Milestone 1 Report

2020 Fall CmpE451 Group 9

29 November 2020

Contents

1 Executive Summary

1.1 Project Description

Our task is to develop an e-commerce website from scratch. The site should have a complete backend with functioning connections to the frontend part. Also, we are supposed to develop a mobile application connecting to the backend part. We choose to develop an iOS application for the mobile part, since our members have more experience developing applications for iOS.

1.2 Work Done So Far

We have started by reviewing the work done last year. We have assessed the requirements, design documents and the project plan, determined the parts that needed improvement. We incorporated our 2 new members to the team, provided them resources and access to our previous works, assisted them by any means possible. Their input was valuable while reviewing the last year's work, since they had fresh eyes and were able to detect any shortcomings and inconsistencies. Then, we decided on the subgroup members, according to our team members' past experience on the field and their preferences.

We have worked for 3 weeks to deliver a website and an iOS application with basic functionalities. We have decided to implement similar parts of the frontend and iOS applications in order to have a consistent development path between them. Backend part of the application is implemented keeping these in mind. At the end, we have finished the home page, signin/signup forms, profile page and some product displays for both iOS and frontend applications. Accordingly, API calls enabling these functionalities are implemented in the backend part. Lastly, since we were meant to present our progress, we had to dockerize the project.

1.3 Road Ahead

We are going to reflect on ourselves in terms of our deliverables and customer feedback to the presentation. We will detect and improve our shortcomings and try to integrate any new aspects the customer demands from us.

We plan to continue our development process as soon as possible. As for the workflow of our project in general, we plan to deliver our deliverables by the end of December, according to our project plan and requirements. We plan to implement a large portion of the functionalities of the website by the 2nd customer milestone.

2 List and Status of Deliverables

Deliverable	**Status**
Revised Requirements	Delivered
Revised Project Plan	Delivered
Frontend	
Home page design	Delivered
Home page backend connection	Delivered
User Login screen design	Delivered
User Login screen backend connection	Delivered
User Sign-up screen design	Delivered
User Sign-up screen backend connection	Delivered
User Profile screen design	Delivered
User Profile screen backend connection	Delivered
Backend	
Database and ER Diagram	Delivered
User Login Endpoint	Delivered
User Sign Up Endpoints	Delivered
User Detail and List Endpoints	Delivered
User Mail Activation Endpoint	Delivered
Product Detail and List Endpoints	Delivered
Location Detail and List Endpoints	Delivered
Mobile	
Home page design	Delivered
User Login screen design	Delivered
User Login screen backend connection	Delivered
User Sign Up screen design	Delivered
User Profile screen design	Delivered
User Profile screen backend connection	Delivered

3 Evaluation of the Status of Deliverables

1. **Milestone 1 - Group Deliverable:**

2. We have reviewed requirements. Although we haven't done many modification on requirements We made small changes, after asking our customer, and uploaded it to GitHub

3. We have reviewed our project plan. We have added customer milestones. Then we have decided our own milestones to control out process. We have uploaded new version of project plan to the GitHub

4. **Backend:**

   - User Sign up: Sign up functionality implemented with respect to requirements 1.1.1.1, 1.1.1.2, 1.1.1.5. Also in 1.1.1.6 we have impelemented the requirement for verifying email but users will be asked to accept Bazaar contract later. Other requirements in Sign up will be implemented later.
   - User Login: Requirements 1.1.2.1 and 1.1.2.3 are completed. However other requirements are in progress
   - User, Product, Location: All user list, product list and location lists are returned as endpoints. Also details of user, product and location are returned as endpoints

5. **Frontend:**

   - User sign up: All related requirements (in 1.1.1. Sign Up part of the requirements document) for the user sign up has been completed except for 1.1.1.3,1.1.1.4, 1.1.1.5, 1.1.1.6. The backend connection for user sign up is also completed.
   - User profile: User profile is done for the customer and vendors, but not for admins yet. The design and basic backend connectivity of the user profile is done, however this view needs more work in order to be done for now
   - User login: All related requirements (1.1.2. Sign In part of the requirements document) for the user login has been completed except for 1.1.2.3 and 1.1.2.4. The backend connection of user sign up view is done.
   - Home page: Home page design is completed. Products and their details are coming from the backend properly. Categories did not work now. Right now, product details are not shown upon clicking on a product, this was beyond the scope of Milestone 1. Backend connection for the home page is not implemented yet since there are some functionalities that aren't available on the backend side that is needed for the home page backend.

6. **iOS:**

   - Home page: Home page design is completed with category filtering functionality for the "Products of the Day". However, searching for products functionality has been decided to be completed after the Categories View is completed since they will be linked to each other in a way. Right now, product details are not shown upon clicking on a product, this was beyond the scope of Milestone 1. Backend connection for the home page is not implemented yet since there are some functionalities that aren't available on the backend side that is needed for the home page backend.
   - User login: All related requirements (1.1.2. Sign In part of the requirements document) for the user login has been completed except for 1.1.2.3 and 1.1.2.4. The backend connection of user sign up view is not done yet.
   - User sign up: All related requirements (in 1.1.1. Sign Up part of the requirements document) for the user sign up has been completed except for 1.1.1.3, 1.1.1.5, 1.1.1.6. The backend connection for user sign up is also completed.
   - User profile: User profile is done for the customer and vendors, but not for admins yet. The design and basic backend connectivity of the user profile is done, however this view needs more work in order to be done for now.

   As can be seen, we have mostly been conformant to our project plan except for some exhaustive requirements and backend connections for some views. The work that has been done on the mobile part except for those who are stated above as "not completed" don't need further improvement, which is a sign that mobile project is proceeding as expected.

7. **API documentation:** API documentation is essential to the project as both frontend and mobile have to call for API requests, and there needs to be a solid basis of what backend can supply. The supply shall be shown in a way that everyone involved in the project should understand and use easily, hence an API documentation is needed.

   We managed to deliver the API documentation of the as project plan dates required, and we plan to improve the documentation as time goes on. We also have a Postman collection that easily shows what our API is capable of, which puts the abstract API documentation into practice.

8. **GitHub Page:**

   GitHub page is very essential for a development team consisting of 12 people, because without sufficient communication between the developers, a project easily can go wrong in any way possible. Therefore, our

team always made our GitHub page stay up-to-date, used issues very effectively to make decisions, especially about design, meeting times, tools and processes.

Our GitHub wiki is also up-to-date with team meeting notes and requirements.

4 Challenges We Met

Our project requires critical effort in order to achive our needs.

The main issue was that all of our team members were beginners for such project. We had to learn all the tools and processes we needed to know and it took a long time to do.

For example, Django is a framework which is helpful for various needs, if one knows how to handle it. It was tough to both learn and apply such a diverse framework. Since we are still beginners and we still have so much to learn, we hope for our project to increase in quality as time goes on.

Dockering the project was one of the lesser problems we met since we deployed our previous project similarly. However, the server we deploy our Docker containers, Amazon EC2 instances might be a problem in long term. It is hard work to manage a EC2 server while trying to be a free AWS user.

Since we deploy our project manually, automating the deployment might be a good idea; however, there are more critical things we prioritize so we plan to continue deploying manually for now.

5 Requirements

In this part, we will try to highlight and reason all changes that we have done in requirements.

GLOSSARY

Admin: A type of user who has special privileges to maintain the order of the e-commerce platform.

All Users: Guests, admins, vendors and customers.

Customer: A person/profile who buys goods using the e-commerce platform. A customer must be logged-in and has specific privileges.

Dockerize: Contain all software in a folder such that everything the program uses is contained in that folder. This eases portability.

Guest: A type of user who is not logged-in and has the following privileges: searching products and vendors, filtering through products, reading comments.

List: A list of items on the platform that a customer is interested in but is not ready to purchase them yet.

Modern Web Browsers: Browsers supporting HTML5 and Bootstrap 4.0.

Order Processing: A sequence of processes starting from the customer creating an order request and ending at customer receiving the order.

Semantic Search: Searching the intent of the typed keywords of the user instead of the literal string match.

Shopping Cart: A list of items on the platform that contains the goods selected by a customer or a guest for purchase until the transaction is completed.

Stock: The number of available product.

Transaction: A financial interchange of money for goods between a vendor and a customer, completion of payment for goods.

User: A person or profile who uses or operates on the e-commerce platform. A user may or may not be logged-in. It can be of type customer, vendor, guest or admin.

Valid payment information: Payment information that expresses a credit card with a 16-digit card number and a valid expiration date.

Vendor: A person/profile who sells goods using the e-commerce platform. A seller must be logged-in and has some specific privileges.

1. FUNCTIONAL REQUIREMENTS

1.1. User Requirements

1.1.1. Sign Up

1.1.1.1. Guests should sign up by providing their email address, name, surname and by also providing a valid email-password pair.
1.1.1.2. Guests should sign up either as a vendor or a customer.
1.1.1.3. Guests who want to register as a vendor are expected to specify the location of their store(s) through Google Maps as well as the specifications specified at 1.1.1.1..
1.1.1.4. Guests should be able to sign up with their Google account.
1.1.1.5. All users should delete their accounts if they want.
1.1.1.6. In order to fulfill the registration process, users shall verify their email addresses and aggreeing the Bazaar contract, a contract where users approve to the legal terms of the website.

