Bilkent University

Department of Computer Engineering

Senior Design Project

Project short-name: Outletter

Project Specifications Report

Muhammad Bilal Bin Khalid, Muehammad Saboor, Mian Usman Naeem Kakakhel, Daniyal Khalil, Muhammad Arham Khan

Supervisor: Hamdi Dibeklioğlu
Jury Members: Çigdem Gündüz-Demir and Can Alkan

Progress Report
Oct 12, 2020

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Contents

1 Introduction

In today's capitalistic age, shopping is one of the most time-consuming activities that people from all walks of life participate in on a regular basis. From shopping for some clothing articles, to a new pair of shoes to even buying the latest electronic gadget, people spend hours trying to validate their shopping decisions and confirming that they're getting the best possible bargain. But considering the inherent information gap in the manual store-by-store checking process [1], we spend many hours only to buy what we got the best deal on at the three stores we visited.

So seeing the potential for improvement in the process, we decided to introduce Outletter, A one-stop shopping companion mobile application that allows users to point their mobile phones towards items like fashion articles, electronics etc. Our app would automatically recognize the product in realtime and render metadata like price comparison with nearby stores, product reviews and other available variants in nearby stores, right in the users field of view using Augmented Reality. This way, whenever a user is out for shopping, they can rely on Outletter to help them make the most reliable decision about a product and be sure to get the best deal in a very easy and fast way. Considering that Outletter will be able to identify almost any product that is available online and that there aren't any steps with manual data entry or button clicks (the user can access all information just by pointing the mobile phone at a product), the application would revolutionize the shopping process for the mass user-base including the elderly.

Apart from this, Outletter will also provide the user with the latest deals and offers going on in a particular store as soon as they are near that store. This way, users will never regret missing out a good deal from a store just because they didn't watch an ad campaign and, as promised by our app, will always be making the most well-informed shopping decision possible.

1.1 Description

This will be a hybrid application developed using React-Native[2] to target the Android and iOS markets (totalling to cover almost the entire market share). The application aims to make the shopping process more well-informed, faster and easier for users from all walks of life and hence, the expected features will be as follows:

1. **Real-time product recognition**: The application will utilize the device camera's video input to identify the item frame instantaneously and then sends only the item frame to the cloud-based server for the item to be identified and the relevant data to be retrieved. The cloud-based application first tries to match the item against currently indexed products in our database (through some brand affiliations), the product elements on the internet using techniques like image comparison and text search from image. As soon as the item is found, returns all relevant data including different prices, product reviews, available variants in current and nearby stores etc. in real time. This process takes under 3 seconds and hence, provides users the liberty to view information about any product in a very intuitive and straightforward way.
2. **Product Information in field-of-view**: The application will render relevant information retrieved from the backend service in the device's field of view using ARCore [3]. So whenever the user points their phones at an object, they will be able to see anchored data like best price, product reviews and other metadata about the object anchored to it in the user's field of view.
3. **Location map for product options**: There will be a map feature for the user to see locations of other stores where the object they're currently pointing the phone at is available and their relevant prices.
4. **Product recommendations**: Display the user with relevant options to shop alongside the currently selected product. So, the app utilizes previous shopping patterns of users to identify relevant products that can be bought alongside the current product.
5. **Product Feedback**: Users will be able to like, dislike and leave reviews on products that users in the future will be able to have a look at when they scan the relevant product.
6. **Share products**: Users will be able to share the current product's information using a custom web link using the sharing app of their choice. This includes sending the link over WhatsApp, Messengers or other social media applications.
7. **Geo-tagged deals for nearby stores**: Whenever the user enters or is near a store, the user will see a popup in their application containing the latest deals and promotions going on at that store. This would require stores that have a brand affiliation with our application and have provided us with their latest

deals, or have made a deal or campaign made available over their websites for our backend engine to index. Hence, during shopping, users will never miss a great deal on articles and will be able to make more informed decisions.

If we have time after completing our project, we will look into adding the following features as well.

1. **Voice recognition to identify product:** In case if the application was unable to identify an item, users will have the option to narrate the item name so that the application can parse that and display information about the article. Hence, even if the application is unable to identify by visual analysis, speech-to-text conversion and search will be able to display relevant information about any item.
2. **AR directions to nearby stores**: If an item is available at a cheaper price in a nearby store, the application will be able to show directions in AR to the user so that they may navigate their way to another store. This way, users will not have to waste time finding their way to a store they are getting a better deal at.

We plan to implement most of the listed features from scratch while also utilizing trained models where available. Since these features require substantial computation and the mobile-end will already be running an AR rendering service, most computations will be done on our cloud server.

1.2 Similar Products and Technologies

Google Lens: Uses a similar approach to identify the product. But unlike our idea, Google lens does not show better prices and other relevant metadata about the product and instead just identifies it.

