IZMIR UNIVERSITY OF ECONOMICS

FACULTY OF ENGINEERING

COMPUTER ENGINEERING

FENG 498 FINAL REPORT

Personal Appointment Assistant

Authors: Sercan Kavdir - Emre Serbest

Supervisor: Asst. Prof. Dr. Kaya Oğuz

Contents

Abstract

Time is the most precious thing that all people have equally in the world and how efficiently it is used takes people further in life. By creating this platform, we aim to help people to manage their time by improving appointment processes. Another advantage of the platform is including a variety of business services from different sectors. Also, people struggle to find a specific service among a variety of services. The recommendation would be helpful to find the desired service that fits them best. The most important feature that distinguishes our platform from others is personalised suggestions, to do this Collaborative Filtering is one of the used methods in recommendation systems. There are different techniques that we have researched such as, Graph-based method and Matrix Factorisation method. See the details in Section 2. So far, we focused on the main functionalities of the platform in the first development process. In the following processes, we are going to use this knowledge in order to implement the Collaborative Filtering mechanism into our platform. Also, we have researched about which factors affect users while they are shopping online.

The project has an MVC(Model-View-Controller) architecture. Also, the latest technologies were chosen while developing this project such as Node.js, MongoDB, React, and React-Native. We also used a variety of packages and latest libraries to make the application more efficient and elegant. We tested some unit of the back-end endpoints and the obtained results are reported. Apart from that, we faced problems and difficulties while developing this project and we overcame these problems by working in a planned and organised manner.

1 Introduction

Time is a resource that people cannot save but only spend. It is free but also priceless. Everyone has it equally but only few people are aware of its worth. People have to use the time efficiently to make a difference. Efficiently used time brings along various benefits and takes them a step further. Thus, managing and planning time properly lead people to success. People who can not manage their time, may complain about not having enough time to fulfil their responsibilities and they delay them. Additionally, these people may not be able to spend time with their families, friends or themselves and not be able to focus their work due to not using the time productively.

With the advancing technology, a variety of digital calendars, planners, and similar applications are available. In business life top firms strongly believe in the necessity of using that type of software in order to coordinate and track their flow of work which leads to an increase their value. SAP(Systems Analysis and Program Development) is the most known program among these types of programs, however, small and medium-scaled businesses may not afford it because of costs. Also, creating similar program may be challenging, in terms of, hiring qualified developers and having financial difficulties for small and medium-scaled businesses.

So far, we have discussed from the perspective of businesses and their management but when we look at the time management and planning issues from the perspective of individuals, there are not many products that help them to organise their time. Even though there are lots of digital calendars and planners, they do not serve as a bridge between users and businesses, thus people do not have a chance to compare with their personal calendar with an available appointment time. Hence, these digital calendars and planners become useless at some point. In addition, when people wish to get a service from large business industry, they find it difficult to locate the most fitting service to be pleased. For example, when people would like to get a service, first, they wish to see photographs of a business, then, they read comments about how that business served to other people in the past.

So, this process may take long time during the busyness of the day. To sum up, it would be great to have a common program that helps people and businesses to manage their time and above all, to connect people and businesses.

1.1 Problem Statement

There does not exist such a platform that brings variety of business services and their customers together to organise their appointment processes. Similar platforms do not provide personal recommendations for helping users to find services that fit them best. Therefore, people need more on the experience and recommendation of their friends when they are looking for a service. Also, the places people prefer to get services are directly related to their income levels. People with good income can choose to get served in luxury places and vice versa. Considering the social-economic situation of people, there is no system that can help them according to their preferences.

1.2 Motivation

We planned to create an online personal appointment assistant software which includes a variety of private business services in one place. It offers users recommendations that fit them best when they search a service. There does not exist such a platform that brings all described features together.

Nowadays, due to pandemic conditions, people need to keep social distancing in public places. Governments published some instructions about keeping social distance. One instruction is not to be in the same place more than the stated number of people. For example, to prevent the crowd in beauty salons, the best way is to organise appointments online. So, this type of appointment management applications will gain more value in the following days.

In addition to the standard applications, we would like to help users by recommending services that fit them best. We aim to provide a good user experience to people by using Collaborative Filtering (CF) method. Main idea of collaborative filtering method is that users who like similar things earlier, may like common things in the future [1]. The details of the CF will be discussed in the Literature Survey. When users would like to get an appointment by using Randeu, they had to give some information about themselves such as age, location, interests, and hobbies. Afterwards, users will have a chance to choose a service that fit them best among a variety of businesses from large business industry throughout Randeu.

Another reason of doing this project is there does not exist an available common online appointment platform. Some businesses have their own appointment application but it may not be preferable to download an application for each business by users. Therefore, businesses try to organise their appointments via phone calls or face to face. In this case, customers have to reach business owners when they need to make an appointment, however, business owners may not be available to answer phone calls all the time or even if they answer them, these phone calls may require a long time to agree on an available time. On the other hand, these phone calls decrease the productivity of work and it leads to complexity in their private life. Moreover, businesses may not be able to reach a wide range of customers. Also, people may have difficulties finding alternatives for a specific service. From these problems, people and businesses may suffer. Obviously, it is an inevitable consequence of not having a common appointment application.

2 Literature Review

There are already existing hub services that bring several products and their customers together. For example, in Amazon, businesses sell variety of products, such as computers, phones, clothes and books in one and common platform. A domestic example is Yemeksepeti which brings different types of food companies together and people who use the platform are able to select desired food. For instance, a larger company Delivery Hero acquired Yemeksepeti for 589 million dollars in 2015 [2], which shows a work opportunity in this field.

