Le Tuan Hiep - 001074343

Supervisor: **PhD. Doan Trung Tung**

November 2019

Fitness System

Word count: **27.221 words**

A dissertation submitted in partial fulfilment of the University of Greenwich's BSc Computing

ABSTRACT

People are living in 4.0 industry with the development of artificial intelligence, internet of things and big data. Nowadays, there are many systems that are applying machine learning, for example, Google, Amazon, Apple, Facebook and so on.

The main purpose of the online system fitness is to connect coaches and users across the world. This report will demonstrate different technologies to build the system including frontend technologies (Angular, React, etc.), backend technologies (Node, Spring Boot), different software architectures (Monolithic and Microservices) and recommendation system algorithms (Content-based, Collaborative Filtering, etc.). Aside that, development methods will be compared to choose the most suitable one to develop the online fitness system.

The report will include reviewing about similar products and requirement analysis. Following that, the document will demonstrate how the developer design the online fitness system and how the developer apply frontend technologies and backend technologies to solve specific problems.

PREFACE

Health is always the first priority of everyone. Therefore, fitness takes an important role in human's life. By doing exercises, people can increase mood, health, and reduce stress. On the other hand, the fitness industry is blooming very quickly with revenue is 16.857 million dollars in 2019 and would be increased 5.0% resulting in a market volume of 20.499 million dollars by 2023. Aside that, people are living in 4.0 industry with developing of artificial intelligence, big data and internet of things. For this reasons, it is a great time to create an online fitness system. With developing of 4.0 industry, it is important to make the system become more intelligence by applying recommendation system and chat bot. Recommendation system will work based on rating system so that the system can recommend coaches that users may want to hire. When users have any questions, chat bot will support users 24/24.

ACKNOWLEDGEMENTS

The author wish to thank PhD. Doan Trung Tung for helping the author in developing the online fitness system and writing this document.

CONTENTS

TABLE OF FIGURES

LIST OF TABLES

1 INTRODUCTION

The report investigates about the fitness domain and different technologies. For this reason, people will understand the importance of fitness in human life and can know about which technologies will be chosen to create the online fitness system. Following that, different products will be discussed and then readers could know issues in the design and implementation of current products. From that, requirements for the online fitness system will be produced. On the other hand, different kind of diagrams, that will be used in this report, to clarify the relationship between each entity in the system. The last but not least, the development process will be shown so that readers could understand how the system is developed and proper evaluation about the design and implementation will be created in order to make the system become better in the future.

2 LITERATURE REVIEW

2.1 Introduction

This section will discuss about Fitness domain in order to find effects of fitness in human's life. Following that, current state and trend of fitness market will be specified to determine the potential of fitness market. Therefore, readers could understand why the online fitness system need to be built. In contrast, there are some problems and difficulties that the fitness market has to face with. That problems will be researched in order to find the solution for the online fitness system. The last but not least, the online fitness system is a web-based solution and there are many methodologies and technologies could be used to build a web system. For this reason, different technologies will be compared to choose the most suitable technologies for the system. Hence, the system could bring more user's experiences and help people increase health and lifestyle.

2.2 Fitness Domain

1. Definition

Nowadays, people are caring about how to increase health and lifestyle. It means that fitness takes an important role in human's life. There are so many definitions of fitness. In fact, fitness could be understood that movement of muscles that result in increasing energy (Caspersen, et al., 1985).

2. Benefits

In fact, benefits of fitness cannot be denied. There are some benefits of fitness including people can reduce stress by doing exercises (Ipek Ensari, et al., 2016). Moreover, people can increase mood no matter the itensity of the exercises. It means that people can increase their feelings by doing simple exercises such as walking. According to WHO - World Health Organization (Organization, 2018), number of overweight people has nearly tripled since 1975. Following that, more than 1.9 billion people more than 18 years old and 41 million children who under the age of 5 were overweight in 2016 and 13% were obese. For this reason, fitness comes into play, fitness can help people in losing weight (Mi-Na Gim, 2016). Many studies has shown that metabolic rate will be decreased when people dieting. It will result in losing weight will be delayed. In other words, metabolic rate will be increased when people often doing exercises and people can burn more calories and lose weight. Aside that, fitness is good for muscles and bones. In reality, memory is very important. A person with good

memory can learn new things quickly and memory could be increased by doing exercises frequently. According to a research about health benefits of physical activity (Warburton , et al., 2006), fitness can reduce risks of disease. On the other hand, fitness will help people who want to have a health skin. In reality, sleep quality is very important. People, that have good sleep quality, could feel more energy to work, fitness could help people increase sleep quality and lifestyle.

