North South University

Project report: Junior Design

Project name: Gogonjo

Supervisor

Dr. Mohammad Ashrafuzzaman Khan

Assistant Professor,

Dept. of Electrical and Computer Engineering

North South University

Team

Abu Sadat Md. Sayem

ID: 1521105642

Course: CSE299

Section: 16

Abstract

This report describes the design and implementation of an e-commerce web application. Several e-commerce web application already exist but I have used new technology to implement this project. And my project idea is also bit different.

Acknowledgements

This project has taken a considerable amount of time and resources and I would like to acknowledge the help of all of those who have made the project possible. In particular I would like to thank my supervisor **Dr. Mohammad Ashrafuzzaman Khan** for his time, patience and guidance, and also for allowing the idea to be pursued originally.

Also, I would like to thank all of the many thousands of people who have worked on all of the Open Source projects without whose efforts this project would not have been possible. A list of these projects is given in Appendix A.

Contents

Introduction

This report is about an overview of our project, which is an e-commerce web app. This document contains information about the Use Cases, Basic features of the System, System Architectural Design, Technologies to be used, Development Plans and Schedules, Project Development and Maintenance Cost. It also contains screenshots of the pages and an annex section.

Aim of the Project

- Create web application online
- Aesthetic user interface, making it an enriching experience for creation of web pages.

Features

Nowadays day by day virtual shop increasing rapidly but many of them have no e-commerce platform to flourish and it is very struggling and very puzzling for a new seller or shopkeeper. Many of them are doing their marketing on facebook without any organized way.

Here I have come with a plan is to create a market-oriented website which will be a common platform for various kinds of business globally. Every seller will have a subdomain like [xyz.gogonjo.com or gogonjo.com/xyz] . They can add and sell their products easily.

User Story

Use Case Scenario 1:

Mrs. Shahida Sultana is a housewife. She loves to cook and her cooking is price worthy and well known in her society. Some days ago she has started to cook and sell them. She receives her order in a phone call and by the facebook page. But now her business is growing rapidly and she could not able to take care of all of the order hence many of the order is going miss out. Now she is feeling that if she has a website or app where she can add his recipe and get the order, later on, she can deliver each and every order.

So she has to visit my website where she can make a online shop as well as she will get high opportunity sell more recipe.

Use Case Scenario 2:

Mr. Aminul Haq is very new in Dhaka city. He doesn't know where he can laundry his dress and as well as he is very busy. He wants some sort of service where he can laundry his dress within minimal time and at the same price.

So he has to visit my website and he will get a lot of laundries service. He will also get review and rating so that he can presume which one will be good for him.

Technologies

1. The software stacks that we plan to use for the Minimum Viable Product(Prototype) is provided as follows:
2. Apollo server: It is used for GraphQL Server Query/Mutation processing.
3. Node.js: It is used as the primary language for writing back-end codes.
4. Vue.js (Front-end framework): It is used for front-end data presentation.
5. CSS (Cascading Style Sheets): It is used for front-end design.
6. MongoDB: It is used as the primary Database engine for storing and retrieving data.

Front-end

For this project I have used Vue.js. Vue.js is an open-source JavaScript framework for building user interfaces and single-page applications. Vue.js features an incrementally adoptable architecture that focuses on declarative rendering and component composition. Advanced features required for complex applications such as routing, state management and build tooling are offered via officially maintained supporting libraries and packages.

1. **Home** which contains the products from various shop.
2. **Signup** consists of a form which an unregistered user can fill up to signup for the site.
3. **Signin** consists of a form where the user will provide their unique username and password to login into the system.
4. **Create shop** consists of a form which an registered user can fill up to create new shop.
5. **Profile,** it contains the profile details of a user.
6. **Add product,** shopkeeper can add their product.
7. **Category** contains a list of items category after click on any category app will show all product which match with clicked product.
8. **Product show by category** will show requested categorized product.
9. **Search**, user can search product, shop and user also.
10. **Shops,** all the shops will be shown here including signed in user's shops.
11. **Dashboard**, here shopkeeper will get all order and his product list, he can also delete and edit his product.
12. **Cart,** It is the page where users can pile up what they want to buy from the website and then simply checkout.
13. **Rating and Comment**, user can rate any product and also can comment.
14. **Deployed** on now **Heroku.**

Apollo-server

Apollo Server is the best way to quickly build a production-ready, self-documenting API for GraphQL clients, using data from any source. It's open-source and works great as a stand-alone server, an addon to an existing Node.js HTTP server, or in "serverless" environments.

List of type, query and mutation

1. **Type**
    - 1.1. prevProductRating
    - 1.2. prevShopRating
    - 1.3. productRating
    - 1.4. shopRating
    - 1.5. Comment
    - 1.6. Product
    - 1.7. User
    - 1.8. Shop
    - 1.9. Carousel
    - 1.10. Love
    - 1.11. Kudos
    - 1.12. Order
    - 1.13. Search
    - 1.14. Token
    - 1.15. ProductPage
    - 1.16. OwnRating
    - 1.17. Address
    - 1.18. AddressInput
    - 1.19. PurchaseItemInput
    - 1.20. PurchaseItem

2. **Query**
    - 2.1. getProducts: [Product]
    - 2.2. getUser: User
    - 2.3. getCurrentUser: User
    - 2.4. getShop: Shop
    - 2.5. getAllShopByaUser: [Shop]
    - 2.6. getProducts: [Products]
    - 2.7. getProductByProductId: Product!

2.8. getProductByShopId: [Products]
2.9. infiniteScrollProduct: [Products]
2.10. getOwnProductRating: OwnRating
2.11. getOrder: [Order]
2.12. searchAny: Search

3. Mutations

3.1. addOrder: Order!
3.2. Signup: Token
3.3. updateProductRating: Product!
3.4. addCarousel: Carousel!
3.5. signin: Token
3.6. addProduct: Product!
3.7. createShop: Shop!
3.8. deleteProduct: Product!
3.9. loveProduct: Love!
3.10. unLoveProduct: Love!
3.11. addComment: Comment!

4. **Deployed** on now **Now-ZEIT**

GraphQL

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools.

Main part of GraphQL is Type schema and Resolvers. GraphQL has own GraphQL schema language.

Our project's -

Schema

```
type prevProductRating {
 product: Product
 rating: Float
}

type prevShopRating {
 shop: Shop
 rating: Float
}

type productRating {
 rate: Float!
 totalNumberOfRating: Int!
 ratedProduct: [prevProductRating]
}

type shopRating {
 rate: Float!
 totalNumberOfRating: Int!
 ratedShop: [prevShopRating]
}

type User {
 _id: ID
 firstName: String!
 lastName: String!
 email: String!
```

```
 contactNo: String!
 password: String!
 userName: String!
 dateOfBirth: String!
 dateOfRegistration: String
 profilePic: String
 coverPic: String
 role: String
 love: [Product]
 ratedProduct: [productRating]
 ratedShop: [shopRating]
}



type Comment {
 _id: ID
 body: String!
 commentUser: User!
 commentDate: String!
 like: Int
}

type Product {
 _id: ID
 productName: String!
 price: Int!
 description: String!
 comments: [Comment]
 rating: productRating
 tag: [String]!
 category: [String]!
 size: [String]!
 color: [String]!
 parent: String
 discountPrice: Int
 picture: [String]!
 dateOfAdd: String
 shopId: String!
 love: Int
 createdBy: Shop!
```

```
}

type Shop {
 _id: ID
 shopName: String!
 owner: User!
 email: String!
 ownerEmail: String!
 contactNo: String!
 address: String!
 dateOfCreation: String
 logo: String
 coverPic: [Carousel]
 rating: shopRating
 shopId: String!
 kudos: Kudos
}
type Carousel {
 _id: ID
 src: String!
 title: String
 button: String
 subtitle: String
 titleColor: String
 subtitleColor: String
 buttonColor: String
}

type Love {
 loves: Int
 wishList: [Product]
}

type Kudos {
 _id: ID
 name: String
 description: String
 year: String
}
```

```
type Search {
 product: [Product]
 shop: [Shop]
 user: [User]
}

type Token {
 token: String!
}

type ProductPage {
 products: [Product]
 hasMore: Boolean
}

type OwnRating {
 rating: Float
}


type Address {
 name: String!
 address: String!
 contact: String!
 state: String!
 city: String!
 zip: String!
 country: String!
}

input AddressInput {
 name: String!
 address: String!
 contact: String!
 state: String!
 city: String!
 zip: String!
 country: String!
}
```

```
input PurchaseItemInput {
 productItem: ID!
 quantity: Int!
 color: String!
 size: String!
 shopId: ID!
 price: Int!
 shopName: String!
 picture: String!
 productName: String!

}

type PurchaseItem {
 productItem: ID!
 quantity: Int!
 color: String!
 size: String!
 shopId: ID!
 price: Int!
 shopName: String!
 picture: String!
 productName: String!
 status: String
}

type Order {
 _id: ID
 consumer: ID!
 consumerEmail: String!
 shippingType: String!
 total: Int!
 purchaseItems: [PurchaseItem]!
 address: Address!
}
```

