Business-to-Consumer on Demand Delivery Cloud Application

Submitted by

Roshni Adhikary

Business-to-Consumer on Demand Delivery Cloud Application

This project is submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology in Information Technology at the Department of Engineering and Technological Studies, University of Kalyani.

Submitted by
Roshni Adhikary

Under Supervision of
Dr. Nabin Ghoshal

DEPARTMENT OF ENGINEERING AND TECHNOLOGICAL STUDIES
UNIVERSITY OF KALYANI

JULY 2022

CERTIFICATE

DEPARTMENT OF ENGINEERING AND TECHNOLOGICAL STUDIES

UNIVERSITY OF KALYANI
KALYANI,NADIA
WEST BENGAL-741235

This is to certify that the project report entitled as "**Business-to-Consumer on Demand Delivery Cloud Application**" has been submitted by **Roshni Adhikary (Reg. No- 100646 of 2018 - 2019, Roll No- 90/INT/180003)** in the partial fulfillment for the requirement of the degree of **B.tech** in **Information Technology** at Department of engineering and technological studies, University of Kalyani, under the supervision of undersigned.

The foregoing project is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite for the degree for which it has been submitted. It is understood by the approval that the undersigned does not necessarily endorse or approve statement made, opinion expressed or conclusion drawn therein, but approves the project report only for the purpose for which it is submitted.

_____
Dr. Nabin Ghoshal
Project guide
DETS, University of Kalyani

_____
Head of the Department

Examiners:_____

ACKNOWLEDGEMENT

I would like to express our humble gratitude and sincere thanks to **Dr. Nabin Ghoshal**, professor at Department of Engineering and Technological Studies, University of Kalyani, under whom I executed my project work for giving me the opportunity to undertake this project and his valuable guidance, encouragement, useful suggestions and co-operative behavior throughout the project duration.

I would also like to give special thanks to my junior **Rimpa Halder(3rd year student,Roll No-90/INT/190003)** who helped me directly or indirectly during this project work.

Place: Kalyani

Date: **ROSHNI ADHIKARY :**_____

INDEX

TITLE	PAGE NO
COVER PAGE	1
TITLE PAGE	2
CERTIFICATE	3
ACKNOWLEDGEMENT	4
Abstract	5
Chapter 1:Literature Survey	6
Chapter 2:Road Map	16
2.1 FlowChart	16
2.2 Database design	17
2.3 Design mock-up	18
2.4.Domain & Server Set up	19
2.5.Software Requirements	21
Chapter 3:Coding Details	27
3.1 WEB app file and folder structures are mentioned below	27
3.2 API project file and folder structures	55
Chapter 4:Discussion	68
Chapter 5:Comparative Study	69
Chapter 6:Conclusion	70
Chapter 7:References	72

Abstract

The term **ONLINE APP** is becoming the heart of today's world. Everywhere in the world, we can see the importance of online applications. We are on the same page too. Here we are going to make a **Business-to-Consumer** cloud application that has the privilege of an on-demand location-based delivery solution. It will act on **WEB**, **ANDROID**, and **IOS** platforms.
The first question is why this project is to be considered. This application basically will deal with people's daily life needs like **groceries**, **fish**, **meat**, **education stationeries medicine**, etc.

There are many companies who are ruling the daily essentials on-demand delivery system in India. They already created a sky limit milestone. Now why are we going to make the same kind of app? We see that every company has its inventory system. Everything is delivered from their wirehouse. Here is the key point we started with. We see how local business chains are going to be affected by these online delivery systems, though they are providing the same rate of products. In many cases, we can see some product values are less than the online price. Another thing is that for almost every order we have to wait for a certain time.

So two things are going to be the backbone of our application. 1. **Save the local business chain to prevent the social-economic imbalance**. 2. **Provide super fast delivery solutions for family essentials**.

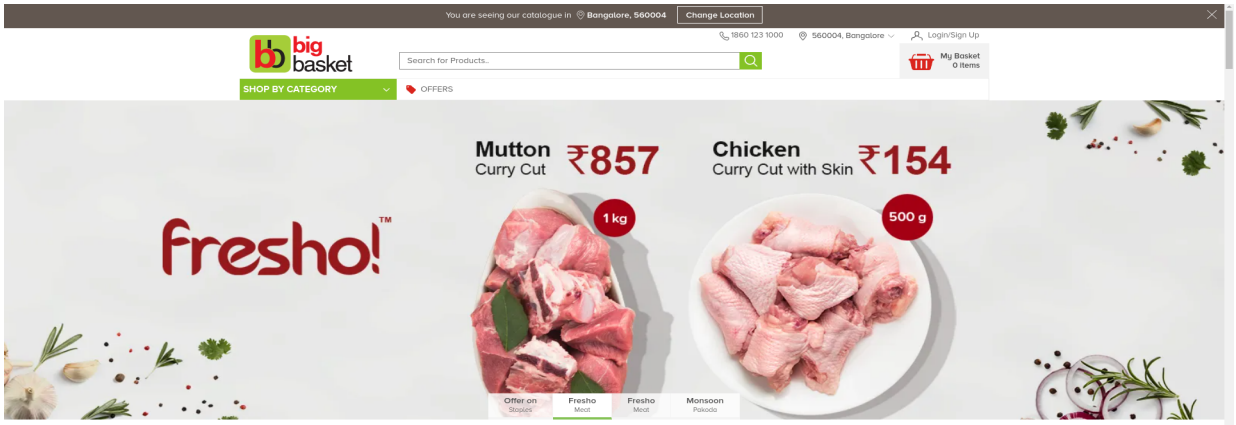
Literature Survey

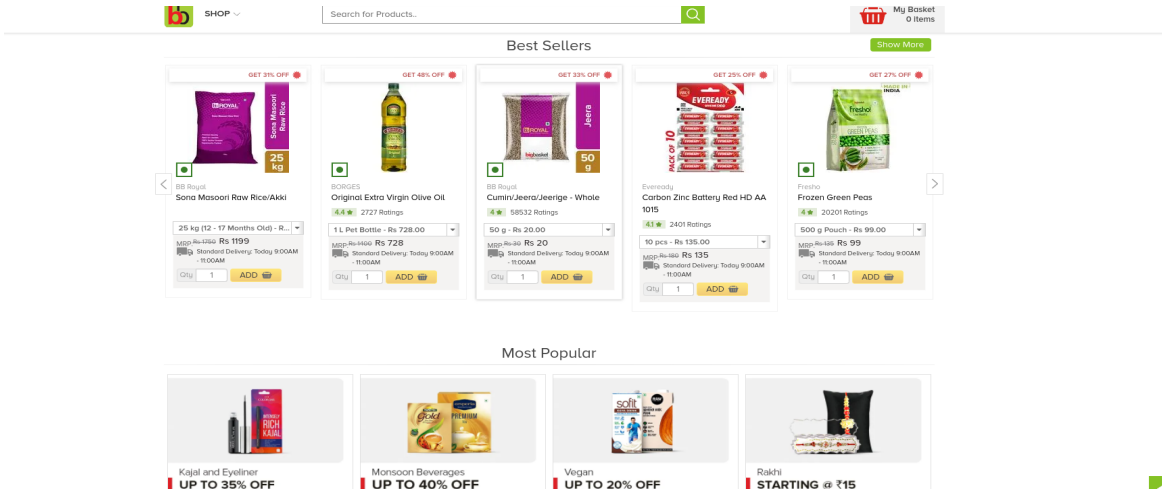
Initially when I started the project it was too hard to understand how to move this project. That's why I reviewed some popular online grocery platforms.Before discussing the details of this application process, let us discuss some worldwide existing giant online delivery apps. The most popular applications in the family essentials fields in India are named below.

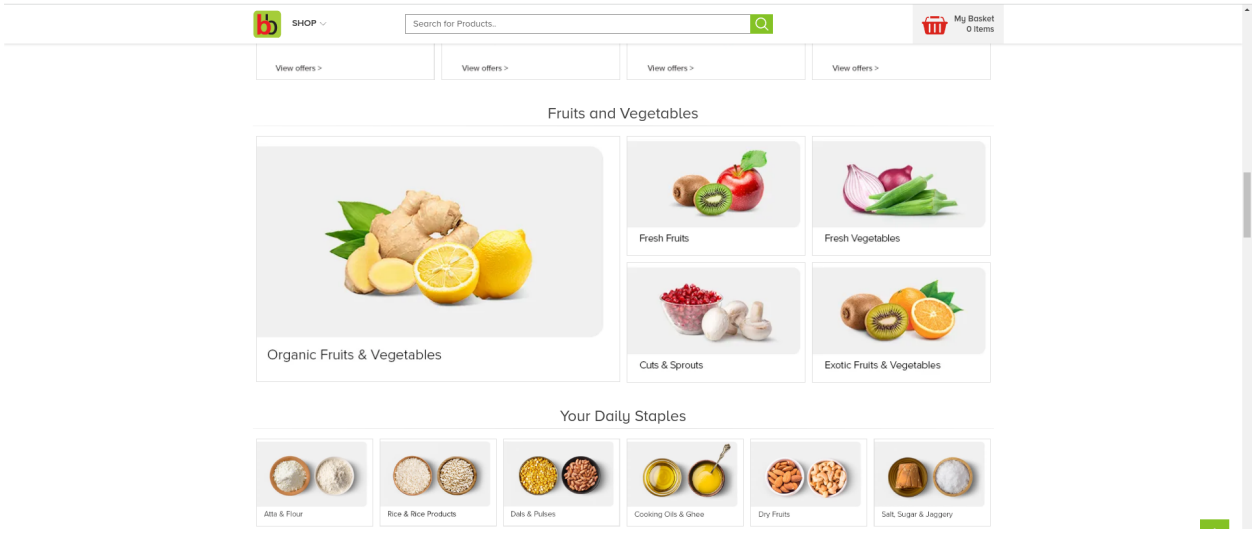
1. BigBasket
2. Blinkit
3. ZopNow
4. Amazon Fresh
5. Flipkart Grocery
6. Nature's Basket
7. Spencer's Online Grocery
8. Paytm Mall
9. DMart Online Grocery Shopping
10. Reliance Smart

1. Bigbasket

BigBasket.com is the India' largest online food and grocery store with over 18,000 products and over 1000 brands in the catalog where one can find everything they are looking for- fresh fruits to vegetables, rice to dals, spices to seasonings or any packaged products, beverages, personal care products, and meats and many more. They offer a wide range of options in every category, handpicked and the best quality available at the lowest prices to patrons in major cities of India.







Business Model of BigBasket.com

When BigBasket started its operations, it adopted a “purchased-to-order” model. In this model, the delivery boy directly picks up the item from the retailer, and directly delivers it to the customer. As the time passed, business continued to achieve success. Later they adopted an ‘inventory model’ for most of their products, and BigBasket now buys products directly from suppliers and keeps stocks in their warehouses. BigBasket sources many perishable food products like fruits and vegetables from the mandis and other places.

To serve the hyper-local sentiment of the people, BigBasket has been implementing hyper-local strategies and has joined hands with more than 1,800 neighborhood grocery stores across India to deliver goods within an hour. Recently the online grocery platform has been funded by Alibaba and has tried to add in more assets in the form of a more efficient delivery fleet.

BigBasket has launched three new segments in the previous two years:

a. **BB Daily:** It is a subscription-based platform for milk, fruits and vegetables, bread, dairy, eggs, breakfast cereals, tender coconut, etc. for every day or alternate days.

b. **BB Instant:** It ensures that you get your basic needs covered 24*7.

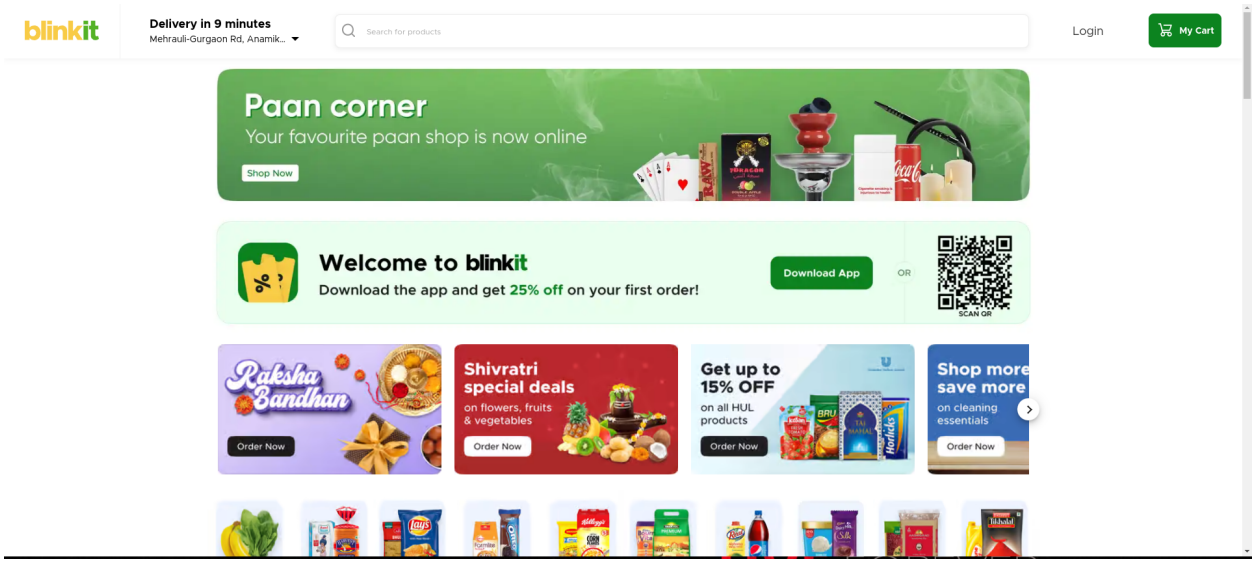
c. **BB Beauty**: It takes care of men and women’s beauty and personal care products at one place.