3. Current state and trend of fitness market

According to the benefits of fitness that was mentioned above, the fitness industry is blooming for some of reasons. According to Statista (statista, none date) Revenue of fitness industry is 16,857 million dollars in 2019, the revenue is increase 7.2%. Revenue is expected to show an annual growth rate of 5.0%, resulting in a market volume of US$20,499m by 2023. The market largest segment is wearables with a market volume of US$14,528m in 2019. In global comparison, most revenue is generated in China with US%5,060m in 2019.

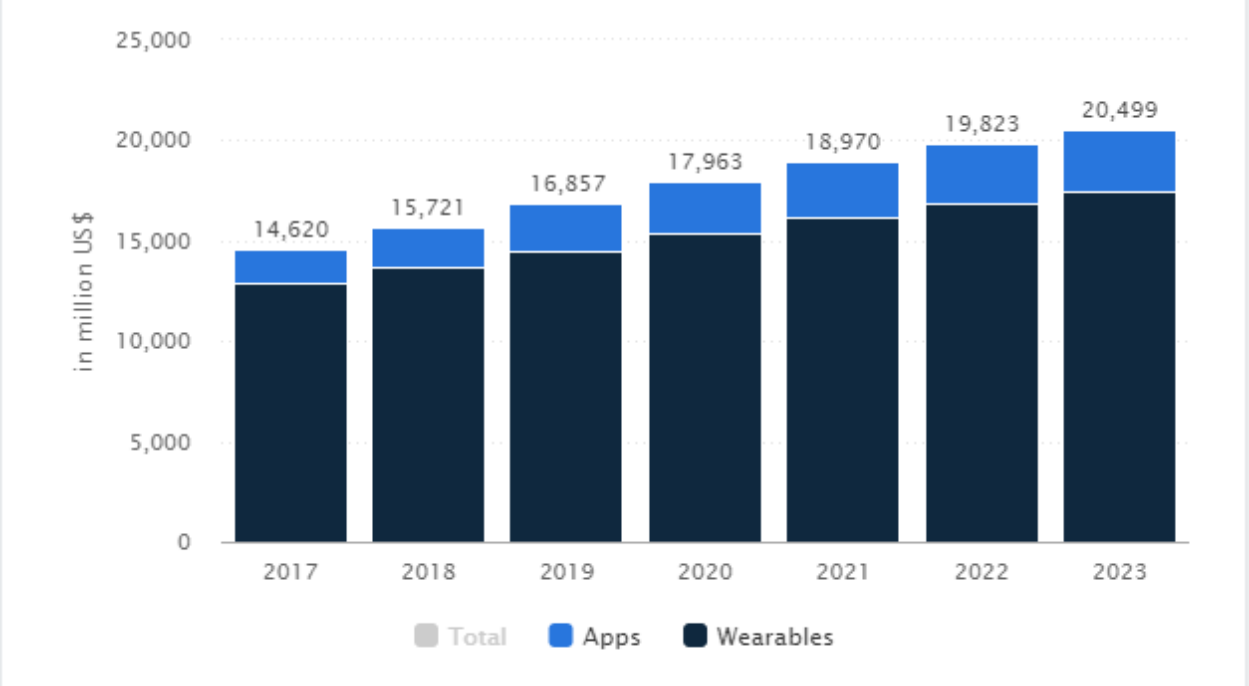


Figure 1. Revenue of fitness market

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

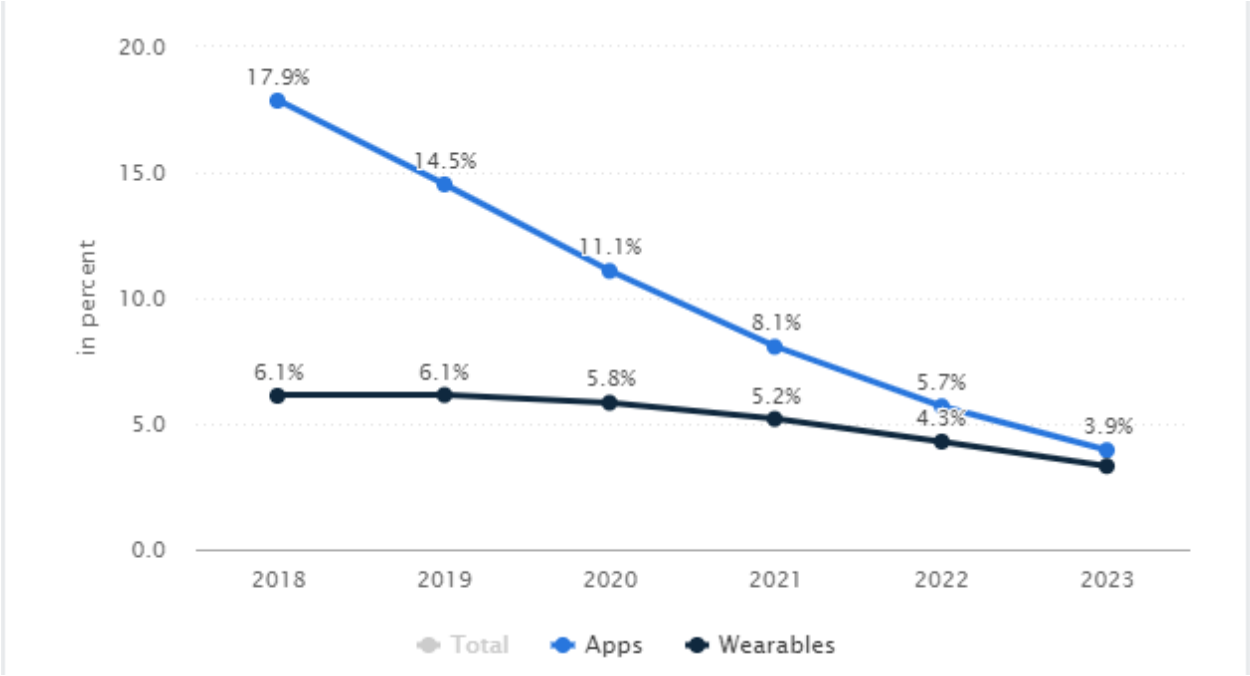


Figure 2. Revenue growth of fitness market