Kolay Randevu, Doktor Takvimi, and Veterinervar are some of our competitors that provide a platform to get appointments from businesses for people. Each of them focus only specific fields. See Figure 1. Kolay Randevu is established on 2015 and it provides appointments from personal care business industry such as hairdresser, massage parlour and beauty parlour. Co-founder of Kolay Randevu Uğur Çivi stated that in the last 2 years since the establishment of operations in Istanbul they get over 30.000 appointments monthly. Kolay Randevu mainly operates in Istanbul and Ankara also they have few business services at Izmir, Antalya and Eskişehir. In 2016 Kolay Randevu took 1 million Turkish Liras investment [3]. Doktor Takvimi is established on 2012 and it provides appointments from health care business industry such as dentist, dermatology, psychiatrist, orthopaedic surgeon. According to, analysis of doktortakvimi.com they reached 22 million user visit in 2015. Also, they increased number of doctors over %200 and they took 10$ million investment from Docplanner [4]. Veterinervar.com is established on July of 2017 and it provides appointments for animal health care. The team of veterinervar.com consisted of 6 people when they established and they agreed with about 80 veterinaries

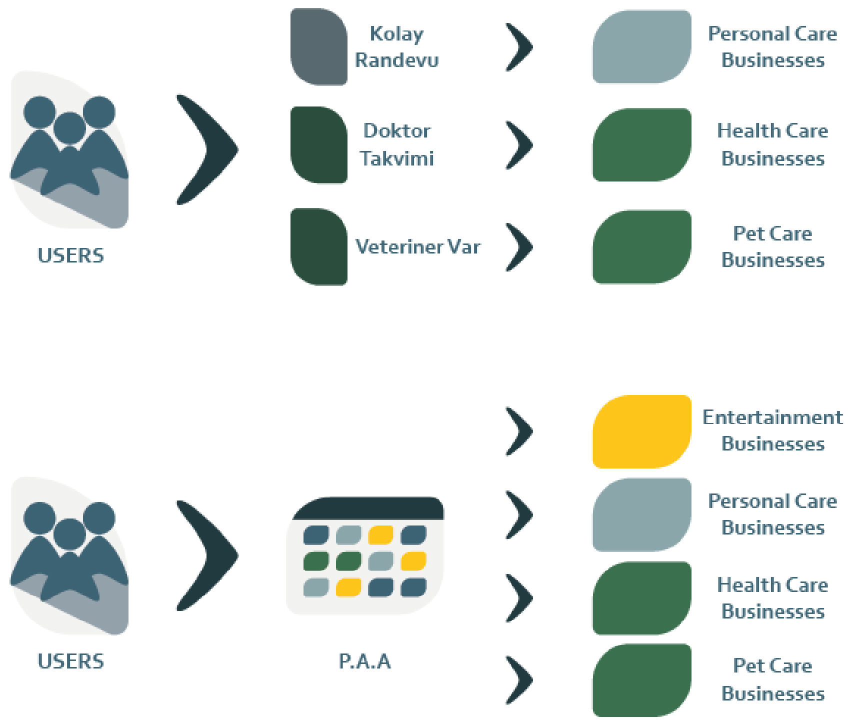
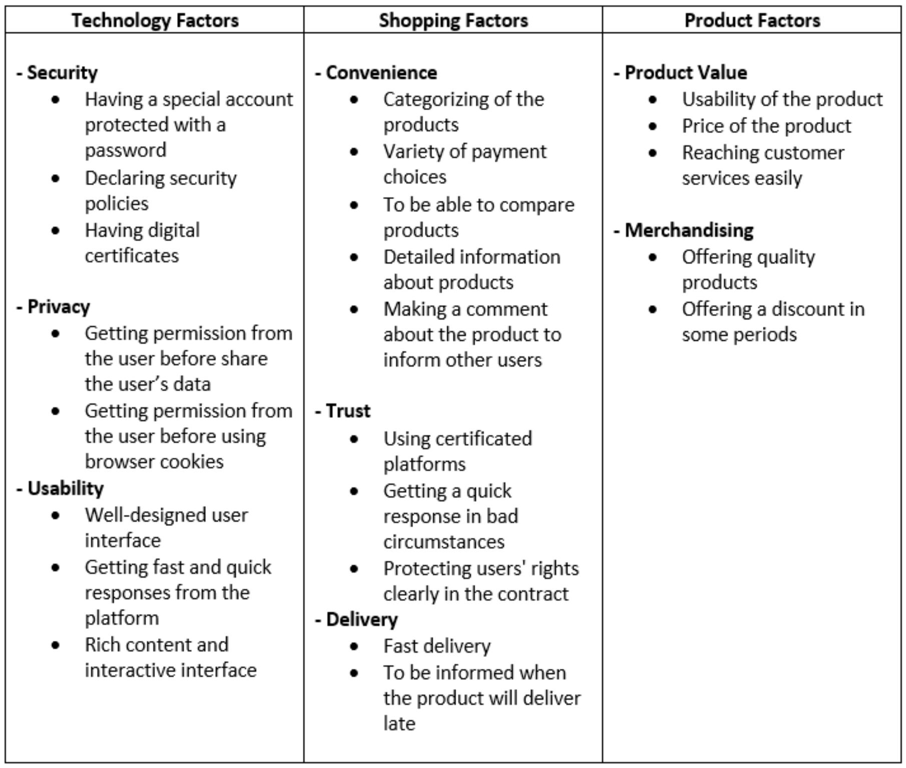


Figure 1: The Differences Between Randeu and Competitors

in first 4 months. In November veterinervar.com reached 500 customer's registrations. Veterinervar.com offers monthly membership and membership fee is 50 Turkish Liras for a month [5].

