Cmpe352 - Milestone Report-2

Spring 2020

CAROUSEL

Afra Akbaş
Ali Alperen Durak
Burak Berk Özer
Göksu Başer
Nursima Çelik
Oğuzhan Özboyacı
Onur Enginer
Ömer Topuz
Yasin Kaya

Contents

1 Executive Summary

1.1 Introduction

Carousel is an e-commerce platform that has different kinds and numbers of products for sale. Users can sell or buy products online for free. Our motivation is to provide our users with a well-designed, safe, and convenient online platform in the e-commerce area. The platform supports Android, and Web so that it is available on both mobile and the web. We also take into consideration ethical concerns while designing our platform. Therefore, we follow the rules of KVKK and GDPR and the standards of W3C as a software project.

Carousel supports several functionalities, and these functionalities differ from user types. There are four types of users in our platform, which are guest, customer, vendor, and admin users. Guest users are the users who did not register the system yet, but they can still buy a product. They can filter products and sort them according to their preferences. After they buy a product, they can also rate the product and comment on it. Customer and vendor users are the registered users of our project. Customer users can do anything that a guest user can do. In addition to that, they can create lists, adds products to them, and get notifications about a product if they want to. Vendors are retailers who add their products to our platform to sell them to other users. They can add their products and their descriptions, which can be Turkish or English, for sale. Admin is the last type of users in our platform. They are also registered users who manage the platform.

1.2 Work Done So Far

Until now, we design our platform in detail. We specified the requirements, both functional and non-functional, and desired outcomes of the system. Then, we created scenarios and mockups for different users and different situations, which helps us to validate requirements. Then, we implemented several API. We created a practice_app in Github which contains our different apps. Names of these apps are Search, Shipment Calculator, Google Shopping, Recommend and Covid Tracker. Some of them are related with our app, but there is also not related app.
For more information, further sections have detailed explanations.

1.3 Road Ahead

We learned how to build an app with Django framework. We practiced it by creating simple apps. We learned Github UI, such as pull request, and Issue section. From now on, we develop our project in a more complicated way together. We will develop apps that more related to our CAROUSEL project.

With this experience, we will plan more carefully CARAOSEL project. We will split this project to Web frontend, Backend, and mobile frontend, then we will proceed on it.

1.4 Challenges You Met As Group

As a group, we met with several challenges. Since all stay at home, we can only communicate online with Hangouts or Zoom. We try to keep in touch with slack and Whatsapp.

Most of the team members used Django for the first time. Therefore, we should have learned and research before starting to implement it. Until we got used to the Django framework, implementing backend and frontend and handling errors were troublesome.

When we deployed our apps to Github, we all encountered some troubles. For instance, some of our friends deployed their apps but forgot to add a configuration path to the main Django projects. And also some of our friends misunderstood and create multiple Django projects inside of the main Django project. They should create a Django application instead of the Django project. Besides, some friends committed the keys file to git without thinking security.

2 List and Status of deliverables

Name	**Delivery Date**	**Status**
Repository Arrangements	Feb 18, 2020	Finalized
Requirements Elicitation	Mar 8, 2020	Finalized
User Story and Mock-Ups	Mar 3, 2020	Finalized
Class Diagrams	Mar 24, 2020	Finalized
Use Case Diagrams	Mar 24, 2020	Finalized
Sequence Diagrams	Mar 24, 2020	Finalized
Project Plan	Apr 21, 2020	Finalized
RAM	Apr 21, 2020	Finalized
API implementation	May 27, 2020	Finalized

3 Evaluation of the status of deliverables

3.1 Our Repositories

We designed our personal repositories and our project repository in a way that we and our project look presentable to the customer. We clearly stated our personal communication information and our short biographies in personal repositories. For the project repository we choose a name, designed a project logo and put a photo of our team together.

3.2 Requirements

Writing requirements from scratch was complicated at first. The most updated deliverable in our project is our requirements because of that. We updated according to the feedback coming from our customer and also while designing stories an even while designing class diagrams we needed to make necessary changes in it. Final product is finalized with the last feedback from customer. In the proceeding periods of project design it most likely will be updated or upgraded again.

Requirements is our map in design. In all of other designs of the project we always make them fit according to the requirements.

3.3 Story and Mock-Ups

We created 3 different personas to simulate usage of our project. In this 3 personas users interact with the system in various ways (a customer returning a product, a vendor answering a message, a customer buying a product). These scenarios then put into mock-ups to show how system look while these interactions. Those updated with the feedback after the initial design. Purpose of these was to show what will happen in the different user scenarios.

3.4 Design Diagrams

Diagrams are done via task sharing. We designed first use case diagram then class diagram and lastly sequence diagram. It is done sequentially because every design is dependent one after another. After design we reviewed designs with the coming feedback.

With the diagrams in hand we can now have a more clear pathway in the project in terms of coding and software developing.

3.5 API implementation

We come up with 5 independent APIs. To implement the APIs, we have used the Django framework of python. Since we were all unexperienced in backend, we had rough times while learning it. But after some inital commits, we were able to implement what we are trying to. Then, we deployed our project to an AWS EC2 instance. Now we have learned basic idea of APIs, how to call them and implementing one in Django.

4 Evaluation of tools and processes

- **Slack:** Slack is our main communication app that can easily integrated with many different apps. It is used widely by professional teams around the world. We have several channels for different topics that helps us to identify each problem or update in a classified manner. We are still in the learning phase, therefore we can not use its all features easily.

- **Discord:** We are using discord while we working as a group with 2 or more members. It allows us to communicate and take decisions fast during finishing assignments, thanks to its voice channels.

- **Google Hangouts:** Due to the pandemics, we are not able to held our weekly meetings in person. Therefore, we switched to the Google Hangouts for our weekly meetings. In addition to video and voice sharing, it also has screen sharing which can be helpful during meetups.