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

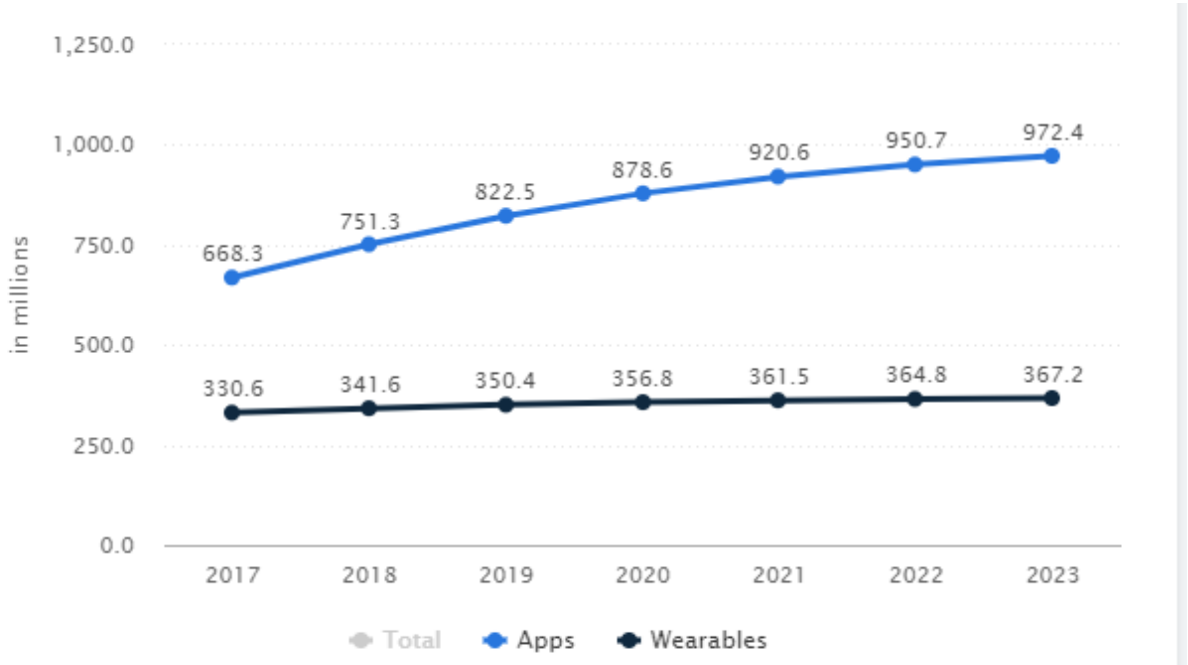


Figure 3. Number of users - fitness market

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

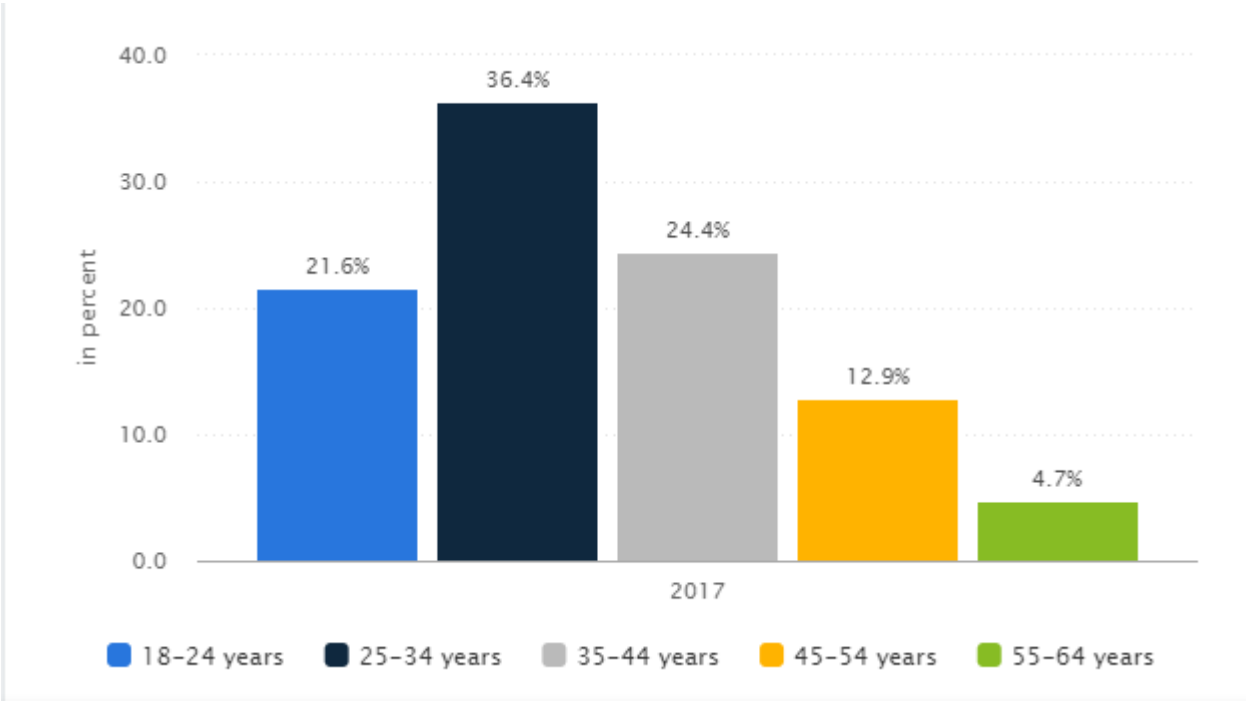


Figure 4. Users by age - fitness market

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

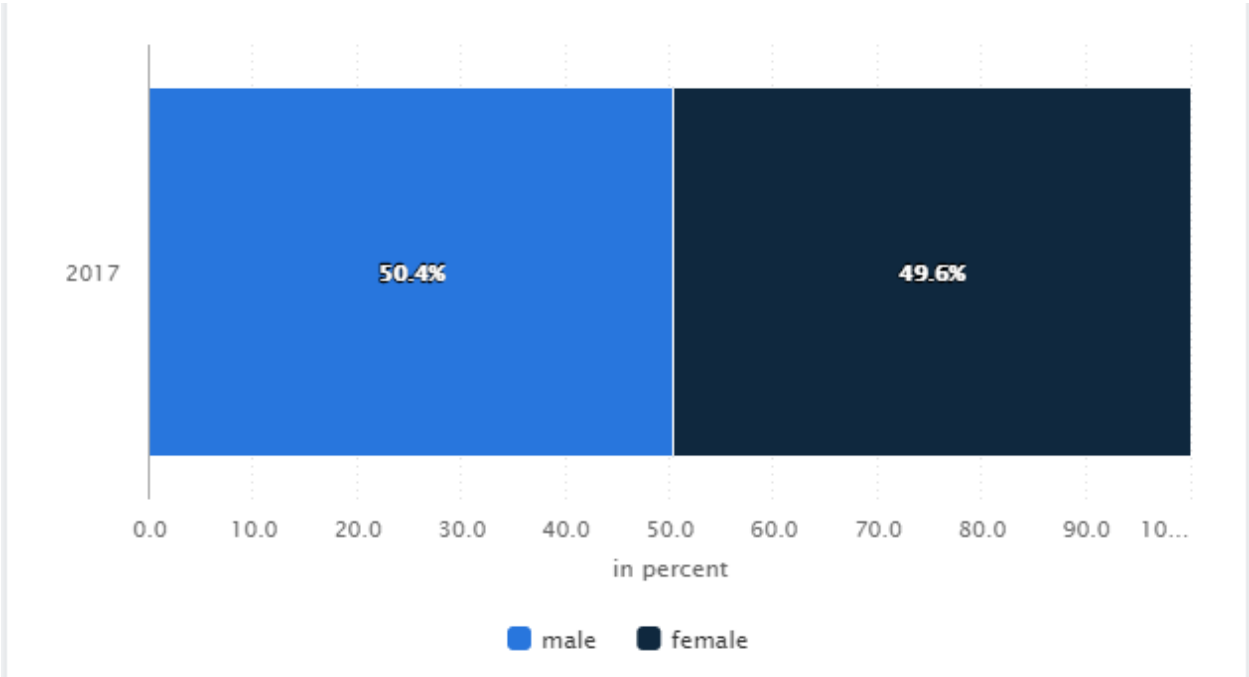


Figure 5. Users by gender - fitness market

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

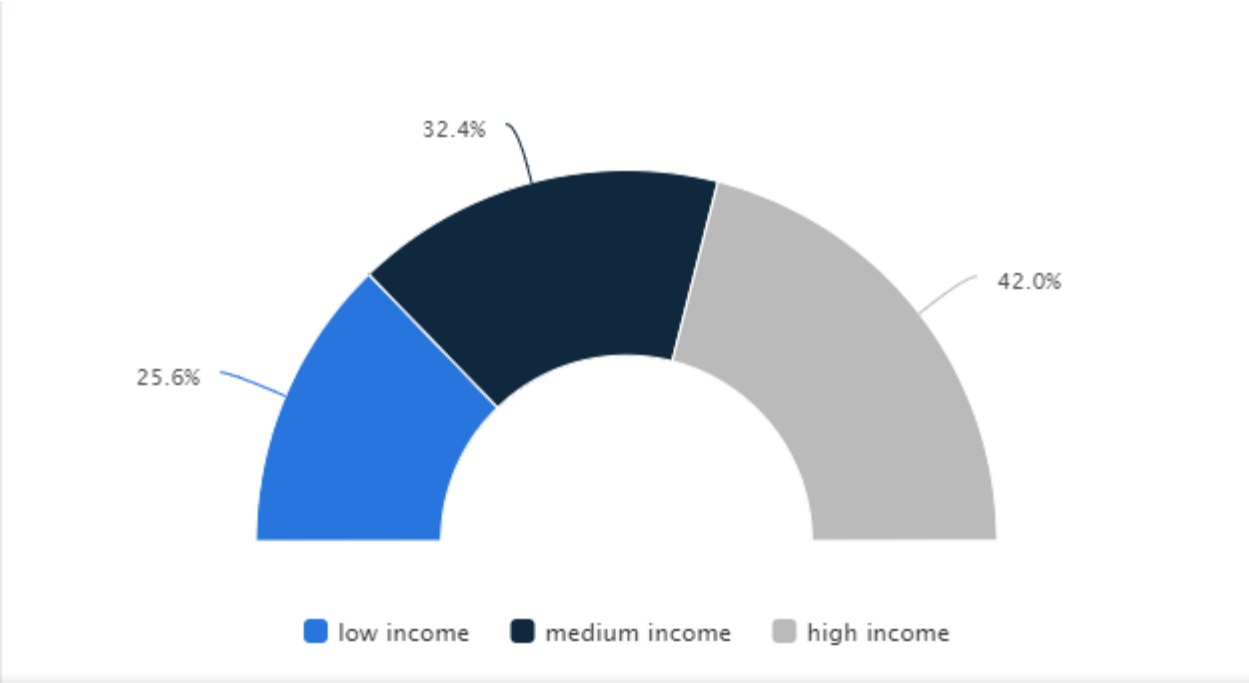


Figure 6. Users by income - fitness market

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

As readers could see at these image above, there are some interesting statistics that could explain why the fitness market is blooming very quickly. In 2019, number