CmpE 451
Fall 2020
Milestone 1
Report

Group 8
Afra Arslan
Ahmet Yiğit Gedik
Burak Berk Özer
Göksu Başer
Muhammet Yazıcı
Nazım Berke Metin
Nursima Çelik
Onur Enginer
Öncel Keleş
Özgürcan Öztaş
Yasin Kaya

November 29, 2020

Contents

1 Executive Summary of Milestone 1:

1.1 Project Summary

Our project is an e-commerce application, where people can sign up as a customer or vendor and buy or sell stuff online. Apart from that, people can also use the app as a guest user.
Customers and guest users can search products, add them to their shopping cart and buy them with credit card information given that they provided an address. Then they can see the details about the order and may return the product or request missing/flawed parts. Customers also can put products on their watchlist, save their payment and address information.
Given that the provided a company name, location and contact information Vendors can put products on their pages and sell them to other users. They can also message a customer back, if that customer purchased a product from that vendor.
Apart from that, the system includes an admin profile. The admin users can see, edit and confirm changes about product and also message with both the Customers and the Vendors.

1.2 Project Status

The deliverables apart from the coding work are already finished. The requirements, class diagrams, user scenarios and sequence diagrams are updated prior to the start of coding, however they can still receive an update if we see a necessary addition.
For this milestone, we finished the login and sign up functionalities of the project, where in sign up we verify the user by sending an e-mail to the given e-mail address. We have also provided a forgot password and reset password addition in order to provide a fully reliable login functionality. Apart from that, we prepared Google login/sign up integration to make users login easier.
The functionalities we stated in our project plan that will be completed for this milestone were up and running. Besides these functionalities, we also worked on the visual and experience design parts of the project and the design of the backend structure.

1.3 Planned Changes

We are going to redesign the pages within the scope the same design and color theme so that the application will look more compact and easier to look at, understand and use experience-wise. We also going to finish the implementation google maps to the Vendor sign up in order to complete the Vendor-side registration in form of location.
Other than code changes, we came up with some solutions to our time management and communication between teams. One member from each team will

attend a weekly meeting and briefly explain the progress of their respective team and discuss what necessary for them to progress further in the project plan.

2 Status of the deliverable:

Deliverable	**Status**
Project Plan (Front-end)	DONE
Login-Sign up	HALF-DONE
Google Login-Sign up	NEED REVIEW
Home Page	DONE
Dockerize	DONE
Deployment	DONE

Table 1: Status of Deliverable for Front-end

Deliverable	**Status**
Project Plan (Backend)	DONE
Login-Sign up	DONE
Google Login-Sign up	DONE
Dockerization	DONE
Deployment	DONE
Database instance for development	DONE
Database instance for production	DONE
Automated integration	HALF-DONE
Automated deployment	HALF-DONE

Table 2: Status of Deliverable for Backend

Deliverable	**Status**
Project Plan (Mobile)	DONE
Login-Sign up	HALF-DONE
Google Login-Sign up	HALF-DONE
Home Page	DONE

Table 3: Status of Deliverable for Mobile

3 Evaluation of the deliverable:

3.1 Project Plan

Before planning our project, we split our team into three groups, which are the Frontend, Backend, and Android team. After that, we began to plan our project. First of all, we determined the essential features that we implemented during this semester. We have listed the tasks by taking into account their importance and dependencies. Then we created a project plan with start-day and deadline. The size of the work and the dates of the customer meetings played a role in determining the deadlines. When we finished our project plan, we started to implement it. We added our project plan to GitHub.

3.2 Login-Signup

Our first goal was to implement login and sign up functionalities for our website and mobile app. To start things off, backend created the necessary endpoints such as /client/login, /client/signup, /client/forgotPassword etc.(All provided endpoints can be seen at API Documentation section of this report or http://18.198.51.178:8008/api-docs domain.) Then after implementing login/sign-up functionalities with e-mail and passwords on both platforms, we went on to implement sign-in with Google. There were some complications in that department, such as Android login with Google was working, but sign-up with Google was not, which will be fixed after this Milestone. Also, all of these features are only functional for clients for now and need to be implemented for vendors as well in the future to be complete. For the Front-end side of the project, a vendor and a customer can sign up or log in. However, the field in the sign-up page that a vendor can add an address is missing. We will add that field after the Milestone-1. For backend, due to nature of an e-commerce website, we had lots of IO operations and uncountable number of simultaneous pings to the server. In order to handle these we have decided on asynchronous programming. Also, since we are creating a online shopping platform these two points are important: clean codebase, dependable runtime. In order to tackle spagetti code, we have created a utility function for every possible event. Even someone with no experience can understand the workflow. To prevent server runtime faults, we have used number of above mentioned utility functions to check the validity of sign up and login requests. These requests flow through different filters and if they successfully pass all of them, they get to interact with the database.

3.3 Home Page

Both the Front-end and the Mobile team designed the home page. The web team designed a header and a list of products for the home page. The header component contains our logo, a search bar, and side buttons, which are login,

list, and cart. The mobile team designed a navigation bar to navigate between homepage, account/login and cart. For homepage we added mock products and listed them categorized as deal of the day, new arrivals and top sellers.

3.4 Dockerization & Deployment:

Dockerization was a crucial part in our development cycle since Docker has brought isolation from all environments and created a environment free, a deterministic development container which contains our product in it. Both Front-end and Back-end divisions have provided their Docker configuration file to inject in our development and deployment machines to work if needed. In back-end, we've also had a deployment script written in Ansible to control and scale or machines in Amazon Web Services an also deploy our product at our will.

4 Summary of the Coding Work Done:

Name	Coding Work Done
İbrahim Özgürcan Öztaş	In Milestone 1, I've organized our repository structure and added branch protection rules for safe and secure code push and pull operations. Also, I've initialized our repository code and I've implemented 2 endpoints that regulates forgetting password and resetting password. In addition to those, I've implemented one of our GitHub Actions automation workflow which is related to code formatting via Prettier. Last but not the least, I've added our Docker configuration file to containerize our API and ensure scalability.
Afra Arslan	After the initialization of our React project, I built the structure of our repository. I grouped files by file types and created folders in that manner. My main job for the Milestone-1 was integrating Google Authentication in our project. I implemented a button providing Login&Signup via Google API. I also created POST and GET methods for our API interactions that can be used easily from anywhere with a single line of code.
Yasin Kaya	I've initialized the React Application and pushed to the our repository. Then, I've created component to show the products in the Homepage. Also, I've added Categories to the Headbar. Lastly, I've added functionality to check whether the given password is strong or not as we've touched on the requirements.
Nazım Berke Metin	I have created the main page structure. Initialized navigation bar and top bar. Added three empty fragments and their buttons which are home page, cart page and account page to the navigation bar. Implemented the account page which changes according to whether a user login already or not. Implemented logout on the account page but not the other buttons for this milestone. Finally, I've added the forgot password page.
Öncel Keleş	I was assigned the header component of our website. I built the header compoenent which consists of the logo, search bar and the login/list/cart buttons. I also created 2 button UIs to be used anywhere of our project on the frontend. Then I designed the forgot password and reset password pages which were part of the login/signup features of our project and connected them to our API accordingly.