1.3 Constraints

1.3.1 Development Constraints

- The application will be targeted towards Android OS and iOS. Thus, we will be using React-Native to develop the front end of our application allowing us to generate native code for both Operating Systems.
- The backend Application Programming Interface (API) of the application will be developed using Django Framework [4] in Python 3 language because we

will also be using multiple Machine Learning libraries (Scikit) [5] which are available for Python.

- For storing the user and product data, we will be using PostgreSQL [6] as our Database Management System.
- For efficient searching of products, we will be indexing our data using Elasticsearch [7].
- ARCore will be used to augment the virtual content, on top of the camera view, on the mobile's screen.
- We will use Machine Learning models based on academic papers such as "DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations" [8] to recognize the attributes of clothes.
- We will use OpenCV [9] to preprocess the images and remove the noise features from them.
- We will use Google's Custom Search JSON API [10] to search for the attributes and features of certain products to obtain the provided information from their respective websites.
- We will use Google's Map API [11] to pinpoint the location of the store corresponding to the product currently in the camera's view.
- We will use Google's Cloud Text-to-Speech [12] API to get the description of the product from the user.
- We will use a recommender system, for generating recommendations for our customers, based on academic papers such as "HCRS: A hybrid clothes recommender system based on user ratings and product features." [13]
- We will use the Object Oriented paradigm to design the software in our project.
- We will use GitHub as our version control and source code management system. We will also use it for issue tracking purposes.
- We will use Microsoft Teams to communicate between the team.

1.3.2 Economic Constraints

- Google's Custom Search JSON API provides 100 search queries per day for free. Any additional requests will cost $5 per 1000 queries and are limited up to 10k queries per day.

- Publishing the android version of the app on Google Play Store will require a one-time payment of 27 USD. For publishing an application on Apple App Store a subscription fee of 99 USD is required.

1.3.3 Technological Constraints

- For Android, the minimum SDK version should be 24 (Nougat 7.0 or above).
- For IOS, the Cloud Anchors SDK is supported on all ARKit-compatible devices running iOS 11.0 or later.
- The device using this application needs a functioning camera and a stable internet connection.

1.3.4 Reliability Constraints

- The application will be tested in multiple environments and conditions to ensure that our methods are robust.
- The APIs used for searching products will be tested to ensure we reach a satisfactory level of accuracy.

1.3.5 Social Constraints

The use of this application may encourage people to rely more on their phones and reduce social interactions, such as asking for help from employees of the particular stores.

1.3.6 Language Constraints

The application will be in English, however, the Google search queries made will be in both English and Turkish to get the most accurate results possible.

1.3.7 Time Constraints

The reports related to this project must be completed within their respective due dates.

1.4 Professional and Ethical Issues

1.4.1 Ethical Issues

- The application will be in accordance with the Code of Ethics and General Data Protection Regulation (GDPR). [14]

- While scanning for products with the application it is possible to take unconsented photos of people which may cause privacy issues. For example, a user can unintentionally take a photo of a person who is in the background of the product. This can be solved by filtering the noise and unwanted parts of the photos.
- The application will not store any photos which are not consented by the user.

1.4.2 Professional Issues

- The source code of the application will not be disclosed to the public and will be kept in a private repository.
- We will have equal work division amongst the team members and will adopt a democratic process for decision making.
- Our team will have scheduled meetings at least once a week.

2 Requirements

2.1 Functional Requirements

2.1.1 System Requirements

The system should:

- Ask permissions to access local storage, camera, microphone and internet connection of the mobile device.
- Display a bounding box around the focused object in the camera view.
- Provide an option to capture the image in the camera view.
- Provide an option to record the product description verbally.
- Send the picture or the verbal description to the backend server for processing.
- Be able to receive the products which are present in the market and the recommended items for the user from the backend server.
- Display the top price, URL, and reviews of the selected object in the captured image on top of the current camera view.
- Provide an option to view top products in different shops with different prices.
- Display the top products in different shops with different prices as a list.
- Provide an option to view recommended items.

- Provide an option to view the location and directions of the store the current product is available at.
- Display the recommended items as a list.
- Provide an option to like/dislike an item in the camera view.
- Provide an option to share an item in the camera view to the social media.
- Provide an option to save an offer to the wish list to view the product later.
- Provide an option to open the wish list.
- Provide an option to delete an item from the wish list.
- Provide an option to open the URL of an item in the wish list.
- Provide an option to share an item in the wish list to social media.
- Provide an option to leave a review of an item in the wish list.
- Provide an option to view the reviews of an item in the wish list.
- Provide an option to like/dislike an item in the wish list.
- Provide an option to view geo-tagged deals in the camera view.
- Display the selected geo-tagged deal.