Advanced information technologies led to significant changes in markets and businesses in recent years. In addition, some communication systems built, in order to provide a channel between business to business (B2B) and business to customer (B2C) [6]. With these channels the needs of some business processes eliminated [7]. In the literature, the intermediary platforms which are defined as cyber, online, and electronic, act as a bridge between business and their customers [8]. We claim responsibility for providing a communication line between businesses and their customers. Also, these types of platforms can be described as Software as a Service (SaaS). SaaS, instead of buying all the rights of the software, it can be defined as paying as much as the software is used. In this way, the business does not need to carry out the cost of software maintenance [9].

Building the communication line is crucial but challenging. End-users are



Technology Factors	Shopping Factors	Product Factors
- Security • Having a special account protected with a password • Declaring security policies • Having digital certificates **- Privacy** • Getting permission from the user before share the user's data • Getting permission from the user before using browser cookies **- Usability** • Well-designed user interface • Getting fast and quick responses from the platform • Rich content and interactive interface	**- Convenience** • Categorizing of the products • Variety of payment choices • To be able to compare products • Detailed information about products • Making a comment about the product to inform other users **- Trust** • Using certificated platforms • Getting a quick response in bad circumstances • Protecting users' rights clearly in the contract **- Delivery** • Fast delivery • To be informed when the product will deliver late	**- Product Value** • Usability of the product • Price of the product • Reaching customer services easily **- Merchandising** • Offering quality products • Offering a discount in some periods

Figure 2: Factors affecting the intention of buying online

picky and cautious while they buy products or services on the internet. Various factors may affect the decision of the user. These factors are defined by Constantinides into three main categories like technology, shopping, and products [10]. These factors are shorty described in the table below, see Figure 2 [11, 12]. In the development time period of this platform, these factors will be considered carefully.

The filtering mechanism that we plan to implement in our project is based on Collaborative Filtering (CF). CF is defined basically as a method that suggests an item to people who are similar to each other, based on their feedback [13]. Generally, CF can be divided into two groups which are memory-based collaborative filtering and model-based collaborative filtering. Memory-based collaborating filtering is also divided as user-based and item-based [1]. User-based CF anticipates score of an item by a user among similar users that has already scored on that item [14]. Item-based CF is gives suggestions according to earlier ratings on similar items [15]. To determine similarity between users or items usually used as "k-nearest-neighbours" method. Previously trained relational user and item matrix is used for new recommendation in model-based collaborative filtering [16]. For better recommendation these techniques are not enough so, we need side information for users and items. Side information can be collected in two ways: user-contributed information and social networks. We will focus more on user-contributed information. A detailed survey about social networks is available in [1].

User-contributed information became an important role for better recommendations in these days. This information can be collected in four different ways which are tags, geotags, multimedia content and reviews and comments. In the project, in order to improve our recommendations, we will consider geotags, reviews and comments. More information about tags and multimedia content is available in [1]. Geotags are used for determining the position of users and items. From social media posts, we can track footprints of a user [17]. This geotag information will help us to recommend a business service to a user according to location range. In addition, the number of comments and reviews of an item may not be enough also, the content of a comment and reaction on this comment by other community members also plays an important role. This is crucial side information for better recommendations [18, 19, 20, 21]. We will be considering reviews and comments in our project to provide reliable recommendations.

After collecting side information about users and items, predictions while recommending will be more acquired and precise. Traditional CF algorithms can differ in three forms such as extending memory-based collaborative filtering, extending model-based collaborative filtering and graph-based approaches. See more detailed information about extending memory-based collaborative filtering and extending model-based collaborative filtering in [1]. The graph-based approach combines side information of users and items into CF algorithms. All connection between user/item and their side information assets bringing together into one graph and then reconsider these in a graph mining technology [22].

When we expand our research, we found that in 2006, Netflix announced a competition which aims to improve the accuracy of predictions for its movie recommendation system [23]. In 2009, BellKor's Pragmatic team has won the competition. We examined their approach to improve our recommendation system. As we mentioned above there is a well-known technique which is used in CF is neighbourhood methods. The weak side of collaborative filtering is cold start problem. When a new user or item added to the system, classical CF methods become unpractical because of no previous information about it. There are two main types of collaborative filtering which are the neighbourhood methods and latent factor models. Latent factor models are a different touch in collaborative filtering. Latent factor models are lean on matrix factorisation which gives more accurate results than classic nearest neighbour methods. Matrix factorisation characterises both items and users vectors of factors deduced from item ratings. One of the powerful sides of matrix factorisation is, when a direct rating is not existing, the system can comprehend by analysing user interaction history. See more detailed information in. For example, when recommending a movie, rated movies by a user can be related with genres of movies. So, the preferences of the user can be identified from these reactions. In our project, we aim to develop such a technique that a service will be recommended to other users who have similar side information, based on preferences and ratings of similar users in terms of age, gender and location.