- **WhatsApp:** During the emergent situations, we are using WhatsApp to receive quick responses. But, since it does not support to have multiple channels, it is very easy to miss some important messages. Therefore, we try to avoid using it.

- **GitHub:** GitHub is where our all works done get together. We keep ourselves up to date by following our wiki page. Also, it has some great features for tracing individuals. For example, by checking GitHub's commit history we are able to track each individual's efforts to the project. Also, using issue system we can easily identify the undone issues, prioritize them and then focus on them.

- **ProjectLibre:** We are using ProjectLibre to maintain our Project Plan. It allows assigning start date, finish date, resources (people in our case) or predecessor task to a specific task. It also shows a Gannt Chart with tasks which is crucial for team management. But, it is not user friendly to use. It works slowly and transferring data from one external app to ProjectLibre is not easy as expected.

- **Processes While Managing Project:** Firstly, we read weekly assignments during the meeting. Then, we try to divide the tasks into independent tasks. After that, we assign one member or group of members to each tasks. We leave the decision of communication channel (Discord, Hangouts, etc.) to groups. After finishing tasks, we try to rearrange the groups in a way that there is at least one member that knows the task and one member that is new to task. In that way, we are hoping to keep everyone up to date. It works good in theory, but in reality grouping people sometimes does not work. People in groups may tend to wait until somebody to take action. And when nobody takes action, we are not able to finish tasks by complying deadlines.

- **Django:** We chose to use Python as our development language and Django as our framework. Django makes a lot of task automatically for people. Also, it has the concept of apps, so that we can develop little pieces in isolation to build the big product.
- **Postman:** Some of us have used Postman to test the API they have chosen. It is easy to install and use.
- **Amazon Web Service:** We deployed our project to the AWS, by getting a free instance. Although it is not so easy to use at first, it provides crucial qualifications such as high availability, scalability, security.

5 A summary of work done by each team member

Names	Work Done
Afra Akbaş	I implemented a simple search engine for products. I took an API key from eBay development. I added a search app branch in the practice_app. I implement the search method in views.py file I gave the url pattern of this app into urls.py files both in the practice_app directory and search directory. I created the search.html file for the front-end of the app. I created a target directory and added this directory into .gitignore to hide secret keys and API keys. I stored my API key into keys.py file. I pulled a request and Yasin reviewed it. I did the review of Shipment Calculator app with Nursima.
Alperen Durak	
Burak Berk Özer	I discussed with Onur to help us better understand the technicalities of this assignment. I researched on publicly available COVID-19 apis. I utilized Coronavirus COVID19 API. Created getCountry function in the covid tracker api which gets statistics of a given country. By using django and bootstrap, I have created a basic frontend for my api. By using python unittest library, I have created unittests for my api. Throughout the development process, I have commited my changes delibaretly. Created a pull requested to main branch. Reviewed Onur's pull requested and merged it into main. Changed my codes according to review I got from my peers.
Göksu Başer	
Nursima Çelik	I created a project in Spotify and got client id and client secret. I created the app named recommend.
Oğuzhan Özboyacı	I decided to implement an api that translate website. I took a google cloud api key. I couldn't implement api due to my default about time management.
Onur Enginer	
Ömer Topuz	
Yasin Kaya	I initialized the project and push it to the GitHub. I implemented the Product Model in shipment_calculator app. I implemented the MapsAPI Class in shipment_calculator/google_api/utils.py. We used as helper functions to connect Geocoding APIs implemented by Google. I implemented the post_product() and list_products() functions in shipment_calculator/views.py. Goksu and I deployed the application to the AWS. I reviewed and commented on the pull request from "search" branch.

6 API Documentation and URL

We used an AWS EC2 instance as our server. Here is our application's URL: http://35.176.181.234/

6.1 Search

/search
This search URL lists the products cards by cards if their names contains the given keyword. The products are listed in the search page with their images, and their titles. Method: POST

6.2 Shipment Calculator

/shipment_calculator/add_vendor
This URL is used to create a vendor. After filling the form, a new vendor will be added to the vendor list.
Method: POST

/shipment_calculator/add_product
This URL is used to create a product. After filling the form, a new product will be added to the product list.
Method: POST

/shipment_calculator/<latitude>_< longitude>

This URL is used to list all product with estimated shipment price. Notice that, customer should give his/her latitude and longitude values as parameters.
Method: GET
Example Usage: http://35.176.181.234/shipment_calculator/40_30/ (A customer in Bilecik)

6.3 Recommend

'album/'
Retrieves a recommended album.
Method: GET

'customer/<int:pk>/'
Retrieves the most related 3 products special to customer with given id.
Method: GET

'customers/'
Lists all customers.
Method: GET

'customer/detail/<int:pk>/'
Get detailed page about customer.
Method: GET

'customer/new/'
Create a new customer.
Method: POST

'products/'
List all products.
Method: GET

'product/<int:pk>/'
Get detailed page about product.
Method: GET

'product/new/'
Create a new product.
Method: POST

'product/<int:pk>/edit/'
Edit a product with given pk.
Method: POST

'orderedproducts/<int:pk>
Get list of ordered products by a customer identified by pk.
Method: GET

6.4 Covid Tracker

/covid_tracker
This URL is used to search for a countries statistics. The requested country should be typed in the form. A new statistics requests also will be directed to this URL.
Method: POST

/covid_tracker/results
This URL is used to list searched countries COVID-19 statistics. Data will be in a tabular format and quite extensive. In order to be able to comprehend the data better, statistics for million population are included in this table. Users can request a new countries statistics by pressing the "Make a new search!" link in the bottom of the page (This will redirect them to /covid_tracker URL).