1.1.2. Sign In

1.1.2.1. Customers and vendors should sign in with their email address and password.
1.1.2.2. Customers and vendors should be able to sign in with their

Google account if they have signed up via their Google account.
1.1.2.3. Customers and vendors shall verify their account in order to use their account via the email sent to their address.
1.1.2.4. Customers and vendors have option to reset password if it is forgotten.

1.1.3. Social Interactions and Communication

1.1.3.1. Customers should communicate with vendors via a direct messaging system. Vendors should receive the messages and answer the customers.
1.1.3.2. Customers should start a conversation with the vendor of a certain product from that product's page, whether it is before customer buys the product or after the product has been ordered and delivered.
1.1.3.3. Customers and vendors should see and manage their direct messages on the 'Messages' page.

1.1.4. Products

1.1.4.1. All users should display a product's page including the following information: image, label, brand, price, vendor info, stock status, comments, rating.
1.1.4.2. Customers should comment on the products they have ordered.
1.1.4.3. Customers should rate the products they have ordered.

1.1.5. Categories

1.1.5.1. All users should discover the categories.
1.1.5.2. All users should search for products in the given category and filter/sort a category.
1.1.5.3. Admins should add, edit or delete categories as they want.

1.1.6. Search

1.1.6.1. All users should search for products by their name, label or brand.
1.1.6.2. All users should filter searched products based on several properties: average customer review, brand, vendor, price range, stock status.
1.1.6.3. All users should sort searched products based on several properties: bestsellers, newest arrivals, price, average customer review, number of comments, ratings.
1.1.6.4. All users should search for vendors by vendors' name and location.

1.1.7. Lists

1.1.7.1. Customers should create private lists.

1.1.7.2. Customers should name, edit and delete a list.
1.1.7.3. Customers should add a product to a list.
1.1.7.4. Customers should search for products in a list and filter/sort the lists.

1.1.8. Cart

1.1.8.1. Customers should add products to their cart.
1.1.8.2. Customers should remove the products from their cart.
1.1.8.3. Customers should change the amount of products in their cart.

1.1.9. Buying

1.1.9.1. Customers should buy the products in their cart by proceeding to the 'Payment' page.
1.1.9.2. Customers shall provide valid payment information and shipping address in order to complete the payment.
1.1.9.3. Customers should be able to save their payment and shipping information for future use. These information are private, can be seen only by the customer.
1.1.9.4. Customers should be able to select their address as the shipping address.

1.1.10. Vendor Profile

1.1.10.1. Vendors shall have a profile page that provides all sufficient information about the vendor: name, address(es), products, rating, comments made to their products.
1.1.10.2. All users should see all the products of the vendor in his profile.
1.1.10.3. All users should reach a certain product page via the vendor's profile.

1.1.11. Selling

1.1.11.1. Vendors should put their products on their profile by providing all the information which is given to the customers and which is displayed on the product's page.
1.1.11.2. Vendors should receive an order with all sufficient information: customer name, address and product IDs.

1.1.12. Orders (Customers)

1.1.12.1. Customers shall have an 'Orders' page. On this page, customers should follow their orders.
1.1.12.2. Customers should see their active orders with the following information: order id; products with their image, their label, their price, and the vendors; total price; order time; estimated delivery

time.
1.1.12.3. Customers should cancel their active orders.
1.1.12.4. Customers should see their delivered orders with such details:

* Order ID, date and current status.
* Product ID, name, label, price, picture, vendor etc.

1.1.12.5. Customers should return their delivered orders when an approved reason is specified.

1.1.13. Orders (Vendors)

1.1.13.1. Vendors shall have an 'Orders' page where vendors should see their active and delivered orders with such details:

* Order ID, date and current status.
* Product ID, name, label, price, picture, vendor etc.

1.1.13.2. Vendors should be able to modify the order status as 'in preparation','cancelled' ,'shipped' and 'delivered'.
1.1.13.3. Vendors should be able to cancel an order during the order processing stage in case of a problem.
1.1.13.4. Vendors should communicate with the platform admins about a certain order.

1.1.14. Notification

1.1.14.1. Customers should choose to get notifications for price changes of a certain product.
1.1.14.2. Customers should set alerts for a certain price of a certain product. Customers shall be notified if the price of that product goes below the chosen amount.

1.1.15. Admin Panel

1.1.15.1. Admins shall have an admin panel to administrate the platform where he can:

* ban users.
* delete comments.
* communicate with other users if needed.
* see ongoing transactions(Orders)

1.1.15.2. Admins should communicate with vendors about orders if they want to.

1.2 System Requirements

1.2.1. Searching/ Filtering/ Sorting

1.2.1.1. The system shall let guests, admins and customers search for products by the properties: label, brand, price.

1.2.1.2. The system shall let all users search for vendors the properties: name, location.
1.2.1.3. The system shall support semantic search, which is used for finding semantically similar products and vendors based on the context information.
1.2.1.4. The system shall let guests, admins, vendors and customers filter the searched products or the products of a certain category. The filtering shall be based on several properties:

* Average customer review.
* Brand.
* Vendor.
* Price range.
* Stock status.

1.2.1.5. The system shall let all users sort the searched products or the products of a certain category. The sorting shall be based on several properties: bestsellers, newest arrivals, price, average customer reviews, number of comments, rating.

1.2.2. Transaction

1.2.2.1. The system shall support transactions from customers' banking account to the sellers' account.
1.2.2.2. The system shall provide a secure medium for the transactions.

1.2.3. Recommendation

1.2.3.1. The system shall give recommendations to customers based on their search and order history.

1.2.4. Notification

1.2.4.1. The system shall notify the customers for the price changes of products that they wish to get notified.
1.2.4.2. The system shall notify the customers if a product they ordered is out of stock or the order is cancelled for any other reason.
1.2.4.3. The system should notify the vendors if an active order to buy their product is cancelled by the customer.
1.2.4.4. The system shall notify the vendors if admins report a problem about selling.

1.2.5. Interactions

1.2.5.1. The system shall let customers and vendors communicate via direct messaging mechanism.
1.2.5.2. The system shall provide a messaging mechanism for vendors to communicate with admins about an order or a product.

2. NON-FUNCTIONAL REQUIREMENTS

2.1. Availability

2.1.1. The platform shall work as a web site via modern web browsers.
2.1.2. The platform shall be available for use as an IOS application.
2.1.3. The platform shall be in English.
2.1.4. The platform shall be deployed on Amazon EC2 server

2.2. Reliability

2.2.1. Unless intentionally shut down, the system should run 24/7.
2.2.2. The platform shall use a dedicated server, be portable and dockerized.
2.2.3. The server shall scale if necessary.
2.2.4. The server shall additionally backup user data every 24 hours.

2.3. Security

2.3.1. All website interactions shall use SSL.
2.3.2. All money transactions shall use 3-D Secure.
2.3.3. All user data, especially email-password shall be encrypted.
2.3.4. Admins should keep track of transaction in case of a problem.

2.4. Response

2.4.1. The server shall respond to every request under 5 seconds.
2.4.2. The server should be able to handle up to 100 requests every minute.

2.5. Privacy

2.5.1. The platform shall follow the rules defined by GDPR and KVKK in terms of collecting and storing any personal information, contact information, copyrighted contents and everything related to these paradigms.
2.5.2. The platform shall follow the standards introduced by the World Wide Web Consortium (W3C), as well as the W3C Activity Streams Protocol.
2.5.3. Users should be able to read privacy policy and they must accept it for using the platform.

Changes we made

- We add 1.1.2.3. for account verification. It can prevent opening more than one for the same mail address and spam accounts.

- We add 1.1.2.4. for forgotten password. So it will be more user friendly . It is highly possible to forgot password for a while. Users tend to forgot

something.

- We change 1.1.4.1. , 1.1.5.1. , 1.1.5.2. , 1.1.6.1. , 1.1.6.2 , 1.1.6.3. , 1.1.6.4 , 1.1.10.2 and 1.1.10.3. by All users. Because it is more rational to all users have see ,search, filter option without sign up.

- We add stock status to 1.2.1.4. .Because if product is out of stock, this should be indicated and the user should be able to filter. Otherwise system will accept to order.