During the first semester, we did a variety of researches about Collaborative Filtering to gain an advantage over our competitors and make this project idea valuable. After the first semester, we applied to some enterprise development events, and also, we interviewed some businesses to define the requirements of possible customers and users. According to our project plans, our main goal was to implement the fundamental functionalities. Therefore, we did not put into practice our researches about Collaborative Filtering. In the upcoming progress, we will definitely need that researches to implement for providing personal recommendations for helping users to find services that fit them best.

3 Methodology

Front and back-end technologies help developers to build web and mobile applications. As the applications become more complex, the architecture becomes more vital to have a more maintainable and scalable solution. MVC is one of the most preferred architectures. MVC stands for model, view, and controller. Developing an application according to the MVC structure provides developers to easily modify a unit of the application without knowing all units of it [24]. Model layer represents database entities and their relationships. View layer visualises models to users. Controller layer is the bridge between model and view and it manages to create, read, update and delete processes. See Figure 3

3.1 Architecture of the Project

In our project, we used the MVC architecture. In model layer, MongoDB is used for collecting application data. MongoDB is a document-based database storage unit. Data structure in MongoDB consists of field-and-value pairs and set of pairs are described as document. These documents are very similar to JSON objects and they are stored in collections which are similar to tables in relational databases. Also, MongoDB provides flexibility because of schema-less architecture. Behind the scenes, on the server side, MongoDB converts JSON data to BSON which is its binary version that can be stored and can queried more efficiently. Also, we used Redis to store user activation key. Redis is an open-source in-memory data structure store, can be used as database, cache and message broker. In local registration flow, a new user profile is created as inactive status. To determine activation flow, the server creates a unique key with expires time for the user and it is stored in the Redis cache database, then server sends an email to the user. If the user clicks to the activation link, his/her status changes to active state.

In view layer, there are two different views which are web and mobile. In the web side of the project, React was used which is a JavaScript library and developed by Facebook for developing interactive user interfaces. React is

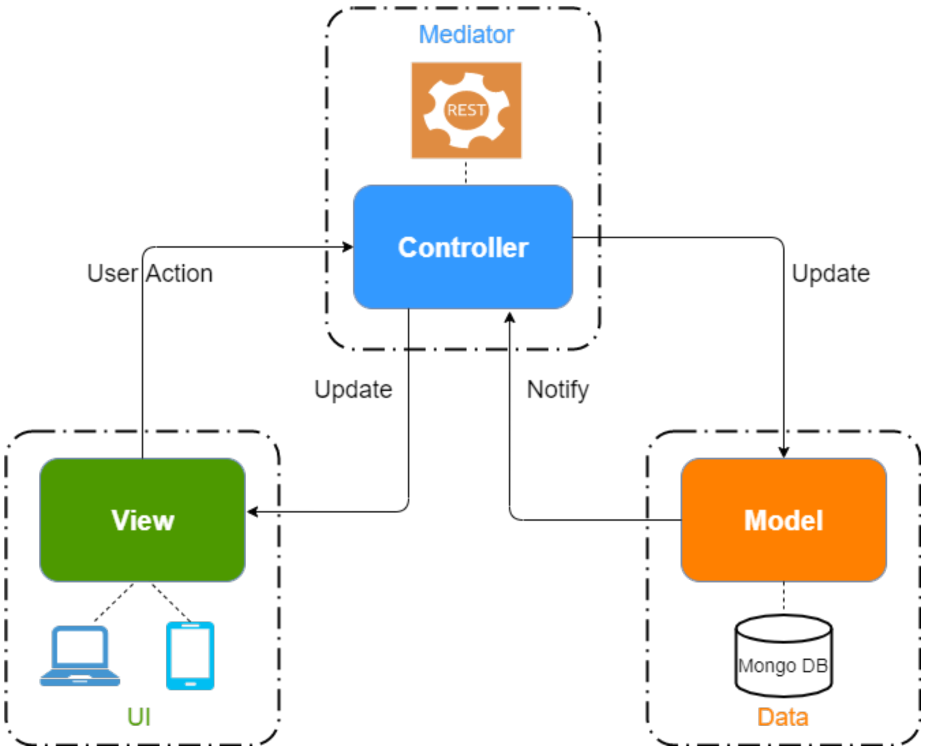


Figure 3: Model - View - Controller

component-based and it uses Virtual DOM (Virtual Document Object Model) which is a blueprint of real DOM of the HTML document. The components are rendered in Virtual DOM before they synced in real DOM. In the mobile side of the project, React-Native was used. React-Native is a JavaScript library used to develop native Android and iOS mobile applications. React and React-Native using the same design principles, except designing interfaces. Controller layer was implemented as a RESTful API which uses GET, PUT, POST or DELETE HTTP methods to retrieve, change, create or remove a resource.

3.2 State Management

Nowadays, users interact with applications more than in the past. In the past, after every user action/request the whole page was downloaded from the server and was rendered again. Thanks to Asynchronous JavaScript and XML(AJAX), it is now possible to load the page dynamically without having to reload the complete page. Behind the scenes, React also uses AJAX and these technologies increased the user experiences in a good way. In React, only updated components are rendered by the browser and it gives the feeling that we are using a single-page-application. In these components, there are dynamic data to be