Query

```
type Query {
 getProducts: [Product]
 getUser(userName: String!): User
 getCurrentUser: User
 getShop(id: String!): Shop
 getAllShopByaUser(id: String!): [Shop]
 getProductsByShopId(id: String!): [Product]
 infiniteScrollProduct(pageNum: Int!, pageSize: Int!): ProductPage!
 getProductByProductId(id: String!): Product!
 searchAny(any: String): Search
 getOwnProductRating(productId: ID!, userId: ID!): OwnRating
 getOrder: [Order]

}
```

Mutation

```
type Mutation {
 addOrder(
 consumer: ID!,
 consumerEmail: String!,
 shippingType: String!, total: Int!,
 purchaseItems: [PurchaseItemInput],
 address: AddressInput!
): Order!

signup(
 firstName: String!,
 lastName: String!,
 contactNo: String!,
 email: String!,
 password: String!,
 userName: String,
 dateOfBirth: String
): Token

updateProductRating(
 productId: ID!,
 userId: ID!,
 givenRating: Float
): Product!

addCarousel(
 src: String!,
 title: String,
 button: String,
 subtitle: String,
 titleColor: String,
 subtitleColor: String,
 buttonColor: String,
 shopId: ID!
): Carousel!

signin(
userName: String,
password: String!
): Token
```

```
addProduct(
 productName: String!,
 price: Int!,
 description: String!,
 tag: [String],
 category: [String]!,
 size: [String]!,
 color: [String]!,
 parent: String,
 picture: [String]!,
 creatorId: ID!, shopId: String!
): Product!

createShop(
 shopName: String!,
 ownerId: ID!,
 ownerEmail: String!,
 email: String!,
 contactNo: String!,
 address: String!
): Shop!

deleteProduct( id: String! ): Product!

 loveProduct(productId: ID!, userName: String!): Love!

 unLoveProduct(productId: ID!, userName: String!): Love!

 addComment(body: String, userId: ID, productId: ID): Comment!

}
```

Resolvers

```
const bcrypt = require("bcrypt");
const jwt = require("jsonwebtoken");
const mongoose = require("mongoose");

const createToken = (user, secret, expiresIn) => {
  const { userName, email, role } = user;
  return jwt.sign({ userName, email, role }, secret, { expiresIn: "30d" });
};

module.exports = {
  Query: {
    searchAny: async (_, { any }, { User, Product, Shop }) => {
      if (any) {
        const searchProduct = await Product.find(
          { $text: { $search: any } },
          { score: { $meta: "textScore" } }
        )
          .sort({
            score: { $meta: "textScore" },
            rating: "desc"
          })
          .limit(15);
        const searchShop = await Shop.find(
          { $text: { $search: any } },
          { score: { $meta: "textScore" } }
        )
          .sort({
            score: { $meta: "textScore" },
            rating: "desc"
          })
          .limit(15);

        const searchUser = await User.find(
          { $text: { $search: any } },
          { score: { $meta: "textScore" } }
        ).limit(15);

        return {
          shop: searchShop,
          product: searchProduct,
          user: searchUser
        };
```

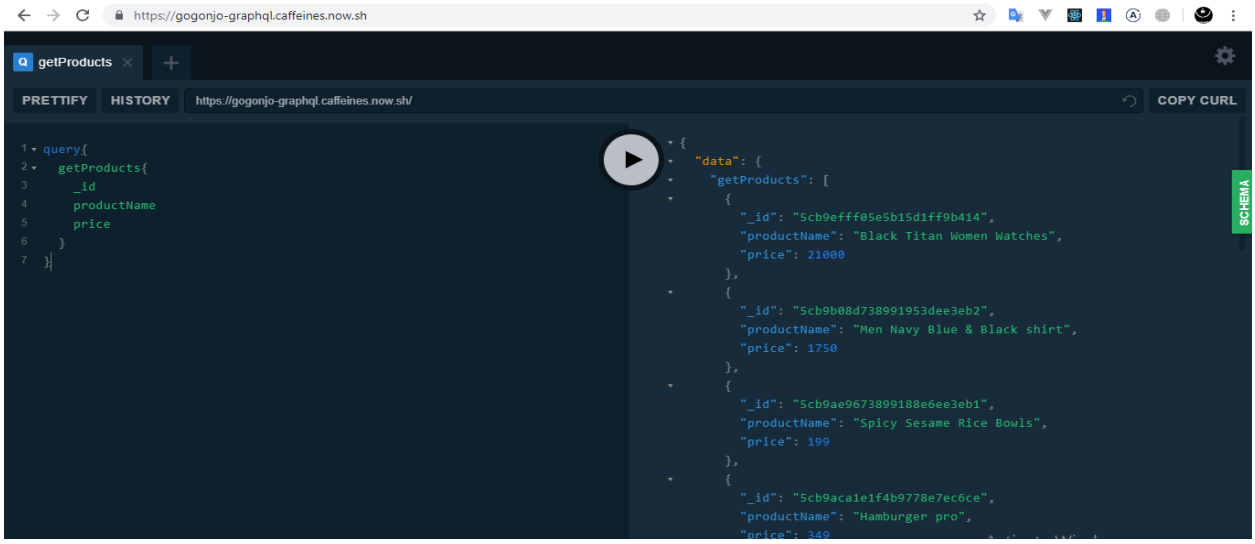
```
    }
  },
  getOrder: async (_, {}, { Order }) => {
    const order = await Order.find({})
      .sort({
        dateOfAdd: "desc"
      })
      .populate({
        path: "createdBy",
        model: "Shop"
      });
    return order;
  },
  /*
   ** Product fetching
   */
  getProducts: async (_, args, { Product }) => {
    const product = await Product.find({})
      .sort({
        dateOfAdd: "desc"
      })
      .populate({
        path: "createdBy",
        model: "Shop"
      });
    return product;
  },
  /*
   ** Product fetching by shop id
   */
  getProductsByShopId: async (_, { id }, { Product }) => {
    const product = await Product.find({ shopId: id })
      .sort({
        dateOfAdd: "desc"
      })
      .populate({
        path: "createdBy",
        model: "Shop"
      });

    return product;
  }
}
```

Query from client

```
query {
 getProducts {
  _id
  productName
  price
 }
}
```

Output

```
{
 "data": {
  "getProducts": [
   {
    "_id": "5cb9efff05e5b15d1ff9b414",
    "productName": "Black Titan Women Watches",
    "price": 21000
   },
   ]
}
```