of users is 825.9 million and increased 9.4%. On the other hand, percentage of users from 18 - 24 years old is 21.6%, percentage of users from 25 – 34 years old is 36.4%. Following that, percentage of users from 35-44 years old is 24%, percentage of users from 45-54 years old is 12.9%. Percentages of users from 55-64 years old is 4.7%. Aside that, percentage of make users and female users is 50.4% and 49.6, respectively. The statistics also demonstrates that percentage of users with low income is 25.6% and with medium income is 32.4% and high income is 42.0%.

In conclusion, readers could see that revenue and statistic of fitness market are very impressed. As mentioned above, the revenue of fitness market is increased 7.2% in 2019 The main purpose of the online fitness system is to connect coaches and users across the world. For this reason, the author wants to bring more user's experiences to customers so that it is great time to build the online fitness system.

4. Problems and difficulties

The number of revenue and users are really great. However, everything has two sides. There are some problems and difficulties that the fitness industry has to face with. There are so many generic competitors such as fitness clubs and many apps, wearables and so on. It means that people can install many apps and do training at home without going to the gym or buying any equipment. With the

development of social media, organization have to take care about the marketing strategy. If the organization want to get more potential users, the organization should make good marketing strategy plan on different platforms such as Google, Facebook, Twitter and so on. There are many trends and more choice out there. For example, customers can join CrossFit club or Yoga club and so on. For this reason, the developer has to make the system become more special and different and still can suit needs from multiple users. The price of personal trainer is not cheap. Therefore, it is not suitable for users with low income. In fact, the fitness clubs have to pay more for experience trainers. For this reason, people do not feel free to hire personal trainer. The last but not least, many people who using fitness app or joining fitness club do not like what they are doing. Therefore, the developer should take care about that and bring more user's experiences.