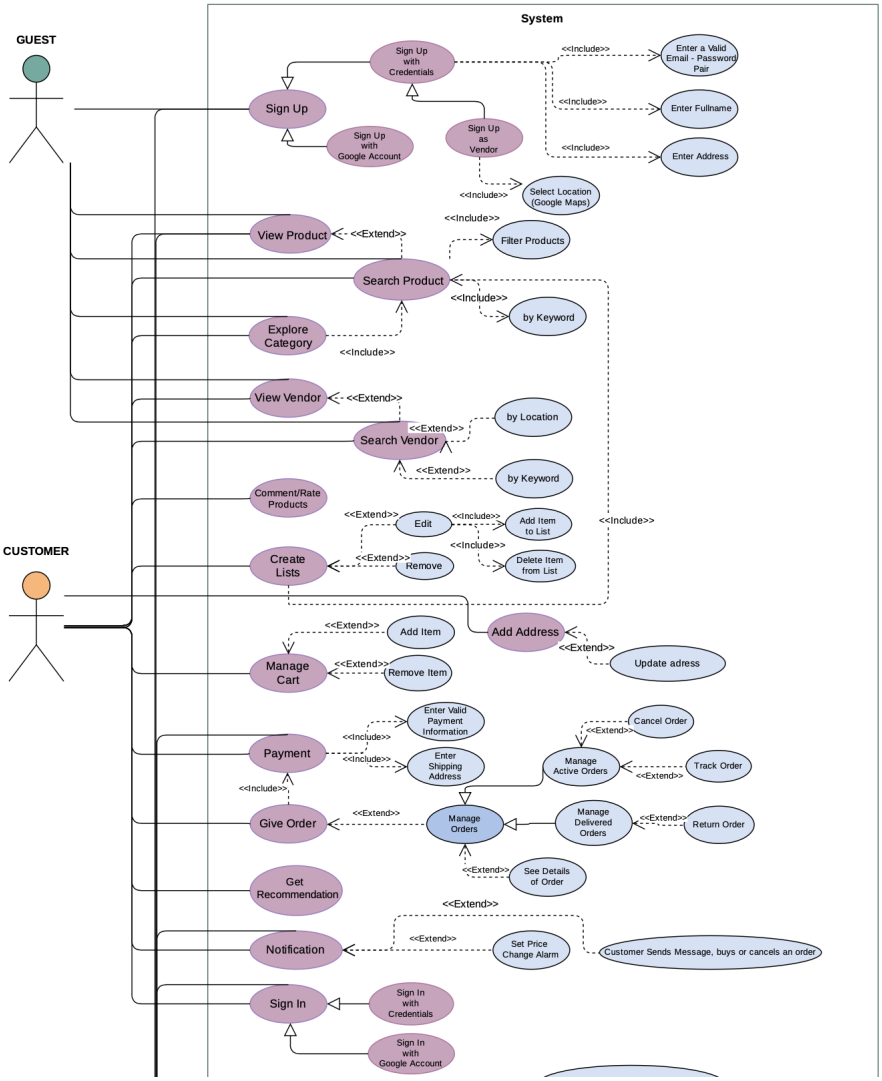
- We change 2.1.2. because mobile app team was more experienced in IOS.

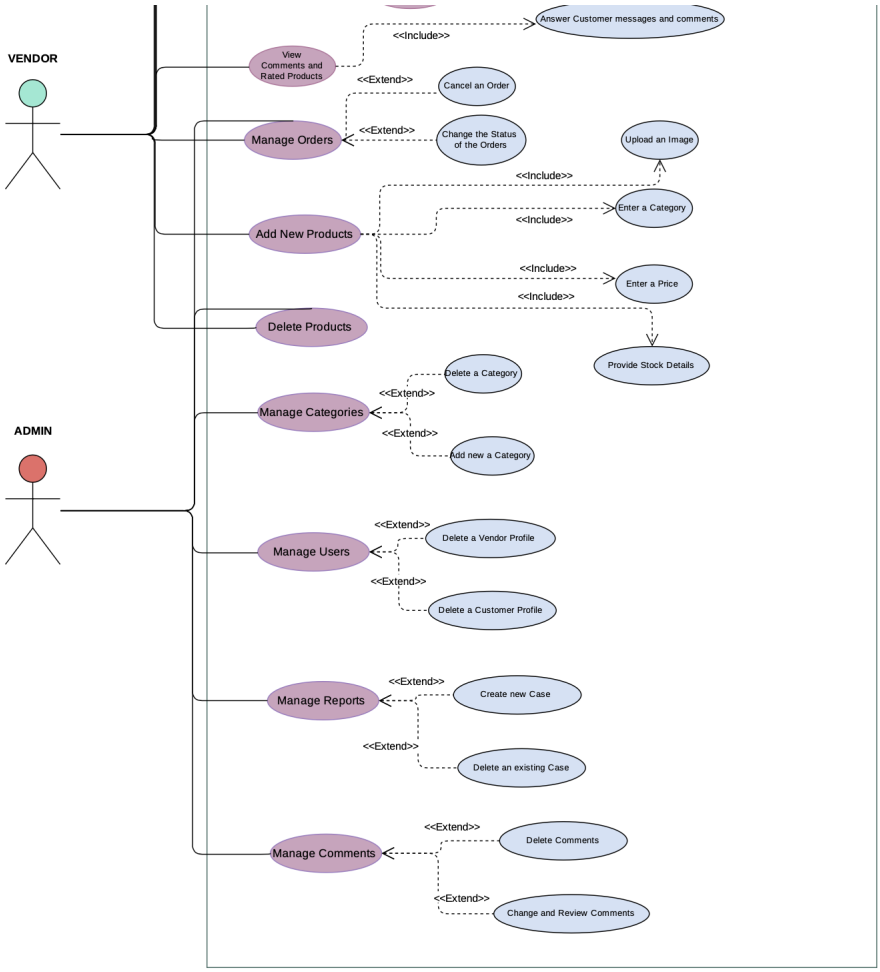
- We add 2.3.3. in order to increase security.

6 Design documents

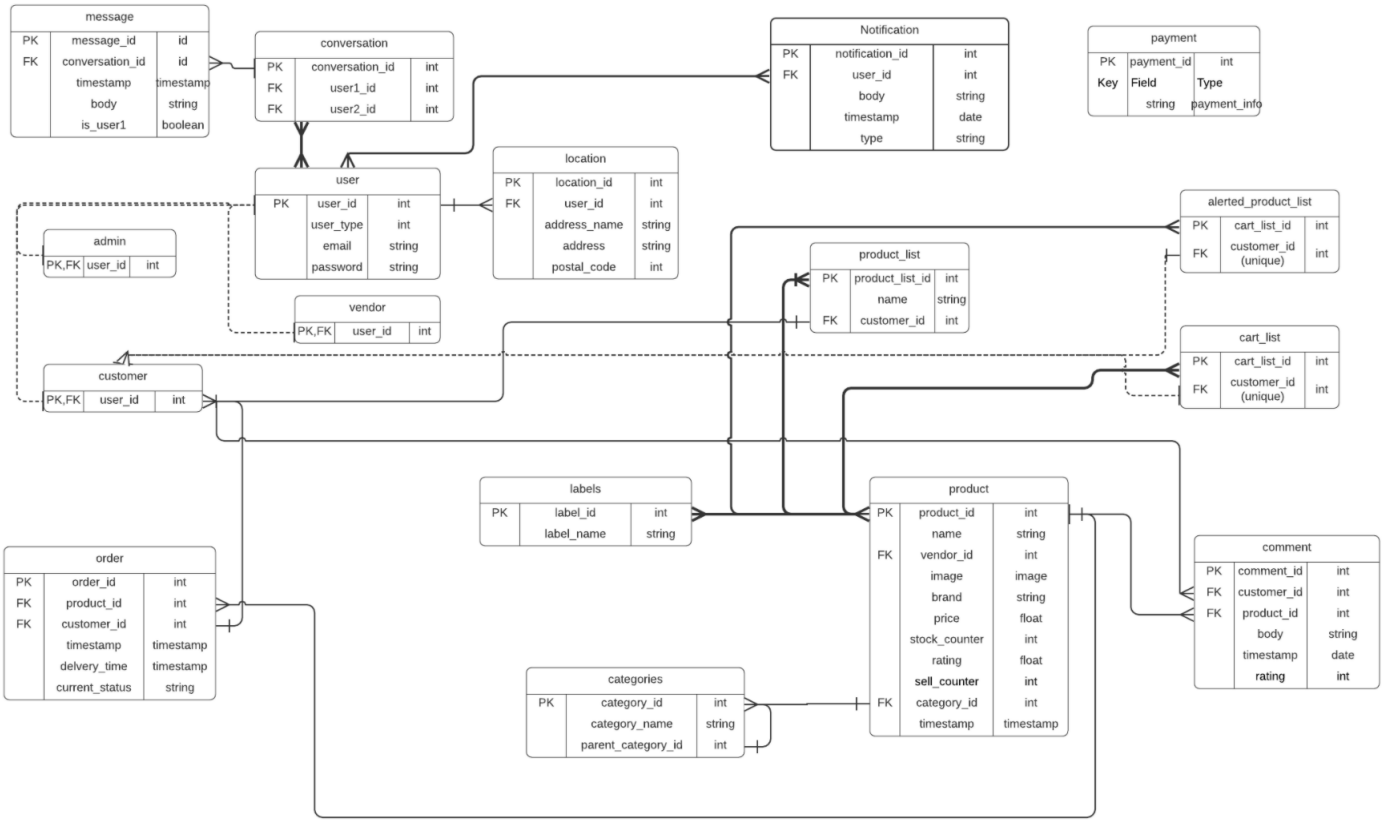
In this part, we will try to highlight and reason all changes that we have done in design documents