Screenshots

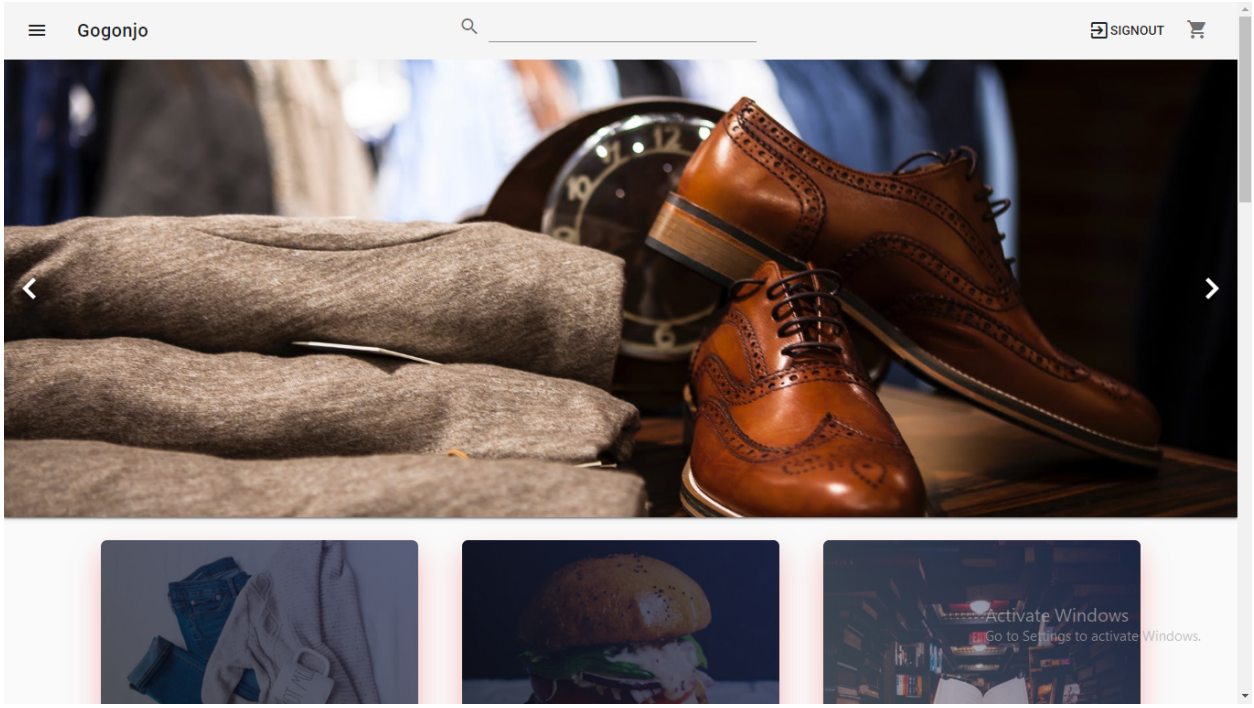


Fig 1: home page -1

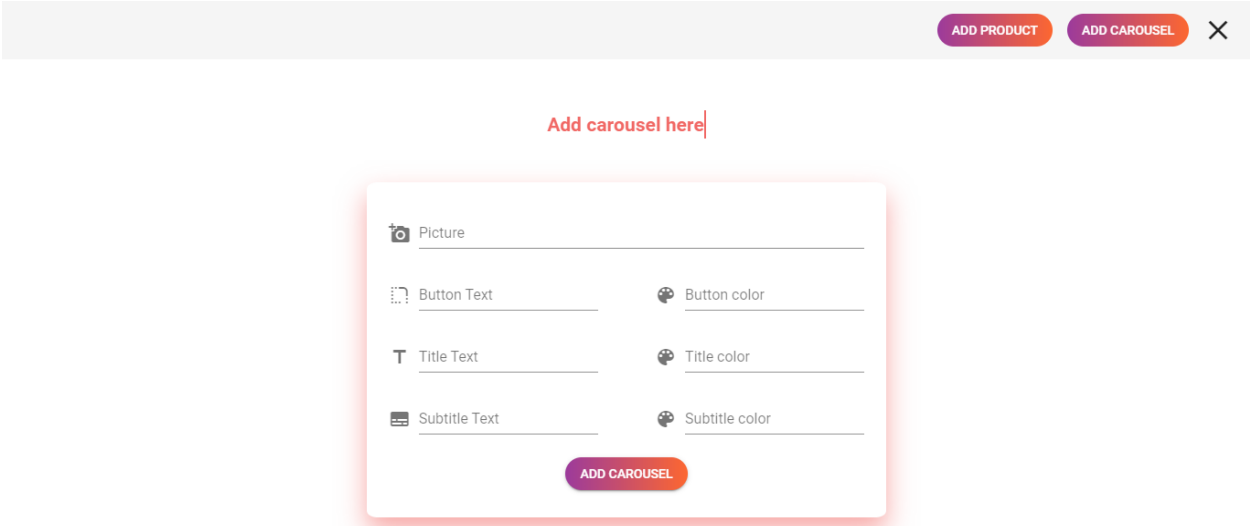


Fig 2: Add carousel page

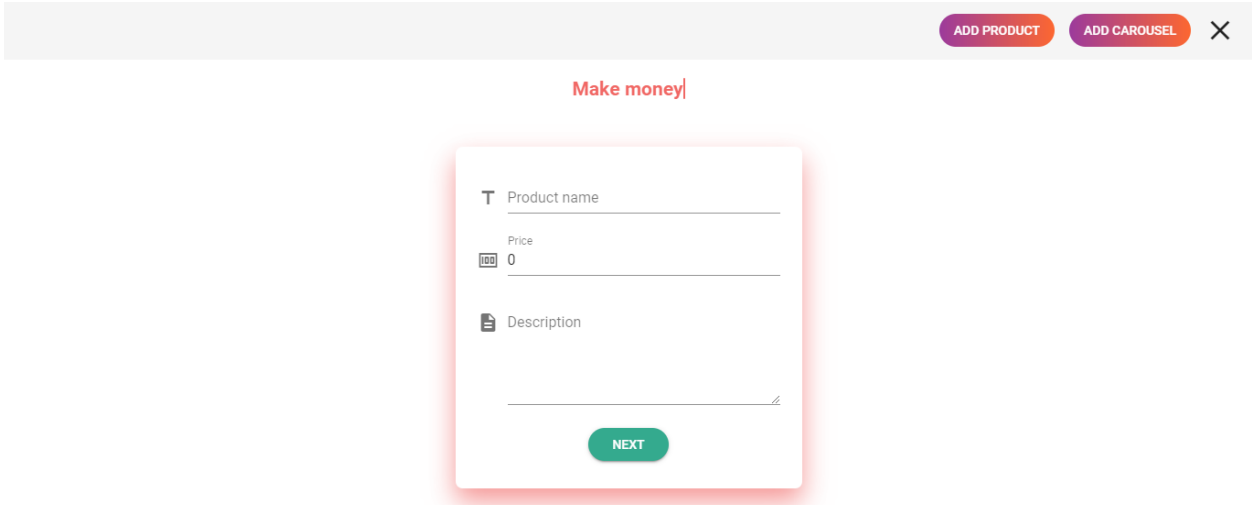