Name	Coding Work Done
Onur Enginer	I have initialized the Anrdoid project and created the login page. I have configured the back-end connections for logging in and google sign-in. I have also created a theme which we are using in Web and Android projects right now.
Ahmet Yiğit Gedik	Firstly, I have created a docker file in order to run react-app in the docker container. After constructing the docker container, I have moved on to implement a login and signup page for the web. In order to provide login and signup features properly, I have used login and signup Apis that have been already created by the backend team. Additionally, I have written the Nginx config files in order to serve the web app as it should be. Finally, I have deployed our web app to an Aws Ec2 instance
Nursima Çelik	I was responsible for Android home content. I added the search bar, and the products as divided into 3 sections (deal of the day, new arrivals, top sellers).
Muhammet Tayyip Yazıcı	My first job was to create two endpoints for signup and email verification. I also wrote the mailer util function and created our mail account. I deployed our backend AWS EC2 instances using Terraform. I also wrote an Ansible playbook for auto configuration of our instances. I tried to integrate the Ansible playbook to Github Actions to create continuous delivery pipeline, it is under test and not committed yet. I also did some bugfixes.
Göksu Başer	I was responsible for the sign-up page in the Android application. I have wrote and designed the form sign-up page for customer and vendor users using Kotlin language. and connected them to the back-end endpoints using OkHttp technology. I have also helped pull request and merge operations on the GitHub branch management. In addition to that I needed to debug forget password operation after my teammate finished coding.
Burak Berk Özer	I was responsible with creating backend google sign up and google login. I have started online nodeJS series from scratch. Unfortunetaly, I had unrelated issues and couldn't continue with the development. I was planning on continuing later on but due to miscommunication with the team, task was assigned to someone else before I could contribute to it.

5 Challenges on the road of Milestone 1:

Name	Challenges encountered while coding
İbrahim Özgürcan Öztaş	In Milestone 1, I've encountered a few challenges since I've experienced in software development process in my previous experience as a "Junior Software Developer". One of these challenges is language difference. Although I've familiarized with RESTful APIs, I've a little experience about implementing a RESTful API by using NodeJS. The infrastructure of the language is way more different than I've expected. I've been surprised. Also, I've encountered challenges while using VCS with branch protection rules, since we've agreed to disable the force push into the repository, some tasks were harder to finish and commit to our repository.
Afra Arslan	There are several challenges that I met during Milestone-1. First of all, we have experience time issues and the running of the plan. In our project plan, the deployment was two days before the customer meeting. However, we had missing features on that day. Therefore, we have deployed it on the last day, and it was not a good thing for us since we got some unexpected errors. Besides, I implemented a Google button for our Milestone, but it was dependent on to Login page. To integrate it and test it, I should wait for the login page. The login page has finished before the day of the customer meeting, so I had to do it on the last day. It was such a challenge for me.
Yasin Kaya	Firstly, I had very limited knowledge about the React, HTML and CSS before the Milestone-1. It was a bit challenge for me to get used to use them. Especially, there were some design problems mainly caused by not being able to implement what I imagined.
Nazım Berke Metin	I didn't have a specific challenge in this milestone since I worked for a company as an Android developer for eight months. My biggest challenge was with time and my computer. Four days before milestone my computer started to give bsod whenever I run an emulator. Because of this reason I couldn't debug my code properly so my group had to help me debugging my code. I thank them for this. I hope I can solve this problem quickly so I can help my group more.

Name	Challenges encountered while coding
Öncel Keleş	The main challange for me was that I knew nothing about React and had to learn it enough to implement what I was assigned. Using github was also not something that I was familiar with, but these 2 main challanges that I thought were the biggest ones, were not the biggest ones. The time management was clearly become the biggest issue. Even though I completed my assigned issue days before the presentation, we as a team did not finish our tasks so we couldn't test the project as a whole. Then in the night before, we finally got to see how the project was running as a whole but as expected, we missed out some features that were needed. So even though I got days before the presentation, I just got one night to finish the forgot/reset implementations which was really exhausting. So, we did not have enough time to revise our pages design-wise.
Nursima Çelik	First challenge I encountered was imagining how home page should look like. Certain things were for sure, like a search bar, preview of some products (with image, title, and price). But I had to decide whether there should be a banner at the beginning, under which titles we should present the products, which products I should use this milestone, whether there should be some brands at the beginning to get the attention of the user, etc. Also for a good impression we would need to come up with nice colors. After deciding what should be in the home page, I tried to realize it, but next challenge was creating such a page in Android Studio. Sometimes I spent too much time on easy tasks because of my lack of experience. Lastly, using git was not so easy, but I managed to push my code to correct place thanks to its user friendliness.
Ahmet Yiğit Gedik	The hardest part for the Milestone 1 was learning how to implement a login and signup page in the react-app because I do not have any experience on react-app, even on implementing a front-end part. I have tracked react tutorials to overcome this challenge. Another challenge was understanding the docker environment. I had some difficulties understanding how the docker container works because it was a new and different concept for me. The last challenge that I have encountered was writing the Nginx config and using it in the docker container.

Name	Challenges encountered while coding
Onur Enginer	My first issue was with Git, which was also the biggest issue for this milestone. It was hard to configure the project as a subfolder in our repository while also having my IDE being able to build the Android project. Coding itself didn't present many challenges, but time was also a major issue since we are a new team and need to figure some things out before we could begin working on our code.
Muhammet Tayyip YAZICI	My biggest issue was with using Express and Node. Although I had used Node before I was not really familiar with Backend development that much. Getting used to the structure of controllers, models and services at first caused some struggles but as I coded I saw its benefits right away. I have also had some struggles with deployment side. I was familiar with Terraform so creating AWS EC2 instances wasn't a big problem. However configurations of those VMs were harder to automate. So I learned Ansible to automate this process. I also tried to write Github Actions trying to create a continuous delivery pipeline. However testing the workflow caused a lot problems because there is no way to run a workflow on our local. I faced with a lot more errors than I anticipated, causing me to unable to commit the Github Workflow I created on our Project repo.
Göksu Başer	My biggest challenge was not knowing Kotlin language and have not written any Android project until now. I have used Intellij IDEA in my internship; therefore, getting used to branch operations was easier comparing. To learn the language I get help from my teammate. Because Onur started login page before, I have observed his project and it accelerated my learning process. Another challenge was debugging my other teammates code. Because he has issues on Android emulator on his computer. I took the job of merging the pull request. However, after the merge, the project crashed and it took me several hours to learn his part of the project and debugging it.
Burak Berk Özer	As I explained in the code summry, I can only discuss about cloud services. The greatest issue I had was with Amazon Web Services. I have never used Amazon or similiar cloud technology provider. First, I was tasked with initating MongoDB database. Even though I had started with 0 knowledge, I have quickly gained insight by their guides. After meddling with their online panels, finally I have created the database. Then, shared necessary login information with backend team but there was a problem. Database was using an highly priced plan. I had to terminate every part of it and since I couldn't trust Amazon to stick to their free plan, task was assigned to an another member of the backend team.

6 Requirements

6.1 Glossary

- **User:** A person who is using the platform.
- **Customer:** A registered user who wants to buy and verified their e-mail.
- **Profile:** An outline of user or customer's information. Holds personalized settings.
- **Vendor:** A registered user who wants to sell and verified their e-mail.
- **Shopping Cart:** List of products that user's intending to buy. Can proceed to checkout.
- **Guest:** A user who is using the platform but has not signed in yet.
- **Admin:** A registered user who can manage the platform.
- **Admin Panel:** A tool that enables platform managers to control products/categories.
- **Product:** Thing that is sold or bought.
- **List:** A list that customers can add products that they want to buy.
- **Active Order:** An order that is not delivered yet.
- **Inactive Order:** An order that is either delivered or canceled.
- **Two-factor Authentication:** Requiring more than one method to grant access. More secure than traditional methods.
- **Product Page:** Individual page that is showcasing one special product.
- **Uptime:** The percentage of time the platform has been working and available.
- **Order Tracking Number:** A unique 9 digit number associated with an order.
- **Orders Page:** A page where all the order history can be seen. Customers and vendors have an orders page.
- **Order Details Page:** A page associated with one order, that contains Order Details.
- **Order Details:** Order Tracking Number, products' names, products' pictures, products' prices, total price, cargo brand, customer/vendor name, status (being prepared, on the way, delivered, canceled by the customer, canceled by the vendor, returned), order date, delivery date if delivered.