Use Case Diagram


System
GUEST
CUSTOMER
Sign Up with Credentials
Sign Up
Sign Up with Google Account
Sign Up as Vendor
<<Include>>
Enter a Valid Email - Password Pair
Enter Fullname
Enter Address
Select Location (Google Maps)
View Product
<<Extend>>
Filter Products
Search Product
by Keyword
Explore Category
View Vendor
Search Vendor
by Location
Comment/Rate Products
Edit
Add Item to List
Delete Item from List
Remove
Create Lists
Add Address
Update adress
Add Item
Manage Cart
Remove Item
Enter Valid Payment Information
Payment
Enter Shipping Address
Cancel Order
Manage Active Orders
Track Order
Give Order
Manage Orders
Manage Delivered Orders
Return Order
See Details of Order
Get Recommendation
Notification
Set Price Change Alarm
Customer Sends Message, buys or cancels an order
Sign In
Sign In with Credentials
Sign In with Google Account

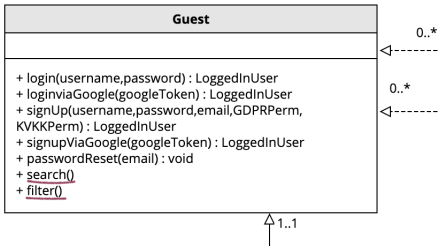


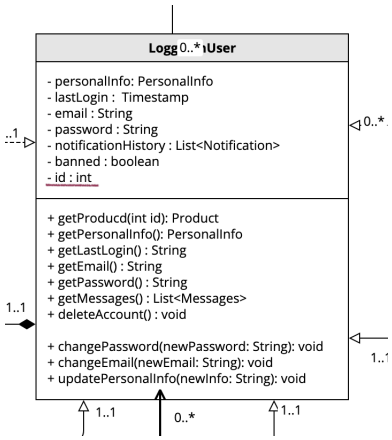
Database ER Diagram



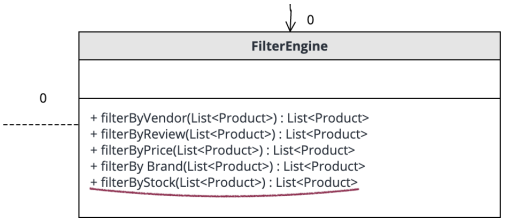
Class Diagram

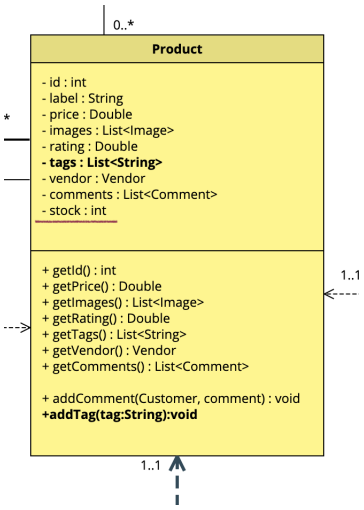
For detailed diagram see the link Class Diagram
Due to change in 1.1.6 requirements. We change Guest class and remove User class.





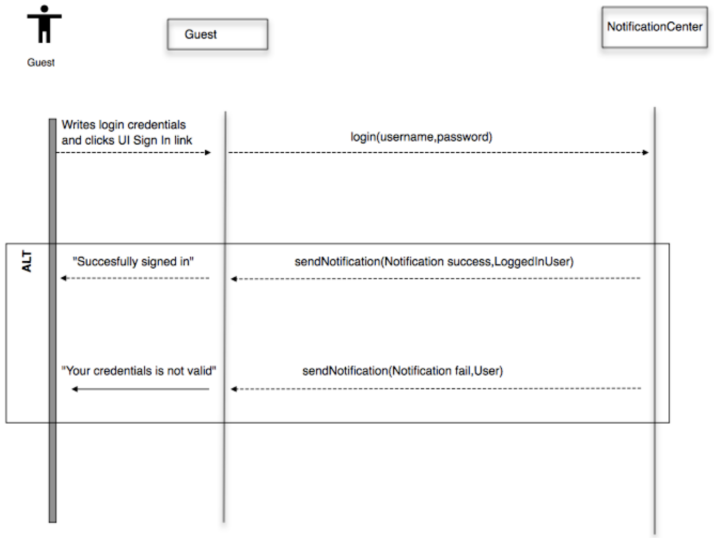
Due to change in 1.2.1.4 Requirements. We add stock status and filtering options according to stock .