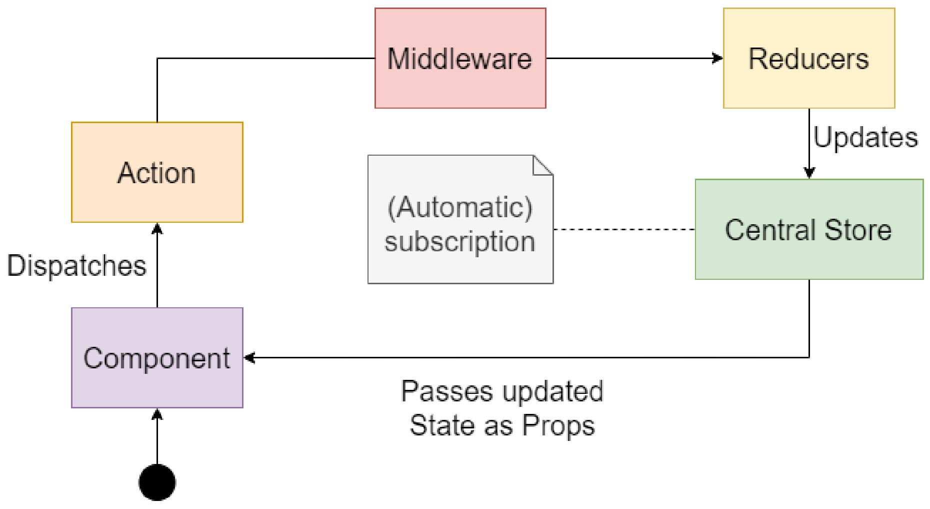


Figure 4: State Management with Redux

displayed and they should be changed after if there is new content in that data. In React, this management is handled by the state and provides distribute data between components. This distribution is provided by props. Briefly, the prop is the object that holds the data from parent component in React and the data are read-only in the respective component.

React and React-Native handle re-rendering processes of the components in the same way for every state and prop change. In some cases, application states are needed to be shared most of the application components such as; users' authentication states. In such cases, handling the state becomes challenging. In order to manage the shared states from one place, we used a Redux external package that uses by a large community.

In the Redux there are actions and reducers. Actions tell what to do in the specific action to the reducer. Reducers decides how to change the state according to given action type. In our application, we grouped reducers and actions into different classes to organise various user actions. These reducers are combined in a root reducer to update the central state. For example, to fetch all businesses three different action types are described, these are "start, success, fail". In the related component, we dispatch the asynchronous **InitFetchBusiness()** middleware method. In this method, we dispatch immediately the start action to change to the loading state. Then, we make a network request to the backend server. If there is no error, we dispatch the success action and send it to response data as a parameter to the success action. Otherwise, fail action is dispatched and send the error as a parameter to the fail action. Later, the reducer catches the dispatched action and manipulates the state according to the defined action type. Finally, the component which listens to the central store gets the updated state and re-render on the screen 4.

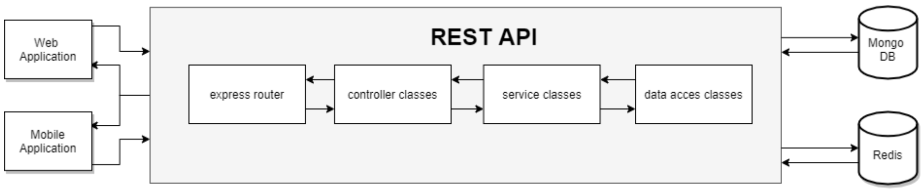


Figure 5: Closer Look into RESTful API

3.3 RESTful API with Node.js

Node.js provides to run JavaScript code on the server as opposed to being forced to run it on the client. Node.js is built on Chrome V8 engine that is an open source project supported by Google. The job of V8 or other engines is to take JavaScript code and compile it down to machine code that a machine can execute. Both node.js and Chrome V8 engines are written in C++. Advantages of the Node.js are non-blocking I/O model, event-driven approach to handle with the processes more efficiently. Node.js is mostly used for JSON API based, single page, I/O bound, data streaming applications. Most known companies that use Node.js are Netflix, Paypal, Linkedin, Uber, Yahoo and this shows that Node.js is powerful to be used serious real world applications. Node.js is open-source and free. It has great community around the world. It has its own built-in packages that comes when it is installed. Also, developers in the community share and borrow their packages via Node.js Package Manager(NPM).

Express is a Node.js web application framework which is an NPM package which is also powerful and flexible for creating mobile and web applications. Developers can use HTTP methods and middle-ware utilities, such as authorisation, in a quick and easy way. The server is listens to the port with the Express method. When a request is made, the related router catches it and before it redirects to its controller middle-ware checks the user authorisation. VerifyToken middleware class attaches the user information into the request if the token is valid, then send it to the controller. Controllers parse the needed information from the request and then, calls the related service layer which helps to connect database layer and whole result checks about queries are done in this service layer. If there is no error, requested data is sent to the client in JSON format and if any error occurs, it throws the error to the controller layer and it is caught in the related controller's try-catch block. Later, the result of the request is returned in JSON format to be handled in the view layer. See figure 5.

Apart from the Express NPM package so far, we included mongoose, passport, validator, bcrypt, and jsonwebtoken. Mongoose is a package that helps to use methods of MongoDB in Node.js environment. Validator package helps to validate model schemes to avoid invalid attempts from a user. Bcrypt package hashes the entered password by a user. Passport is an authentication middleware that works well with the express and it uses different strategies to authenticate user requests. In our project, we used three strategies such as Google,

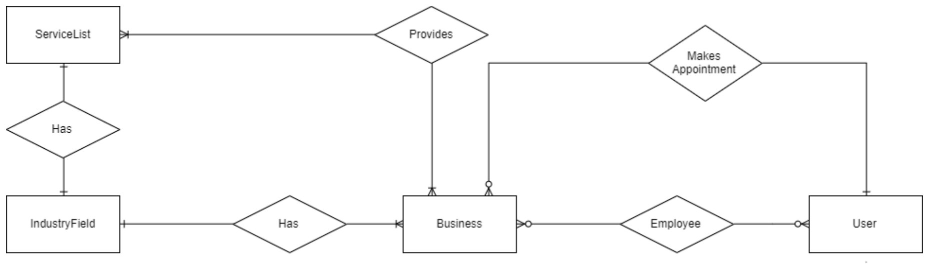


Figure 6: Entity Relationships Diagram

Facebook and local. Jsonwebtoken package creates a web token that identifies user access certification.