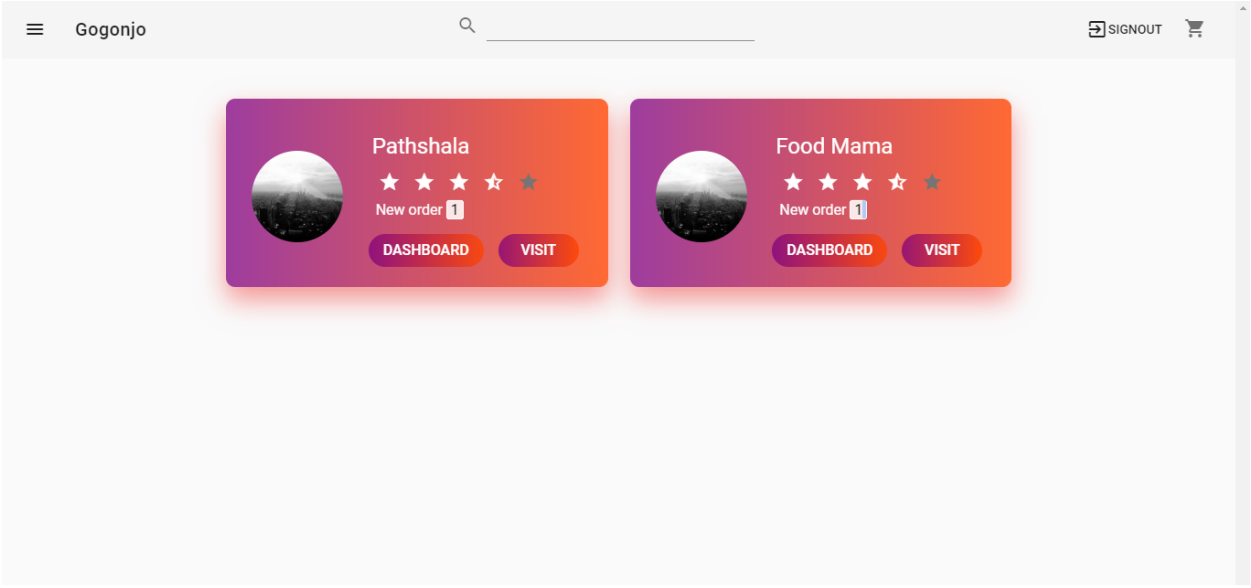
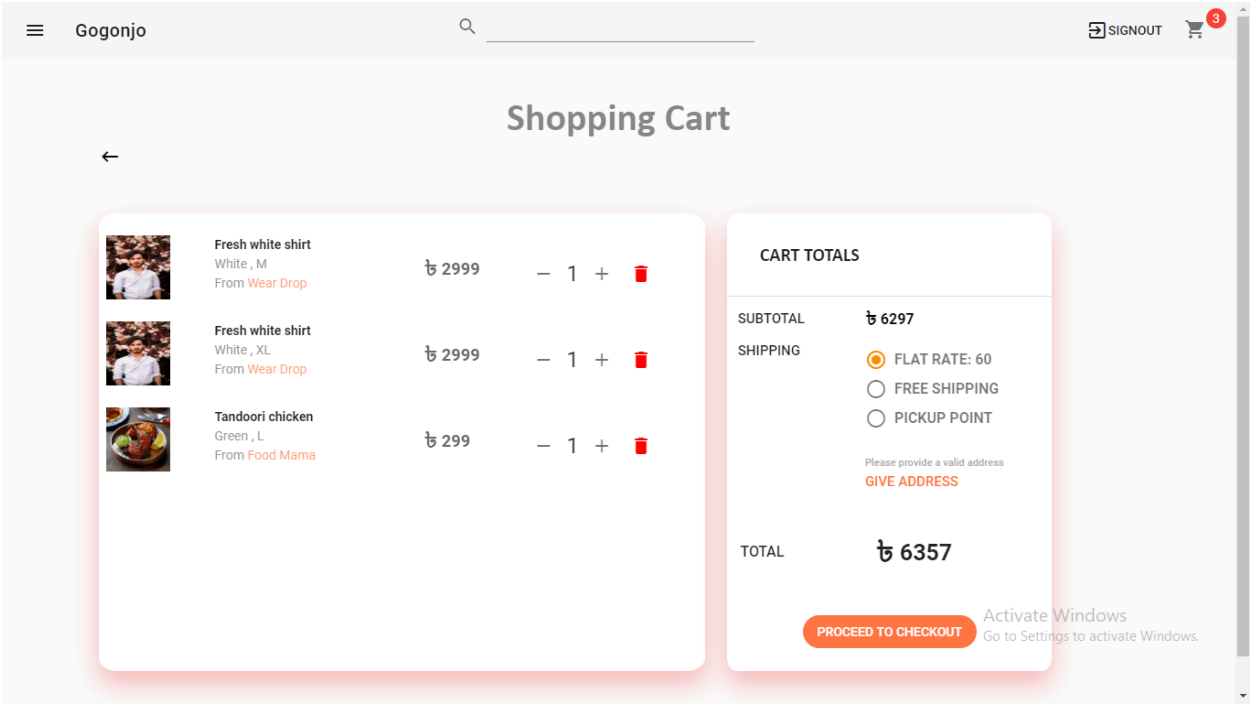
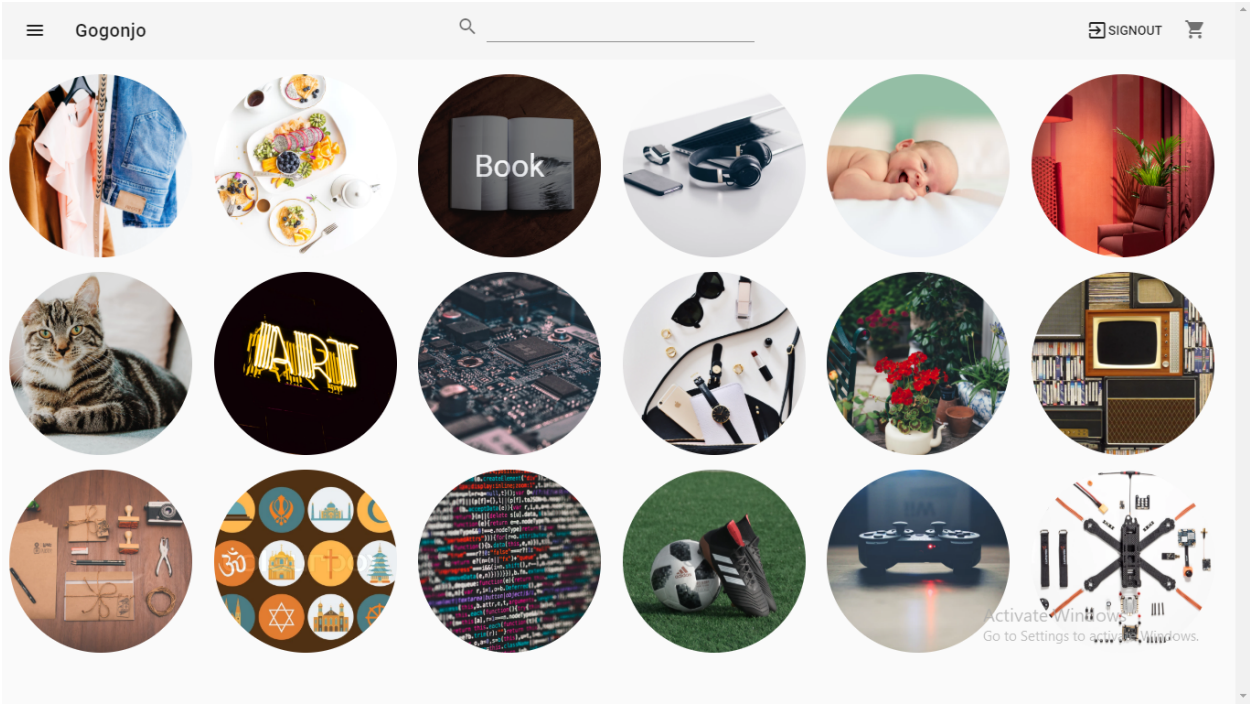
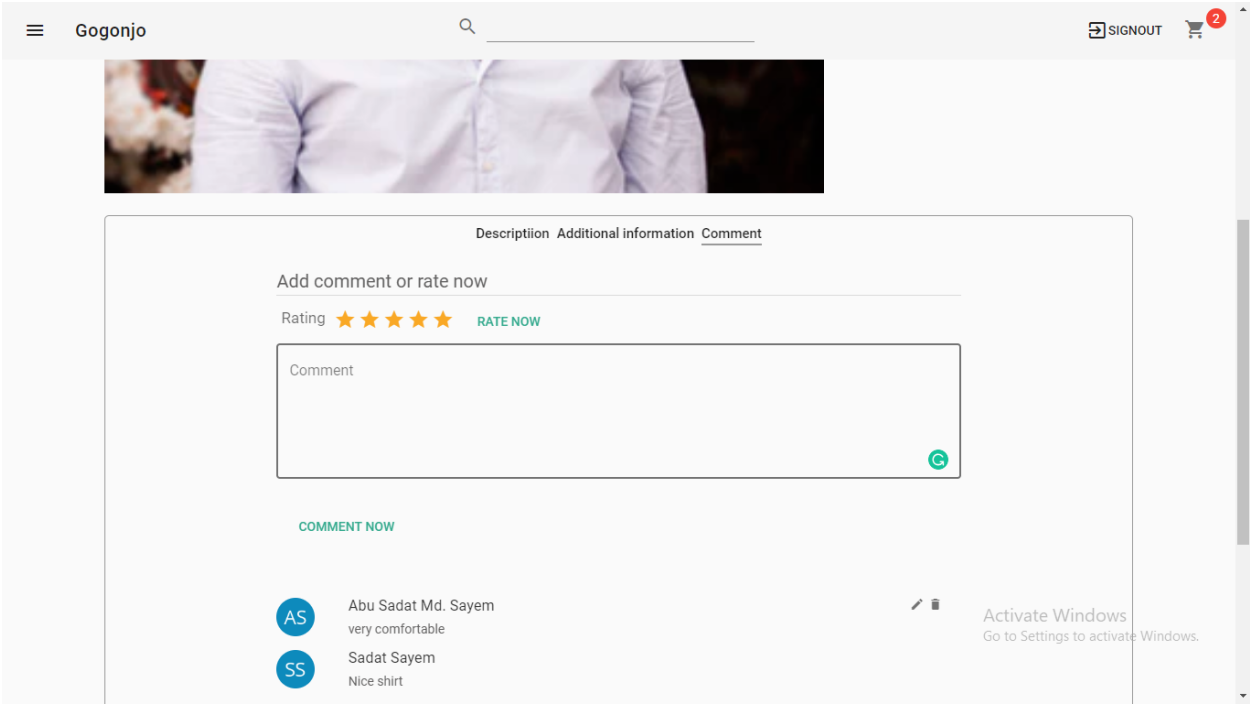