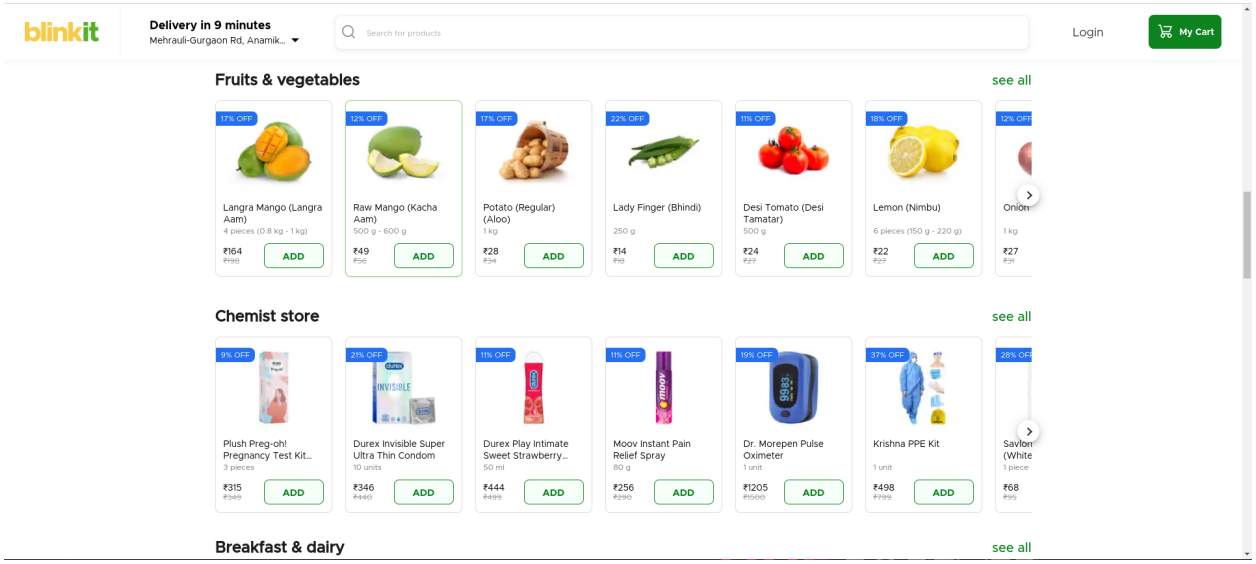
2.Grofers(blinkit)

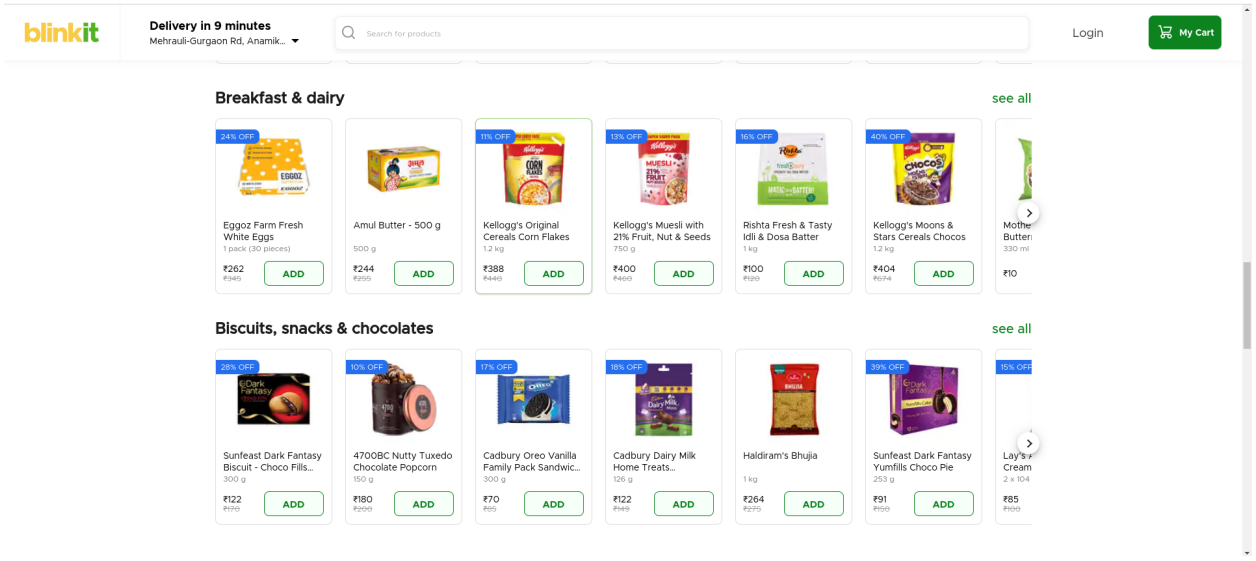
Grofers is an Indian on-demand grocery delivery service. The company was launched in 2013 and is based out of Gurugram. As of 2019, the company has raised about $643.52 million across all four tranches of the Series F round, which was organized in the same year. The investors of the company are Tiger Global, SoftBank, and Sequoia Capital.

The company name “Grofers” is a portmanteau of the term “grocery gophers.” Customers use mobile apps and websites to place grocery orders online. Grofers employees check their warehouse to secure the requested items and deliver them to customers’ doorsteps. Customers can schedule the delivery as per their convenience at any time from the platform while ordering the products or items they require.

Grofers presently operates in more than 29 cities, including Delhi, Pune, Mumbai, Jaipur, Bangalore, Kolkata, Chennai, Hyderabad, Ahmedabad, Gurugram, Lucknow, Aligarh, Agra, Kanpur, Noida, Ghaziabad, Rohtak, Faridabad, Meerut, Sonipat, Guwahati, Bhiwadi, Moradabad, Mathura, Vadodara, Panipat, Durgapur, Allahabad, and Modinagar. During the Covid-19 outbreak, Grofers and Big Basket, and Amazon continued to operate in India. The online grocery delivery platforms efficiently cater to customers' needs during the tough time of social distancing and disease spread.







Revenue Model (How Grofers Make Money)

Inventory based business model of Grofers helps the company to provide outstanding delivery service to end-users. The process involves technology, partners, suppliers, alliance partners, etc. The company offers more than thousands of products such as bakery products, meats, flower delivery, etc.

All the items are stored at Grofers' warehouse; the company uses its infrastructure to deliver and connect with an end number of brands.

Grofers charges around 8 to 15% on every order placed using the platform.
Recently, Grofers has invested in its original labels ensuring its active participation in the online ordering and delivery market. Its private brand accounts for around 20% of the total revenue generated.

Grofers changed its brand name to Blinkit **in line with its vision to embrace quick-commerce.**

3.**Amazon Fresh**

Amazon Fresh is a grocery delivery service available to Amazon Prime and Amazon Prime Student members in select cities.

Note that even if you're a current Prime member, you do not automatically have access to Amazon Fresh. Members need to receive or request an invitation before they can use the service.

Once you have access to Amazon Fresh, simply sign in to your account in the app or online and start adding items to your shopping cart.

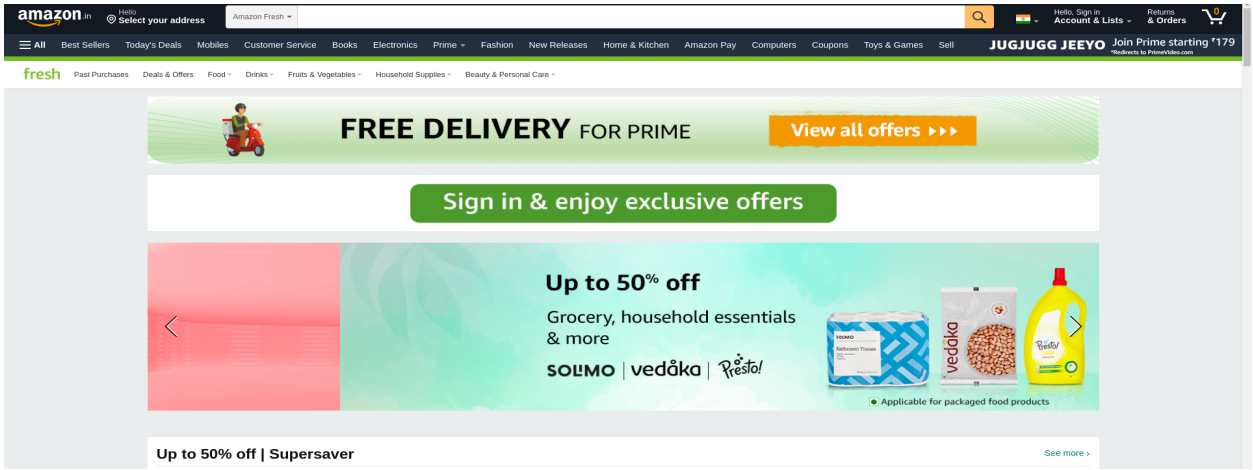
Shoppers have access to over 500,000 products, including fresh produce, meat and seafood, dairy products, canned and prepackaged goods, and even home essentials like toilet paper and cleaning supplies.

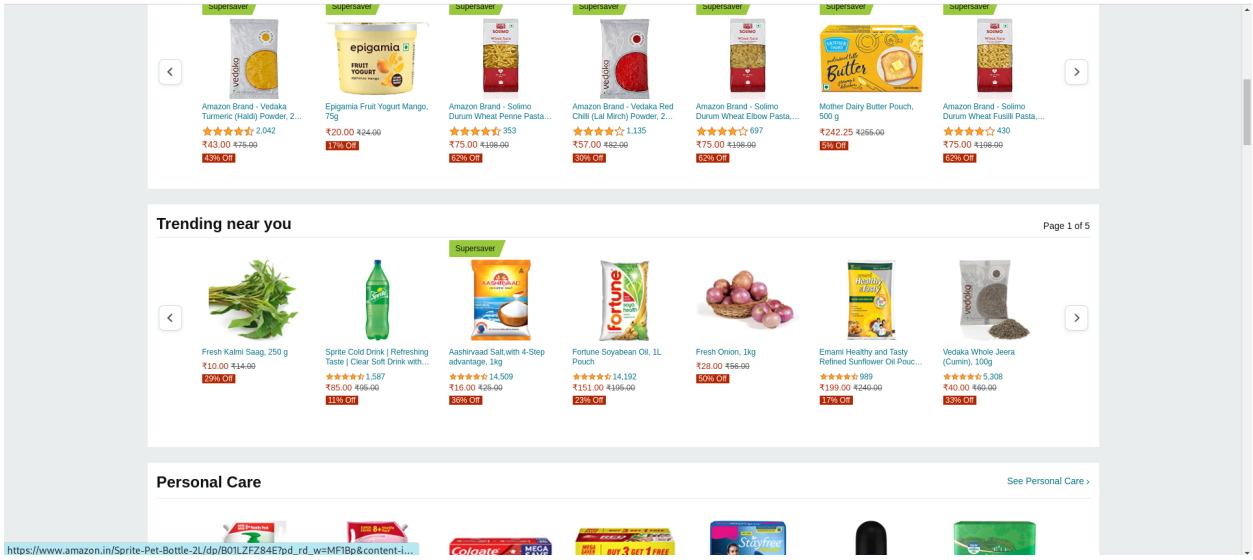
When you checkout, you choose a date and 2-hour delivery window. Amazon Fresh then packages items at an Amazon fulfillment center and delivers them directly to your door. A pickup option may also be available, depending on the region.

Through this discussion we can conclude that these big tycoons send their products from their warehouse at their own pace.

Hence we can say that the growth of local business chains is getting hampered.

Our project's motivation is to grow the local business chains, that is why we are trying to provide essential products required by the customer directly from the local market and deliver it in minimum required time.





4.Flipkart Grocery

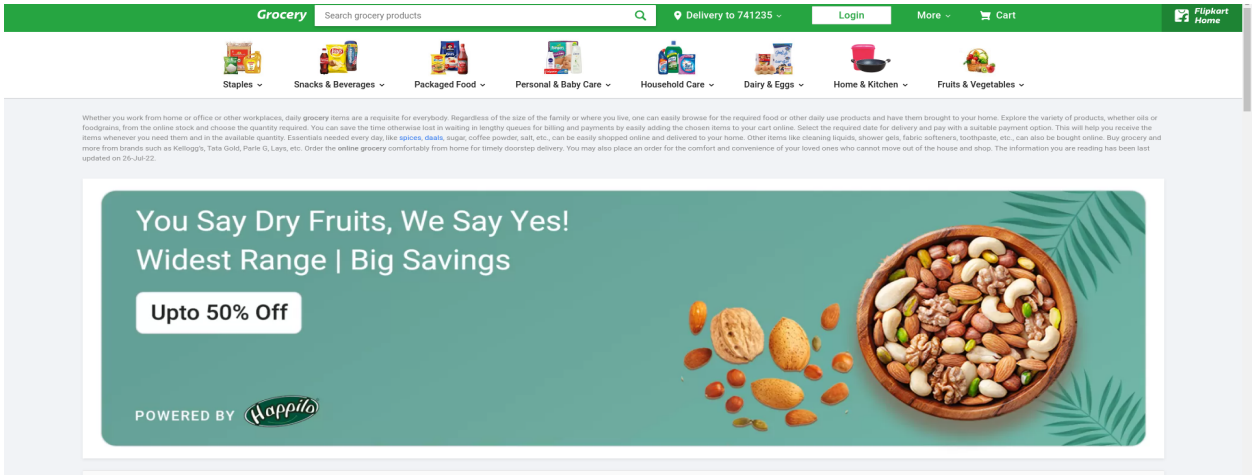
Flipkart, one of India's largest e-commerce platforms, quietly dived into the grocery market once again with the launch of Supermart in Bengaluru. The new initiative looks to take on the likes of Amazon India and grocery delivery app BigBasket, to name a few. The launch of Supermart comes following a pilot project that began a few months ago.

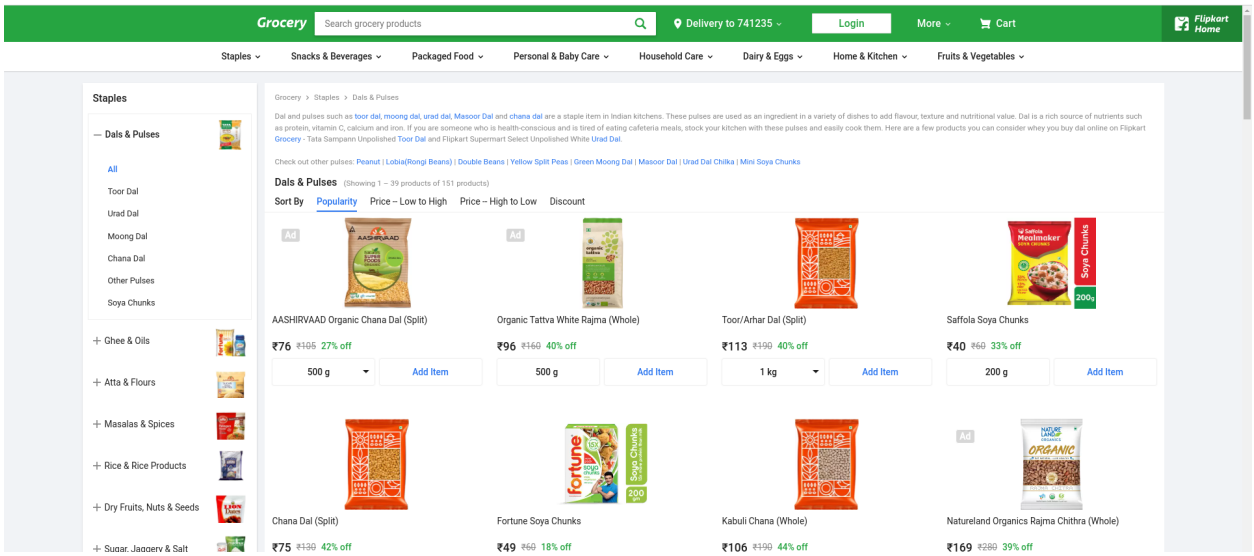
Economic Times reports that the new Supermart section in the Flipkart app lets you order groceries for a minimum of Rs 500 and will offer free of cost delivery for orders above Rs. 1000. "We have done a soft launch for grocery categories at Flipkart. Currently, it is available to select customers in Bengaluru. In line with our aim of transforming ecommerce in the country through technology, we want to make shopping for everyday essentials convenient for our customers, we intend to scale it up to all customers in Bengaluru and take it to other cities in future," a Flipkart spokesperson told Economic Times.

