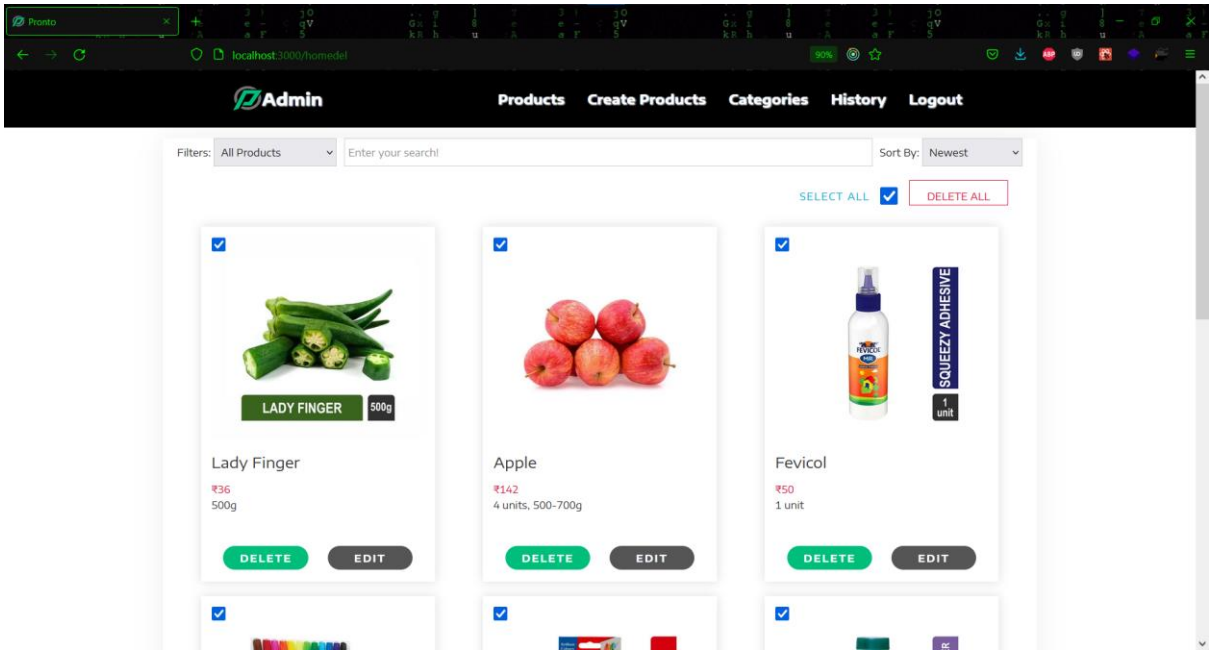


Figure 6.45 Selecting products

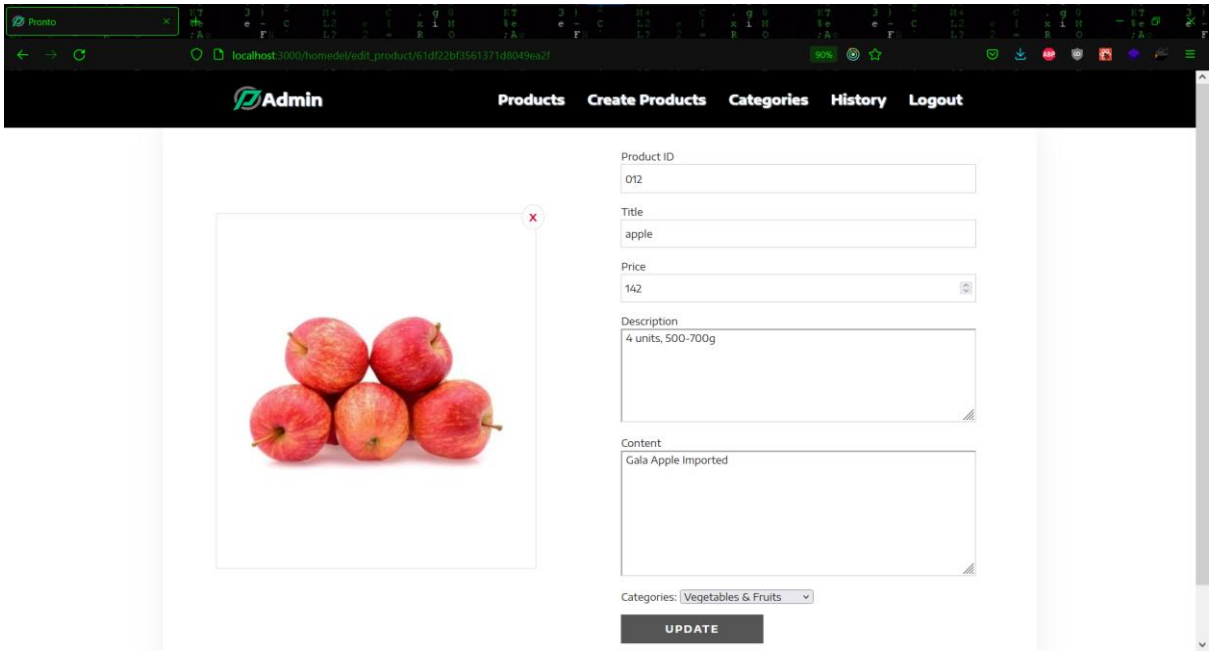


Figure 6.46 Update product detail

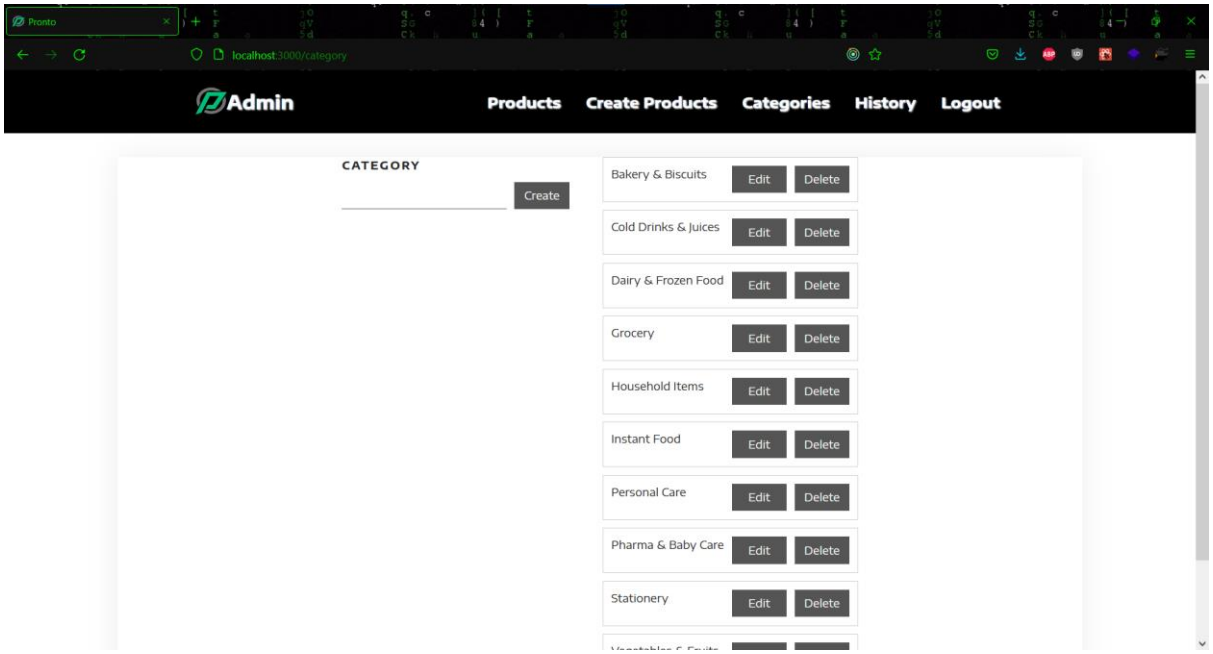


Figure 6.47 Category list

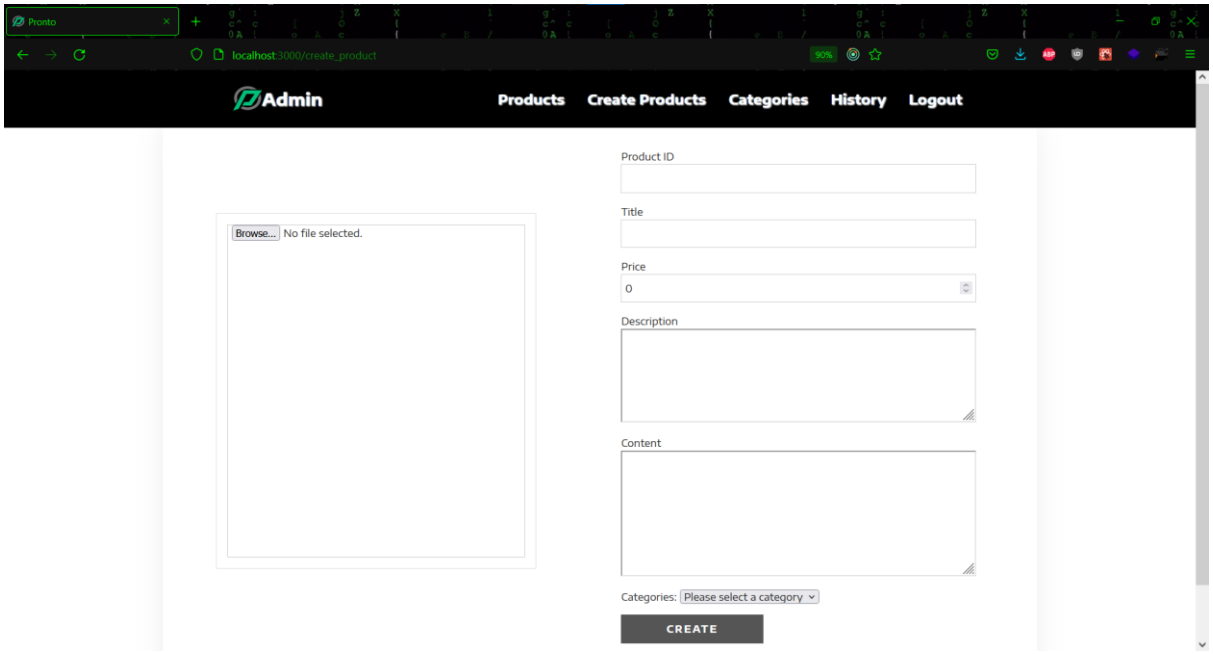