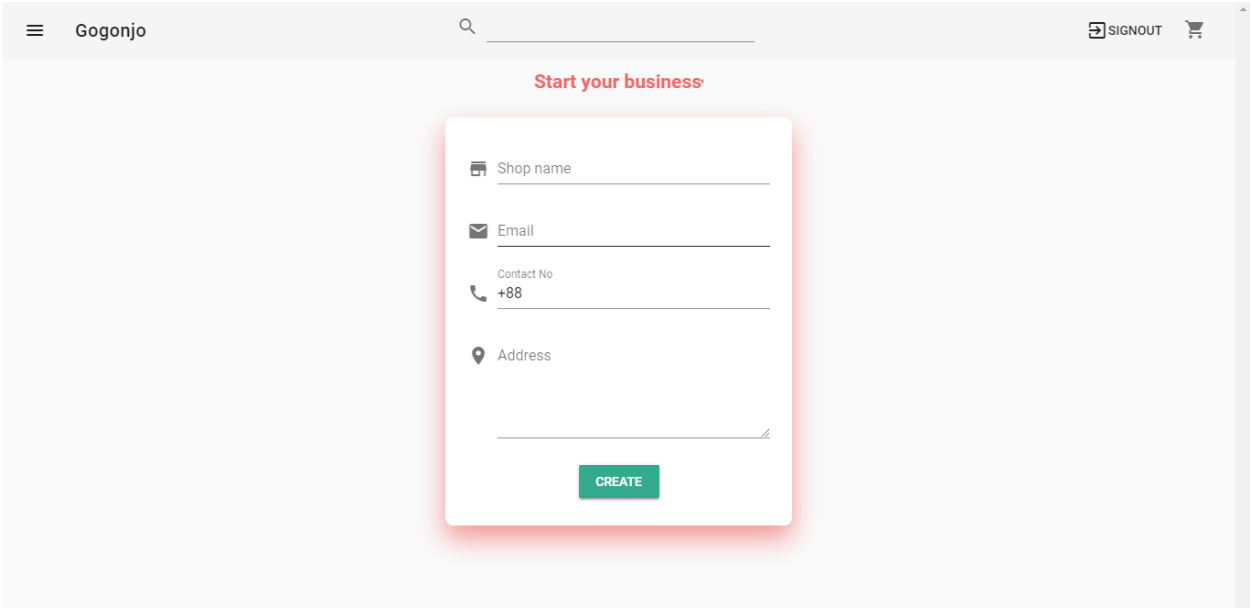
Fig 3: Add product form

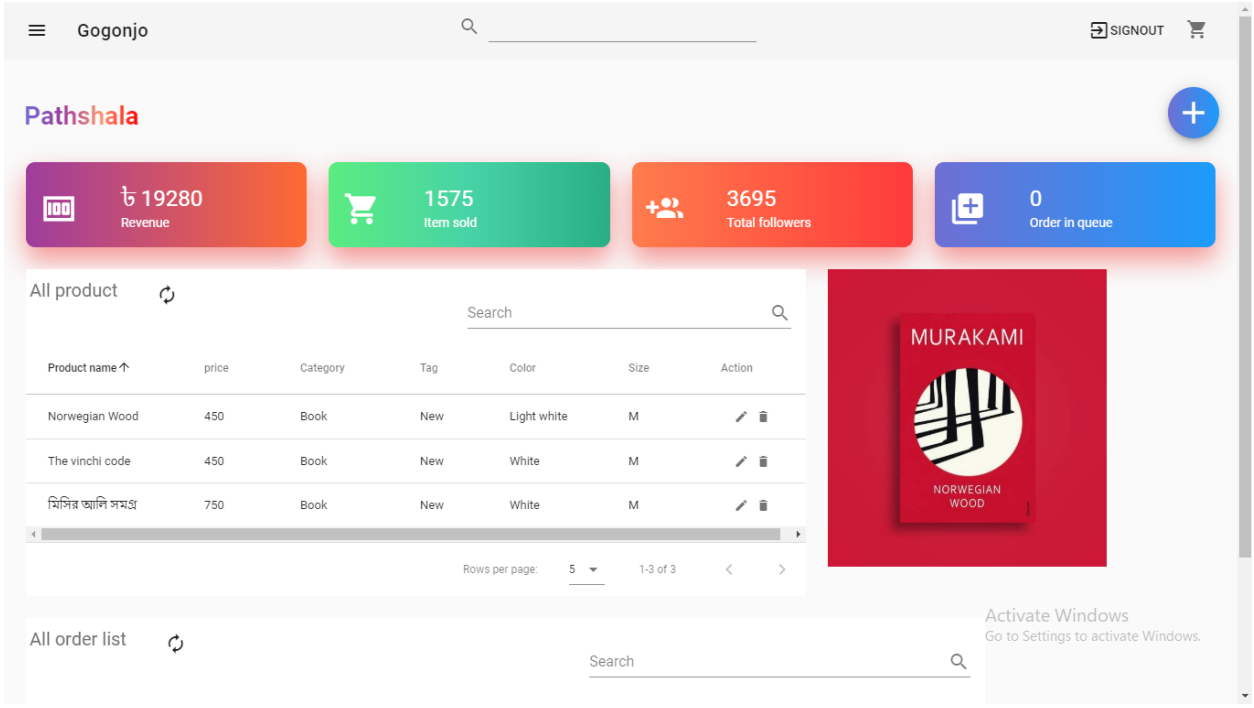


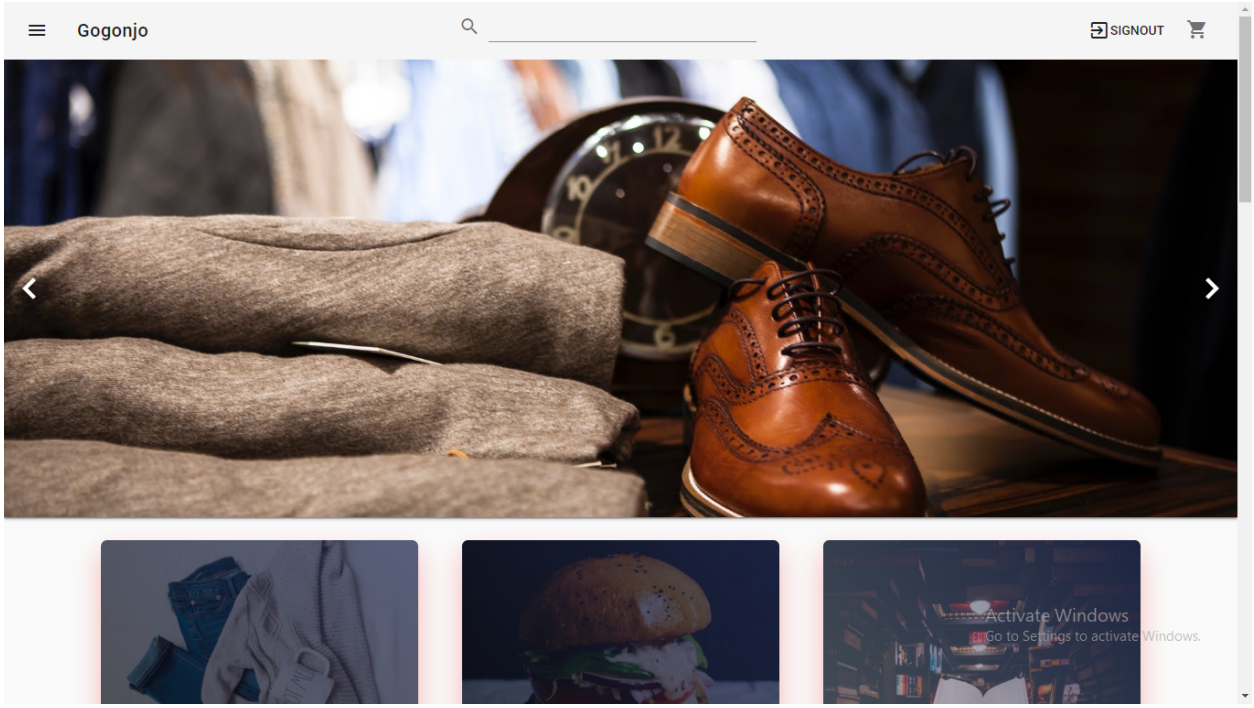

Gogonjo
SIGNOUT
3
Shopping Cart
Fresh white shirt
White , M
From Wear Drop
৳ 2999
1
Fresh white shirt
White , XL
From Wear Drop
৳ 2999
1
Tandoori chicken
Green , L
From Food Mama
৳ 299