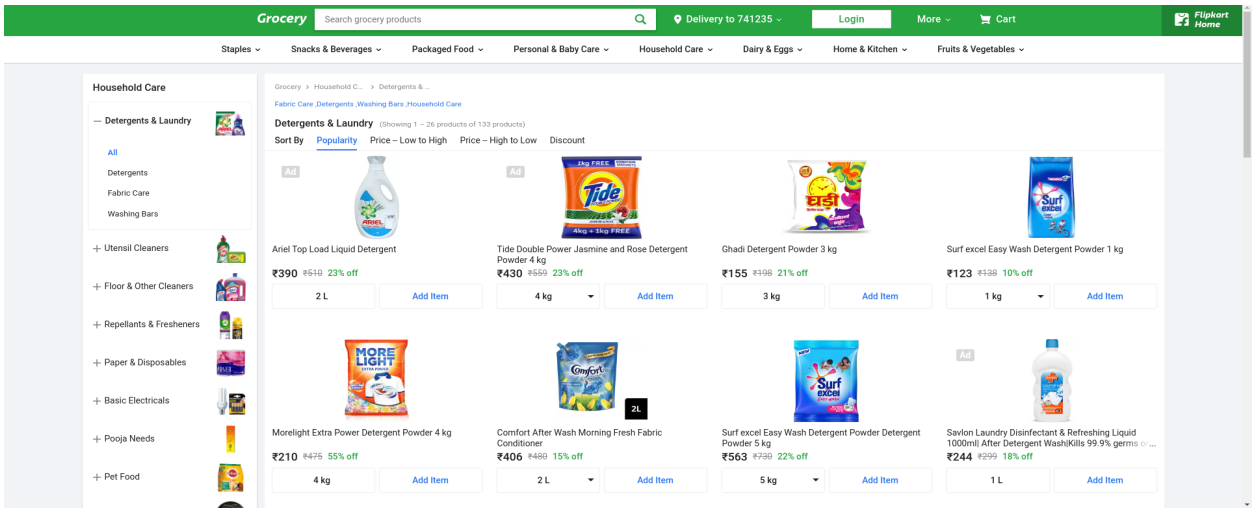
Flipkart's grocery delivery service comes a year after Amazon India launched the Amazon Now app that guarantees delivery in two hours. Amazon's service is available in Bengaluru as well as in Delhi and Mumbai. The company also has a service called Amazon Pantry that offers next-day delivery for groceries. Meanwhile, Paytm Mall along with its investor Alibaba is also planning to expand into the grocery segment and is in talks to buy a large stake in BigBasket. These companies see a great potential in the grocery delivery segment in India that has already proved fruitful for companies like BigBasket and Food Panda. However, apps like Nearby and Grofers in the past have proved that competing in the segment won't be easy.

Notably, Flipkart has tried once before to enter the grocery delivery market with the Nearby app, which was launched as a pilot in Bengaluru. However, the service shut down a few months later early last year as the app was unable to attract users due to poor marketing. From what we understand, Supermart will not be a standalone app and will be a service offered within the Flipkart app. The company will hope the grocery service works this time around and baking the service into Flipkart's own app will likely prove more successful as far as reaching the masses is concerned.

As of now, Flipkart's grocery marketplace is only available in Bengaluru with no word yet on when the company plans to expand to other cities.

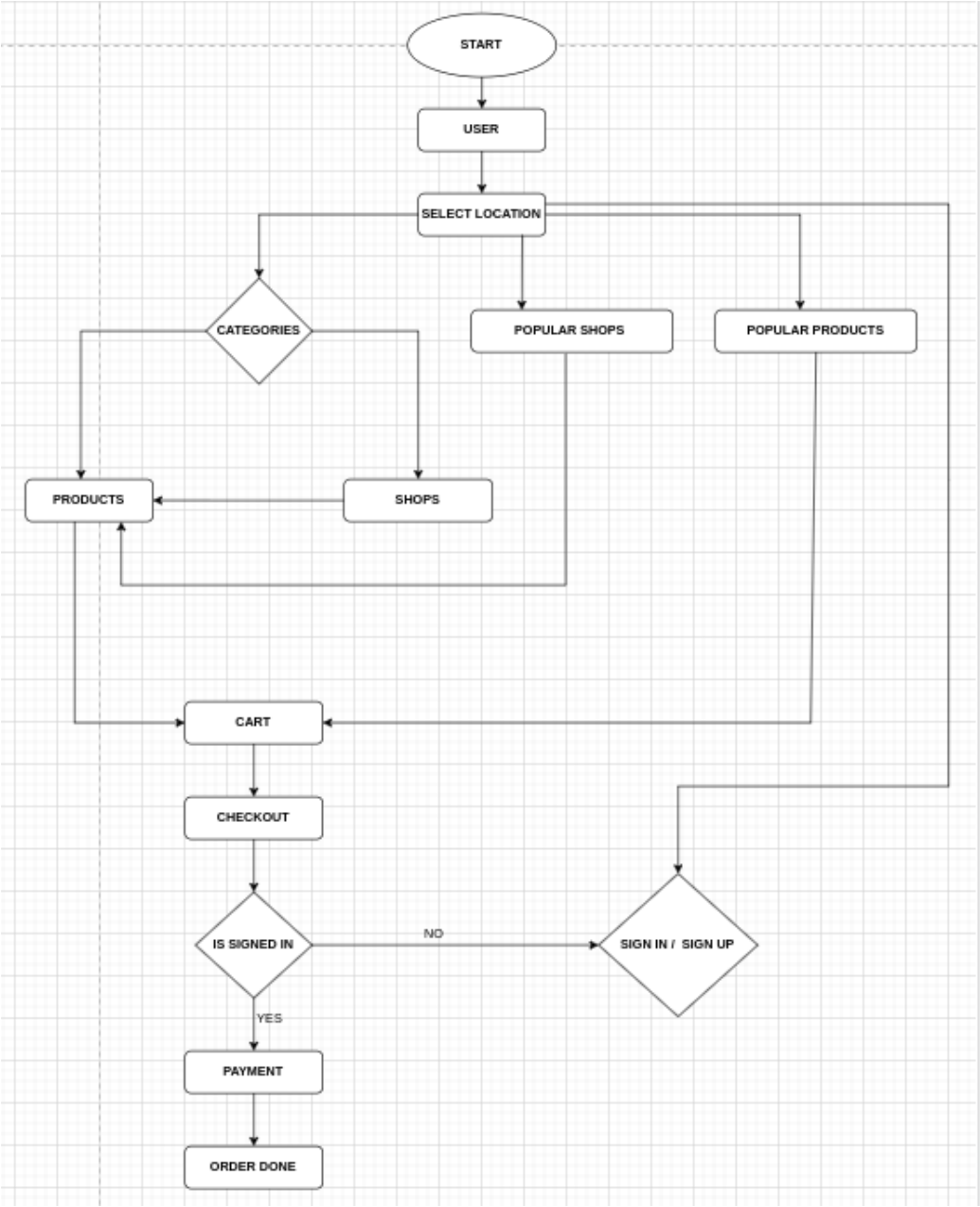

Grocery
Search grocery products
Delivery to 741235
Login
More
Cart
Flipkart Home
Staples
Snacks & Beverages
Packaged Food
Personal & Baby Care
Household Care
Dairy & Eggs
Home & Kitchen
Fruits & Vegetables
You Say Dry Fruits, We Say Yes!
Widest Range | Big Savings
Upto 50% Off
POWERED BY Happilo


Grocery
Search grocery products
Delivery to 741235
Login
More
Cart
Flipkart Home
Staples
Snacks & Beverages
Packaged Food
Personal & Baby Care
Household Care
Dairy & Eggs
Home & Kitchen
Fruits & Vegetables
Staples
Dals & Pulses
All
Toor Dal
Urad Dal
Moong Dal
Chana Dal
Other Pulses
Soya Chunks
Ghee & Oils
Atta & Flours
Masalas & Spices
Rice & Rice Products
Dry Fruits, Nuts & Seeds
Grocery > Staples > Dals & Pulses
Dals & Pulses (Showing 1 – 39 products of 151 products)
Sort By Popularity Price – Low to High Price – High to Low Discount
AASHIRVAAD Organic Chana Dal (Split)
₹76 ₹105 27% off
500 g Add Item
Organic Tattva White Rajma (Whole)
₹96 ₹160 40% off
500 g Add Item
Toor/Arhar Dal (Split)
₹113 ₹190 40% off
1 kg Add Item
Saffola Soya Chunks
₹40 ₹60 33% off
200 g Add Item
Chana Dal (Split)
₹75 ₹130 42% off
Fortune Soya Chunks
₹49 ₹60 18% off
Kabuli Chana (Whole)
₹106 ₹190 44% off
Natureland Organics Rajma Chithra (Whole)
₹169 ₹280 39% off

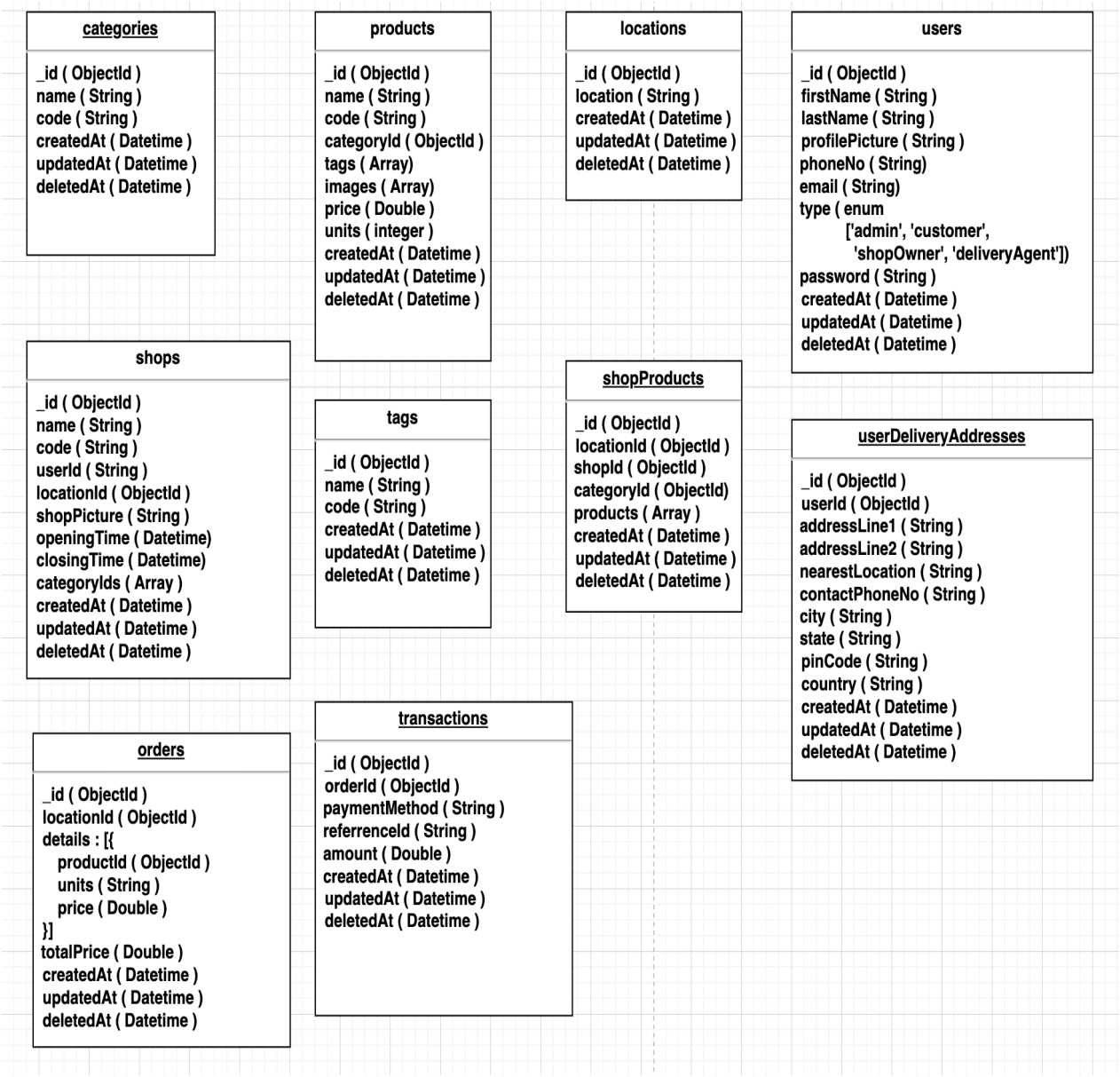

Grocery
Search grocery products
Delivery to 741235
Login
More
Cart
Flipkart Home
Staples
Snacks & Beverages
Packaged Food
Personal & Baby Care
Household Care
Dairy & Eggs
Home & Kitchen
Fruits & Vegetables
Household Care
Detergents & Laundry
All
Detergents
Fabric Care
Washing Bars
Utensil Cleaners
Floor & Other Cleaners
Repellants & Fresheners
Paper & Disposables
Basic Electricals
Pooja Needs
Pet Food
Fabric Care ,Detergents ,Washing Bars ,Household Care
Detergents & Laundry (Showing 1 – 26 products of 133 products)
Sort By Popularity Price – Low to High Price – High to Low Discount
Ariel Top Load Liquid Detergent
₹390 ₹510 23% off
2 L Add Item
Tide Double Power Jasmine and Rose Detergent Powder 4 kg
₹430 ₹559 23% off
4 kg Add Item
Ghadi Detergent Powder 3 kg
₹155 ₹198 21% off
3 kg Add Item
Surf excel Easy Wash Detergent Powder 1 kg
₹123 ₹138 10% off
1 kg Add Item
Morelight Extra Power Detergent Powder 4 kg
₹210 ₹475 55% off
4 kg Add Item
Comfort After Wash Morning Fresh Fabric Conditioner
₹406 ₹480 15% off
2 L Add Item
Surf excel Easy Wash Detergent Powder Detergent Powder 5 kg
₹563 ₹730 22% off
5 kg Add Item
Savlon Laundry Disinfectant & Refreshing Liquid 1000ml| After Detergent Wash|Kills 99.9% germs
₹244 ₹299 18% off
1 L Add Item

