GatorMart Documentation

March 4th, 2022

Sprint 2

Team Members:

Nitin Ramesh
Bhanu Prakash Reddy Palagati
Vamsi Krishna Reddy Komareddy
Gowtham Reddy Eda

Introduction:

The GatorMart application is an online market build with Angular and Go language. Built from the ground up, the application aims to seamlessly connect buyers to their products and sellers to their target audience. Its standout features include allowing the users to select a target audience while buying or selling. This allows to buy products or sell products in single or wholesale quantities while specifying a target audience such as students, professionals, farmers, etc.

Tech Stack:

Frontend: Angular 11.0 with TypeScript
Frontend Test cases: Cypress and Jest Framework
Backend: Go Language
Database: MySQL

Frontend:

The front end of the application is built using Angular 11. Angular is a framework rather than a package that provides all the essential functional requirements out of the box. This will force the developers to follow a pattern, as a result, we have fewer decisions to take for the organization and spend time more on what matters.

We have used material design and there is a package named Angular Materials which helps us to implement the material style components easily and efficiently.

Test cases have been written using Jest for unit testing and Cypress for integration testing, this has ensured the smooth integrated working of the frontend and backend.

The application demo consists of two main views: the “List view” and the “Detailed view”:

1) List View:
   The List view is the main screen that greets the user when opening the GatorMart application. It is a block view of all the products, that have been hand-picked by our algorithms to suit the user’s taste.

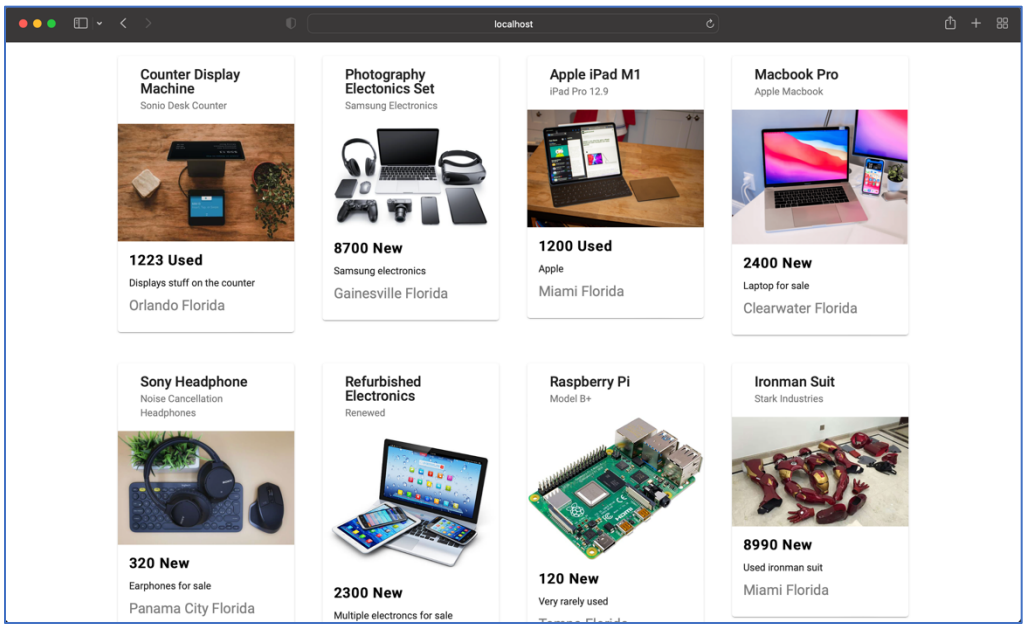


Image: List View

The products are displayed here in a small window shaped material design as shown below:

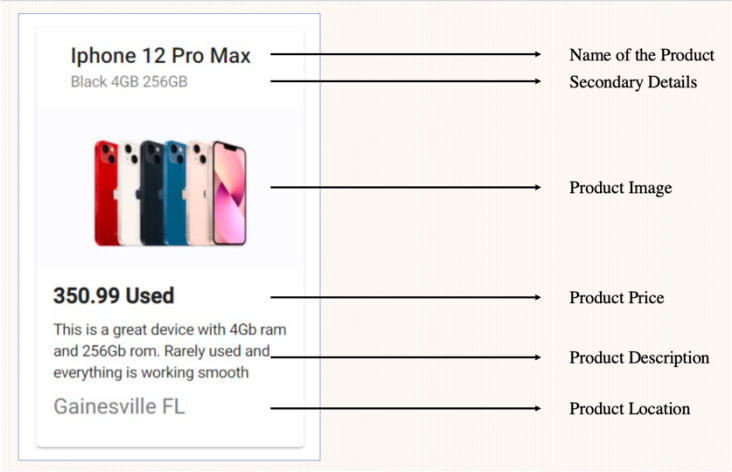


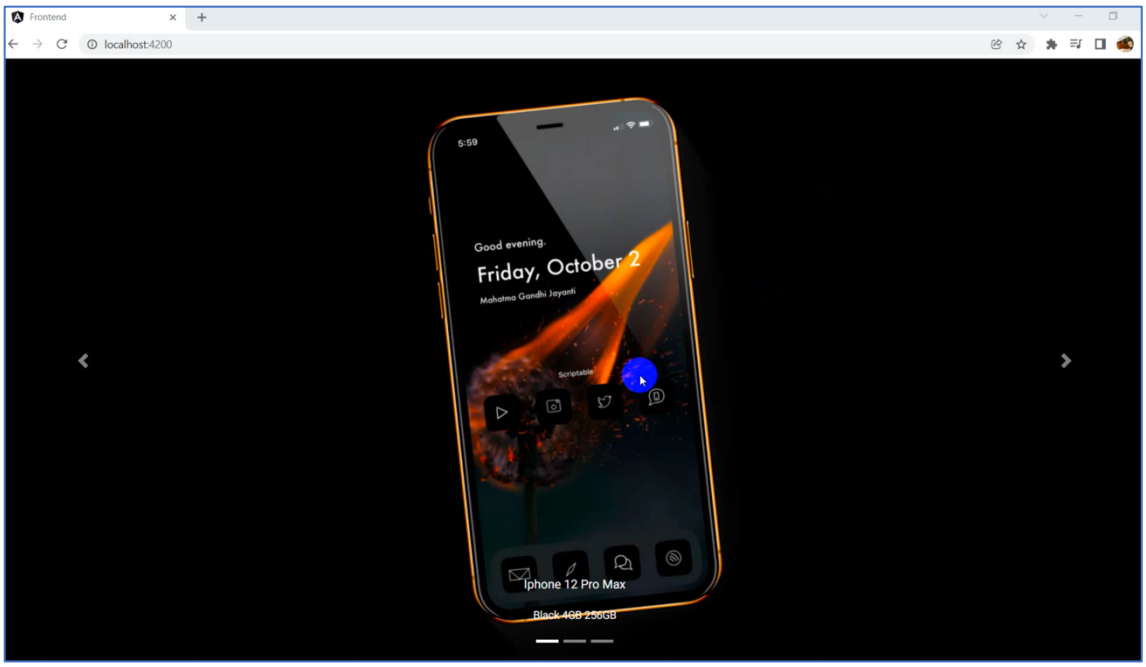
Image: Product Details

This method favors readability, as the most important product information is conveyed to the user directly as they scroll the different listings on the site.

2) Detailed View:

The Detailed View is presented when the user clicks on a product in the List view. This view, as the name suggests, gives more detailed information regarding the selected product such as:

- A carousel of images of the product
- Age of the product
- Detailed description
- Price etc.
- Option to Edit/Remove the Add



Pic: Image carousel

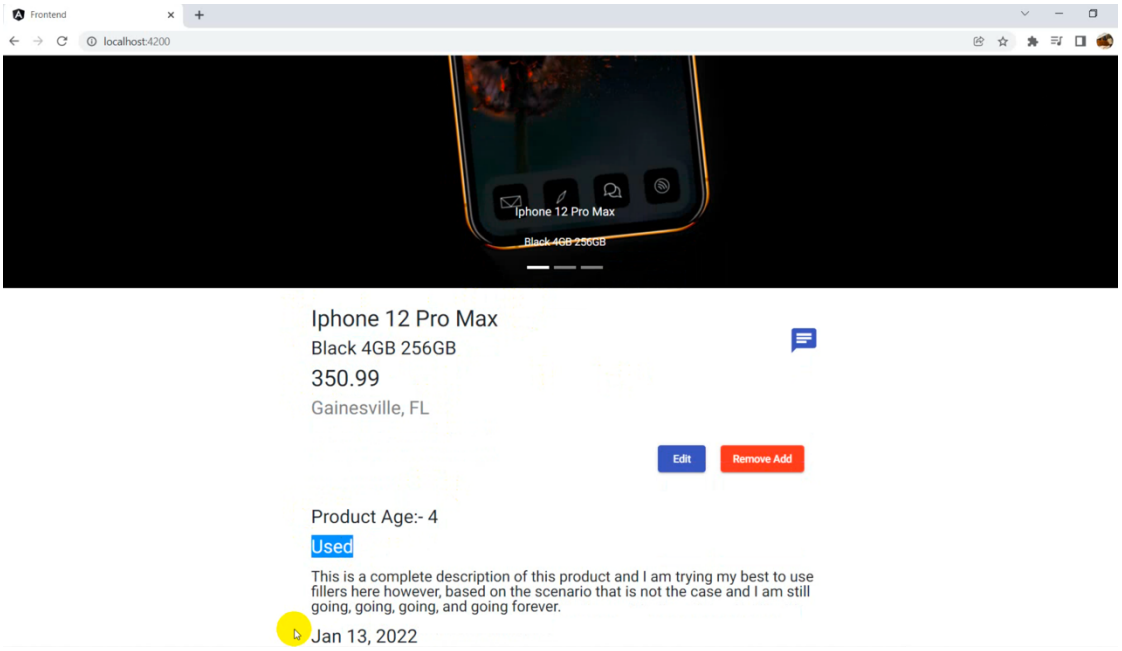


Image: Detailed View

All details are conveniently placed in one area with multiple images to give a better overall understanding of the product.