1
CART TOTALS
SUBTOTAL
৳ 6297
SHIPPING
FLAT RATE: 60
FREE SHIPPING
PICKUP POINT
Please provide a valid address
GIVE ADDRESS
TOTAL
৳ 6357
PROCEED TO CHECKOUT
Activate Windows
Go to Settings to activate Windows.

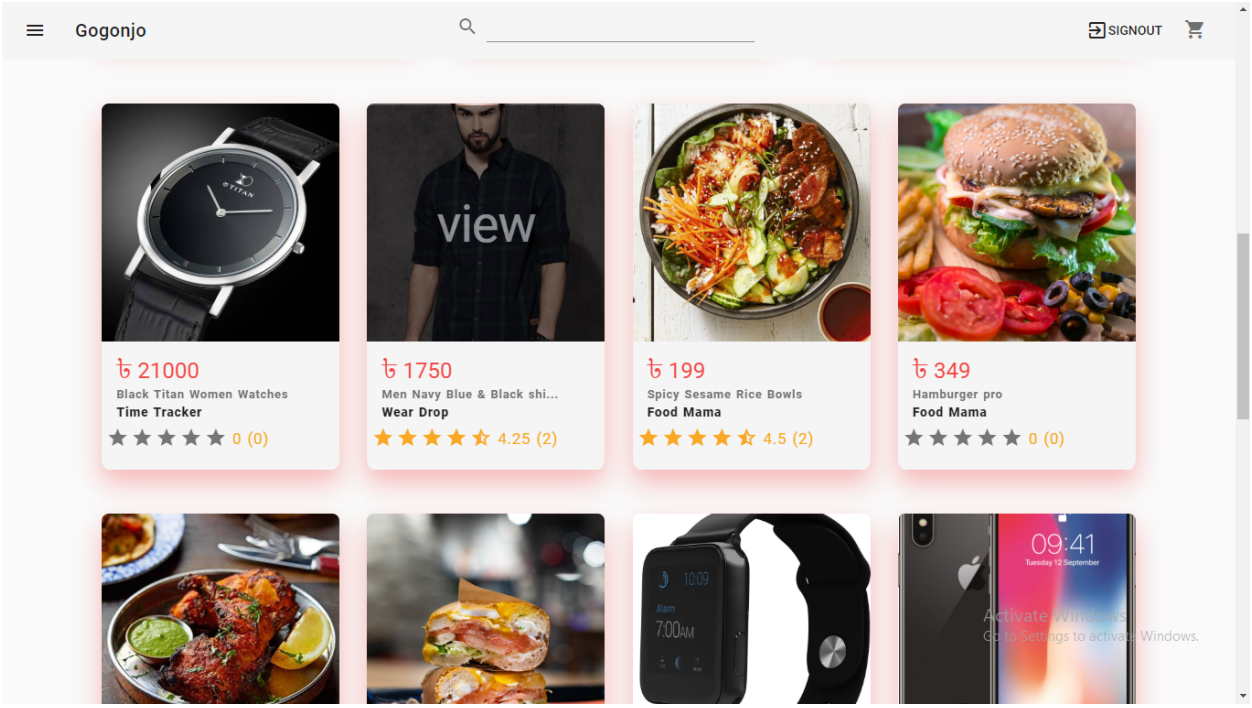

Gogonjo
SIGNOUT
Book
ART
Activate Windows
Go to Settings to activate Windows.