Figure 6.48 Creating new category

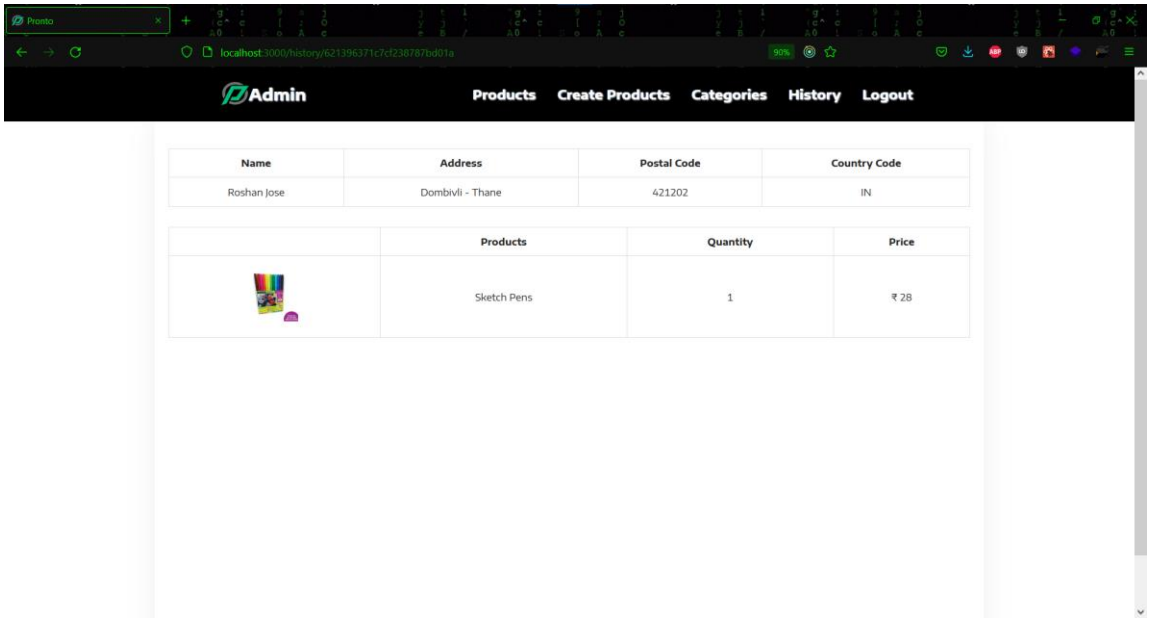


Figure 6.49 Ordered product detail

- **Mechanic View**

This is the website's mechanic view. When a mechanic logs in, he is directed to the Mechanics homepage, which features a common navigation bar. All assigned orders that are in process can be seen under the In-Process Order section. The order's status can be changed suitably by the mechanic. Under the My Orders page, the mechanic may see all of the orders. Mechanics may also save My Orders as a csv or pdf file using the download option.