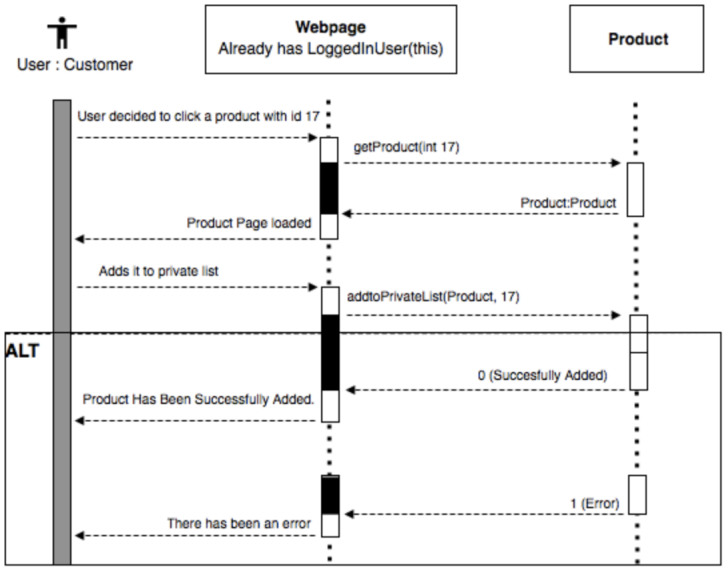


Sequences Diagrams

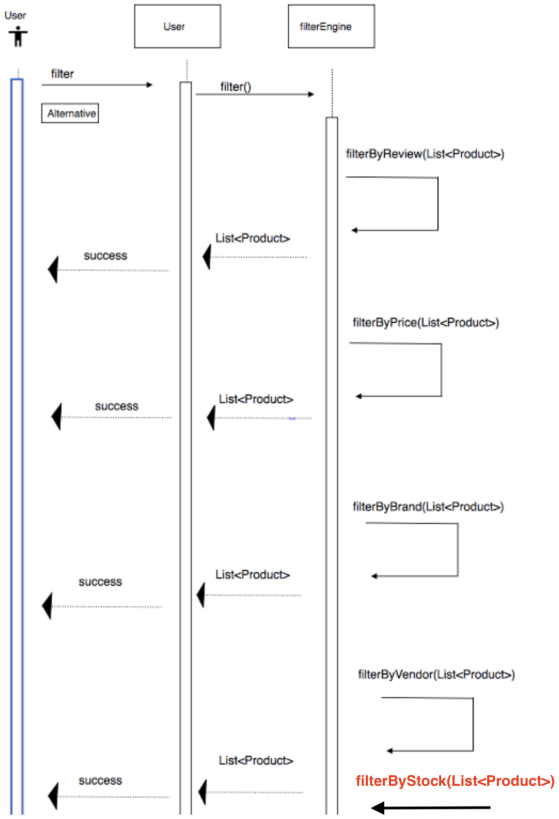
- **Login via Username**



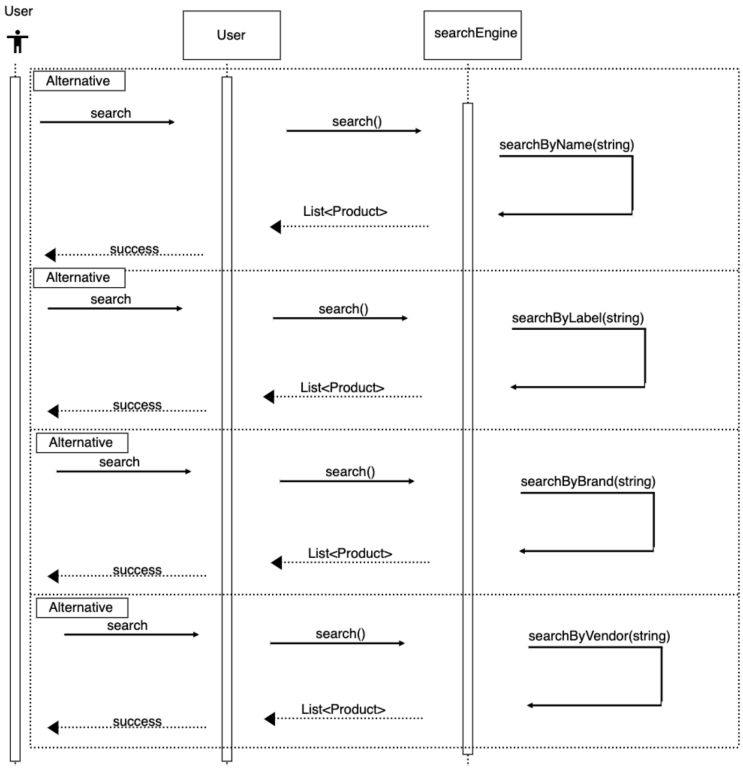
- **Add to List**



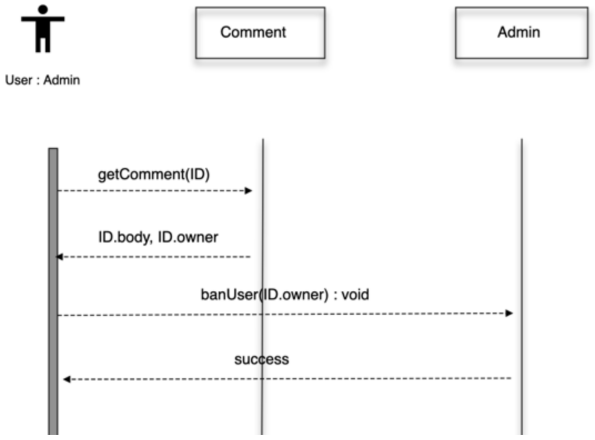
- **Filter**



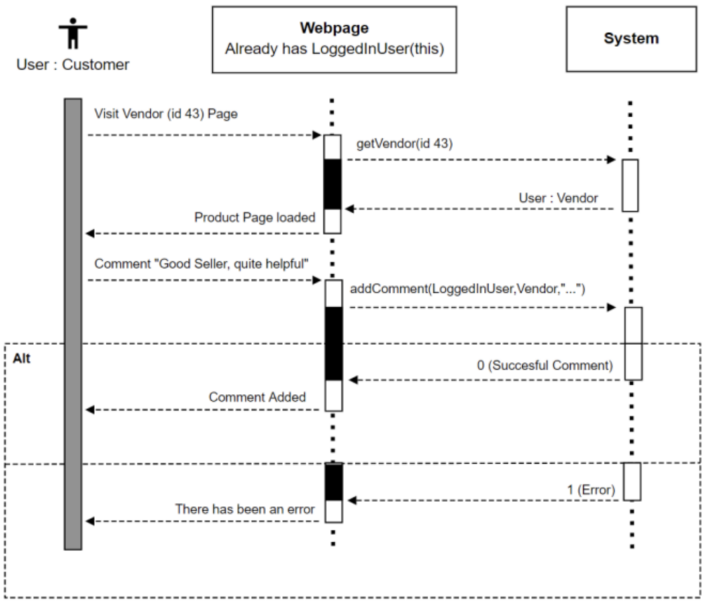
- **Search**


User
User
searchEngine
Alternative
search
search()
searchByName(string)
List<Product>
success
Alternative
search
search()
searchByLabel(string)
List<Product>
success
Alternative
search
search()
searchByBrand(string)
List<Product>
success
Alternative
search
search()
searchByVendor(string)
List<Product>
success

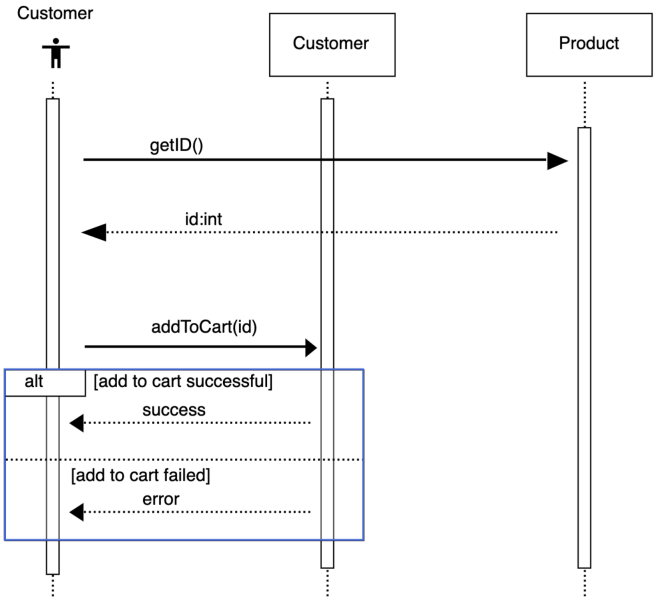
- **Ban User**


User : Admin
Comment
Admin
getComment(ID)
ID.body, ID.owner
banUser(ID.owner) : void
success