- **Semantic Search:** It denotes search with meaning, as distinguished from lexical search where the search engine looks for literal matches of the query words or variants of them, without understanding the overall meaning of the query.
- **Credential:** An e-mail and password pair, or a Google account; used to authorize a user while logging in.
- **Strong Password:** A string that contains numbers or characters of length between 6 and 20. It must contain at least one lowercase letter, one uppercase letter, one numeric digit, but cannot contain white space.
- **Gift Card:** A special code used in place of a specified amount of money. This code can be obtained in several different ways. Such as from special events or returned orders if the customer agrees with the vendor or some vendors may sell gift cards as gifts for other customers.
  - We decided this was as one of our Nice to Have features in the last term. However while we are updating requirements with new team. We concluded, we can remove this feature because it is not clear enough to accomplish and unnecessary.
- **Discount Code:** A special code that provides a discount to certain items. This code can be obtained from special events
  - We decided this was as one of our Nice to Have features in the last term. However while we are updating requirements with new team. We concluded, we can remove this feature because it is not clear enough to accomplish and unnecessary.
- **Popular Vendor:** Vendor who has high ratings.
  - We added this definition because it was in our requirements and it was not clear enough. Therefore, we felt that we need to explain in detail in the glossary part.

6.2 Functional Requirements

6.2.1 User Requirements

6.2.1.1 Registration and Login

- **6.2.1.1.1** Users shall be able to register as a customer by providing an e-mail address, a strong password and name/surname.
- **6.2.1.1.2** Users shall be able to register as a vendor by providing an e-mail address, a strong password, one or more locations of their stores through Google Maps, and name and title of the company.
- **6.2.1.1.3** Users should be able to use a Google account instead of e-mail and password while registering.

- **6.2.1.1.4** Registered users shall be able to log in as a customer or a vendor.
- **6.2.1.1.5** Registered users shall be able to login with their e-mail and password.
- **6.2.1.1.6** Registered users should be able to login with their Google account.
- **6.2.1.1.7** A customer shall be able to create a vendor account with the same credentials and vice versa.
- **6.2.1.1.8** A customer shall accept system to store the password, the email, and the phone of the user to sign up.
- **6.2.1.1.9** A vendor shall accept system to store the location, the name, and their contact information to sign up.
- **6.2.1.1.10** Registered users shall be able to verify their e-mail by clicking the link that is sent to their e-mail address.
- **6.2.1.1.11** Users who are in the verification process shall not login the application unless complete the verification process.
  - **-** A safety measure for the unregistered email registration in our system
- **6.2.1.1.12** Customers or vendors shall be able to select "Forgot your password" when they have forgotten their password and select their new password.
  - **-** A necessary functionality for clients who are unable to reach out their password or cannot login with the current password.

6.2.1.2 Profile

- **6.2.1.2.1** Customers shall be able to change their name/surname or password after a confirmation of current password.
  - **-** Since name and surname are the most definitive information fields for our customers, it should not be changed by someone else except the person itself.
- **6.2.1.2.2** Customers shall have a profile settings page where after they confirm their current passwords they can edit their:
  - **6.2.1.2.2.1** Address
  - **6.2.1.2.2.2** Phone Number
  - **6.2.1.2.2.3** Payment Method
  - **6.2.1.2.2.4** Birthday date

- **6.2.1.2.3** Vendors shall have a profile settings page where they can change their password after they confirm their current passwords.
- **6.2.1.2.4** Vendors shall have a public profile page where they can edit fields:
  - **6.2.1.2.4.1** Name of the company.
  - **6.2.1.2.4.2** Contact information of the company.
  - **6.2.1.2.4.3** Location of the store through Google Maps.

6.2.1.3 Products

- **6.2.1.3.1** Users shall be able to view the page of products.
- **6.2.1.3.2** Page of the products shall contain price, expected delivery date, images of the product(if possible), technical features, availability at the moment, comments, rating, and its current vendors.
  - **-** Older version did not specify the feature set of a product.
- **6.2.1.3.3** Vendors shall be able to request to create, update or delete the page of products from admins.
  - **-** It is a control measure for more than one reasons: Registering a product more than once, deleting a product until each instance of it has been delivered etc.

6.2.1.4 Search

- **6.2.1.4.1** Users shall be able to choose to search for products or vendors.
  - **6.2.1.4.1.1** Users shall be able to search for products based on one of the followings: product title, category, brand, vendor, color.
    - **-** Previous versions did not specify the search fields.
  - **6.2.1.4.1.2** Users shall be able to search for vendors based on vendor name.
    - **-** It is inserted to enable vendor-based product search.
- **6.2.1.4.2** Users shall be able to do a semantic search for products.
- **6.2.1.4.3** Users shall be able to filter results according to:
  - **6.2.1.4.3.1** The average customer review
  - **6.2.1.4.3.2** Brand name
  - **6.2.1.4.3.3** Vendor name
  - **6.2.1.4.3.4** Minimum and maximum price
  - **6.2.1.4.3.5** Color

- **6.2.1.4.4** Users shall be able to sort results according to:
  - **6.2.1.4.4.1** Popular sellers" and "Popular vendors" is defined in "Glossary" as "Vendors who has high ratings".
    - **-** It is changed to reform a term in the glossary.
  - **6.2.1.4.4.2** Newest ones
  - **6.2.1.4.4.3** Price, increasing or decreasing
  - **6.2.1.4.4.4** Average customer review
  - **6.2.1.4.4.5** Number of comments

6.2.1.5 Shopping Cart

- **6.2.1.5.1** Customers and guests shall be able to add products to their shopping cart.
- **6.2.1.5.2** Customers and guests shall be able to see the price of each product in the cart.
- **6.2.1.5.3** Customers and guests shall be able to increase or decrease the quantity of the products.
- **6.2.1.5.4** Customers and guests shall be able to delete products from their carts.
- **6.2.1.5.5** Customers and guests shall be able to see the total price of the cart.
- **6.2.1.5.6** Customers shall be able to redirect their current page to the page of the product they have clicked on.
  - **-** It enables a customer to check his/her products and their set of features without any effort.

6.2.1.6 Lists

- **6.2.1.6.1** Customers shall perform these operations on the lists: creating, naming, deleting.
- **6.2.1.6.2** Customers shall be able to add/remove products to/from their lists.
- **6.2.1.6.3** Customers are provided with one list namely "Watchlist" and one list namely "Favlist" by default and Requirements (6.1.6.1) is not applicable to these two lists.
  - **-** It is one of our "Nice to Have!" features which provides two default lists to a customer to get notifications easily or collect his/her favorite products in one list.

- **6.2.1.6.4** Watchlist is the default list given by the application which enables user to get notified if one or many events happen in Requirements (6.2.1.10.1).
  - **-** Watchlist is a default list which enables customer to get notified if a product in watchlist gets changed at any condition.
- **6.2.1.6.5** Favlist is the default list given by the application which lets user to collect their liked products into one list.
  - **-** Favlist is a default list which enables customer to collect his/her favorite products in one collection.