ROAD MAP

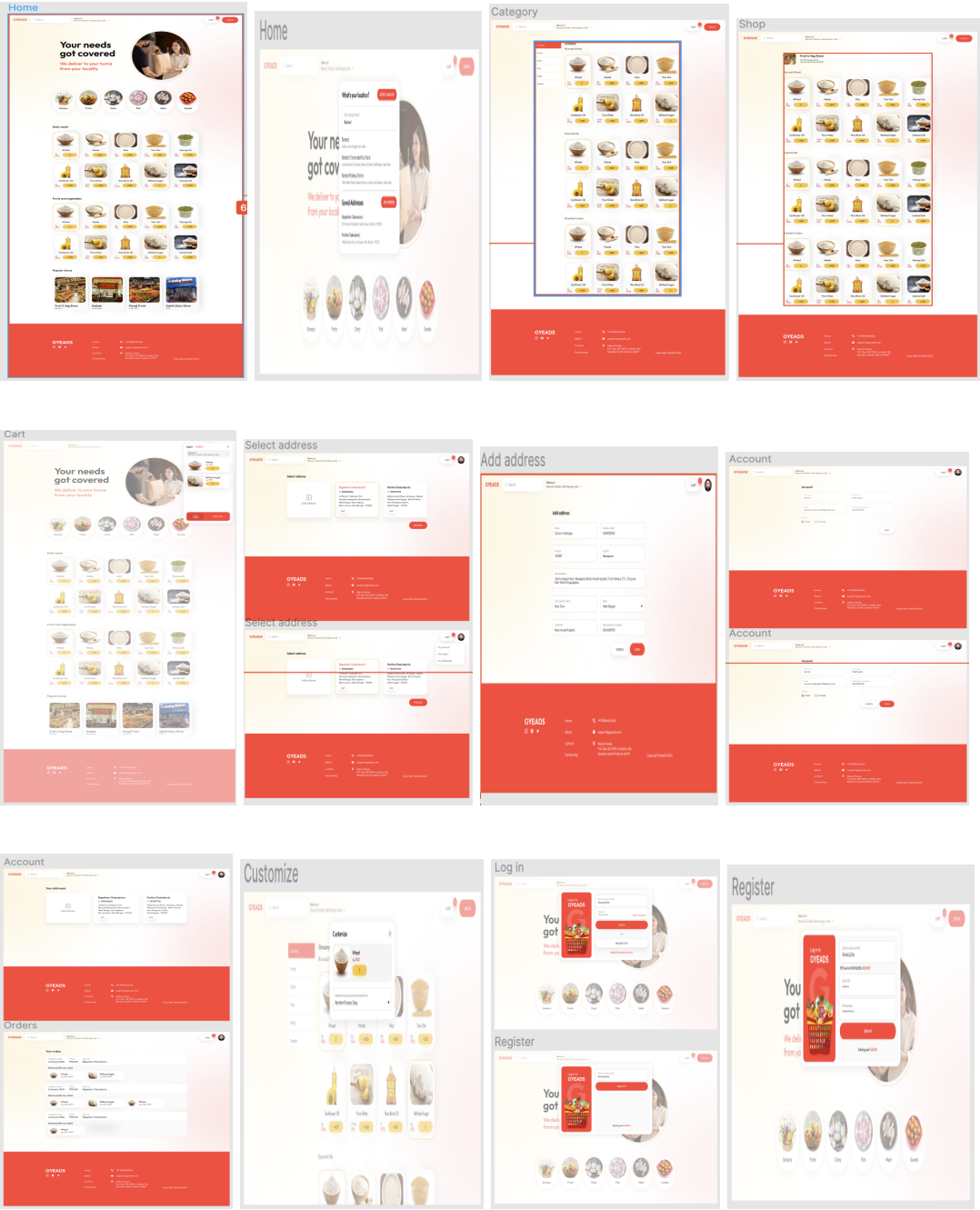
1.FLOWCHART



2.DATABASE DESIGN



3.DESIGN MOCK-UP



4.DOMAIN & SERVER SETUP

When referring to an Internet address or name, a domain or domain name is the location of a website. For example, the domain name "google.com" points to the IP address "216.58.216.164". Generally, it's easier to remember a name rather than a long string of numbers. A domain name contains a maximum of sixty three characters, with one character minimum, and is entered after the protocol in the URL, as shown in the following example.

gyeads.com is the domain which represents this application in the cloud.

AWS (Amazon Web Services) **VPC** (Virtual Private Cloud) will be used as a cloud server. That domain will be mapped into this **VPC.** Here amazon linux operating system will host the application. **Nginx** will be used to handle server requests. **Mongo atlas** cloud mongodb used as database provider.

(i) API (**Application Programming Interface)**

API development is the backbone of any application.APIs are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols. For example, the weather bureau's software system contains daily weather data. The weather app on your phone "talks" to this system via APIs and shows you daily weather updates on your phone.

API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses. Their API documentation contains information on how developers are to structure those requests and responses.

I'll use **express js (a node js library)** to create all the APIs of this application.

Node JS is a fast javascript runtime environment that we use to build server-side applications, but it does not know how to serve files, handle requests and handle HTTP method so this is where express JS comes in.

Express JS is a node js framework designed to build APIs web applications cross-platform mobile apps quickly and make node js easy.

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy. Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.

(ii) WEB

The Web is the common name for the World Wide Web, a subset of the Internet consisting of the pages that can be accessed by a Web browser. Many people assume that the Web is the same as the Internet, and use these terms interchangeably. However, the term Internet actually refers to the global network of servers that makes the information sharing that happens over the Web possible. So, although the Web does make up a large portion of the Internet, they are not one and the same.

Web pages are formatted in a language called Hypertext Markup Language (HTML). It is this language that allows users to click through pages on the Web via links. The Web uses HTTP protocol to transmit data and share information. Browsers such as Internet Explorer, Google Chrome or Mozilla Firefox are used to access Web documents, or Web pages, which are connected via links.

The Web is just one of the ways that information is shared over the Internet; others include email, instant messaging and File Transfer Protocol (FTP).

Nextjs (Reactjs framework) will be used to create client side WEB applications.

React (also known as **React.js** or **ReactJS**) is a free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by **Meta** (formerly Facebook) and a community of individual developers and companies.React can be used as a base in the development of single-page, mobile, or server-rendered applications with frameworks like **Next.js**. However, React is only concerned with state management and rendering that state to the DOM, so creating

React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

Next.js is a React framework that enables several extra features, including server-side rendering and generating static websites.

In other words,**Next.js** is an open-source web development framework built on top of Node.js enabling React-based web applications functionalities such as server-side rendering and generating static websites.

Tailwind CSS framework will be used to meet the look and feel of this application.

Tailwind CSS is basically a utility-first CSS framework for rapidly building custom user interfaces. It is a highly customizable, low-level CSS framework that gives you all of the building blocks you need to build bespoke designs without any annoying opinionated styles you have to fight to override.

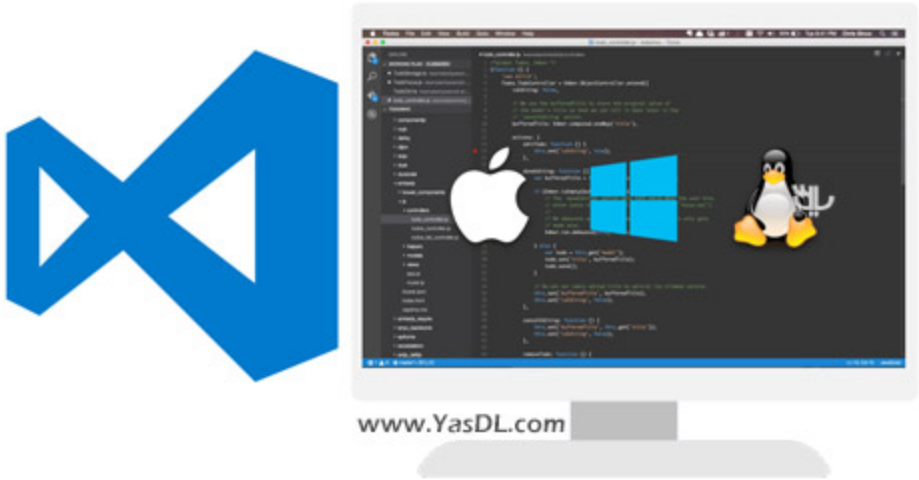
5.SOFTWARE REQUIREMENTS

(i) Visual Studio Code

Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging.

First and foremost, it is an editor that gets out of your way. The delightfully frictionless edit-build-debug cycle means less time fiddling with your environment, and more time executing on your ideas.

Visual Studio Code supports macOS, Linux, and Windows - so you can hit the ground running, no matter the platform.



At its heart, Visual Studio Code features a lightning fast source code editor, perfect for day-to-day use. With support for hundreds of languages, VS Code helps you be instantly productive with syntax highlighting, bracket-matching, auto-indentation, box-selection, snippets, and more. Intuitive keyboard shortcuts, easy customization and community-contributed keyboard shortcut mappings let you navigate your code with ease.
For serious coding, you'll often benefit from tools with more code understanding than just blocks of text. Visual Studio Code includes built-in support for IntelliSense code completion, rich semantic code understanding and navigation, and code refactoring.

And when the coding gets tough, the tough get debugging. Debugging is often the one feature that developers miss most in a leaner coding experience, so we made it happen. Visual Studio Code includes an interactive debugger, so you can step through source code, inspect variables, view call stacks, and execute commands in the console.

VS Code also integrates with build and scripting tools to perform common tasks making everyday workflows faster. VS Code has support for Git so you can work with source control without leaving the editor including viewing pending changes diffs.

Customize every feature to your liking and install any number of third-party extensions. While most scenarios work "out of the box" with no configuration, VS Code also grows with you, and we encourage you to optimize your experience to suit your unique needs. VS Code is an open-source project so you can also contribute to the growing and vibrant community on GitHub.

(ii) Robo 3T

Robo 3T is a free and lightweight GUI for MongoDB. It is a MongoDB management tool that has a shell-centric cross-platform and is supported by JSON i.e. JavaScript Object Notation. This tool is not typical of MongoDB's other administrative tools of user Interface

i.e. its shell could get embedded in Mongo Shell with a whole lot of access in both Mongo CLI and Mongo GUI.

With the help of this mongo shell, a user could view, edit and delete mongo documents. Moreover, Robo 3T is a volunteer open-source project and it's totally free of cost to the public.

It could be re-disseminated and could get re-modified as well by following the TOS of General public license version 3, which has been published by Free Software Foundation.

This software has been promulgated and could be redistributed for the purpose of helping people who could get assistance from it, that's why it bears no Warranty of wholesaling it, as per the rules by GNU.

For more information about GNU, you can check out GNU Licenses

Robo 3T

mongodb

Robo 3T is a free and machine friendly software that uses a small number of resources available on a machine. It's highly appreciated and recognized as the world-famous project with the highest success ratio in giving prime output.

Above all, by Robo 3T, the user doesn't have to go through the messy procedure of using tables and rows, which is typically used in relational databases. Unlike them, it is built on architecture Mongo collections and Mongo documents.

(iii) Postman testing Tool

Postman Testing tool is an impressive tool useful to test web services.

A lot of people use the postman tool for api testing for better results.

This tool facilitates you to send and receive a request from a service.

To receive a response from the service, framework building is not necessary while using this tool.

Know what is API testing before starting the tool.

Requests are substantially created as per the API specifications.

Several response parameters are prosecuted like the headers, status code and the response body which helps in getting API endpoints.

Postman creators have also added a feature called Jetpacks, which consists of a Javascript testing library and also some automation tools.

This tool is extremely valuable for developers and functional testers as it enhances the testing features of the application.

Features of Postman

Let's walk you through some of the features of Postman which makes it so beneficial:

1.User-Friendly

Having a user-friendly interface enables one to send a simple request by only entering it, filling in the necessary headers, selecting the HTTP method and pressing the 'send' button.

If you are new to writing scripts, Postman even provides you with code snippets.

These code snippets consist of examples of validations for response code, response time, etc.

2.Automation

Automated tests can be easily set up using the Postman API testing tool.

You can automate your test cases in Postman in javascript with impressive features like write test suites, build requests that consist of dynamic parameters, pass on data in between the request, etc.

In this way tests and requests can be aggregated into a single automated test sequence.

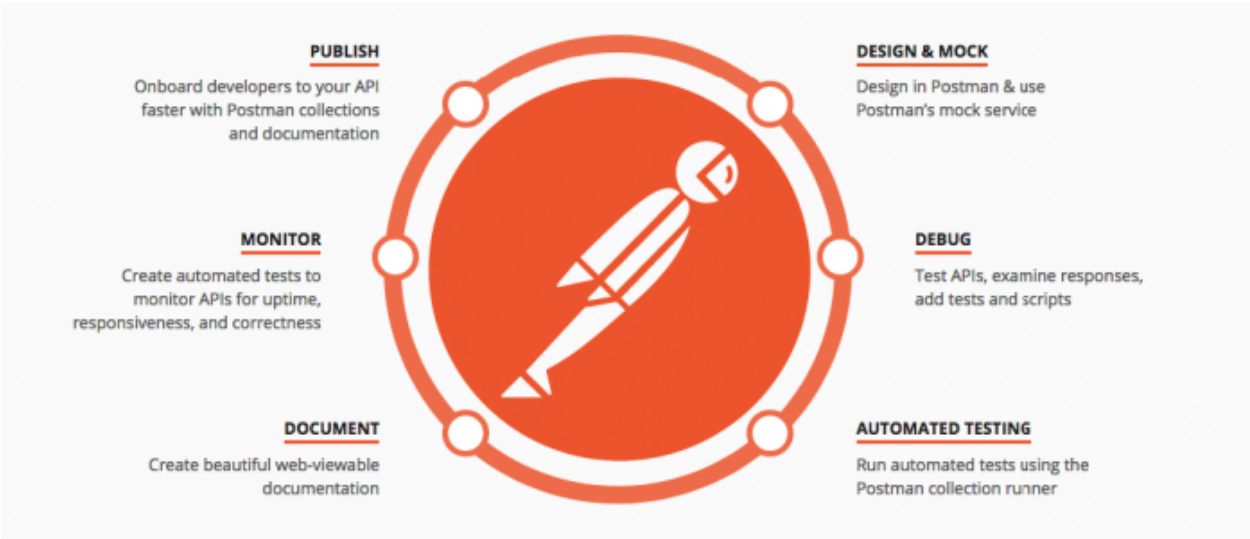
3.Cost Effective

Apart from the many benefits which this tool provides us, this tool is also very cost effective.