3.4 Model of the Project

Our model represents how our data structures are stored in our application. There are industry fields that have the main industry title and service type. For example, health care is an industry and it has service types such as dentists or medical doctors. Service lists are defined for each service type by the system administrators. Each business has to choose provided services from the service list which is defined by the system administrator.

There are different user types. A user of the application may work at a business. These users are defined as an employee of the business. Also, each business has to have at least one business owner which is also a user of the application. If a user is not an employee of a business and is not a business owner, then this user is an end-user.

All types of users can make an appointment with a business. When making an appointment, a user has to select a service that is provided by an employee who works at that business.

3.5 Designing Mock-up Screens

Designing is a crucial point for an application before starting to code and it helps to visualise the whole working application. It helps to focus on coding the application without distraction and also, provides coding of the application in a more organised way. For designing mock-up screens, we planned to work with a graphic designer student from our university. We could not work together with him because of his Erasmus during the project development. As we mentioned in the risk analysis section of the first report we designed the mock-up screens without getting any help. To solve this problem, we talked with our supervisor and he managed a meeting with a professional graphic designer to talk about the essential points while designing the mock-up screens. In order to design application pages for both mobile and web, we used a program called as Balsamiq.

3.6 Publishing Applications

Github is a version control system that we used in our project development. After we finished the development of the back-end of the application, we deploy it to a cloud server to not to work locally while developing view layers of the application. In the past, we planned to work with AWS but it became complicated when we tried to deploy the application on an AWS server. Therefore, we looked for another solution and saw that it is much easier to deploy a Node.js application to Heroku cloud server. Github and Heroku are working smoothly to manage the application releases. After we created an account on Heroku, we connected to the GitHub account to manage the master branch of the application. When the connection established with the Heroku and GitHub repository, we created an application on Heroku and deploy the master branch. Heroku server bundled the application automatically. Lastly, we imported the NodeJS environment variables to the Heroku server and the application went to the working state without a problem.

4 Results and Discussion

Software testing is the process to examine whether a software can meet the features expected of it. In this way, bugs and errors in the software can be found and corrected and adapted to emerging requirements. There are a variety of testing software types such as unit testing, integration testing, regression testing, system testing, etc. Unit Test is a software development process in which the smallest testable parts of the software are examined to run correctly, individually, and independently. In our project, we implemented different unit test cases for the back-end endpoints. To do that, we used “Jest” framework for testing JavaScript code.

We divided our tests as User, Sector, BusinessType, Business, Service, and Appointments test classes. In some cases, we needed some dummy data to organise database before running tests. To setup the database correctly, we used tear down methods provided by Jest framework such as “beforeEach()”, “beforeAll()”. **BeforeAll** method runs before all test cases in defined class. We used this method when writing Business tests because business creation depends on other data, for example Sector and BusinessType. **BeforeEach** method runs before every test case run. We used this method to prevent confusion by clearing the old dummy data on the database. After a user is registered into the application, we send a “welcome email” and “activation link”. To disable sending these emails while testing these endpoints we used mock functions provided by Jest.

4.1 Test Cases

4.1.1 User Tests

Test Procedure: Should create a user with valid fields.
Fields:

1. **fullName:** required, cannot be empty and only string values
2. **email:** required, cannot be empty and valid email input
3. **password:** required, at least 6 characters and cannot be empty
4. **passwordCheck:** required, at least 6 characters and should match with password value

Expected Results: Registering the user successfully. Otherwise, proper status code will be displayed for the request.

Input Values:

1. **fullName:** “Emre Serbest”
2. **email:** “emre@test.com”
3. **password:** “123456”
4. **passwordCheck:** “123456”

- **Expected Output:** Successful Register Status Code (200)
- **Pass / Fail:** PASS

Input Values:

1. **fullName:** “Sercan Kavdır”
2. **email:** “sercan@test.com”
3. **password:** “123”
4. **passwordCheck:** “123”

- **Expected Output:** Bad Request Status Code (400)
- **Pass / Fail:** FAIL

4.1.2 Sector Tests

Test Procedure: Should create a sector with valid fields and credentials
Fields:

1. **sectorName:** required, cannot be empty and only string values
2. **header:** required and cannot be empty

Expected Results: Successful sector creation. Otherwise, authorisation error will be displayed

Input Values:

1. **sectorName:** "Sağlık"
2. **header:** "Authorisation Bearer token"

- **Expected Output:** Successful Sector Creation Status Code (201)
- **Pass / Fail:** PASS

Input Values:

1. **sectorName:** "Sağlık"
2. **header:** ""

- **Expected Output:** Forbidden Status Code (403)
- **Pass / Fail:** FAIL

4.1.3 Business Type Tests

Test Procedure: Should create a businessType with valid fields and credentials
Fields:

1. **businessTypeName:** required, cannot be empty and only string values
2. **sectorId:** required, cannot be empty and should be MongoDB ID
3. **header:** required and cannot be empty

Expected Results: Successful business type creation. Otherwise, authorisation error will be displayed

Input Values:

1. **businessTypeName:** "Güzellik Merkezi"

2. **sectorId:** “dummy sector data id”
3. **header:** “Authorisation Bearer token”

- **Expected Output:** Successful Business Type Creation Status Code (200)
- **Pass / Fail:** PASS

Input Values:

1. **businessTypeName:** “Güzellik Merkezi”
2. **sectorId:** “dummy sector data id”
3. **header:** “”

- **Expected Output:** Forbidden Status Code (403)
- **Pass / Fail:** FAIL

4.1.4 Service Tests

Test Procedure: Should create a service with valid fields and credentials
Fields:

1. **serviceName:** required, cannot be empty and only string values
2. **businessTypeId:** required, cannot be empty and should be MongoDB ID
3. **header:** required and cannot be empty

Expected Results: Successful service creation. Otherwise, authorisation error will be displayed

Input Values:

1. **serviceName:** “Saç Kesimi”
2. **businessTypeId:** “dummy businessType data id”
3. **header:** “Authorisation Bearer token”

- **Expected Output:** Successful Service Creation Status Code (200)
- **Pass / Fail:** PASS

Input Values:

1. **serviceName:** “Saç Kesimi”
2. **businessTypeId:** “dummy businessType data id”
3. **header:** “”

- **Expected Output:** Forbidden Status Code (403)
- **Pass / Fail:** FAIL

4.1.5 Business Tests

Test Procedure: Should create a business with valid fields and credentials
Fields:

1. **businessName:** required, cannot be empty and only string values
2. **address:** required and cannot be empty
3. **sectorId:** required, cannot be empty and should be MongoDB ID
4. **businessTypeId:** required, cannot be empty and should be MongoDB ID
5. **businessOwnerId:** required, cannot be empty and should be MongoDB ID
6. **header:** required and cannot be empty

Expected Results: Successful business creation. Otherwise, authorisation error will be displayed

Input Values:

1. **businessName:** “Saç Kesimi”
2. **address:** “İş yeri adresi”
3. **sectorId:** “dummy sector data id”
4. **businessTypeId:** “dummy business type data id”
5. **businessOwnerId:** “dummy user data id”
6. **header:** “Authorisation Bearer token”

- **Expected Output:** Successful Business Creation Status Code (201)
- **Pass / Fail:** PASS

Input Values:

1. **businessName:** “Saç Kesimi”
2. **address:** “İş yeri adresi”
3. **sectorId:** “dummy sector data id”
4. **businessTypeId:** “dummy business type data id”
5. **businessOwnerId:** “dummy user data id”
6. **header:** “”

- **Expected Output:** Forbidden Status Code (403)
- **Pass / Fail:** FAIL

4.1.6 Appointment Tests

Test Procedure: Should get business calendar with valid fields and credentials
Fields:

1. **businessId:** required, cannot be empty and should be MongoDB ID
2. **startDate:** required, cannot be empty and should be valid date(YYYY-MM-DD HH:mm). It gets the current date and displays the appointments in a week.

Expected Results: Get appointments for the specified business successfully . Otherwise, date error will be displayed

Input Values:

1. **businessId:** “dummy business data id”
2. **startDate:** “dummy date”

- **Expected Output:** Successful Get Appointments Status Code (200)
- **Pass / Fail:** PASS

Input Values:

1. **businessId:** “dummy business data id”
2. **startDate:** “”

- **Expected Output:** Bad Request (400)
- **Pass / Fail:** FAIL

Test Procedure: Should request an appointment from a business with valid fields and credentials
Fields:

1. **customerId:** required, cannot be empty and should be MongoDB ID
2. **buinessId:** required, cannot be empty and should be MongoDB ID
3. **employeeId:** required, cannot be empty and should be MongoDB ID
4. **serviceId:** required, cannot be empty and should be MongoDB ID
5. **startDate:** required, cannot be empty and should be date(YYYY-MM-DD HH:mm)
6. **endDate:** required, cannot be empty and should be date(YYYY-MM-DD HH:mm)

Expected Results: Request an appointment from a business successfully . Otherwise, if the selected date is not available, proper error code will be displayed.

Input Values:

1. **customerId:** “dummy user data id”
2. **businessId:** “dummy business data id”
3. **employeeId:** “dummy user data id”
4. **serviceId:** “dummy service data id”
5. **startDate:** “dummy available start date”
6. **endDate:** “dummy available end date”

- **Expected Output:** Successful Request Appointment Status Code (201)
- **Pass / Fail:** PASS

Input Values:

1. **customerId:** “dummy user data id”
2. **businessId:** “dummy business data id”
3. **employeeId:** “dummy user data id”
4. **serviceId:** “dummy service data id”
5. **startDate:** “dummy not available start date”

6. **endDate:** “dummy not available end date”

- **Expected Output:** Conflict (409)
- **Pass / Fail:** FAIL

5 Conclusions

In conclusion, we considered this project as the flagship product of a possible startup company. The current pandemic disease that we experience nowadays, has an incredible effect on this business field, and our motivation increases every other day. Later, with the Collaborative Filtering mechanism, we feel that this project would be useful in real life by helping people to choose a service by recommending them the best fitting services. So far, we focused on the main functionalities of the project but the features that we have mentioned will be implemented when in the future works. We applied for a few entrepreneurship programs and we wait to get their feedback. Our plan for the future is to get investment by improving the application quality and ease of use. Also, our aim is to make our name known in the market by applying to these entrepreneurship programs.