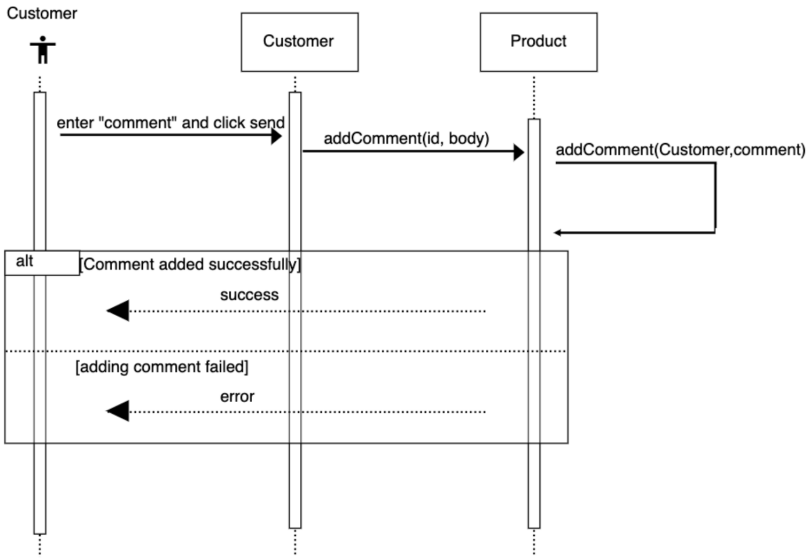
- **Comment to Vendor**



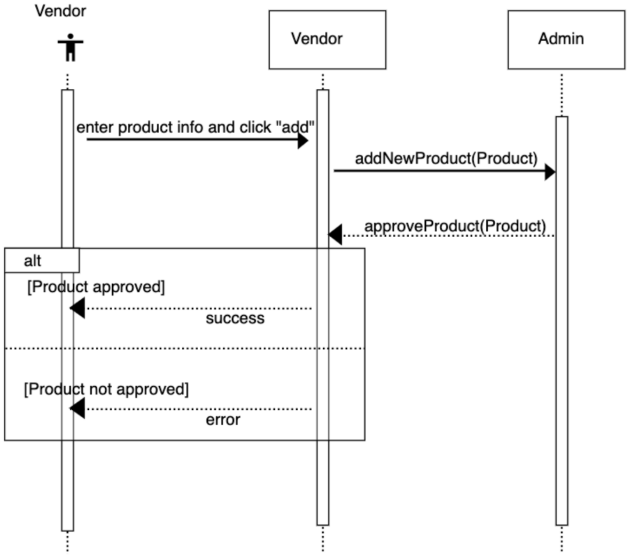
- **Add to Cart**



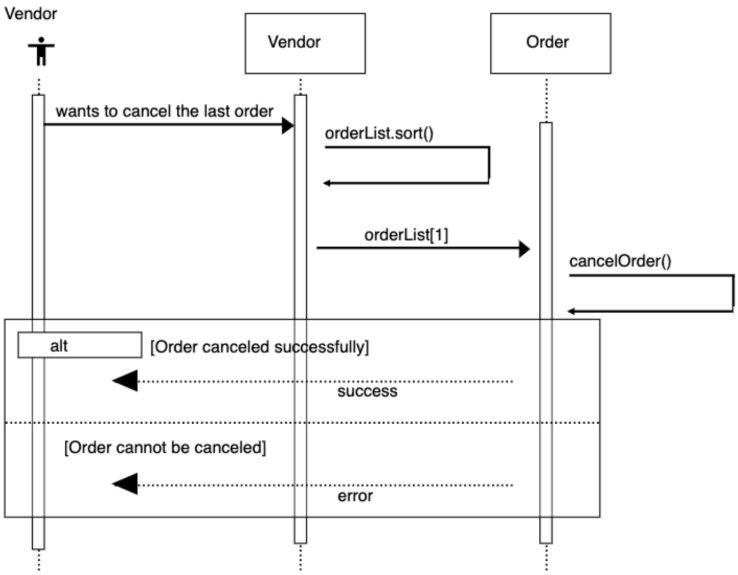
- **Comment on Product**



- **Add New Product**



- **Vendor Cancels 2nd to Last Received Order:**



7 Project Plan

	Name	Duration	Start	Finish	Predecessors	Resource Names
46	***** Backend**	15 days?	**11/3/20 8:00 AM**	**11/23/20 5:00 PM**	**45**	
47	Database Tables for Users	8 days	11/3/20 8:00 AM	11/12/20 5:00 PM		Backend team
48	Sign-in & Sign-up & Password Features	8 days?	11/3/20 8:00 AM	11/12/20 5:00 PM		Backend team
49	Custom Milestone #1	0 days?	11/13/20 8:00 AM	11/13/20 8:00 AM	47;48	
50	Product Profile	7 days?	11/13/20 8:00 AM	11/23/20 5:00 PM	49	Backend team
51	User Profiles	7 days?	11/13/20 8:00 AM	11/23/20 5:00 PM	49	Backend team
52	Unit Tests	2 days?	11/20/20 8:00 AM	11/23/20 5:00 PM		Backend team
53	***** Frontend**	15 days?	**11/3/20 8:00 AM**	**11/23/20 5:00 PM**	**45**	
54	Home Page	8 days	11/3/20 8:00 AM	11/12/20 5:00 PM		Frontend team
55	Signup/Login	8 days	11/3/20 8:00 AM	11/12/20 5:00 PM		Frontend team
56	Navbar	8 days	11/3/20 8:00 AM	11/12/20 5:00 PM		Frontend team
57	Custom Milestone #1	0 days?	11/13/20 8:00 AM	11/13/20 8:00 AM	54;55;56	
58	Header/Footer	7 days	11/13/20 8:00 AM	11/23/20 5:00 PM	57	Frontend team
59	Profile Page	7 days	11/13/20 8:00 AM	11/23/20 5:00 PM	57	Frontend team
60	Unit Tests	2 days	11/20/20 8:00 AM	11/23/20 5:00 PM		Frontend team
61	***** iOS**	15 days?	**11/3/20 8:00 AM**	**11/23/20 5:00 PM**	**45**	
62	Home Page	8 days	11/3/20 8:00 AM	11/12/20 5:00 PM		iOS team
63	Custom Milestone #1	0 days?	11/13/20 8:00 AM	11/13/20 8:00 AM	62	
64	Signup/Login	7 days	11/13/20 8:00 AM	11/23/20 5:00 PM	63	iOS team
65	Sign Up via Google	7 days	11/13/20 8:00 AM	11/23/20 5:00 PM	63	iOS team
66	User Profiles	7 days	11/13/20 8:00 AM	11/23/20 5:00 PM	63	iOS team
67	Unit Tests	2 days	11/20/20 8:00 AM	11/23/20 5:00 PM		iOS team
68	Customer Milestone #1	0 days?	11/24/20 5:00 PM	11/24/20 5:00 PM	46;53;61	
69	***** Backend**	24 days?	**11/25/20 8:00 AM**	**12/28/20 5:00 PM**	**68**	
70	Implementing Order Structure	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Backend team
71	Implementing Customer Lists	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Backend team
72	Implementing Shopping Cart	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Backend team
73	Implementing Product Adding	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Backend team
74	Custom Milestone #2	0 days?	12/11/20 5:00 PM	12/11/20 5:00 PM	70;71;72;73	
75	Implementing Comment Structure	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	74	Backend team
76	Implementing Search & Sort Functionality	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	74	Backend team
77	Implementing Messaging System	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	74	Backend team
78	Unit Tests	2 days	12/25/20 8:00 AM	12/28/20 5:00 PM		Backend team
79	***** Frontend**	24 days?	**11/25/20 8:00 AM**	**12/28/20 5:00 PM**	**68**	
80	Implementing Order Structure	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Frontend team
81	Implementing Customer Lists	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Frontend team
82	Implementing Shopping Cart	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Frontend team
83	Implementing Product Adding	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		Frontend team
84	Custom Milestone #2	0 days?	12/11/20 5:00 PM	12/11/20 5:00 PM	80;82;83;81	
85	Implementing Comment Structure	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	84	Frontend team
86	Implementing Search & Sort Functionality	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	84	Frontend team
87	Implementing Messaging System	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	84	Frontend team
88	Unit Tests	2 days	12/25/20 8:00 AM	12/28/20 5:00 PM		Frontend team

		Name	Duration	Start	Finish	Predecessors	Resource Names
89		***** iOS**	24 days?	**11/25/20 8:00 AM**	**12/28/20 5:00 PM**	**68**	
90		Implementing Order Structure	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		iOS team
91		Implementing Customer Lists	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		iOS team
92		Implementing Shopping Cart	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		iOS team
93		Implementing Product Adding	12 days	11/25/20 8:00 AM	12/10/20 5:00 PM		iOS team
94		Custom Milestone #2	0 days?	12/11/20 5:00 PM	12/11/20 5:00 PM	90;92;93;91	
95		Implementing Comment Structure	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	94	iOS team
96		Implementing Search & Sort Functionality	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	94	iOS team
97		Implementing Messaging System	11 days	12/14/20 8:00 AM	12/28/20 5:00 PM	94	iOS team
98		Unit Tests	2 days	12/25/20 8:00 AM	12/28/20 5:00 PM		iOS team
99		Milestone #4	0 days?	12/29/20 5:00 PM	12/29/20 5:00 PM	69;79;89	
100		***** Backend**	10 days	**12/30/20 8:00 AM**	**1/12/21 5:00 PM**	**99**	
101		Recommendation Engine	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Backend team
102		Notification Engine	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Backend team
103		Annotations	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Backend team
104		Optimization	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Backend team
105		***** Frontend**	10 days	**12/30/20 8:00 AM**	**1/12/21 5:00 PM**	**99**	
106		Recommendation Engine	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Frontend team
107		Notification Engine	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Frontend team
108		Annotations	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Frontend team
109		Optimization	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		Frontend team
110		***** iOS**	10 days	**12/30/20 8:00 AM**	**1/12/21 5:00 PM**	**99**	
111		Recommendation Engine	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		iOS team
112		Notification Engine	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		iOS team
113		Annotations	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		iOS team
114		Optimization	10 days	12/30/20 8:00 AM	1/12/21 5:00 PM		iOS team
115		End-to-End Testing	4 days	1/13/21 8:00 AM	1/18/21 5:00 PM	100;105;110	To be assigned
116		Milestone #5 Deployement	0 days?	1/19/21 5:00 PM	1/19/21 5:00 PM	115	

8 User Scenarios from the Presentation

8.0.1 Scenario 1: Mobile

Persona:

- İbrahim
- 21 years old
- University student
- He wants to be a IOS developer.

Preconditions:

- He is already registered to the Bazaar application.
- He likes online-shopping a lot.
- He has a phone which is iPhone 8.

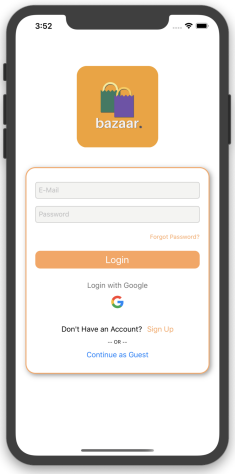
- He frequently looks new apps from AppStore

Goals:

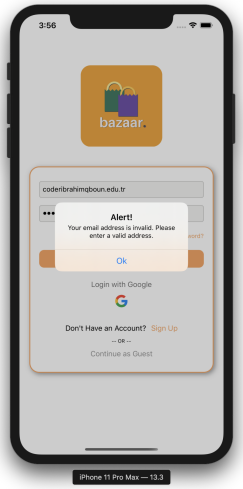
- He doesn't like his current phone. Therefore he wants change his phone with iPhone 12.

Scenario:

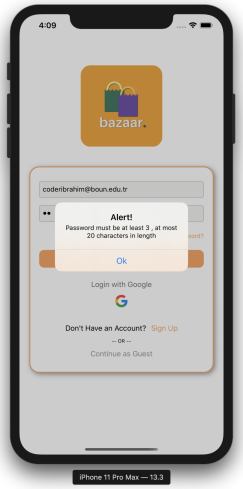
1. İbrahim opened the Bazaar app from his iPhone.
2. He confronted the sign-in page.



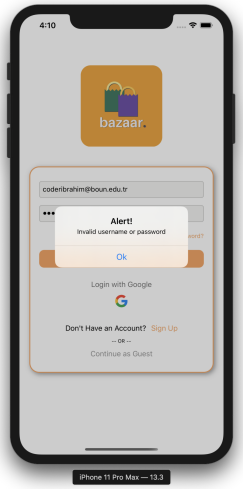
3. Initially, he enters "coderibrahimqboun.edu.tr" as e-mail, however, it is not an email so there is an error.



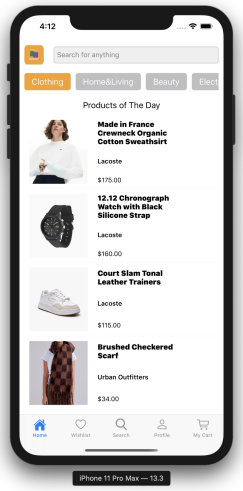
4. Then, he enters "coderibrahim@boun.edu.tr" as e-mail and a password which contains 2 characters. He gets an error which warns him to enter a password which has more than 3 characters.



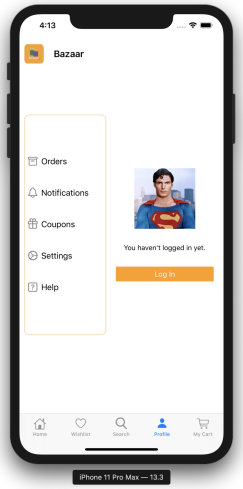
5. This time he enters a password which has more than 4 characters. But he gets "Wrong Password" error.



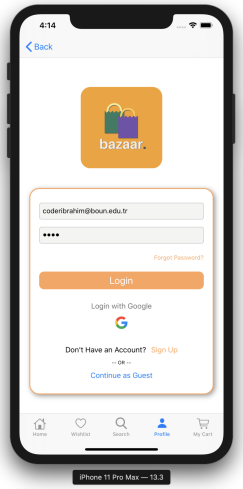
6. He doesn't remember his password and enters to the main page as a guest for now.



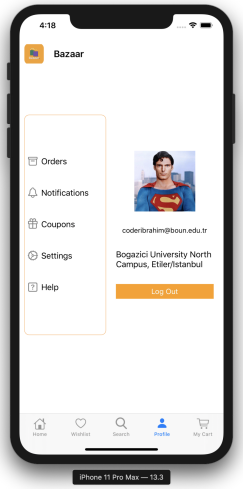
7. İbrahim looks at the main page, clicks on his profile page.