You can use the Postman in their free plan, which consists of limited features but most helpful.

The paid version of Postman with high end features are also very affordable and cost effective.

This feature makes this tool light on our pockets.



Postman for API testing

Postman provides you with a lot of features which makes it the perfect choice for usage.

To ensure that the API is working as expected, Postman allows you to create a collection of integration tests.

The right environment configuration is automatically inserted into your test and also different environments specific information can be stored in Postman.

You can also store previous tests data into global variables. These variables can be precisely used as environmental variables.

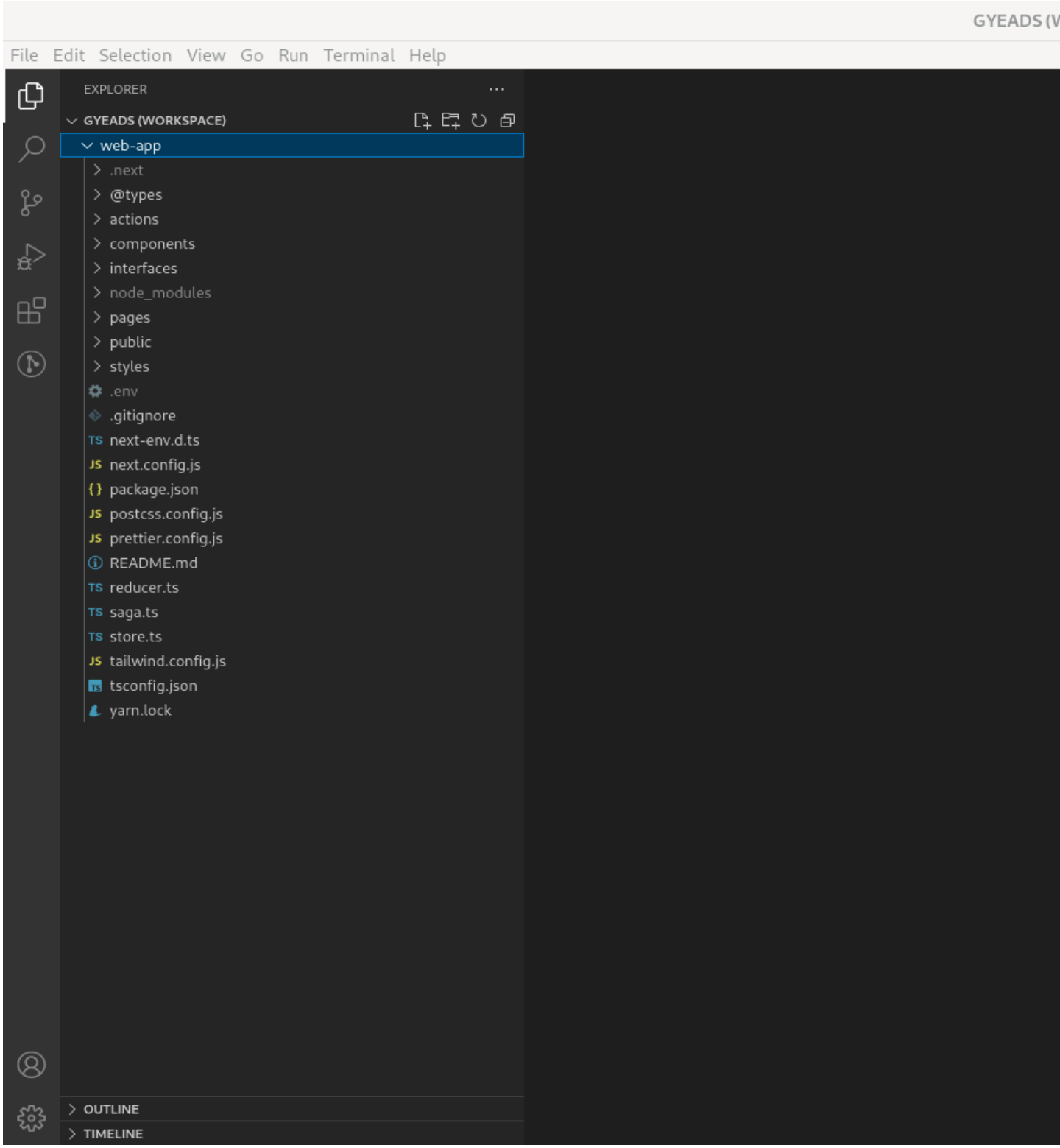
Postman's command-line interface, Newman, makes it accessible to run a set of tests right from the command line.

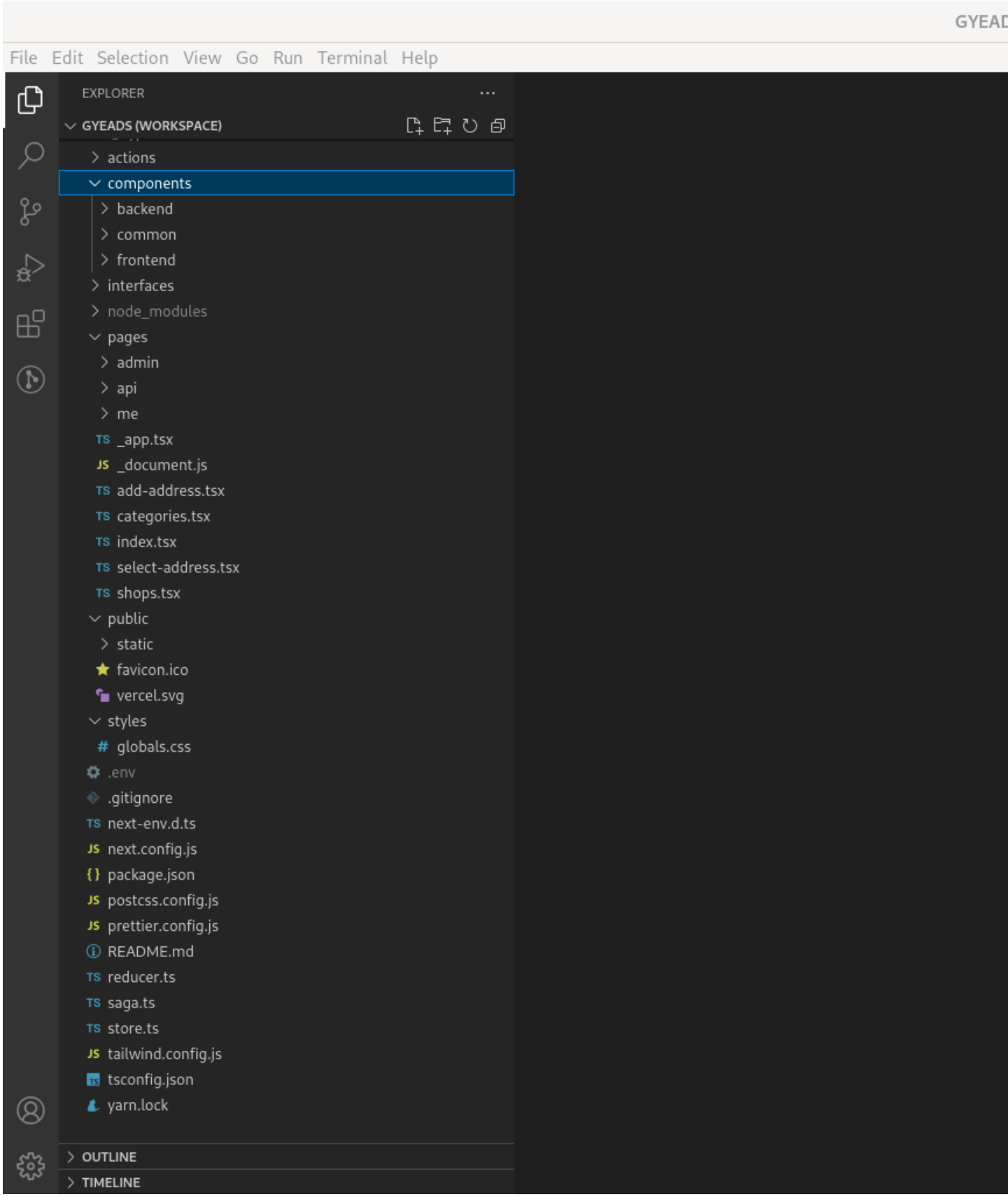
It is easy to move and share tests into different environments and tests using Postman.

This can be done by encapsulating collections in an "environmental configuration" or a "test" file.

CODING DETAILS

WEB app file and folder structures are mentioned below:




GYEAD
File Edit Selection View Go Run Terminal Help
EXPLORER
GYEADS (WORKSPACE)
actions
components
backend
common
frontend
interfaces
node_modules
pages
admin
api
me
_app.tsx
_document.js
add-address.tsx
categories.tsx
index.tsx
select-address.tsx
shops.tsx
public
static
favicon.ico
vercel.svg
styles
globals.css
.env
.gitignore
next-env.d.ts
next.config.js
package.json
postcss.config.js
prettier.config.js
README.md
reducer.ts
saga.ts
store.ts
tailwind.config.js
tsconfig.json
yarn.lock
OUTLINE
TIMELINE

Few files code sample are mentioned below:

Index.tsx (WEB Application entry point)

```
import { useState } from "react";
import type { NextPage } from "next";
import Banner from "../components/frontend/Banner";
import Categories from "../components/frontend/Categories";
import Footer from "../components/frontend/Footer";
import Header from "../components/frontend/Header";
import ProductTile from "../components/frontend/ProductTile";
import ShopTile from "../components/frontend/ShopTile";

const Home: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);
  return (
    <>
      <Header setScrollOff={setScrollOff} />

      <div className="container mx-auto py-10 px-8 xl:px-48">
        <Banner />
        <Categories />

        <p className="py-6 text-3xl">Daily needs</p>
        <div className="mb-24 grid grid-cols-2 gap-8 md:grid-cols-3 lg:grid-cols-4
xl:grid-cols-5">
          <ProductTile />
        </div>

        <p className="py-6 text-3xl">Popular shops</p>

        <div className="grid grid-cols-1 gap-8 sm:grid-cols-2 md:grid-cols-4">
          <ShopTile />
        </div>
      </div>

      <Footer />
    </>
  );
};

export default Home;
```

Category module

 # categories/index

```
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

import AdminLayout from "../../../components/backend/AdminLayout";
import List from "../../../components/backend/categories/List";

const { publicRuntimeConfig } = getConfig();

const pageTitle = "Categories";
const pageEntity = "categories";
const pageUrlParam = "categories";

const Index: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);

  return (
    <>
      <Head>
        <title>
          {publicRuntimeConfig.APP_TITLE + " | " + pageTitle}
        </title>
      </Head>
      <AdminLayout>
        <div className="lg:col-span-3">
          <p className="mb-6 text-3xl">{pageTitle}</p>

          <div className="mb-4 flex items-center justify-between">
            <p className="text-xl">List</p>

            <Link href={"/admin/" + pageUrlParam + "/create"}>
```

```
                <button className="rounded-full bg-[#FF4236] px-4 py-2 text-sm font-bold
text-white hover:bg-[#FF6536]">
                  Create
                </button>
              </Link>
            </div>

            <List />
          </div>
        </AdminLayout>
      </>
    );
};
export default Index;
```

 # admin/categories/create

```
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

import AdminLayout from "../../../components/backend/AdminLayout";
import CreateEditForm from "../../../components/backend/categories/CreateEditForm";

const { publicRuntimeConfig } = getConfig();

const pageTitle = "Categories";
const pageEntity = "categories";
const pageUrlParam = "categories";

const Create: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);

  return (
    <>
      <Head>
        <title>
          {publicRuntimeConfig.APP_TITLE + " | " + pageTitle}
```

```
            </title>
          </Head>
          <AdminLayout>
            <div className="lg:col-span-3">
              <p className="mb-6 text-3xl">{pageTitle}</p>

              <div className="mb-4 flex items-center justify-between">
                <p className="text-xl">Create</p>

                <Link href={"/admin/" + pageUrlParam}>
                  <button className="rounded-full bg-[#FF4236] px-6 py-2 text-sm font-bold
text-white drop-shadow-md hover:bg-[#FF6536]">
                    List
                  </button>
                </Link>
              </div>

              <CreateEditForm formType={"create"} />
            </div>
          </AdminLayout>
        </>
    );
};
export default Create;
```

 # categories/edit

```
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

import AdminLayout from "../../../../components/backend/AdminLayout";
import CreateEditForm from "../../../../components/backend/categories/CreateEditForm";

const { publicRuntimeConfig } = getConfig();

const pageTitle = "Categories";
```

```
const pageEntity = "categories";
const pageUrlParam = "categories";

const Edit: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);

  return (
    <>
      <Head>
        <title>
          {publicRuntimeConfig.APP_TITLE + " | " + pageTitle}
        </title>
      </Head>
      <AdminLayout>
        <div className="lg:col-span-3">
          <p className="mb-6 text-3xl">{pageTitle}</p>

          <div className="mb-4 flex items-center justify-between">
            <p className="text-xl">Edit</p>

            <div className="flex flex-row gap-2">
              <Link href={"/admin/" + pageUrlParam}>
                <button className="rounded-full bg-[#FF4236] px-8 py-2 text-sm font-bold
text-white drop-shadow-md hover:bg-[#FF6536]">
                  List
                </button>
              </Link>
              <Link href={"/admin/" + pageUrlParam}>
                <button className="rounded-full bg-[#FF4236] px-6 py-2 text-sm font-bold
text-white drop-shadow-md hover:bg-[#FF6536]">
                  Create
                </button>
              </Link>
            </div>
          </div>

          <CreateEditForm formType={"edit"} />
        </div>
      </AdminLayout>
    </>
  );
};
```

```
export default Edit;
```