We combined the theoretical knowledge we learned at the university and the practical experience while doing the project. In this way, we had the chance to prepare ourselves for real-life job opportunities. It was the first time that we have used technologies such as Node.js, MongoDB, React, React Native, and so on. The reason for choosing these technologies was to learn and practice the latest and mostly used technologies in the field. Besides, we experienced how we can adapt to new technologies during a short time. Finally, we published our current application version to the server so, we plan to improve our application with the feedback we receive from them by showing our application to the business owners we know in our environment.

References

[1] Y. Shi, M. Larson, and A. Hanjalic, “Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges,” *ACM Comput. Surv.*, vol. 47, pp. 3:1–3:45, May 2014.

[2] N. Ayan, “Yemeksepeti.com 589 milyon dolar değerlemeyle delivery hero tarafından satın alındı.” `https://webrazzi.com/2015/05/05/yemeksepeti-delivery-hero-satin-alma/`.

[3] H. Öğütçü, “1 milyon tl’lik yeni yatırım alan kolay randevu, ankara’ya açıldı.” `https://egirisim.com/2017/04/18/1-milyon-tllik-yeni-yatirim-alan-kolay-randevu-ankaraya-acildi/`.

[4] M. Kara, “Docplanner ile birleşen eniyihekim.com adını doktortakvimi.com olarak değiştirdi.” `https://webrazzi.com/2016/02/25/docplanner-ile-birlesen-eniyihekim-com-adini-doktortakvimi-com-olarak-degistirdi/`.

[5] F. Demirel, “Veteriner kliniklerini dijitale taşıyan veterinervar, internetten randevu imkanı sunuyor.” https://webrazzi.com/2017/11/16/veterinervar/.

[6] G. M. Giaglis, S. Klein, and R. M. O’Keefe, “The role of intermediaries in electronic marketplaces: developing a contingency model,” *Information Systems Journal*, vol. 12, no. 3, pp. 231–246, 2002.

[7] Y. Bakos, “The emerging role of electronic marketplaces on the internet,” *Commun. ACM*, vol. 41, p. 35–42, Aug. 1998.

[8] M. B. Sarkar, B. Butler, and C. Steinfield, “Intermediaries and cybermediaries: Sarkar, butler and steinfield,” *Journal of Computer-Mediated Communication*, vol. 1, no. 3, pp. 0–0, 1995.

[9] A. Dubey and D. Wagle, “Delivering software as a service,” *The McKinsey Quarterly*, vol. May, 01 2007.

[10] E. Constantinides, “Influencing the online consumer’s behavior: The web experience,” *Internet Research*, vol. 14, pp. 111–126, 04 2004.

[11] M. Tomaş, “An exploratory study on the reasons of the takeaway customer using e-intermediary for food ordering: Yemeksepeti.com case study,” *Journal of Internet Applications and Management*, vol. 5, pp. 29 – 41, 2014.

[12] Y.-H. Chen, I.-C. Hsu, and C.-C. Lin, “Website attributes that increase consumer purchase intention: A conjoint analysis,” *Journal of Business Research*, vol. 63, pp. 1007–1014, September 2010.

[13] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” *Commun. ACM*, vol. 35, pp. 61–70, Dec. 1992.

[14] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “Grouplens: An open architecture for collaborative filtering of netnews,” in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW ’94, (New York, NY, USA), pp. 175–186, ACM, 1994.

[15] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the 10th International Conference on World Wide Web*, WWW ’01, (New York, NY, USA), pp. 285–295, ACM, 2001.

[16] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, pp. 734–749, 07 2005.

[17] Z. Cheng, J. Caverlee, and K. Lee, "You are where you tweet: A content-based approach to geo-locating twitter users," in *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, (New York, NY, USA), pp. 759–768, ACM, 2010.

[18] S. Aciar, D. Zhang, S. Simoff, and J. Debenham, "Informed recommender: Basing recommendations on consumer product reviews," *IEEE Intelligent Systems*, vol. 22, pp. 39–47, May 2007.

[19] N. Jakob, S. H. Weber, M. C. Müller, and I. Gurevych, "Beyond the stars: Exploiting free-text user reviews to improve the accuracy of movie recommendations," in *Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion*, TSA '09, (New York, NY, USA), pp. 57–64, ACM, 2009.

[20] A. Levi, O. Mokryn, C. Diot, and N. Taft, "Finding a needle in a haystack of reviews: Cold start context-based hotel recommender system," in *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, (New York, NY, USA), pp. 115–122, ACM, 2012.

[21] Y. Moshfeghi, B. Piwowarski, and J. M. Jose, "Handling data sparsity in collaborative filtering using emotion and semantic based features," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, (New York, NY, USA), pp. 625–634, ACM, 2011.

[22] I. Konstas, V. Stathopoulos, and J. M. Jose, "On social networks and collaborative recommendation," in *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, (New York, NY, USA), pp. 195–202, ACM, 2009.

[23] R. Bell, Y. Koren, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, pp. 30–37, aug 2009.

[24] G. E. Krasner and S. T. Pope, "A cookbook for using the model-view controller user interface paradigm in smalltalk-80," *Journal of Object Oriented Programing*, vol. 1, no. 3, pp. 26–49, 1988.