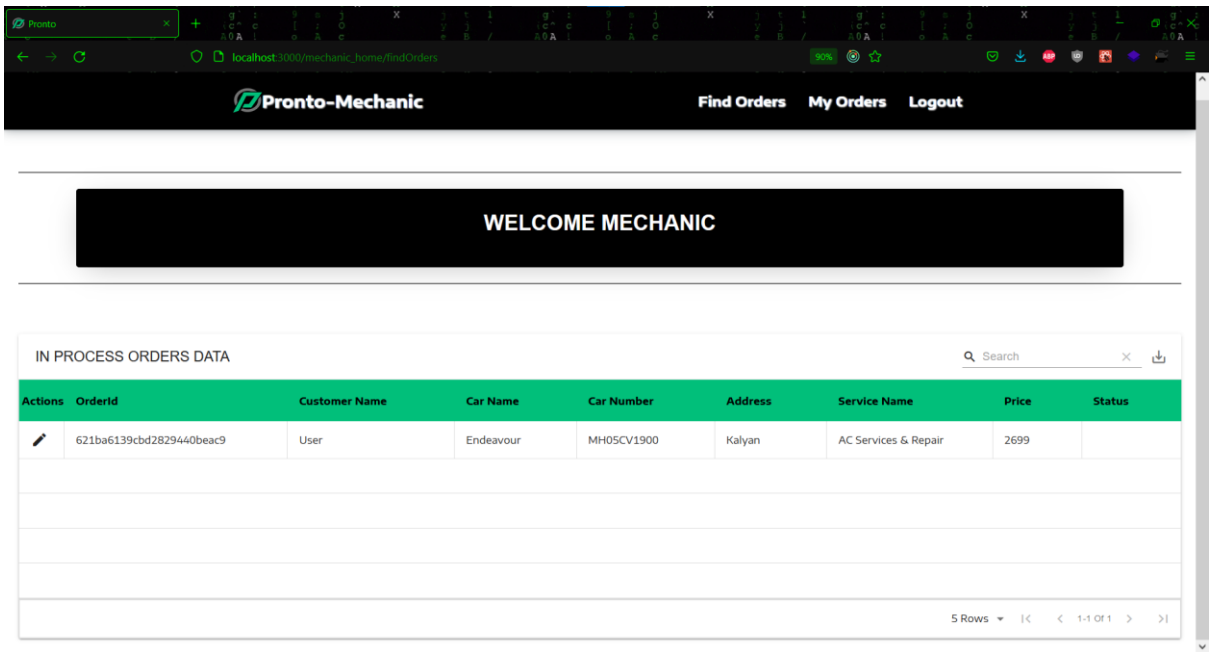


Figure 6.50 Mechanic homepage

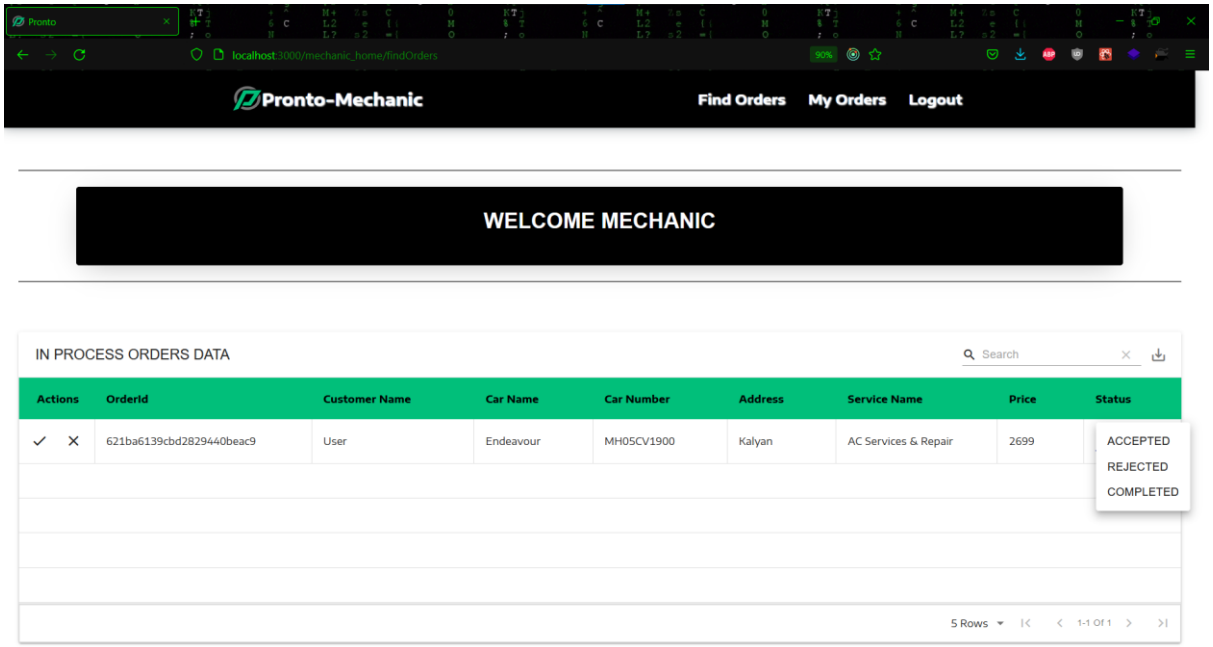


Figure 6.51 In process orders data

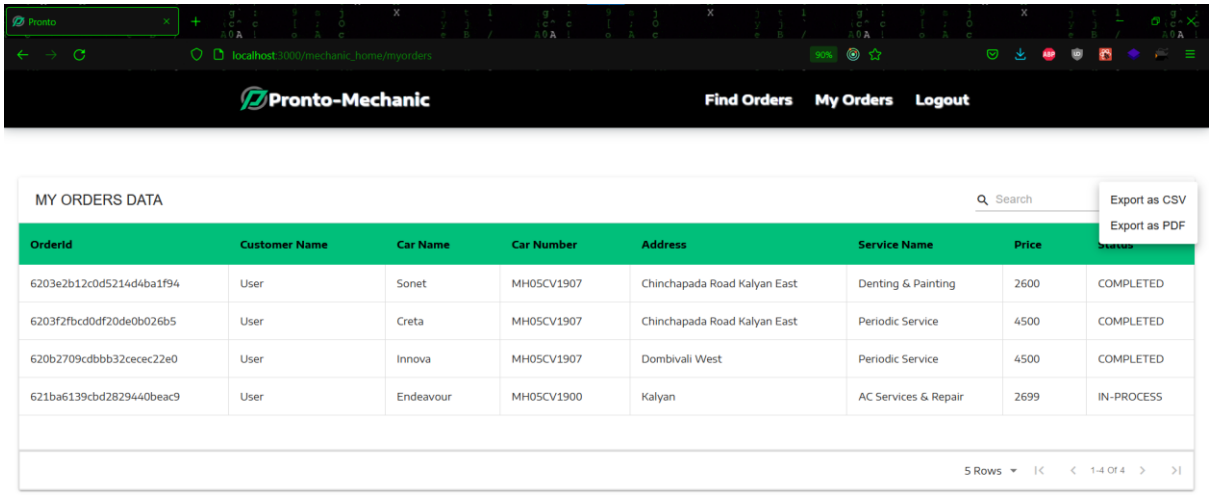


Figure 6.52 Mechanic orders data

- **Feedback**

Users of this website can give feedback or recommendations to the website's admins via email, utilizing the feedback option under Post Login.