6.2.1.7 Order

- **6.2.1.7.1** Registered users shall have an Orders Page where all the active/inactive orders are listed.
- **6.2.1.7.2** Guests who previously placed an order shall be able to enter to the Order Details Page by providing their Order Tracking Number.
  - **-** We cut our e-mail because only order tracking number was enough to track.
- **6.2.1.7.3** All users shall be to see Order Details on the Order Details Page.
- **6.2.1.7.4** Users shall be able to cancel orders that have not been shipped yet on the Order Details Page.
- **6.2.1.7.5** Customers and guests shall be able to return an inactive order within 14 days after delivery.
- **6.2.1.7.6** Customers and guests who want to return a product shall choose a cargo company from the contracted ones for delivery.
- **6.2.1.7.7** Customers and guests who have chosen a company that transports returns for free, shall receive a cargo code to send the product free of charge.
- **6.2.1.7.8** Vendor shall accept the return or not, according to the condition of the product.
- **6.2.1.7.9** Upon acceptance of a return request, the customer/guest shall get a refund.
- **6.2.1.7.10** Upon acceptance of a return request, customers shall also get cancellation of an increase in points due to that product(s).
- **6.2.1.7.11** Upon rejection of a return request, the customer/guest shall receive the products again.

- **6.2.1.7.12** Customers shall be able to see their scores based on the money that they spent so far on the platform.
- **6.2.1.7.13** Vendors shall be able to see the money that they earned so far on the platform.
- **6.2.1.7.14** Customers should be able to return their orders for exchange.
  - **-** We concluded that there should be a product exchange option.
- **6.2.1.7.15** Customers should be able to request missing/flawed parts.
  - **-** We concluded that the costumers should be able to request missing/deformed parts.

6.2.1.8 Purchase

- **6.2.1.8.1** Customers and guests shall be able to purchase the products in their shopping carts.
- **6.2.1.8.2** Guests shall log in, or provide an e-mail address to proceed to payment.
- **6.2.1.8.3** Customers and guests shall choose between those payment methods: credit/debit card, EFT.
- **6.2.1.8.4** Customers who have chosen credit/debit cards shall either provide card information or use a saved one in case there is any.
- **6.2.1.8.5** Guests who have chosen credit/debit cards shall provide card information.
- **6.2.1.8.6** Customers and guests who have chosen EFT shall receive the IBAN.
- **6.2.1.8.7** Customers shall either enter a new address or use a saved one.
- **6.2.1.8.8** Guests shall enter an address.
- **6.2.1.8.9** Customers and guests shall be able to use a 3D secure system.
- **6.2.1.8.10** Customers and guests shall receive an Order Tracking Number upon payment.
- **6.2.1.8.11** Customers and guests shall receive an e-mail about the details of the order.
- **6.2.1.8.12** Customers and guests shall be able to allow the system to store the card information of him/her after he/she made a purchase.
- **6.2.1.8.13** Users shall earn points according to their purchases.
  - **-** We added this bullet because it was missing.

6.2.1.9 Comment and Rate

- **6.2.1.9.1** Customers shall be able to comment on or rate the products they bought.
- **6.2.1.9.2** Users shall be able to view user comments and ratings about products.
- **6.2.1.9.3** Users shall be able to view the average rating of vendors based on the ratings of the products they sell.
- **6.2.1.9.4** Guests shall be able to rate the products they bought when entered with an e-mail address and Order Tracking Number.
  - **-** Guests are no longer able to comment.

6.2.1.10 Notifications

- **6.2.1.10.1** Customers shall be able to choose to get notifications concerning one product or any product in their watchlists, for certain events:
  - **6.2.1.10.1.1** A change in price
  - **6.2.1.10.1.2** Price of the product goes below a threshold
  - **6.2.1.10.1.3** An unavailable product becomes available
  - **-** Customers should only receive notifications for the products in the watchlist, not any other lists.

6.2.1.11 Recommendations

- **6.2.1.11.1** Customer users shall receive product recommendations based on purchase history.
  - **-** Customers won't receive recommandations for all of their activity but only their purchases.
- **6.2.1.11.2** Recommendations will be based on purchases and their associated weights(price, date, etc.).
  - **-** These recommendations will be based on chronological purchase history and other information about those purchases.

6.2.1.12 Messaging System

- **6.2.1.12.1** Vendors and admins shall be able to communicate via messaging anytime.
- **6.2.1.12.2** Customers shall be able to start a conversation with a vendor after purchasing a product from that vendor.

- **6.2.1.12.3** Vendors shall be able to start a conversation with the customer only during an active order.
- **6.2.1.12.4** Vendors shall be able to reply to a customer until the period of order cancellation ends.
- **6.2.1.12.5** Customers should be able to message admins if a problem arises after the delivery within guarantee extent.
  - **-** This bullet was added in order to cover the customers' guarantee if the vendor neglects them.

6.2.1.13 Admin Panel

- **6.2.1.13.1** Admins shall be able to add/remove categories.
- **6.2.1.13.2** Admins shall be able to delete or suspend accounts.
- **6.2.1.13.3** Admins shall confirm or not when a product is added.
- **6.2.1.13.4** Admins shall be able to delete products.
- **6.2.1.13.5** Admins shall be able to delete comments.
- **6.2.1.13.6** Admins shall be able to confirm if a product information is edited by the vendors.
  - **-** This bullet was added so that vendors won't be able to abuse edit product feature and constantly change prices and information about the products.

6.2.2 System Requirements

6.2.2.1 Registration and Login

- **6.2.2.1.1** The system shall allow guest users to register via e-mail and password or their google account as a customer, vendor.
- **6.2.2.1.2** The system shall allow guest users to login via their e-mail and password or their google account.
- **6.2.2.1.3** The system shall require users to verify their e-mail via a link sent to their e-mail.
- **6.2.2.1.4** The system shall send a warning message after 5 unsuccessful login attempts to the mail address that wants to be logged in, about someone else may try to login with his/her account.
- **6.1.2.2.5** The system shall give a warning when the user tries to sign up with shorter than 8 characters, at least an uppercase character, a lowercase character, and a number.

- **6.1.2.2.6** The system shall take consent from the customer user to store the password, the email, birthday information, and the phone of the user while signing up.
  - **-** Birthday of a user is added to the information requested from user. So, the system should take consent on the birthday information too.
- **6.1.2.2.7** The system shall take consent from the vendor user to store the location, the name, and the contact information of him/her while signing up.

6.2.2.2 Profile Management

- **6.2.2.2.1** The system shall store user names, e-mail addresses, passwords, addresses, phone numbers, full names, and birth dates of each registered customer.
- **6.2.2.2.2** The system shall store profile pictures, user names, e-mail addresses, passwords, store address, phone numbers, full names, the location from Google Maps of each registered vendor.
- **6.2.2.2.3** The system shall store profile pictures, user names, e-mail addresses, passwords, store address, phone numbers, full names of each registered vendor; profile pictures, user names, e-mail addresses, passwords, phone numbers, full names of each registered customer.
  - **-** This requirement is split into two pieces for the customer case and the vendor case as 6.2.2.2.1 and 6.2.2.2.1, in order to achieve more atomic requirements.