Products module

```
# products/index
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

import AdminLayout from "../../../components/backend/AdminLayout";
import List from "../../../components/backend/products/List";

const { publicRuntimeConfig } = getConfig();

const pageTitle = "Products";
const pageEntity = "products";
const pageUrlParam = "products";

const Index: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);

  return (
    <>
      <Head>
        <title>
          {publicRuntimeConfig.APP_TITLE + " | " + pageTitle}
        </title>
      </Head>
      <AdminLayout>
        <div className="lg:col-span-3">
          <p className="mb-6 text-3xl">{pageTitle}</p>

          <div className="mb-4 flex items-center justify-between">
            <p className="text-xl">List</p>

            <Link href={"/admin/" + pageUrlParam + "/create"}>
```

```
                <button className="rounded-full bg-[#FF4236] px-4 py-2 text-sm font-bold
text-white hover:bg-[#FF6536]">
                  Create
                </button>
              </Link>
            </div>

            <List />
          </div>
        </AdminLayout>
      </>
    );
};
export default Index;
```

 # product/create

```
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

import AdminLayout from "../../../components/backend/AdminLayout";
import CreateEditForm from "../../../components/backend/products/CreateEditForm";

const { publicRuntimeConfig } = getConfig();

const pageTitle = "Products";
const pageEntity = "products";
const pageUrlParam = "products";

const Create: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);

  return (
    <>
      <Head>
        <title>
          {publicRuntimeConfig.APP_TITLE + " | " + pageTitle}
```

```
            </title>
          </Head>
          <AdminLayout>
            <div className="lg:col-span-3">
              <p className="mb-6 text-3xl">{pageTitle}</p>

              <div className="mb-4 flex items-center justify-between">
                <p className="text-xl">Create</p>

                <Link href={"/admin/" + pageUrlParam}>
                  <button className="rounded-full bg-[#FF4236] px-6 py-2 text-sm font-bold
text-white drop-shadow-md hover:bg-[#FF6536]">
                    List
                  </button>
                </Link>
              </div>

              <CreateEditForm formType={"create"} />
            </div>
          </AdminLayout>
        </>
    );
};
export default Create;
```

 # products/edit

```
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

import AdminLayout from "../../../../components/backend/AdminLayout";
import CreateEditForm from "../../../../components/backend/products/CreateEditForm";

const { publicRuntimeConfig } = getConfig();

const pageTitle = "Products";
const pageEntity = "products";
```

```
const pageUrlParam = "products";

const Edit: NextPage = () => {
  const [scrollOff, setScrollOff] = useState(false);

  return (
    <>
      <Head>
        <title>
          {publicRuntimeConfig.APP_TITLE + " | " + pageTitle}
        </title>
      </Head>
      <AdminLayout>
        <div className="lg:col-span-3">
          <p className="mb-6 text-3xl">{pageTitle}</p>

          <div className="mb-4 flex items-center justify-between">
            <p className="text-xl">Edit</p>

            <div className="flex flex-row gap-2">
              <Link href={"/admin/" + pageUrlParam}>
                <button className="rounded-full bg-[#FF4236] px-8 py-2 text-sm font-bold
text-white drop-shadow-md hover:bg-[#FF6536]">
                  List
                </button>
              </Link>
              <Link href={"/admin/" + pageUrlParam}>
                <button className="rounded-full bg-[#FF4236] px-6 py-2 text-sm font-bold
text-white drop-shadow-md hover:bg-[#FF6536]">
                  Create
                </button>
              </Link>
            </div>
          </div>

          <CreateEditForm formType={"edit"} />
        </div>
      </AdminLayout>
    </>
  );
};
export default Edit;
```

```
# app/sidebar
import type { NextPage } from "next";
import Head from "next/head";
import Link from "next/link";
import getConfig from "next/config";

import { useEffect, useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { useRouter } from "next/router";

const Sidebar: React.FC = () => {
  const router = useRouter();
  const { asPath } = useRouter();

  const [scrollOff, setScrollOff] = useState(false);
  return (
    <>
      <div className="hidden divide-y lg:block">
        <Link href="/admin/dashboard">
          <button
            className={
              (asPath &&
                asPath.includes("dashboard") &&
                "w-full bg-[#FF4236] py-4 pl-4 text-left text-white first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05]") ||
              "w-full bg-[#FFFFFF] py-4 pl-4 text-left text-slate-800 first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05] hover:text-white"
            }
          >
            Dashboard
          </button>
        </Link>

        <Link href="/admin/categories">
          <button
            className={
              (asPath &&
                asPath.includes("categories") &&
                "w-full bg-[#FF4236] py-4 pl-4 text-left text-white first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05]") ||
```

```
                "w-full bg-[#FFFFFF] py-4 pl-4 text-left text-slate-800 first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05] hover:text-white"
              }
            >
              Categories
            </button>
          </Link>

          <Link href="/admin/products">
            <button
              className={
                (asPath &&
                  asPath.includes("products") &&
                  "w-full bg-[#FF4236] py-4 pl-4 text-left text-white first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05]") ||
                "w-full bg-[#FFFFFF] py-4 pl-4 text-left text-slate-800 first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05] hover:text-white"
              }
            >
              Products
            </button>
          </Link>

          <Link href="/admin/shops">
            <button
              className={
                (asPath &&
                  asPath.includes("shops") &&
                  "w-full bg-[#FF4236] py-4 pl-4 text-left text-white first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05]") ||
                "w-full bg-[#FFFFFF] py-4 pl-4 text-left text-slate-800 first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05] hover:text-white"
              }
            >
              Shops
            </button>
          </Link>

          <Link href="/admin/shopProducts">
            <button
              className={
                (asPath &&
```

```
                    asPath.includes("shopProducts") &&
                    "w-full bg-[#FF4236] py-4 pl-4 text-left text-white first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05]") ||
                  "w-full bg-[#FFFFFF] py-4 pl-4 text-left text-slate-800 first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05] hover:text-white"
                }
              >
                ShopProducts
              </button>
            </Link>

            <Link href="/admin/tags">
              <button
                className={
                  (asPath &&
                    asPath.includes("tags") &&
                    "w-full bg-[#FF4236] py-4 pl-4 text-left text-white first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05]") ||
                  "w-full bg-[#FFFFFF] py-4 pl-4 text-left text-slate-800 first:rounded-t-lg
last:rounded-b-lg hover:bg-[#E53B05] hover:text-white"
                }
              >
                Tags
              </button>
            </Link>
          </div>
        </>
    );
};
export default Sidebar;
```

 # app/login

```
import { useEffect, useState } from "react";
import type { NextPage } from "next";
import { Formik, Form, Field, ErrorMessage } from "formik";
import * as Yup from "yup";
import {
  ChevronDownIcon,
  ChevronRightIcon,
  XIcon
} from "@heroicons/react/solid";
```

```
// Props interface
interface Props {
   loginModalStatus: Boolean;
   setLoginModalStatus: Function;
}

const validationSchema = Yup.object({
   emailOrPhoneNumber: Yup.string().required().label("Phone Number"),
   password: Yup.string().required().label("Password")
});

interface FormValues {
   emailOrPhoneNumber: string;
   password: string;
}

const LoginModal: React.FC<Props> = ({
   loginModalStatus,
   setLoginModalStatus
}) => {
   const initialValues: FormValues = { emailOrPhoneNumber: "", password: "" };

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

   return (
      <div
         className={
            (loginModalStatus === true &&
               "fixed inset-0 z-40 flex items-start justify-center bg-white bg-opacity-50") ||
            "fixed inset-0 z-40 hidden items-start justify-end bg-white bg-opacity-50"
         }
         id="overlay"
      >
         <div className="m-16 w-full rounded-2xl bg-white p-16 py-2 px-3 text-gray-800
drop-shadow-2xl md:w-1/2">
            <div className="flex flex-row gap-4">
               <div className="hidden basis-1/4 rounded-2xl bg-[url('/static/assets/login.jpg')]
bg-cover bg-center py-16 text-center text-2xl text-white lg:block">
                  <p>Log in to</p>
                  <p className="text-4xl">GYEADS</p>
               </div>
```

```
            <div className="w-full rounded-2xl bg-gray-100 p-4 lg:basis-3/4">
              <div className="mb-2 flex justify-end">
                <XIcon
                  className="h-6 w-6 cursor-pointer rounded-full p-1 hover:bg-gray-300"
                  id="close-modal"
                  onClick={() => {
                    setLoginModalStatus(false);
                  }}
                />
              </div>

              <div className="mb-4 rounded-2xl border-2 border-gray-200 bg-white px-6 py-3">
                <label className="text-sm text-slate-400">
                  Email or Phone Number
                </label>
                <input
                  className="w-full bg-transparent py-2 text-base font-thin text-slate-800
placeholder:text-base placeholder:text-slate-500 focus:outline-none"
                  placeholder="Email or Phone Number"
                  type="text"
                  name="emailOrPhoneNumber"
                />
              </div>

              <div className="mb-4 rounded-2xl border-2 border-gray-200 bg-white px-6 py-3">
                <label className="text-sm text-slate-400">
                  Password
                </label>
                <input
                  className="w-full bg-transparent py-2 text-base font-thin text-slate-800
placeholder:text-base placeholder:text-slate-500 focus:outline-none"
                  placeholder="Password"
                  type="password"
                  name="password"
                />
              </div>

              <button
                type="submit"
                className="mb-4 w-full rounded-full bg-[#FF4236] p-4 text-white
drop-shadow-md duration-300 hover:bg-[#FF4236]"
              >
```

```
                SUBMIT
              </button>

              <p className="mb-4 text-center">OR</p>

              <button className="mb-4 w-full rounded-full bg-white p-4 text-slate-800
drop-shadow-md duration-300 hover:bg-gray-200">
                REQUEST OTP
              </button>

              <p className="mb-4 text-center">
                <a href="#" className="text-[#FF4236]">
                  CREATE NEW ACCOUNT
                </a>
              </p>
            </div>
          </div>
        </div>
      </div>
    );
};

export default LoginModal;
```

 # app/cart

```
import { useEffect, useState } from "react";
import type { NextPage } from "next";
import {
  ChevronDownIcon,
  ChevronRightIcon,
  XIcon
} from "@heroicons/react/solid";

// Props interface
// with openModal set to Boolean
interface Props {
  cartModalStatus: Boolean;
  setCartModalStatus: Function;
}

const CartModal: React.FC<Props> = ({
  cartModalStatus,
```

```
    setCartModalStatus
}) => {
    // const [openModal, setOpenModal] = useState(false);

    useEffect(() => {
        console.log("SAMPLE MODAL", cartModalStatus);
    }, [cartModalStatus]);

    return (
        <div
            className={
                (cartModalStatus === true &&
                    "fixed inset-0 z-40 flex items-start justify-end bg-white bg-opacity-50") ||
                "fixed inset-0 z-40 hidden items-start justify-end bg-white bg-opacity-50"
            }
            id="overlay"
        >
            <div className="m-5 max-w-sm rounded-2xl bg-white py-2 px-3 text-gray-800
drop-shadow-2xl">
                <div className="flex items-center justify-between">
                    <div className="flex justify-start gap-4">
                        <h4 className="text-xl">Cart</h4>
                        <span className="rounded-full bg-[#f0b6b1] px-3 py-1 text-sm text-[#FF4236]">
                            2 items
                        </span>
                    </div>
                    <XIcon
                        className="h-6 w-6 cursor-pointer rounded-full p-1 hover:bg-gray-300"
                        id="close-modal"
                        onClick={() => {
                            setCartModalStatus(false);
                        }}
                    />
                </div>

                <div className="mt-5 mb-5 rounded-xl bg-slate-100 p-2 drop-shadow-2xl">
                    <p className="hidden text-sm text-slate-900 lg:block">
                        Delivery At
                    </p>
                    <div className="flex flex-wrap items-center gap-1">
                        <p className="text-xs text-slate-500 md:text-sm">
                            Barasat, Kolkata, West Bengal, India
```

```
            </p>
            <ChevronDownIcon className="h-4 w-4 text-[#FF4236] lg:h-6 lg:w-6" />
          </div>
        </div>

        <div className="mt-5 mb-5 h-96 overflow-y-auto">
          <div className="flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
            <div className="shrink-0">
              <img
                className="w-18 h-20"
                src="/static/assets/rice.png"
                alt="Cat6"
              />
            </div>
            <div className="">
              <div className="text-lg font-medium text-black">
                Wheat
              </div>
              <div className="text-xs text-slate-500">
                1kg
                <span className="ml-2 text-xs font-bold text-red-600">
                  ₹40
                </span>
              </div>

              <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  +
                </button>
                <span className="px-3">3</span>
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  -
                </button>
              </div>
            </div>
          </div>

          <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
            <div className="shrink-0">
              <img
```

```
                    className="w-18 h-20"
                    src="/static/assets/flour.png"
                    alt="Cat6"
                  />
                </div>
                <div className="">
                  <div className="text-lg font-medium text-black">
                    Wheat
                  </div>
                  <div className="text-xs text-slate-500">
                    1kg
                    <span className="ml-2 text-xs font-bold text-red-600">
                      ₹40
                    </span>
                  </div>

                  <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
                    <button className="h-6 w-6 rounded-full bg-yellow-300">
                      +
                    </button>
                    <span className="px-3">3</span>
                    <button className="h-6 w-6 rounded-full bg-yellow-300">
                      -
                    </button>
                  </div>
                </div>
              </div>

              <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
                <div className="shrink-0">
                  <img
                    className="w-18 h-20"
                    src="/static/assets/toorDal.png"
                    alt="Cat6"
                  />
                </div>
                <div className="">
                  <div className="text-lg font-medium text-black">
                    Wheat
                  </div>
                  <div className="text-xs text-slate-500">
```

```
                1kg
                <span className="ml-2 text-xs font-bold text-red-600">
                  ₹40
                </span>
              </div>

              <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  +
                </button>
                <span className="px-3">3</span>
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  -
                </button>
              </div>
            </div>
          </div>

          <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
            <div className="shrink-0">
              <img
                className="w-18 h-20"
                src="/static/assets/moongDal.png"
                alt="Cat6"
              />
            </div>
            <div className="">
              <div className="text-lg font-medium text-black">
                Wheat
              </div>
              <div className="text-xs text-slate-500">
                1kg
                <span className="ml-2 text-xs font-bold text-red-600">
                  ₹40
                </span>
              </div>

              <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  +
                </button>
```

```
                <span className="px-3">3</span>
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  -
                </button>
              </div>
            </div>
          </div>

          <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
            <div className="shrink-0">
              <img
                className="w-18 h-20"
                src="/static/assets/flour.png"
                alt="Cat6"
              />
            </div>
            <div className="">
              <div className="text-lg font-medium text-black">
                Wheat
              </div>
              <div className="text-xs text-slate-500">
                1kg
                <span className="ml-2 text-xs font-bold text-red-600">
                  ₹40
                </span>
              </div>

              <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  +
                </button>
                <span className="px-3">3</span>
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  -
                </button>
              </div>
            </div>
          </div>

          <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
```

```
            <div className="shrink-0">
              <img
                className="w-18 h-20"
                src="/static/assets/flour.png"
                alt="Cat6"
              />
            </div>
            <div className="">
              <div className="text-lg font-medium text-black">
                Wheat
              </div>
              <div className="text-xs text-slate-500">
                1kg
                <span className="ml-2 text-xs font-bold text-red-600">
                  ₹40
                </span>
              </div>

              <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  +
                </button>
                <span className="px-3">3</span>
                <button className="h-6 w-6 rounded-full bg-yellow-300">
                  -
                </button>
              </div>
            </div>
          </div>

          <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
            <div className="shrink-0">
              <img
                className="w-18 h-20"
                src="/static/assets/flour.png"
                alt="Cat6"
              />
            </div>
            <div className="">
              <div className="text-lg font-medium text-black">
                Wheat
```

```
            </div>
            <div className="text-xs text-slate-500">
              1kg
              <span className="ml-2 text-xs font-bold text-red-600">
                ₹40
              </span>
            </div>

            <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
              <button className="h-6 w-6 rounded-full bg-yellow-300">
                +
              </button>
              <span className="px-3">3</span>
              <button className="h-6 w-6 rounded-full bg-yellow-300">
                -
              </button>
            </div>
          </div>
        </div>

        <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
          <div className="shrink-0">
            <img
              className="w-18 h-20"
              src="/static/assets/flour.png"
              alt="Cat6"
            />
          </div>
          <div className="">
            <div className="text-lg font-medium text-black">
              Wheat
            </div>
            <div className="text-xs text-slate-500">
              1kg
              <span className="ml-2 text-xs font-bold text-red-600">
                ₹40
              </span>
            </div>

            <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
              <button className="h-6 w-6 rounded-full bg-yellow-300">
```

```
                +
              </button>
              <span className="px-3">3</span>
              <button className="h-6 w-6 rounded-full bg-yellow-300">
                -
              </button>
            </div>
          </div>
        </div>

        <div className="mt-5 flex max-w-sm items-center space-x-4 rounded-xl bg-white p-4
drop-shadow-lg">
          <div className="shrink-0">
            <img
              className="w-18 h-20"
              src="/static/assets/flour.png"
              alt="Cat6"
            />
          </div>
          <div className="">
            <div className="text-lg font-medium text-black">
              Wheat
            </div>
            <div className="text-xs text-slate-500">
              1kg
              <span className="ml-2 text-xs font-bold text-red-600">
                ₹40
              </span>
            </div>

            <div className="mt-1 rounded-full bg-yellow-500 px-1 py-1">
              <button className="h-6 w-6 rounded-full bg-yellow-300">
                +
              </button>
              <span className="px-3">3</span>
              <button className="h-6 w-6 rounded-full bg-yellow-300">
                -
              </button>
            </div>
          </div>
        </div>
      </div>
```

```
            <div className="mt-5 flex items-center justify-between rounded-xl bg-[#FF4236] p-2
text-white drop-shadow-2xl">
                <div className="rounded-xl bg-[#E53B31] px-8 py-2 ">
                    <p>2 items</p>
                    <p>₹ 190</p>
                </div>
                <div className="flex items-center justify-between">
                    <p>Checkout </p>
                    <ChevronRightIcon className="h-7 w-7 text-white" />
                </div>
            </div>
        </div>
    </div>
  );
};

export default CartModal;
```