2.1.2 User Requirements

The user can:

- Allow or deny permissions to access local storage, camera, microphone and internet connection of the mobile device.
- Select an object to focus in the camera view.
- Capture the image in the camera view.
- Describe the product verbally.
- View the top price, URL, and reviews of the selected object in the captured image on top of the current camera view.
- Select the option to view top products in different shops with different prices.
- View the top products in different shops with different prices as a list.
- View the location and directions of the store where the current product is available.
- Select the option to view recommended items.
- View the recommended items as a list.
- Like/dislike an item in the camera view.
- Share an item in the camera view to social media.

- Save an offer to the wish list to view the product later.
- Open the wish list.
- Delete an item from the wish list.
- Open the URL of an item in the wish list.
- Leave a review of an item in the wish list.
- View the reviews of an item in the wish list.
- Like/dislike an item in the wish list.
- Share an item in the wish list to social media.
- Select the option to view geo-tagged deals in the camera view.
- View geo-tagged deals in the camera view.
- Allow or deny permissions to share his/her preferences for the purpose of improvement of the system.

2.2 Non-Functional Requirements

2.2.1 Availability

- **Service:** The server downtime should not be more than 1% on a monthly regulation cycle otherwise the server should be available for requests 24 hours a day, 7 days a week.
- **Location:** The Application will only be available to users in Turkey, due to the limited scope of websites that can be parsed.
- **Operating System:** The Application should be available in both Androids 7.0 or higher and iOS 11.0 or higher operating systems.

2.2.2 Response Time

The Application Response Time for user queries and searches should be less than 3 seconds. The Response Time for User Recommendations should be less than 1 second.

2.2.3 Scalability

The Application usage is dependent upon the user demand. Hence the application should be scalable to meet with increasing user demands by increasing the number of users and database storage. For now, due to not having proper production servers, the number of concurrent users that would be able to use the application would not be more than 100.

2.2.4 Accuracy

The product results given by the application should have at least have an Accuracy of 90% and the recommendations should all be based on the personalized tastes of the user.

2.2.5 Extensibility

The application should be extendible to different areas and categories to meet the user demand if needed. Currently, as mentioned before, the application is available only in one location which is Turkey.

2.2.6 License

All the datasets used in training the models, and all the libraries used in the development should be properly licensed and accredited.

2.2.7 Privacy

The application should ask the user before using their data in improving the recommendation system to more suit their tastes.

2.2.8 Maintainability

The application should adopt a low coupling modular design to allow the modules to not affect each other during alterations.

2.2.9 Network & Server

The Application should run smoothly on 1mbps internet smoothly without any issues. The server should be able to handle 100 concurrent users concurrently. The server should be running on a minimum 2.2 GHz.

2.2.10 Usability

The Application should be user-friendly and have an intuitive design to help promote user-functionalities.

3 References

[1] Mohsin, Maryam. “10 Online Shopping Statistics You Need to Know in 2020 [Infographic].” [Infographic]. Oberlo, September 29, 2020. https://www.oberlo.com/blog/online-shopping-statistics.

[2] “React Native · A Framework for Building Native Apps Using React.” Accessed October 12, 2020. https://reactnative.dev/.

[3] “Build New Augmented Reality Experiences That Seamlessly Blend the Digital and Physical Worlds.” Accessed October 12, 2020. https://developers.google.com/ar.

[4] Django. Accessed October 12, 2020. https://www.djangoproject.com/.

[5] “Learn.” scikit. Accessed October 12, 2020. https://scikit-learn.org/stable/.

[6] “The World's Most Advanced Open Source Relational Database.” PostgreSQL. Accessed October 12, 2020. https://www.postgresql.org/.

[7] “Open Source Search: The Creators of Elasticsearch, ELK Stack & Kibana.” Elastic. Accessed October 12, 2020. https://www.elastic.co/.

[8] DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations - IEEE Conference Publication. Accessed October 12, 2020. https://ieeexplore.ieee.org/abstract/document/7780493.

[9] OpenCV. Accessed October 12, 2020. https://docs.opencv.org/.

[10] “Custom Search JSON API | Programmable Search Engine.” Google. Google. Accessed October 12, 2020.https://developers.google.com/custom-search/v1/overview..

[11] Google Maps API. Google. Accessed October 12, 2020. https://developers.google.com/maps/documentation/javascript/overview.

[12] “Cloud Text-to-Speech API | Cloud Text-to-Speech Documentation.” Google. Google. Accessed October 12, 2020. https://cloud.google.com/text-to-speech/docs/reference/rest.

[13] Hu, Xiaosong, Wen Zhu, and Qing Li. “HCRS: A Hybrid Clothes Recommender System Based on User Ratings and Product Features.” arXiv.org, November 25, 2014. https://arxiv.org/abs/1411.6754.

[14] “Code of Ethics.” Accessed October 12, 2020. https://www.nspe.org/resources/ethics/code-ethics.