6.2.2.3 Search

- **6.2.2.3.1** The system shall allow keyword and semantic search for users.
- **6.2.2.3.2** The system shall allow users to filter products by product-rate, brand, vendor-rate, price, color, and shipment information.
- **6.2.2.3.3** The system shall allow users to sort by search results by their selling numbers, date of release, price, amount of user interaction and the average rating.
- **6.2.2.3.4** The system shall have categories for products that have a similar feature.
- **6.2.2.3.5**The system shall allow users to search for products categorically.
- **6.2.2.3.6** The system shall distinguish the same products which are sold by different vendors.

6.2.2.4 Purchase

- **6.2.2.4.1** The system shall support the usage of gift cards and discount codes.
  - **-** This bullet is a "Nice to Have" feature and can be considered in future versions. We prioritize other functionalities.
- **6.2.2.4.1** The system shall allow customers and guest users to buy the products in their cart after they supply the information about their payment and address.
  - **-** The order of this requirement has changed due to deletion of requirement one before.
- **6.2.2.4.2** The system shall support different types of purchase methods such as credit card, debit card, and EFT.
- **6.2.2.4.3** The system shall give the registered users the option to save their credit card information.
- **6.2.2.4.4** The system shall give options for using a new address or a saved address for the purchase.
- **6.2.2.4.5** The system shall allow users to add products to their carts.
- **6.2.2.4.6** The system shall allow customers and guest users to remove products from their carts.
- **6.2.2.4.7** The system shall allow customers and guest users to buy products that are in their carts only.
- **6.2.2.4.8** The system shall take consent from the customer and guest user to store the card information of him/her after he/she made a purchase.
  - **-** Storing user's card information could be problematic for KVKK, as Amazon instances we will be using are abroad.

6.2.2.5 Comment and Rate

- **6.2.2.5.1** The system shall have a rating system out of five for the products.
- **6.2.2.5.2** The system shall allow customers and guest users, who ordered a product, to comment on those products.
- **6.2.2.5.3** The system shall allow customers and guest users, who ordered a product, to rate on those products.
- **6.2.2.5.4** There should be a list of banned words and the system should not allow users to use a banned word in their comments.

  - **-** We noticed that we should be checking what is written in comments. They should contain proper words for both public's sake and the prestige of the platform.

- **6.2.2.5.5** Users should not be able to add links to comments.
  - **-** Since allowing links may pave the way for malicious links, we decided to disallow links in comments to make our site safer.

- **6.2.2.5.6** If the 6.2.2.5.4 and 6.2.2.5.5 are not selected for development, there shall be a confirmation system for the comments.
  - **-** Above two requirements are not mandatory, so, in case banned words and disallowing links are not realized, then a comment confirmation system is a must.

6.2.2.6 Product pages

- **6.2.2.6.1** The system shall hold product information on the product's page.
- **6.2.2.6.2** The system shall have pictures of products.
- **6.2.2.6.3** The system shall support comments for products.
- **6.2.2.6.4** The system shall support product rate.
- **6.2.2.6.5** The system shall hold the property of the products.
- **6.2.2.6.6** The system shall support vendor rate.

6.2.2.7 Lists

- **6.2.2.7.1** The system shall allow users to create, edit, delete non-default lists.
  - We added default lists to the system for more ease to users. These default lists cannot be created/edited/deleted.

- **6.2.2.7.2** The system shall allow available products to be transferred from lists to cart.
  - **-** Only the available products can be put in the cart. Seeing a product is unavailable after putting it in the cart or, worse, in the purchase step would be an unpleasant experience for the user.

- **6.2.2.7.3** The system shall allow products to be added in lists.
- **6.2.2.7.4** The system shall keep the lists private for each user.
- **6.2.2.7.5** Users shall be notified about unavailable products while transferring product from the list to the shopping cart.

- **-** As a product may become unavailable after the user puts it in a list, and only the available products should be in the cart, user should be informed about the unavailable products at the stage of transferring from list to cart.

6.2.2.8 Google Accounts and Maps Synchronization

- **6.2.2.8.1** The system shall support Google Maps
- **6.2.2.8.2** The system shall support the synchronization of Google accounts and Google Maps.

6.2.2.9 Recommendation System

- **6.2.2.9.1** The system shall recommend users similar products related to their search history.

6.2.2.10 Messaging System

- **6.2.2.10.1** The system shall have a private chat platform for customer and vendor communication.
- **6.2.2.10.2** The system shall allow customers and vendors to ask help from admins by opening tickets.
  - **-** We converted the messaging system to ticket system.
- **6.2.2.10.3** There shall be a character limit for messages.
  - **-** Users should not be able to send messages that are extremely long.
- **6.2.2.10.4** There shall be a limit on consecutive messages.
  - **-** As too many consecutive messages is not natural (can be nonhuman) and inconvenient, there should be a limit.
- **6.2.2.10.5** The messages shall be deleted after a certain amount of time after delivery.
  - **-** This bullet is added so that database doesn't overflow with accumulating messages.

6.2.2.11 Order Page

- **6.2.2.11.1** The system shall allow customers and vendors to see their active and delivered orders.
- **6.2.2.11.2** The system shall allow customers to cancel an active order before it being shipped.

  - **-** They shouldn't be able to cancel orders at the stage of shipment.

- **6.2.2.11.3** The system shall support returning and refunding of delivered products.

- **6.2.2.11.4** The system shall contain shipment information containing tracking number and customer address accessible by customers and vendors.

  - **-** More details are given about shipment information for the sake of clearness.

- **6.2.2.11.5** The system shall allow customers to see the total amount of money as points they collected.

- **6.2.2.11.6** The system shall allow vendors to see the total amount of money they earn considering canceled and returned orders.

  - **-** This is one of our "Nice to Have" features and can be considered in future versions. For this stage, we omit this feature.

- **6.2.2.11.6** Users shall be able to track return exchanges and part requests.

  - **-** Tracking a shipment is something a user may want. Also we added part requests to the system, considering a case where a product's only one piece is flawed/missing and that should be exchanged/requested.

6.2.2.12 Admin Permissions

- **6.2.2.12.1** Admins shall be allowed to delete or suspend accounts violating.

- **6.2.2.12.2** Admins shall be allowed to create a new category.

- **6.2.2.12.3** Admins shall be able to confirm a product edit.

  - **-** If a product is edited, it needs an admin confirmation before it is presented in the platform, since its images, title, description may contain inappropriate elements in the edited version.

- **6.2.2.12.4** Admins shall be able to see and edit message tickets.

  - **-** In problematic situations, it is useful for admins to have such control; so that they can solve the issue more easily.

6.2.2.13 Notification System

- **6.2.2.14.1** The system shall provide e-mail notification for products in the watchlist if the price of the product decreases.
  - **-** We added a default list called watchlist, and users shall add products they want to notified about in this list. So, notifications are valid for this list.
- **6.2.2.14.2** The system shall allow customers to set alarm for products in the watchlist for a certain price and notify them via e-mail.
  - **-** Similar to the previous requirement, alarm can be set for the products in watchlist.

6.3 Non-functional Requirements

6.3.1 Security

- **6.3.1.1** Each user shall have a strong password.
- **6.3.1.2** All user passwords shall be stored with encryption in the database using SHA-256.

6.3.2 Privacy

- **6.3.2.1** All user data shall be processed according to KVKK and GDPR.
- **6.3.2.2** The system shall take consent from the customer user to store the password, the email, birthday information, and the phone of the user while signing up.
  - **-** This bullet was updated so that users would also give consent for their birthday information too.
- **6.3.2.3** The system shall take consent from the vendor user to store the location, the name, and the contact information of him/her while signing up.