 # next.config.js

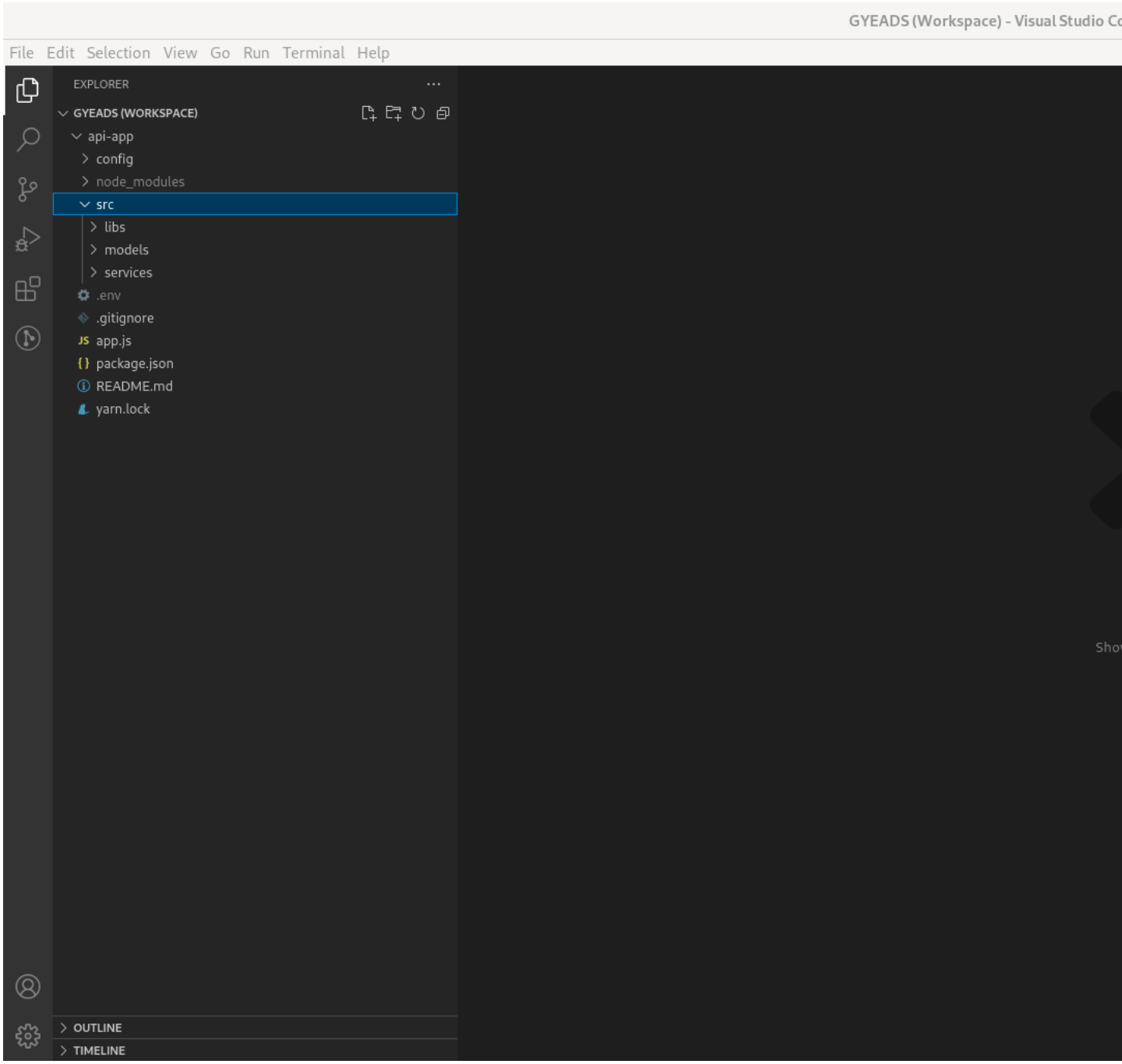
```
const withPlugins = require("next-compose-plugins");
const withImages = require("next-images");
module.exports = withPlugins([withImages], {
  reactStrictMode: true,
  images: {
    domains: [""]
  },
  publicRuntimeConfig: {
    // Will be available on both server and client
    APP_TITLE: "GYEADS",
    APP_URL: process.env.WEB_APP_URL,
    APP_ENV: "development",
    API_URL: process.env.API_APP_URL,
    APP_BRANDING: "GYEADS",
    APP_PAGINATE_LIMIT: 18,
    APP_ISLOADING_TIMEOUT: 500,
    APP_TOAST_TIMEOUT: 2500,
    AWS_S3_BUCKET_NAME: "",
    AWS_S3_REGION: "",
    AWS_S3_ACCESS_ID: "",
    AWS_S3_SECRET: "",
    GOOGLE_CLIENT_ID: "",
```

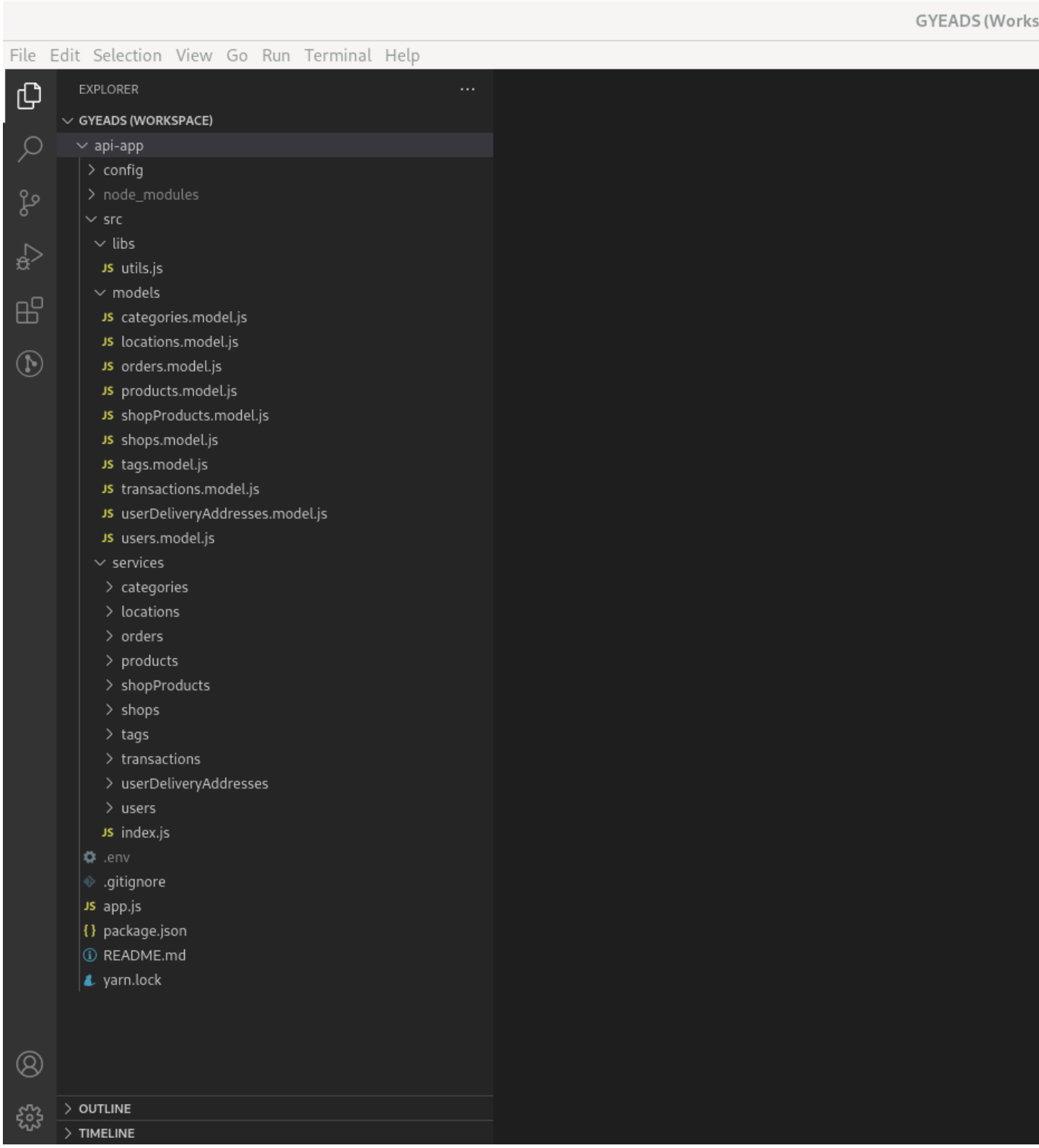
```
        GOOGLE_CLIENT_SECRET: "",
        FACEBOOK_CLIENT_ID: "",
        FACEBOOK_CLIENT_SECRET: ""
    }
});

# package.json
{
    "private": true,
    "scripts": {
        "dev": "next dev",
        "build": "next build",
        "start": "next start"
    },
    "dependencies": {
        "@heroicons/react": "^1.0.6",
        "@types/react-redux": "^7.1.24",
        "@types/redux-saga": "^0.10.5",
        "@typescript-eslint/eslint-plugin": "^5.25.0",
        "@typescript-eslint/parser": "^5.25.0",
        "axios": "^0.27.2",
        "formik": "^2.2.9",
        "next": "latest",
        "next-compose-plugins": "^2.2.1",
        "next-images": "^1.8.4",
        "next-redux-wrapper": "^7.0.5",
        "react": "^17.0.2",
        "react-dom": "^17.0.2",
        "react-redux": "^8.0.1",
        "redux": "^4.2.0",
        "redux-saga": "^1.1.3",
        "reselect": "^4.1.5",
        "yup": "^0.32.11"
    },
    "devDependencies": {
        "@types/node": "17.0.4",
        "@types/react": "17.0.38",
        "@types/react-dom": "^18.0.4",
        "autoprefixer": "^10.4.0",
        "postcss": "^8.4.5",
        "prettier": "^2.5.1",
        "prettier-plugin-tailwindcss": "^0.1.1",
```

"redux-devtools-extension": "^2.13.9",
"tailwindcss": "^3.0.7",
"typescript": "4.5.4"
}
}

API project file and folder structures:




GYEADS (Works
File Edit Selection View Go Run Terminal Help
EXPLORER
GYEADS (WORKSPACE)
api-app
config
node_modules
src
libs
utils.js
models
categories.model.js
locations.model.js
orders.model.js
products.model.js
shopProducts.model.js
shops.model.js
tags.model.js
transactions.model.js
userDeliveryAddresses.model.js
users.model.js
services
categories
locations
orders
products
shopProducts
shops
tags
transactions
userDeliveryAddresses
users
index.js
.env
.gitignore
app.js
package.json
README.md
yarn.lock
OUTLINE
TIMELINE

Few files code sample are mentioned below:

 # index.js

```
const router = require("express").Router();

router.use("/", require("./categories/categories.routes"));

router.use("/", require("./products/products.routes"));

router.use("/", require("./locations/locations.routes"));

router.use("/", require("./users/users.routes"));

router.use("/", require("./shops/shops.routes"));

router.use("/", require("./tags/tags.routes"));

router.use("/", require("./shopProducts/shopProducts.routes"));

router.use(
  "/",
  require("./userDeliveryAddresses/userDeliveryAddresses.routes")
);

router.use("/", require("./orders/orders.routes"));

router.use("/", require("./transactions/transactions.routes"));

module.exports = router;
```

Category Module

 # category routes

```
const router = require("express").Router();
const entity = "categories";
const entityServices = require("./" + entity + ".services");

// find
router.get("/" + entity, entityServices.find);

// get
router.get("/" + entity + "/:id", entityServices.get);
```

```
// post
router.post("/" + entity, entityServices.post);

// put
router.put("/" + entity + "/:id", entityServices.put);

// delete
router.delete("/" + entity + "/:id", entityServices.delete);

module.exports = router;
```

 # category services

```
const entityRepository = require("./categories.repositories");
const utils = require("../../libs/utils");

exports.find = async (req, res) => {
  const repoRes = await entityRepository.find(req, res);

  utils.responseHandler(res, "get", repoRes);
};

exports.get = async (req, res) => {
  const { params } = req;
  const { id } = params;
  const repoRes = await entityRepository.get(id);

  utils.responseHandler(res, "get", repoRes);
};

exports.post = async (req, res) => {
  const { body } = req;
  const repoRes = await entityRepository.post(body);

  utils.responseHandler(res, "post", repoRes);
};

exports.put = async (req, res) => {
  const { params, body } = req;
  const { id } = params;
  const repoRes = await entityRepository.put(id, body);
```

```
  utils.responseHandler(res, "put", repoRes);
};

exports.delete = async (req, res) => {
  const { params } = req;
  const { id } = params;
  const repoRes = await entityRepository.delete(id);

  utils.responseHandler(res, "delete", repoRes);
};

# category repositories
const entityModel = require("../../models/categories.model");
const ObjectId = require("mongoose").Types.ObjectId;
const utils = require("../../libs/utils");

exports.find = async (req, res) => {
  try {
    let limit = 10,
      skip = 0,
      searchText = null,
      isActive,
      thisQuery = [];

    // TODO
    const { query } = req;

    limit = (query.limit && parseInt(query.limit)) || limit;
    skip = (query.skip && parseInt(query.skip)) || skip;
    searchText = query.searchText && query.searchText;
    isActive = (query.isActive && query.isActive) || "active";

    // default
    thisQuery = [
      {
        $match: {
          status: isActive,
          name: new RegExp(searchText, "i"),
          deletedAt: null,
        },
      },
      { $limit: limit },
```

```
    { $skip: skip },
  ];

  // adding limit and skip
  // thisQuery.push({ $limit: limit });
  // thisQuery.push({ $skip: skip });

  // for all records
  if (limit === -1) {
    thisQuery = [{ $match: { deletedAt: null } }];
  }

  // sort data
  thisQuery.push({ $sort: { createdAt: -1 } });

  console.log("thisQuery", thisQuery);

  const total = await entityModel.countDocuments({ deletedAt: null });
  const records = await entityModel.aggregate(thisQuery);

  return {
    success: true,
    data: {
      total: total,
      limit: limit,
      skip: skip,
      data: records,
    },
  };
 } catch (error) {
  return {
    error: (error && error.message && error.message) || null,
    message: "Something went wrong for categories (find).",
  };
 }
};

// get
exports.get = async (id) => {
 try {
  const record = await entityModel.findById(ObjectId(id));
  return { data: record };
```

```
  } catch (error) {
    return {
      error: (error && error.message && error.message) || null,
      message: "Something went wrong for categories (get).",
    };
  }
};

// post
exports.post = async (payload) => {
  try {
    // prepare payload for adding code value
    const records = await entityModel.find({});
    Object.assign(payload, {
      code: "GYC-" + utils.zeroPad(records.length + 1, 3),
    });

    const creatableObject = new entityModel(payload);
    const record = await creatableObject.save();

    return { data: record };
  } catch (error) {
    return {
      error: (error && error.message && error.message) || null,
      message: "Something went wrong for categories (post).",
    };
  }
};

//put
exports.put = async (id, payload) => {
  try {
    await entityModel.findByIdAndUpdate(ObjectId(id), payload);

    const record = await entityModel.findById(id);
    return { data: record };
  } catch (error) {
    return {
      error: (error && error.message && error.message) || null,
      message: "Something went wrong for categories (put).",
    };
  }
```

```
};

//delete
exports.delete = async (id) => {
  try {
    // const deleteRes = await entityModel.deleteOne({ _id: ObjectId(id) });

    await entityModel.findByIdAndUpdate(ObjectId(id), {
      deletedAt: new Date(),
    });

    const record = await entityModel.find({ deletedAt: null });
    return { data: record };
  } catch (error) {
    return {
      error: (error && error.message && error.message) || null,
      message: "Something went wrong for categories (delete).",
    };
  }
};
```

 # database.config.js

```
const mongoose = require("mongoose");
require("dotenv").config();

const devConnection = process.env.DB_DEV_URL;
const prodConnection = process.env.DB_PROD_URL;

// Connect to the correct environment database
if (process.env.NODE_ENV === "production") {
  mongoose.connect(prodConnection, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  });

  mongoose.connection.on("connected", () => {
    console.log("Database connected PROD");
  });
} else {
  mongoose.connect(devConnection, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
```

```
    useCreateIndex: true,
  });

  mongoose.connection.on("connected", () => {
    console.log("Database connected DEV");
  });
}
```

 # categories.model.js

```
const mongoose = require("mongoose");

const entitySchema = mongoose.Schema(
  {
    name: { type: String },
    code: { type: String },
    status: {
      type: String,
      enum: ["active", "inactive"],
      default: "active",
    },
    deletedAt: { type: Date, default: null },
  },
  {
    timestamps: true,
  }
);

module.exports = mongoose.model("categories", entitySchema);
```

 # orders.model.js

```
const mongoose = require("mongoose");

const ordersSchema = mongoose.Schema(
  {
    locationId: { type: mongoose.Schema.Types.ObjectId, ref: "locations" },
    details: [
      {
        productId: { type: mongoose.Schema.Types.ObjectId, ref: "products" },
        units: { type: String },
        price: { type: Number },
      },
    ],
```

```
    totalPrice: { type: Number },
    status: {
      type: String,
      enum: ["active", "inactive"],
      default: "active",
    },
    deletedAt: { type: Date, default: null },
  },
  {
    timestamps: true,
  }
);

module.exports = mongoose.model("orders", ordersSchema);
```

 # products.model.js

```
const mongoose = require("mongoose");

const entitySchema = mongoose.Schema(
  {
    name: { type: String },
    code: { type: String },
    categoryId: { type: mongoose.Schema.Types.ObjectId, ref: "categories" },
    tags: { type: Array },
    images: { type: Array },
    price: { type: Number },
    units: { type: Number },
    status: {
      type: String,
      enum: ["active", "inactive"],
      default: "active",
    },
    deletedAt: { type: Date, default: null },
  },
  {
    timestamps: true,
  }
);

module.exports = mongoose.model("products", entitySchema);
```

 # shopProducts.model.js

```
const mongoose = require("mongoose");

const shopProductsSchema = mongoose.Schema(
  {
    name: { type: String },
    locationId: { type: mongoose.Schema.Types.ObjectId, ref: "locations" },
    shopId: { type: mongoose.Schema.Types.ObjectId, ref: "shops" },
    categoryId: { type: mongoose.Schema.Types.ObjectId, ref: "categories" },
    productIds: [{ type: mongoose.Schema.Types.ObjectId, ref: "products" }],
    status: {
      type: String,
      enum: ["active", "inactive"],
      default: "active",
    },
    deletedAt: { type: Date, default: null },
  },
  {
    timestamps: true,
  }
);

module.exports = mongoose.model("shopProducts", shopProductsSchema);
```

 # shops.model.js

```
const mongoose = require("mongoose");

const shopsSchema = mongoose.Schema(
  {
    name: { type: String },
    code: { type: String },
    userId: { type: String },
    locationId: { type: mongoose.Schema.Types.ObjectId, ref: "locations" },
    shopPicture: { type: String },
    openingTime: { type: String },
    closingTime: { type: String },
    categoryId: { type: mongoose.Schema.Types.ObjectId, ref: "categories" },
    tagIds: { type: mongoose.Schema.Types.ObjectId, ref: "tags" },

    status: {
      type: String,
      enum: ["active", "inactive"],
      default: "active",
```

```
    },
    deletedAt: { type: Date, default: null },
  },
  {
    timestamps: true,
  }
);

module.exports = mongoose.model("shops", shopsSchema);
```

 # app.js

```
// express
const express = require("express");
// TODO
const cors = require("cors");
// TODO
const path = require("path");

const mongoose = require("mongoose");

// gives us access to variables from .env file
require("dotenv").config();

// app running port
const port = process.env.PORT || 8080;

// Configures the database and opens a global connection that can be used in any module with
mongoose.connection
require("./config/database.config");

// create express app
const app = express();

// instead of using body-parser middleware, use new express implementation of same thing
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// home route
app.get("/", (req, res) => {
  res.send("API Home Page");
});
```

```
// services
app.use(require("./src/services/index"));

app.listen(port, () => {
  console.log("Server is listening on " + port);
});
```

 # package.json

```
{
  "name": "gyeads-api-app",
  "version": "1.0.0",
  "main": "app.js",
  "repository": "",
  "author": "Roshni Adhikary",
  "license": "MIT",
  "scripts": {
    "start": "node app.js",
    "dev": "nodemon app.js"
  },
  "dependencies": {
    "cors": "^2.8.5",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.3.1",
    "nodemon": "^2.0.15",
    "path": "^0.12.7"
  }
}
```

Discussion

In today's fast paced life of today when everyone is squeezed for time, the majority of people are finicky when it comes to doing physical shopping. A consumer finds a product more interesting and attractive when they find it on the website of a retailer directly and are able to see item's details online.The customers of today are not only attracted because online shopping is very convenient, but also because they have broader selections, highly competitive prices, better information about the product (including people's reviews) and extremely simplified navigation for searching regarding the product. Moreover, business owners often offer online shopping options at low rates because the overhead expenses in opening and running a physical store are higher. Further, with online shopping, their products have access to a worldwide market, which increases the number of customers from different ethnic groups, adds customer value, and is overall sustainable in the marketing.

Online web stores, such as Amazon, Flipkart, Bigbusket, Blinkit etc, have gained huge popularity over the years because one can buy almost everything at these stores. We have already seen in COVID pandemic how much the online delivery system was useful. In that limelight I had planned to create a full fledged family essentials ecommerce system. I hope this application will survive in the real world but it's going to be extremely difficult, whereas a lot of online application giants are already running this industry.

People are habituated to use the most popular application to fulfill their needs. Now it's going to be a hard task to attract public attention into a new application even if it will be good in terms of quality and price. We always can do our best despite thinking about the result or outcome of any process. We are doing the same here. So here we are creating online family essentials delivery solutions. But it's different from the existing popular applications. We will create a local business chain vai our application and people can get their on-demand products in a very small span of time. We believe it will have a great impact in our society. As the gigantic business giants are growing their capital in rapid ratio, the low level local businesses are struggling for their existence. So if we can reach people in our society vai this application, it will help to keep the social business chain. It can also prevent the social economic imbalance which is a very serious matter to consider.

Comparative Study

I had a plan to create a full fledged online application of the family essentials delivery system. There are different modules involved in this application. Let us discuss the modules.

i.**Requirements analysis:** We have reviewed a lot of applications to fix our requirements for this application. Here we create the ERD and the DATABASE design of this application.

ii. **Design the WEB app:** In this step we design the LOOKS & FEEL of the WEB application. We use the FIGMA tool to create the mockup design.

iii. **Creating HTML and CSS:** Once the mockup design is ready we convert them into HTML using css. We use tailwind css (tailwindcss.com) css framework to design our HTML.

iv. **Convert HTML into SPA application**: As we are using ReactJs, we converted HTML into REACT for each module like categories, products, shops, shopproducts etc.

v. **Prepare APIS for each module:** For now we are not able to prepare all the APIS we require. But we created the base API application. We need to work on it to make this application production ready.

vi. **Domain & Server setup:** We have already deployed this application on the AWS server under domain name (https://gyeads.com). We use nginx to create the web server. Mongodb cloud (https://www.mongodb.com/cloud) used for database solutions.

Conclusion

This project is built on a full stack javascript platform. This project is now available at https://gyeads.com. Javascript , typescript, nodejs, reactJS, tailwind css, mongodb (database) and visual studio (code editor) are used here. Let us discuss the tools we used in our project.

Javascript (JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles.

TypeScript is an object-oriented programming language developed by Microsoft Corporation, whereas JavaScript is the programming language for the web. TypeScript is an open-source language to build large-scale web apps, whereas JavaScript is a server-side programming language that helps to develop interactive web pages.

Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project! Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant. A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

Tailwind CSS can be used to make websites in the fastest and the easiest way.

Tailwind CSS is basically a utility-first CSS framework for rapidly building custom user interfaces. It is a highly customizable, low-level CSS framework that gives you all of the building blocks you need to build bespoke designs without any annoying opinionated styles you have to fight to override.
The beauty of this thing called tailwind is it doesn't impose design specification or how your site should look like, you simply bring tiny components together to construct a user interface that is unique. What Tailwind simply does is take a 'raw' CSS file, processes this CSS file over a configuration file, and produces an output.

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.

Declarative views make your code more predictable and easier to debug.

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.

Declarative views make your code more predictable and easier to debug.

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as Node, C++, C#, Java, Python, PHP, Go, .NET).

Pattern recognition is the process of recognizing patterns by using a machine learning algorithm. Pattern recognition can be defined as the classification of data based on knowledge already gained or on statistical information extracted from patterns and/or their representation. One of the important aspects of pattern recognition is its application potential.

Examples: Speech recognition, speaker identification, multimedia document recognition (MDR), automatic medical diagnosis etc. In a typical pattern recognition application, the raw data is processed and converted into a form that is amenable for a machine to use. Pattern recognition involves the classification and cluster of patterns.

We can add so many features to it in the near future using machine learning technology to make it more user-friendly and interactive. I wish this project would continue upto its best level.

References:

https://reactjs.org/
https://nextjs.org/
https://www.mongodb.com/docs/
https://www.mongodb.com/cloud
https://nodejs.org/en/
https://expressjs.com/
https://redux.js.org/
https://react-redux.js.org/
https://redux-saga.js.org/
https://www.typescriptlang.org/
https://vhudyma-blog.eu/add-redux-saga-with-typescript-to-your-react-application-january-2021/