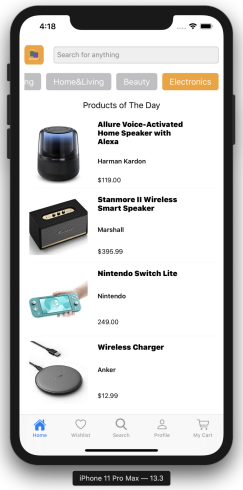
8. Then, he clicks on the Login button in his profile page and gets redirected to sign-in page. This time, he enters his right credentials. Finally, he signs in.



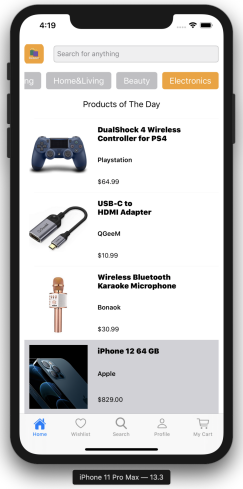
9. Then, he remembers that he recently moved to a new house and wants to check if he changed his address to his new one. He goes to his profile page and sees that his address is the new address.



10. İbrahim then continues to the main page, there he sees the products, he views the categories.



11. He finds a handy product with a good price. His phone is an iPhone 8, he wanted an iPhone12 for a long time, for general purpose. However, even with such a good price, he can't afford it. Therefore he decides to not buy it for now.



12. He closes the app with a sad sigh.

8.0.2 Scenario 2: Web

Persona:

- Şahin Doğan
- 42 years old
- Car Dealer

Preconditions:

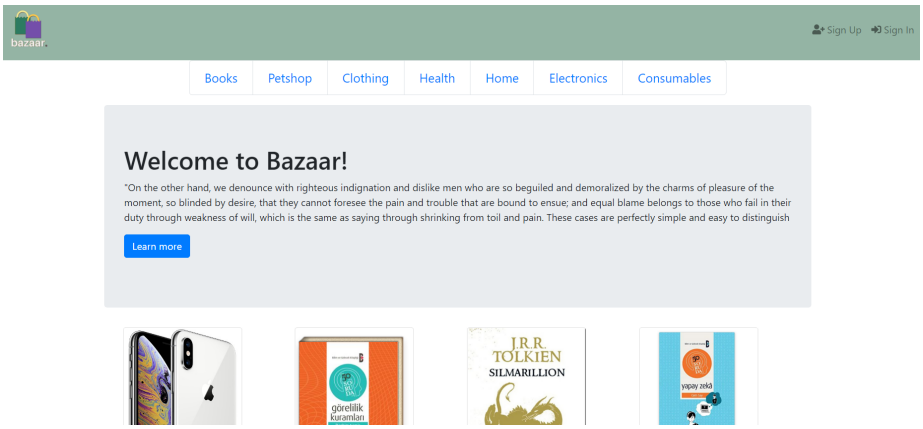
- He uses web browser.
- He is not registered to the Bazaar application.

Goals:

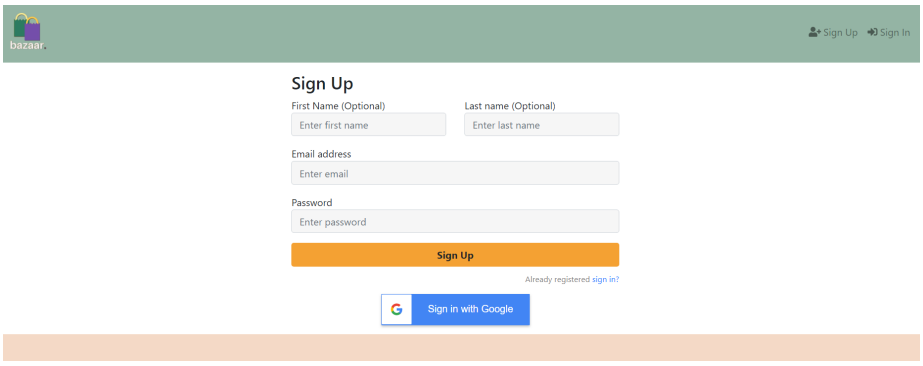
- He wants to create an online shopping account to purchase his needs in the future.

Scenario:

1. After doing some research on net, Mr. Doğan decides to enter Bazaar.
2. He confronts the main page.



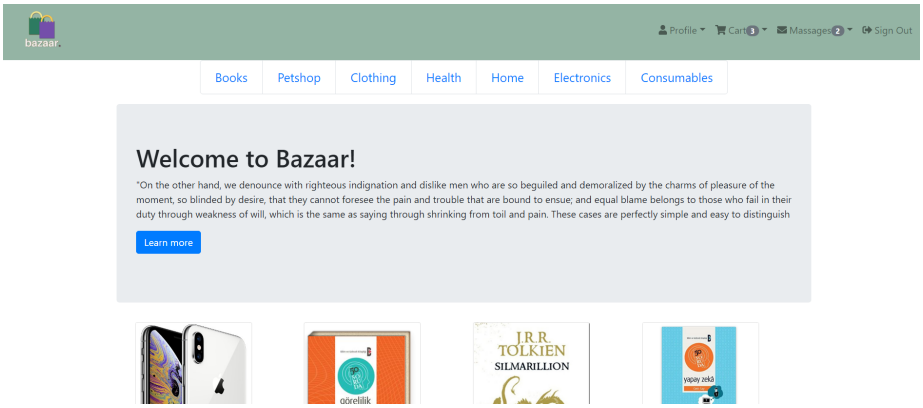
3. He wants to create an account therefore, he clicks on sign-up.



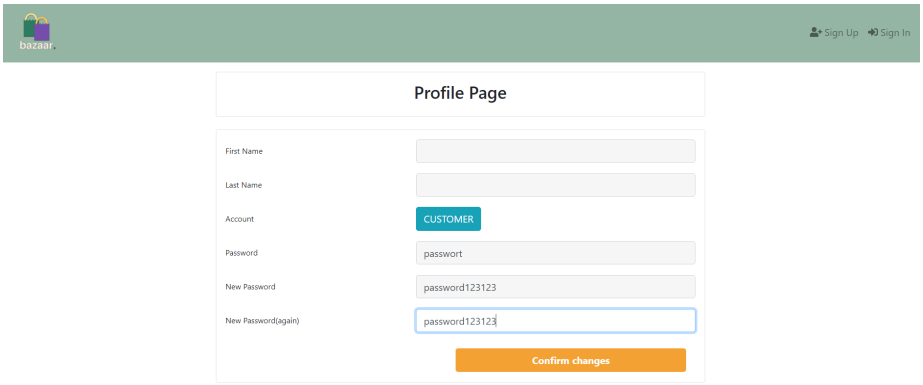
4. Then, he enters wrong e-mail while entering his new credentials. He gets an error that says him to enter proper e-mail.

5. This time he enters a correct e-mail. He enters a password and signs up.

6. He signs in with his new account and confronts with the new main page.



7. Then, he clicks on his profile page.



8. He tries changes his password because he thought it was too simple,however, he gets errors while trying to change password (wrong old password, new passwords does not match).

9. This time he enters correct password and sees the message: "password change success!"

10. He signs out.

11. He signs in with his new password.

9 Tools and Processes

Go to project's GitHub repository.

- GitHub:
  GitHub allows us to easily handle version management. Apart from the Wiki Page which we used in cmpe352; in cmpe451, we started to work around branches more professionally. Each branch has a name that clearly shows it's parent branch in order to keep up with the development. There are 3 master branches excluding the "master branch" itself: **backend_master, frontend_master and ios_master**.

  Additionally, we use GitHub Desktop, which is a GUI that eases working around git, branches, commits, fetches and so on.

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

- Bootstrap:
  Bootstrap contains free HTML and CSS templates for interface components. Our main page's template is based on Bootstrap. It provided us some very sleek templates and allowed us to have an eye-appealing design.

- Django:
  Django is a powerful python framework, used to create our backend part of the project. It is quite spanning, and it can be extended with extra frameworks. For example, we used **Django REST Framework**, which is an extention of Django, especially designed for RESTful backend projects.

- SQLite: Initially, we planned to use MySQL as our database server. However, we decided that for now, SQLite manages what we need and we can easily swap to a MySQL server if needed since all table-relation management is done via Django.

- Dockers:
  In order to deploy our project, we benefited from Docker images/containers. Since we were already experienced with Docker, this was a lesser time-consuming part of the project.

- XCode:
  We chose the iOS operating system for mobile development. We are developing the application using swift language with the help of the Xcode software development tool. Xcode is an integrated development environment for macOS, developed by Apple to develop software for iOS. It is used not only for iOS development, but also for macOS, watchOS and tvOS development, but we will only use it for iOS development.

- IDEs/Editors:
  Every project member uses his/her own project development environment they are familiar with. Some examples are: VSCode, PyCharm, WebStorm etc..

- Sourcetree: Apart from GitHub desktop, the iOS team needed an external version control system because Sourcetree offers better functionalities for this purpose than GitHub desktop when it comes to iOS development. In Atlassian's own wording, Sourcetree provides a graphical interface for repositories, between users and Git, in which simplifies its use for beginners, who haven't mastered Git, and experts, who can be more productive focusing solely on the code.