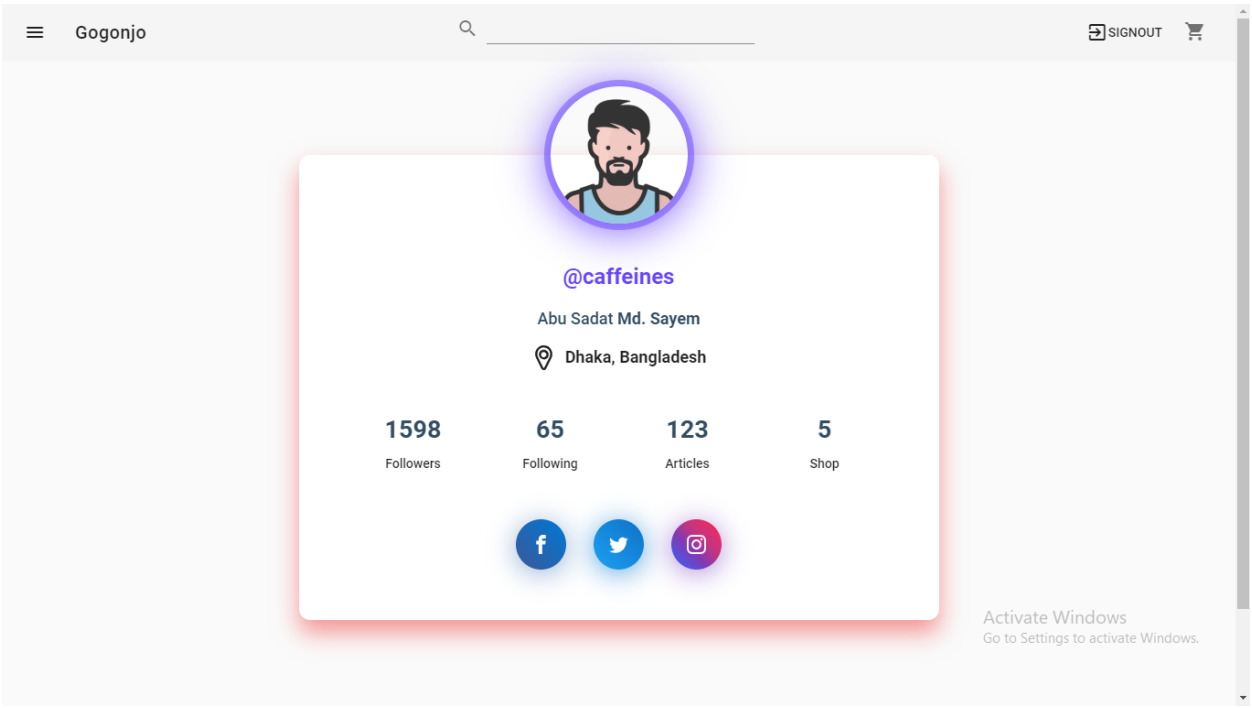

Gogonjo
SIGNOUT
2
Descriptiion
Additional information
Comment
Add comment or rate now
Rating
RATE NOW
Comment
COMMENT NOW
AS
Abu Sadat Md. Sayem
very comfortable
SS
Sadat Sayem
Nice shirt
Activate Windows
Go to Settings to activate Windows.

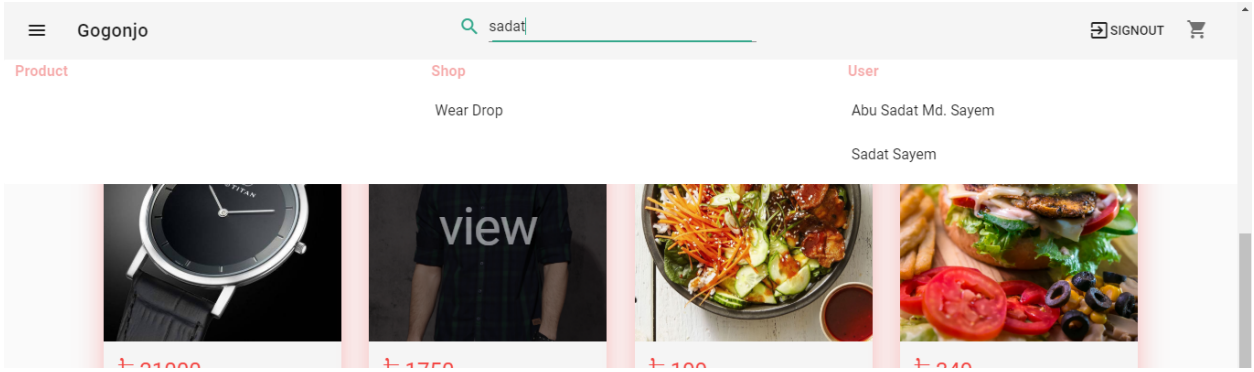

Gogonjo
SIGNOUT
Start your business
Shop name
Email
Contact No
+88
Address
CREATE

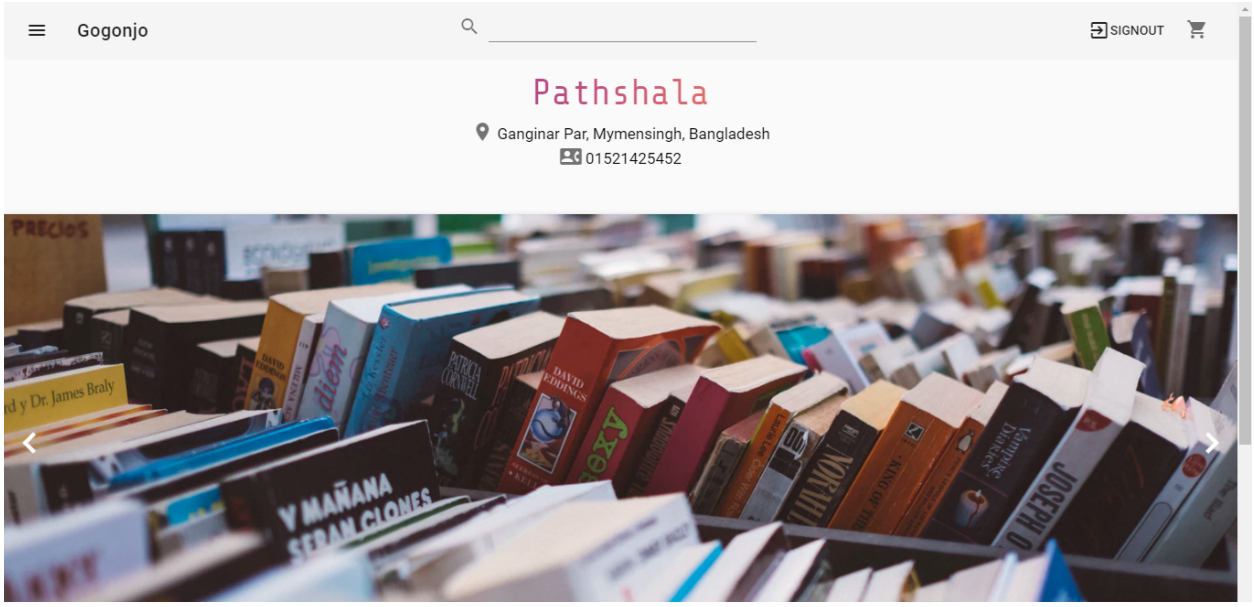

Gogonjo
SIGNOUT
Pathshala
৳ 19280
Revenue
1575
Item sold
3695
Total followers
0
Order in queue
All product
Search
Product name
price
Category
Tag
Color
Size
Action
Norwegian Wood
450
Book
New
Light white
M
The vinchi code
450
Book
New
White
M
মিসির আলি সমগ্র
750
Book
New
White
M
Rows per page:
5
1-3 of 3
MURAKAMI
NORWEGIAN WOOD
All order list
Search
Activate Windows
Go to Settings to activate Windows.