6.3.3 Accessibility

- **6.3.3.1** Product shall be accessible from any Android device that has Android 6.0+.
- **6.3.3.2** Product shall support Google Chrome 79+, Safari 10+, Firefox 71+.
  - **-** This bullet was updated so that we would only support browser versions dating from 1 year before.

6.3.4 Availability

- **6.3.4.1** System shall have at least %99 uptime.
- **6.3.4.2** In every month there shall be maintenance lasting 2 hours.
- **6.3.4.3** Product information shall be available in English.
  - **-** We removed Turkish support to focus solely on international market.

6.3.5 Performance

- **6.3.5.1** The system shall respond to any user under 3 seconds.
- **6.3.5.2** The system should guarantee to work for at least 10000 users simultaneously, without crushing.
  - **-** We do not want share our performance metrics at this stage before the 3rd stage of our product.

6.3.6 Standards

- **6.3.6.1** The system shall follow the standards introduced by the World Wide Web Consortium (W3C).
- **6.3.6.2** Platform should support the W3C Activity Streams protocol.

7 Changes in the design documentations:

7.1 Class diagram

Class Name	Changes
Registered User	Firstly, we've removed the phone information from the class as it was already included in the address information. Also, we decided to add a functionality to check the available recipients, in other words registered users will be able know whom they can send a message to.
Product	In this class we changed vendorId to vendors, which is a dictionary list. We decided multiple vendors should be able to sell the same product with their custom information added in the dictionary. We added tags field for search engine. We added category field for making classification easier. We changed price to float.
Order	For fields, since order's will be given in bulks; we have added bulkProductID and defined products as BulkProduct. We have added getBulkProductID to identify bulk products. Also, we have added setter and getter functions for the refund process.
Admin User	We added functionality to cleanse users, when a user's ban could be revoked. Also, we realised that admin users should reject a product, therefore we added another function for that purpose.
SearchEngine	We have added filter by category and filter by tag functionalities to satisfy project requirements.
Vendor	First of all, to keep its location information (longitude and latitude) we've added a new variable. Also, we've realised that IBAN of the vendor is also needed to transfer their money. On the other hand, we noticed that vendors should not be able to remove a product as they wish, but they should make a request to the admins with a valid excuse. Lastly, we've added missing setter/getter functions.
Message	We have changed name of MessageInterface to Message for redundancy.

Class Name	Changes
Bulk Product	This class's old name was ordered product. Because it defines its purpose more clear.
Cart	We have renamed OrderedProduct to productsIn for clarity.
Shopping List	We added a function called addToCart for user to easily buy their shopping list.
DocumentationEngine	We have created DocumentationEngine. It stores endpoints as a key end returns their documentation as value.
Customer	For customer class, we decided to add birthday information, as we decided that it will be useful to recommend more accurate products.

You can find the updated class diagram here.

7.2 Use Case diagram

We've inspected our Use-Case Diagram and we've realized that our project plan and decided actions support this version of the Use-Case diagram thus we've decided to accept the document as it is for now. If a major change occurs in the coding phase, one of our group member will consult to all members to apply changes if necessary.

7.3 Sequence diagram

7.3.1 Comment

The first sequence diagram was about a customer that comments on a product. The scenario and the flow are the same as the last semester, but there are some class changes. In the class diagram review, the OrderedProduct class changed into BulkProduct class. Therefore, when a user writes a comment, it invokes the getProduct function in the Order class, and it returns a bulk of products associated with the order. BulkProduct class takes this product list and returns the product that the customer wants to add a comment. Product class adds the comment that product and 'Comment has been added successfully' message returns the actor, which is a customer in this scenario.

7.3.2 Purchase Order

Purchase Order Diagram is almost the same as before. The only change is the return message from the payment engine. When a customer purchases his/her cart, it invokes the buy function in the Cart class. Then, it sends the credit card information to the PaymentEngine. If the payment engine gets the money successfully, it returns a message that says 'Successful'! This message was True before, but because of the changes in the Class diagram, it turns into successful now. Everything else is the same. If it is successful, it gets the order from the

Order class and shipping information from the ShippingInfo class and returns the order. In the other case, if the payment engine could not get the money, it returns the message 'Could not draw money from the account.'

7.3.3 Cancel Order

Cancel order sequence starts with the request from logged in customer. Customer class runs with cancelOrder(orderId) request and calls Order class's cancelOrder(). Then order class checks if order is shipped from shippingInfo class. If the order is not shipped yet order class sends refund request to the paymentEngine with the orderId. If order is already shipped user gets "Order is already shipped" info. When paymentEngine gets refund request user gets the info if the refund is successful or not.

7.3.4 Add Product

Add product sequence diagram has changes according to new class diagram. It now sends a addProduct() request to the Vendor class and Vendor Class creates Product with given information. Then both returns responses.

7.3.5 Confirm Products

Confirm products sequence diagram change reflected the changes in class diagram, which is in this case renaming function makeNotConfirmed() to rejectProduct(). Also an additional call to a pre-existing function removeWaitingProduct(productId) is removed since it seemed reduntant and its function could be achieved with rejectProduct() function.

7.3.6 Add to Cart

Only change made in add to cart sequence diagram was to rename OrderedProduct to BulkProduct as it is changed in class diagram.

7.3.7 Check Messages

Only change made in check messages sequence diagram was to rename MessageInterface[] to Message[] to reflect changes in the class diagram.

7.3.8 Search for a Product

Search a Product sequence diagram was missing in the last semester because I was assigned to create search for a product sequence diagram. There are many alternatives inside search. The program calls appropriate function according to selections of the user. Called function returns the filtered products.

7.3.9 Report User

Report User sequence diagram was missing and we as a team fell like it was necessary for our future implementation journey. To make coding easier I created Report User sequence diagram and added it to the GitHub repository.

8 API Documentation:

Our domain of API Documentation: http://18.198.51.178:8080/api-docs

8.1 Authenticity:

Functionality	**URI**	**Method**	**Parameters**
Initialize	/client/init	POST	tokenCode: String
Login	/client/login	POST	password, email, type
Verify Email	/client/verifyEmail	GET	verifyToken
Sign-up	/client/signup	POST	name, lastName, email, password, passwordConfirm
Change Password	/client/changePassword	POST	tokenCode, newPassword, newPasswordRepeat
Google Sign-up	/client/signupWithGoogle	POST	email, googleID, type
Google Login	/client/loginWithGoogle	POST	email, googleID, type
Forgot Password	/client/forgotPassword	POST	email, type
Reset Password	/client/resetPassword	POST	resetPasswordToken, newPassword, newPasswordCheck

9 Our Project Plan:

We've inserted our complete project plan in the next page.

		Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Revision**	**22 days?**	**10/27/20, 8:00 AM**	**11/17/20, 5:00 PM**		
2		Requirements - revisit	8 days?	10/27/20, 8:00 AM	11/3/20, 5:00 PM		Everyone
3		Scenarios & mockups - revisit	8 days?	10/27/20, 8:00 AM	11/3/20, 5:00 PM		Oncel Keles
4		Class diagram - revisit	5 days	11/4/20, 8:00 AM	11/8/20, 5:00 PM	2	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Yasin Kaya;Ozgurcan Oztas
5		Sequence diagram - revisit	3 days?	11/9/20, 8:00 AM	11/11/20, 5:00 PM	2;4	Afra Arslan;Ahmet Yigit Gedik;Goksu Baser;Nazim Berke Metin;Onur Enginer
6		Project plan - revisit	5 days?	11/3/20, 8:00 AM	11/7/20, 5:00 PM		Nursima Celik
7		W3C Standards Research	8 days	11/10/20, 8:00 AM	11/17/20, 5:00 PM		Everyone
8		Activity Streams 2.0 Research	8 days?	11/10/20, 8:00 AM	11/17/20, 5:00 PM		Everyone
9		**Back end**	**15 days?**	**11/10/20, 8:00 AM**	**11/24/20, 5:00 PM**		
10		Login-Signup	12 days	11/10/20, 8:00 AM	11/21/20, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
11		Homepage	12 days?	11/10/20, 8:00 AM	11/21/20, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
12		Dockerize	3 days	11/22/20, 8:00 AM	11/24/20, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
13		AWS Deployment - R&D	3 days?	11/22/20, 8:00 AM	11/24/20, 5:00 PM	10;11	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
14		**Front end**	**15 days?**	**11/10/20, 8:00 AM**	**11/24/20, 5:00 PM**		
15		Login-Signup	12 days?	11/10/20, 8:00 AM	11/21/20, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
16		Homepage	12 days?	11/10/20, 8:00 AM	11/21/20, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
17		Dockerize	3 days	11/22/20, 8:00 AM	11/24/20, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
18		AWS Deployment - R&D	3 days?	11/22/20, 8:00 AM	11/24/20, 5:00 PM	15;16	Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
19		**Mobile**	**15 days?**	**11/10/20, 8:00 AM**	**11/24/20, 5:00 PM**		
20		Login-Signup	12 days?	11/10/20, 8:00 AM	11/21/20, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
21		Homepage	12 days?	11/10/20, 8:00 AM	11/21/20, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
22		Dockerize	3 days	11/22/20, 8:00 AM	11/24/20, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
23		AWS Deployment - R&D	3 days?	11/22/20, 8:00 AM	11/24/20, 5:00 PM	20;21	Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
24		Milestone1	0 days?	11/24/20, 5:00 PM	11/24/20, 5:00 PM	9;14;19	Everyone
25		GDPR and KVKK Research	5 days?	11/24/20, 8:00 AM	11/28/20, 5:00 PM		Everyone
26		**Back end**	**34 days?**	**11/24/20, 8:00 AM**	**12/27/20, 5:00 PM**		
27		Profile - Admin Panel	5 days?	11/24/20, 8:00 AM	11/28/20, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
28		Product	7 days?	11/28/20, 8:00 AM	12/4/20, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
29		Search	6 days?	12/5/20, 8:00 AM	12/10/20, 5:00 PM	28	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
30		Activity Streams 2.0	5 days?	12/10/20, 8:00 AM	12/14/20, 5:00 PM	8;9;28	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
31		List-Notification	5 days?	12/10/20, 8:00 AM	12/14/20, 5:00 PM	28	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
32		Shopping Cart-Order-Purchase	11 days?	12/14/20, 8:00 AM	12/24/20, 5:00 PM	28	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
33		Comment and Rate	4 days?	12/24/20, 8:00 AM	12/27/20, 5:00 PM	28	Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
34		**Front end**	**34 days?**	**11/24/20, 8:00 AM**	**12/27/20, 5:00 PM**		
35		Profile - Admin Panel	5 days?	11/24/20, 8:00 AM	11/28/20, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
36		Product	7 days?	11/28/20, 8:00 AM	12/4/20, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
37		Search	6 days?	12/5/20, 8:00 AM	12/10/20, 5:00 PM	36	Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
38		Activity Streams 2.0	5 days?	12/10/20, 8:00 AM	12/14/20, 5:00 PM	8;14;36	Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
39		List-Notification	5 days?	12/10/20, 8:00 AM	12/14/20, 5:00 PM	36	Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
40		Shopping Cart-Order-Purchase	11 days?	12/14/20, 8:00 AM	12/24/20, 5:00 PM	36	Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
41		Comment and Rate	4 days?	12/24/20, 8:00 AM	12/27/20, 5:00 PM	36	Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
42		**Mobile**	**34 days?**	**11/24/20, 8:00 AM**	**12/27/20, 5:00 PM**		
43		Profile - Admin Panel	5 days?	11/24/20, 8:00 AM	11/28/20, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
44		Product	7 days?	11/28/20, 8:00 AM	12/4/20, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
45		Search	6 days?	12/5/20, 8:00 AM	12/10/20, 5:00 PM	44	Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
46		Activity Streams 2.0	5 days?	12/10/20, 8:00 AM	12/14/20, 5:00 PM	8;19;44	Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
47		List-Notification	5 days?	12/10/20, 8:00 AM	12/14/20, 5:00 PM	44	Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
48		Shopping Cart-Order-Purchase	11 days?	12/14/20, 8:00 AM	12/24/20, 5:00 PM	44	Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
49		Comment and Rate	4 days?	12/24/20, 8:00 AM	12/27/20, 5:00 PM	44	Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
50		Milestone2	0 days	12/29/20, 5:00 PM	12/29/20, 5:00 PM	26;34;42	Everyone
51		**Back end**	**22 days?**	**12/29/20, 8:00 AM**	**1/19/21, 5:00 PM**		
52		Messaging System	8 days?	12/29/20, 8:00 AM	1/5/21, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
53		Recommendation System	6 days?	1/5/21, 8:00 AM	1/10/21, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
54		Testing	10 days?	1/10/21, 8:00 AM	1/19/21, 5:00 PM		Burak Berk Ozer;Kaan Dura;Muhammet Tayyip Yazici;Ozgurcan Oztas
55		**Front end**	**22 days?**	**12/29/20, 8:00 AM**	**1/19/21, 5:00 PM**		
56		Messaging System	8 days?	12/29/20, 8:00 AM	1/5/21, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
57		Recommendation System	6 days?	1/5/21, 8:00 AM	1/10/21, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
58		Testing	10 days?	1/10/21, 8:00 AM	1/19/21, 5:00 PM		Afra Arslan;Ahmet Yigit Gedik;Yasin Kaya;Oncel Keles
59		**Mobile**	**22 days?**	**12/29/20, 8:00 AM**	**1/19/21, 5:00 PM**		
60		Messaging System	8 days?	12/29/20, 8:00 AM	1/5/21, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
61		Recommendation System	6 days?	1/5/21, 8:00 AM	1/10/21, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
62		Testing	10 days?	1/10/21, 8:00 AM	1/19/21, 5:00 PM		Goksu Baser;Nazim Berke Metin;Nursima Celik;Onur Enginer
63		Milestone 3	1 day?	1/20/21, 8:00 AM	1/20/21, 5:00 PM	51;55;59	Everyone

10 Milestone Scenarios:

10.1 Scenario 4

10.1.1 Cüneyt Özdemir (Guest)

10.1.1.1 Persona Cüneyt is 50 years old. He is a journalist, who worked for various Turkish Media channels for years. Now he is enjoying his time by making youtube videos about agenda of Turkey and the whole world. One day one of his subscribers recommended him a new e-commerce website and he should check it out. Now, before shooting a video of the site, he decided to check out this website briefly.

10.1.1.2 Preconditions

- He is a guest user, he hasn't logged in to an account yet.

10.1.1.3 Actions

1. He opens the website.
2. He scrolls down to look at the listed products.
3. He clicks the login button.
4. He tries his usual password, but gets an error as it is weak.
5. He decides to sign up using google sign up as a customer.
6. He is redirected to the home page as logged in.