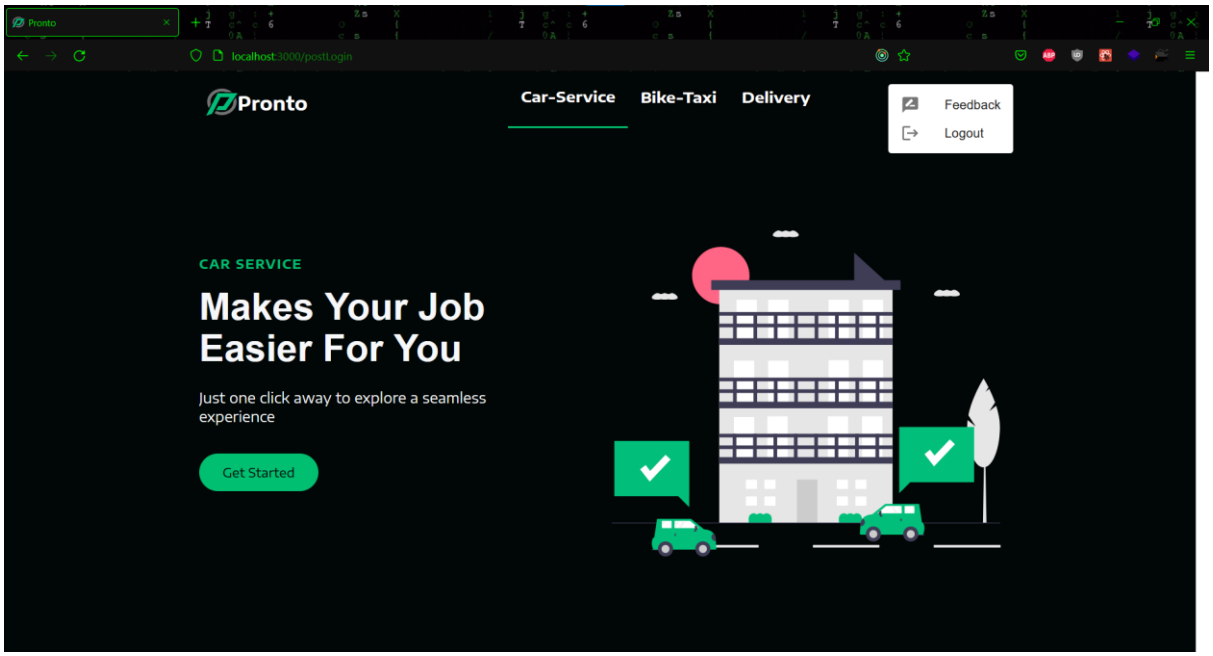


Figure 6.53 Feedback and Log out

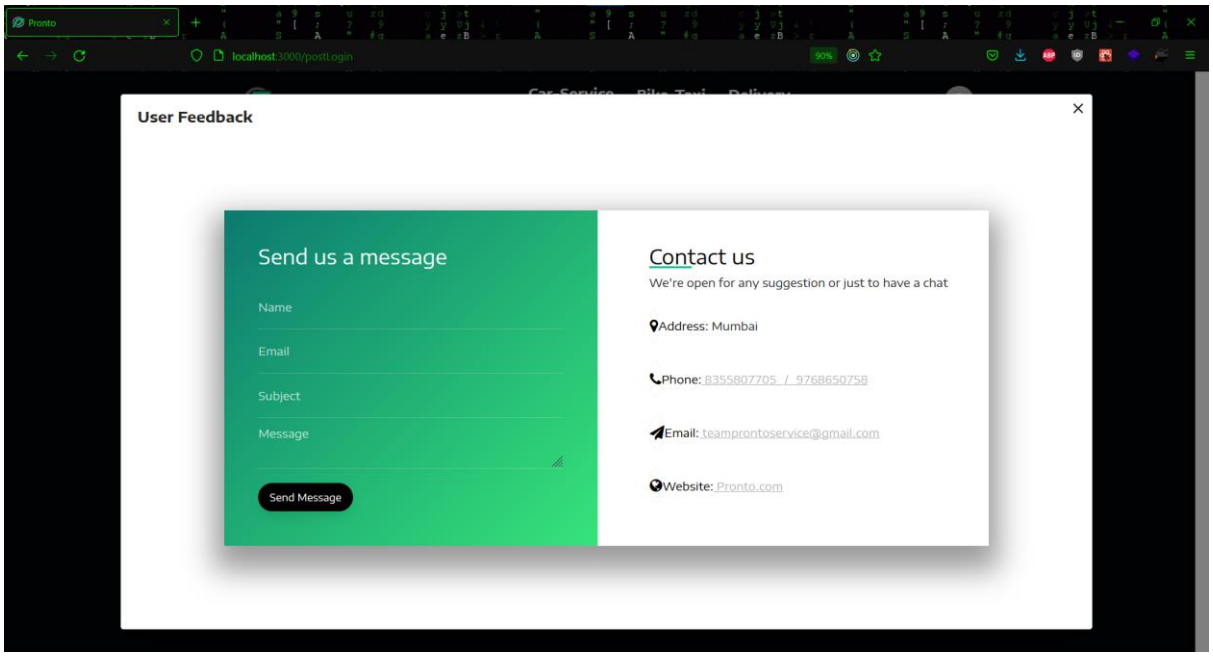


Figure 6.54 Feedback form

Chapter 7: Conclusion

7.1 Significance of the system:

Pronto comes in with the goal of delivering a quick and dependable service that consumers can rely on and utilise as needed. It provides a safe car service with minimal exposure to infectious environments. A user can use the bike taxi service as a substitute means of transportation if they are tired of driving or concerned about traffic.

The home delivery service allows customers to order anything they need and have it delivered from their local markets right away, which is especially useful for older people who are currently living alone due to the pandemic.

7.2 Limitations of the system

Pronto aims to provide the user with an all-in-one solution, but it has some limitations. One of the constraints we discovered during testing was that we couldn't filter the order based on the order dates. Also, a feature allowing consumers to rate products may have been included, making it easier for other customers to choose a product based on its rating. A live tracking system for bike taxi with a driver included could have been implemented to show users the arrival status of the driver in real time. Also, there is presently no driver's view for bike taxi. Furthermore, some user information such as address and contact information is saved in local storage, making it less secure.

7.3 Future Scope of the Project

The project is fully functioning and meets the requirements, although some modules and features could be implemented, which were not possible due to time constraints and a lack of subject knowledge. One of these is a separate page for bike taxi drivers, where they can see all of the bookings that have been assigned to them. Also, for services, a rating system can be implemented in which a person can submit his or her opinion that will assist other users. Furthermore, we can create order filtering based on order dates to provide admin with a better understanding of product sales. The system's security can be enhanced in the future.

References

- https://reactjs.org/docs/getting-started.html
- https://www.w3schools.com/REACT/DEFAULT.ASP
- https://www.tutorialspoint.com/reactjs/index.htm
- https://www.geeksforgeeks.org/reactjs-tutorials/
- https://expressjs.com/en/guide/routing.html
- https://expressjs.com/en/4x/api.html#express
- https://docs.mongodb.com/drivers/node/current/fundamentals/connection/
- https://docs.mongodb.com/drivers/node/current/quick-start/
- https://www.npmjs.com/package/easyinvoice
- https://www.npmjs.com/package/@paypal/react-paypal-js
- https://stripe.com/docs/stripe-js/react
- https://www.npmjs.com/package/@stripe/react-stripe-js