3) Create Product View:

The create product view confirses of a form where the user can add all relevant details and set the product up for listing. This includes the main image, carousel images and device specific data.

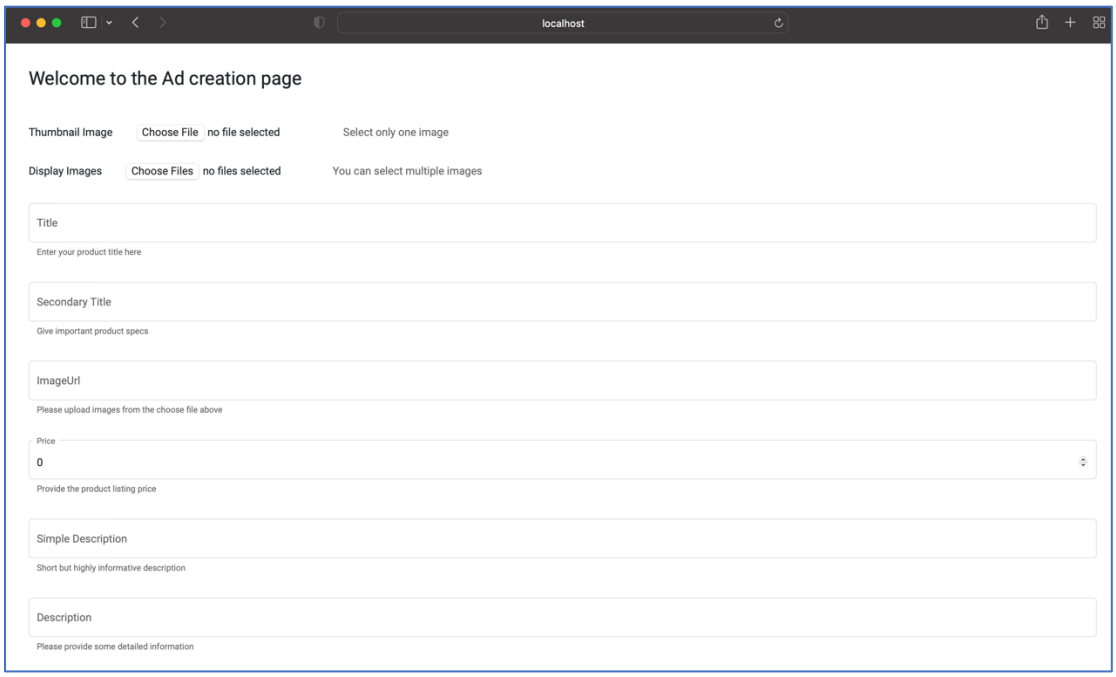


Image: Create Product View

Backend :

1. GetProducts

Method: Get

URL: localhost/products

It displays all the products that are present in the database and displays it to the user.

Possible Response codes: 200,401

Examples: 200 – Success and 400 – User not authorized.

2. SaveProduct

Method – Post

URL: localhost/product

It inserts the data into the database with all the features of the product and the images into the database.

Possible Response codes: 401,400

Examples: 401 – User not authorized and 400 –The product that the user is searching for does not exist.

3. GetProduct

Method: Get

URL: localhost/product/:id

It displays the specific product that the user is searching for.

Possible Response codes: 401,400

Examples: 401 – User not authorized and 400 –The product that the user is searching for does not exist.

4. UpdateProduct

Method: Put

URL: localhost/product/:id

It updates the information about the products, and it modifies the details.

Possible Response codes: 401,400

Examples: 401 – User not authorized and 400 – The product that the user is searching for does not exist.

5. DeleteProduct

Method: Delete

URL: localhost/product/:id

It deletes all the information from the database if in case the product is sold out or for any other reason.

Possible Response codes: 401,400

Examples: 401 – User not authorized and 400 – The product that the user is searching for does not exist.

6. UploadImage

Method: Put

URL: localhost/product/upload

It helps in posting the images from the user to Amazon S3 bucket

Possible Response codes: 401,500,201

Examples: 401 – User not authorized , 400 – Server Error and 201 – Image uploaded successfully.

7. Register

Method: Post

URL: localhost/gatormart/register

Register the user and adds the data into the database. The id of the user will be increasing by 1. Mandatory fields are name, email and password in the user registration page.

Possible Response codes: 200,400

Examples: 200 – Success and 400 -Failure.

8. Login

Method: Post

URL: localhost/gatormart/login

It allows the user to login by checking the username and the password. If it matches it allows the user to login else it displays the failure status.

Possible Response codes: 200,400

Examples: 200 – Success and 400 -Failure.

9. Logout

Method: Post

URL: localhost/gatormart/logut

This method allows the user to log out.

Demo Videos:

FrontEnd→

https://drive.google.com/file/d/18LDNOOYALKxxdG_OW0flN2UZapOGdBGy/view?usp=sharing

Backend → https://drive.google.com/file/d/1H6YM6Oun1QdO3iNhftrhiHto0otlvQDa/view?usp=sharing