10.1.1.4 Acceptance Criteria

1. Users shall be able to register as a customer by providing an e-mail address, a strong password and name/surname
2. Users should be able to use a Google account instead of e-mail & password while registering. (Requirement 1.1.1.3)
3. Registered users shall be able to verify their e-mail by clicking the link that is sent to their e-mail address. (Requirement 1.1.1.10)
4. Users shall choose to login as a customer or a vendor. (Requirement 1.1.1.4)

10.2 Scenario 5

10.2.1 Mehmet Unutkan (Guest)

10.2.1.1 Persona Mehmet is 43 years old. He is a cook, who worked for various kitchens and restaurants for years. He usually enjoys his time by shopping online in mainstream shopping sites. However he uses different password each time. Therefore he often forgets them. One day one of his colleague recommended him a new e-commerce website and he should check it out. Now, he is in his home, decided to download the Carousel app and create a profile.

10.2.1.2 Preconditions

- He is a guest user, he hasn't logged in to an account yet.

10.2.1.3 Actions

1. He opens the app.
2. He scrolls to look at the listed products.
3. He clicks the account button.
4. He clicks the SIGN UP button.
5. He chooses to the SIGN UP As CUSTOMER button.
6. He enters his Name, Surname, Email adress and a password of "111" to remember.
7. He accepts terms and conditions.
8. He clicks the SIGN UP button.
9. He reads the warning message and enters his informations again.
10. He uses "ABC32abc32" as his password this time.
11. He gets an e-mail to verify his sign-up credentials.
12. He verifies the e-mail and directed to the website's login page however he continues with his phone.
13. He enters his login information but enters "111" as his password and clicks login.
14. He reads error message.
15. He clicks the Forgot my password button.
16. He gets an e-mail to reset his password.
17. He resets his password to "MHT11mht11".
18. He uses his phone to login with correct password.

10.2.1.4 Acceptance Criteria

1. Users shall be able to register as a customer by providing an e-mail address, a strong password, and name/surname. (Requirement 1.1.1.1)
2. Registered users shall be able to verify their e-mail by clicking the link that is sent to their e-mail address. (Requirement 1.1.1.10)
3. Users shall choose to log in as a customer or a vendor. (Requirement 1.1.1.4)
4. Registered users shall be able to login with their e-mail and password. (Requirement 1.1.1.5)
5. Registered users shall be able to login with their e-mail and password. (Requirement 1.1.1.5)
6. A customer shall accept the system to store the password, the email, and the phone of the user to sign up. (Requirement 1.1.1.8)
7. Customers or vendors shall be able to select "Forgot your password" when they have forgotten their password and select their new password. (Requirement 1.1.1.12)

11 Structure of our codebase:

- **Branches:** We've decided to use 5 branches in our repository: master, development, backend, web, android. All these 5 branches have branch protection rules for any contributor to regulate the code pushing and code pulling actions within safety measurements. Also, before pushing our code into the repository, first we pull at our division branch(backend, web, android) and retrieve any changes, then we checkout to a new branch from that divisionary branch, implement our feature or add our bugfix and push to that branch on our remote repository. After pushing, we open a pull request from that branch to our divisionary branch and after an approval of sufficient number of revisions and pass of implemented or added checks into the branch, we merge that branch to our divisionary branch. After completing tasks in our project plan with confirmed changes, we merge our code from all divisionary branches to the 'development' branch to merge all work into one place. After that merging, we plan to check our structure and state of the product and if leader of each division approves the changes, we merge development into master branch to have our production image.

- **Pull Requests:** We've added checks and mandatory revision for each pull request to keep our codebase clean and structured. We've implemented a code format checking GitHub Action in our repository and it works fine for backend division. Also, automated unit-testing in GitHub Actions will be implemented in next few days for backend division.

12 Evaluation of the used tools:

12.1 Backend

12.1.1 Express

Express is a back-end web application framework for Node.js. It is free and open source. It has a minimal structure that can be extended using plug-ins allowing us to modify it to our needs. It uses middlewares to create a stack that each request follows. This middleware feature makes it very easy for us to add and delete features for any endpoint we want to change. We think it is going to be very valueble as the code gets more complex.

12.1.2 Docker

We used Docker to isolate our back-end application from the environment it was running. With Docker we could put all dependencies for our application in the image and run it on any computer with Docker installed.

12.1.3 Visual Studio Code

As a team we decided we would use Visual Studio Code as our coding environment. We used Prettier plug-in to format our code to follow a coding standard among team.

12.1.4 Terraform and Ansible

Terraform offers a declarative configuration language for creating resources on a number of platforms. Ansible is similar to Terraform but it is better suited for automatic configuration of VMs. We used Terraform to deploy our AWS EC2 instances. With Terraform we can easily scale our number of nodes in a more structured manner. Terraform has also integrated with Ansible, we are using them together to easily configure each node we create. We are also planning to use Ansible on its own to configure our VMs if ever need be.

12.1.5 Github Actions

We added a workflow for code quality checking using Prettier. This way we could always have our code at the standard level. We also tried to add a workflow for continuous delivery pipeline and it is under test. When it is finished we will have our backend application updated all the time.

12.2 Android

12.2.1 Android Studio

We have used Android studio IDE for coding. It is easy to learn and use. It has code builtin code formatter and git tools. This specifications makes coding and pushing really easy even for a beginner with Android and Git.

12.3 Frontend

12.3.1 Docker

We have used docker container to run our react web application. Thanks to docker environment, we have isolated our program from the system that the program runs on. No matter on which system the program runs, if the the system has the docker, dependency issues are no longer concerns. Additionally, the docker containers are so lightweight, easy to run and containers are constructed in a fast way.

12.3.2 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. However, React is only concerned with rendering data to the DOM, and so creating React applications usually requires the use of additional libraries for state management and routing Redux and React Router are respective examples of such libraries.

12.3.3 Prettier and Visual Studio Code

We have decided to use Visual Studio Code IDE while developing the frontend because it has Prettier code formatter. Thanks to this feature of the Visual Studio Code, our frontend team could work without any conflict about the format of the code and unnecessary code changes in commits.

13 Assessment of the Customer Meeting:

13.1 Experience of Yasin Kaya, our first presenter:

I had limited experience about presenting, especially in English. Thanks to the presentation that I talked on behalf of Cuneyt Ozdemir, I gained confidence, and now I feel more comfortable about presenting. My biggest concern about presenting was, not being able to speak fluently or forgetting the lines that I should talk about during the presentation. But I think, practicing the lines, and doing demos inside the team beforehand helped me to overcome my worries. On the other hand, there were some points that I should improve myself. As an example, the presentation was relatively short, therefore the content of the presentation should be enriched. Also, I will try to get in the persona more, in the future presentations.

13.2 Experience of Göksu Başer, our second presenter:

I tried to do a funny presentation with little jokes such as Mehmet Unutkan forgetting his password. Before actual presentation Yasin and I made a demo for our presentations to overcome anxiety. It helped a lot but I was still anxious before starting. I planned to present Android part of the project but after the presentation customer asked about web and back-end related issues. They caught me unexpected. It showed me I need to prepare not only my presentation but also other divisions jobs to answer these questions. However with the help of the team we managed to answer those questions. I think presentation itself was successful because I was very in control of scenario of mine since I also prepared the scenario to present.

In addition to all of these we learned that we should work on not only the scenarios in the presentation, but we also should work on the other functionalities that are not present in the scenarios. Also, we should be clearer about which functionalities we are supporting and which of them we are not supporting. Last but not least, we should have more deep scenarios.

On the other hand, thanks to the this presentation, we gained acceleration comparing before milestone. We noticed our deficits, and we've already started to work on them. Therefore, we think it will be easier for us to implement future requirements, by taking into account all the feedback given during and after the presentation.