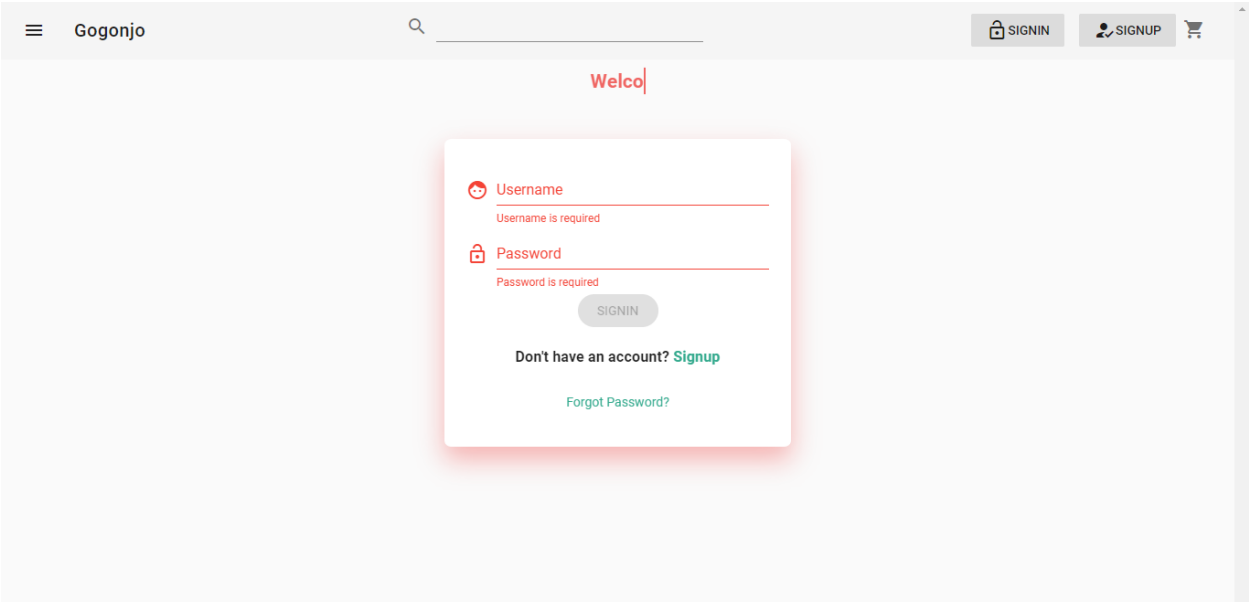

Gogonjo
SIGNOUT
Activate Windows
Go to Settings to activate Windows.

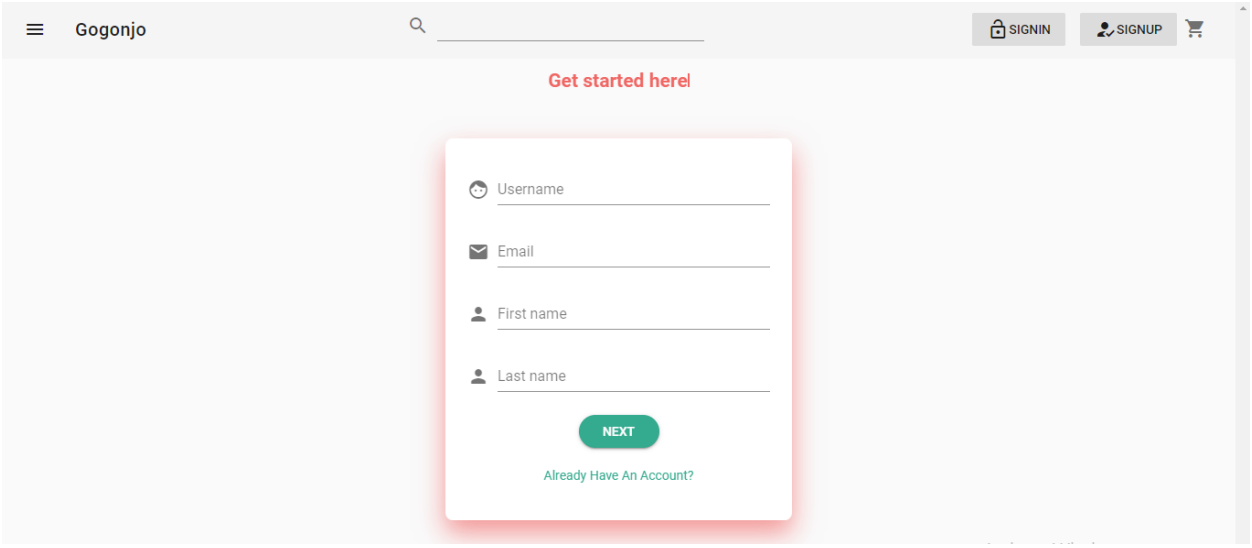

Gogonjo
SIGNOUT
view
৳ 21000
Black Titan Women Watches
Time Tracker
0 (0)
৳ 1750
Men Navy Blue & Black shi...
Wear Drop
4.25 (2)
৳ 199
Spicy Sesame Rice Bowls
Food Mama
4.5 (2)
৳ 349
Hamburger pro
Food Mama
0 (0)
09:41
Activate Windows
Go to Settings to activate Windows.

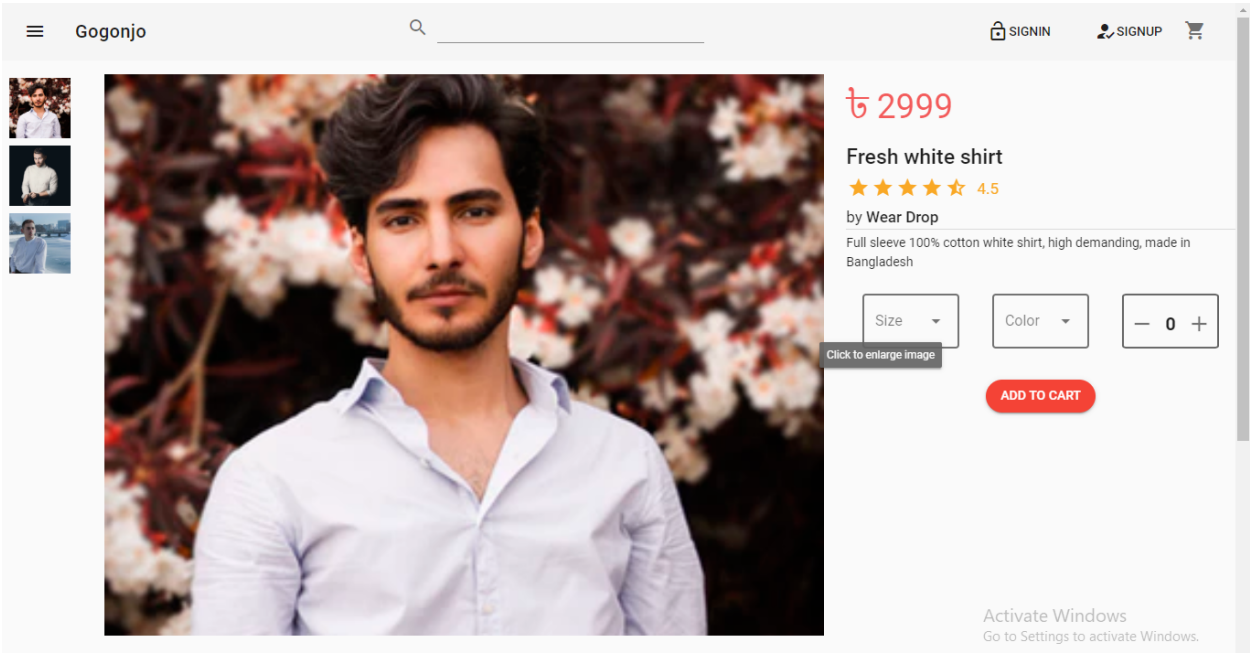

Gogonjo
SIGNOUT
@caffeines
Abu Sadat Md. Sayem
Dhaka, Bangladesh
1598
Followers
65
Following
123
Articles
5
Shop
Activate Windows
Go to Settings to activate Windows.


Gogonjo
sadat
SIGNOUT
Product
Shop
User
Wear Drop
Abu Sadat Md. Sayem
Sadat Sayem
view


Gogonjo
SIGNOUT
Pathshala
Ganginar Par, Mymensingh, Bangladesh
01521425452


Gogonjo
SIGNIN
SIGNUP
Welco
Username
Username is required
Password
Password is required
SIGNIN
Don't have an account? Signup
Forgot Password?


Gogonjo
SIGNIN
SIGNUP
Get started here!
Username
Email
First name
Last name
NEXT
Already Have An Account?


Gogonjo
SIGNIN
SIGNUP
৳ 2999
Fresh white shirt
4.5
by Wear Drop
Full sleeve 100% cotton white shirt, high demanding, made in Bangladesh
Size
Color
0
Click to enlarge image
ADD TO CART
Activate Windows
Go to Settings to activate Windows.

Appendix

GraphQL live: https://gogonjo-graphql.caffeines.now.sh/

Gogonjo live: https://gogonjo.herokuapp.com/