10 API Documentation

10.1 /api/

list of all other requests, useful for exploration:

```
> GET {{url}}/api/

->
{
    "user": "{{url}}/api/product/",
    "product": "{{url}}/api/user/",
    "location": "{{url}}/api/location/"
}
```

10.2 /api/product/

general information of all products :

```
> GET {{url}}/api/product/

->
[
    {
        "id": 3,
        "name": "Iphone XS 64GB",
        "brand": "Apple",
        "labels": [
            "10% off"
        ],
        "categories": [
            "Electronics"
        ],
        "price": 11211.52,
        "stock": 48,
        "sell_counter": 0,
        "rating": 0.0,
        "vendor": 2
    },
    ...
]
```

10.3 /api/product/$< id >$/

detailed information of the specified product:

```
> GET {{url}}/api/product/3/
```

```
->
{
    "id": 3,
    "name": "Iphone XS 64GB",
    "brand": "Apple",
    "labels": [
        "10% off"
    ],
    "categories": [
        "Electronics"
    ],
    "price": 11211.52,
    "stock": 48,
    "sell_counter": 0,
    "rating": 0.0,
    "vendor": 2
}
```

10.4 /api/user/

list of all users:

```
> GET {{url}}/api/user/
```

```
-> [
    ...,
    {
        "id": 28,
        "email": "rajah.faolan@extraale.com",
        "first_name": "",
        "last_name": "",
        "date_joined": "2020-11-26T14:36:15.093169Z",
        "last_login": "2020-11-26T14:45:13.455166Z",
        "user_type": 1,
        "bazaar_point": 0
    },
    ...
]
```

10.5 /api/user/$< id >$/

detailed description of the specified user:

```
> GET {{url}}/api/user/28/
```

```
-> {
        "id": 61,
        "email": "rajah.faolan@extraale.com",
        "first_name": "",
        "last_name": "",
        "date_joined": "2020-11-26T14:36:15.093169Z",
        "last_login": "2020-11-26T14:45:13.455166Z",
        "user_type": 1,
        "bazaar_point": 0
}
```

10.6 /api/user/profile/

detailed description of the specified user(token):

```
> GET {{url}}/api/user/profile/
> HEADER {
"Authorization": "Token f0653a57e1e4686f9e56425f5660b00baa942657"
}

-> {
        "id": 61,
        "email": "rajah.faolan@extraale.com",
        "first_name": "",
        "last_name": "",
        "date_joined": "2020-11-26T14:36:15.093169Z",
        "last_login": "2020-11-26T14:45:13.455166Z",
        "user_type": 1,
        "bazaar_point": 0
}
```

10.7 /api/user/login/

login request that returns the user and the token
requires “username”, “password” to authenticate
returns 400 if failed
returns 200 and “token”, “user_id” as JSON:

```
> POST {{url}}/api/user/login/
> BODY {
    "username": "rajah.faolan@extraale.com",
    "password": "password"
}

-> {
    "token": "f0653a57e1e4686f9e56425f5660b00baa942657",
```

```
    "user_id": 61,
    "email": "rajah.faolan@extraale.com",
    "password": "password"
}
```

10.8 /api/user/signup/

signup request which sends an email to activate the user later
requires "username", "password", "user_type"
returns 400 if requirements not satisfied
returns 201 if succesful, sends mail to the given email:

```
> POST {{url}}/api/user/signup/
> BODY {
    "username": "rajah.faolan@extraale.com,
    "password": "password",
    "user_type": "1"
}

-> {
    "message": "An mail has been sent to your email, please check it"
}
```

10.9 /api/user/activate/< $uidb64$ >/

account activation link for the specified user
returns 200 if verification complete
returns 400 if there has been a problem with a json body explaining the problem, ie: {`"message": "user is already verified"`}:

```
> GET /api/user/activate/MjU/

-> {
    "message": "Your Account, rajah.faolan@extraale.com has been activated"
}
```

11 Assessment of the Presentation

Overall, we had an average presentation. For the frontend part, we only completed the development process a day before the presentation, so we didn't have time to rehearse the aspects we were meant to present. As a result, some functionalities of the frontend part were lacking in the presentation. Despite our best efforts of bug-fixing at the last minute, we couldn't manage to fix all the issues. In the future, we will set a more homogeneous time-table with strict deadlines in order to ensure that this will never happen again before a presentation. This was a time management issue, and could be fixed by better

planning. For the mobile part of the presentation, the scenario went well and there was no significant bug. Log in/ log out procedures were as expected and as planned to be. Home page, the category filtering and the user profile page were okay as well. Obviously the application has some missing features but they were out of the scope of this milestone. As a team, we need to speed up and come with more functionalities in the next milestone.

12 Summary of coding work done

Hasan Demirkiran	• Creating the frontend react project from scratch. • Creating the project plan, screen, component, subcomponent, utils, icons etc. • Integrating the used design libraries which are Bootstrap4 and react-bootstrap. • Creating the routes for the pages in app.js • Creating the home page for guest and signed users. • Fix the bugs, 3rd part footer library to use. • Choosing a color palette and wear it to the website. • Backend connection for the endpoints. For example product list, log-in sign-up etc. • Handle the token issue, store it in the cookies and get when needed. • Change nav-bar when user logged. • Developing a label, change if the logged user vendor or a customer in profile section. • Developing a label, change if the logged user vendor or a customer in profile section.

Muhsin ETKI	• Initial setup of the Bazaar-iOS application • Initial design and development of main screen • Adding the navigation controller, application logo, and search textfield • Creating the UITableView to display the products • Creating custom tableview cell for products • Design and development of the login screen • Design and development of the sign up screen • Updating autolayout design for login and sign up screen • Checking the validity of user email and password • Adding alert view for email and password validity • Developing continue as a guest functionality • Developing sign up via Google button • Developing radio button for customer or vendor selection option in sign up screen • Developing sign up as customer or vendor functionality • Developing category collection view cell to show product categories • Improving category collection view cell to make cells selectable and functional • Backend connection of the login page • Creating basic structures to establish the link between backend and mobile • Developing URLRequestBuilder protocol, thanks to this protocol the basic structure for all api calls is established, any api call operation can be done easily. • Developing the ApiRouter enum, basic functions are created for the data requested from the backend, such as parameters, headers, and methods. • Developing the APIDatas struct, data types from the api response are created in this struct. • Developing APIParse struct, any kind of json data can be decoded . • Developing the APIManager struct, this struct is where the necessary functions will be created to handle all the api calls.

Emine Alcan Unsal	• Final design of the Table View Cells for the products in the profile page. • Final design of the Home Page view as a whole. • Category filtering functionality for the home page. • Header design (logo and search bar) of the home page. • Table view for the Products of the Day section in the Home Page. • Creation of a dummy product database for the products of the day section of the home page with real products consisting of names, prices, brands and images. • Backend connection of the profile page. • Login and log out functionalities for the profile page. • Construction of the base of the navigation schema of the app. • Construction of the tab bar for the whole app. • Struct representing the products in the products database.

Ömer Cihan Benzer	• Creation of Initial Backend Django project • Initial Table(Django.Model) design for: User, Product, ProductImage, Order, Comment, Notification, Category, Label, ProductList, Payment, Conversation, Message, Location • Serializer(Handles Model-Request connection) for: Product • API-url pattern design, forming API Json standarts • API documentation • Dockerizing and deploying both frontend and backend • Various bug fixes, error/conflict handling

Halil İbrahim Orhan	• Creating ER diagram of database • Implementing Database classes on Django models for: User, Product, ProductImage, Order, Comment, Notification, Category, Label, ProductList, Payment, Conversation, Message, Location • Implementing VerificationView class as view • Sending email functionality • Adding API urls

Mehmet Berk Kemaloğlu	• Implementing first login page • Adding login API url • Bug fixing

Fırat Bulut	• Review requirements and design documents from last year. • Update project plan according to our time-table, set custom milestones for the project. • Design signin, signup and profile page of the frontend. h connection with the backend on updating profile page information. • Design product cards of the frontend, establish connection with the backend. • Decide on the color palette of the website. • Work on various minor improvements on the frontend part.

Beste Göger	• Creation and design of the Profile Page • Creating UILabels to display email and address • Construction of the menu on the Profile Page • Adding the menu navigation to other pages. • Creating View Controllers for the pages: Orders, Notifications, Coupons, Settings and Help

Şadi Uysal	• Changed the initial design of the Profile Page • Created and updated constraints for Profile Menu, Stack View and Info View autolayout design on the Profile Page • Created connections between View Controllers • Helped the initial design of main screen • Updated and designed IOS Git branch structure according to naming convention • Rebased certain branches to manage colaboration and reviewed teammate codes

Ahmet Bilal Uçan	• temp

Mehmet Can Ünal	• Review and making changes on requirements and design documents. • Discussion about design issues with front-end team. • Creation and design of Navbar. • Changed navbar was rendered after authentication . • Adding sign up, sign in and sign out parts navigation to other pages. • Profile, massages and cart dropdowns were added and will be rendered for each user.

Gökhan Tekel	• Review requirements and design patterns for the first time • Being involved to the team • Update project plan • Create User API • Being Cognizant of Front and Back side coherence • Review and update Login Signup Endpoints • Decide on the structure of Database tables