5. Conclusion

In conclusion, according to what mentioned above, it is a great time to create fitness system because of the fitness market is blooming very quickly. On the other hand, there are some difficulties that the developer has to face with, but the developer will bring more options and user experiences to overcome these difficulties. The online fitness system will be a web-based solution. Hence, customers can use the system on smartphones, tablet, smart TV, wearables device. Therefore, customers can do exercise anytime and anywhere. Especially, customers do not need to pay if customers do exercises in the system. customers can hire coach with cheaper price, the system can recommend coaches to customers by analysing customer's requirements and applying machine learning algorithms. Although the coaches in the fitness system are online coaches, but the developer will bring convenient ways to help coaches and customers can communicate to each other including chat text and video call.

2.3 Technologies

According to the project proposal, an online fitness system will be created and the system will be a web system. Nowadays, there are so many technologies to create a web system, for example:

- Architectural: Monothelic, Microservice, etc.
- Frontend Technologies: Angular, ReactJs, etc.
- Backend Technologies: Spring boot framework, NodeJs/Express framework, etc.
- Recommendation system algorithms: content-based, collaborative filtering, etc.

Note: That technologies will be compared and discussed to choose the most suitable technologies for the fitness system.

A. Architectural

In modern system, there are some architectural could be used, for example, Monothelic architecture, Microservices architectural and so on. In this section, Monothelic and Microservices will be compared to choose the most suitable one for the fitness system

1. Monolithic

a. Definition

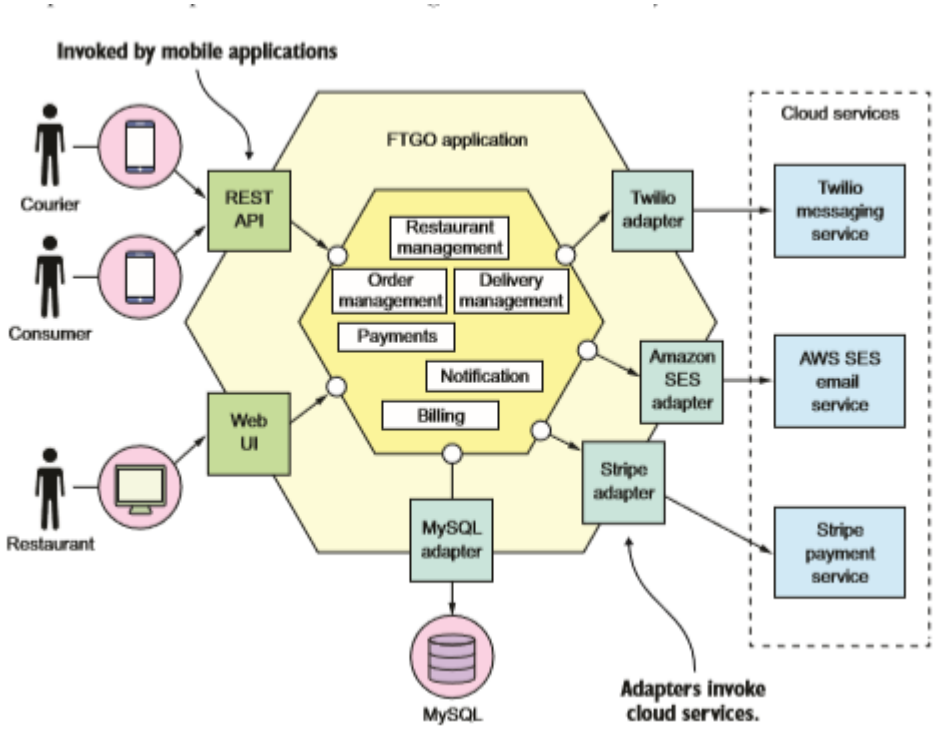


Figure 7. Monolithic Example *(Chris, 2019)*.

The image above demonstrated monolithic software architecture of FTGO application. Although, the system contains different modules including Restaurant management, order management, delivery management, payments, notification, billing, but FTGO application just has a single WAR file. It means that the system using monolithic software architecture. For this reason, according to Chris Richardson (Chris, 2019), monolithic architecture could be understood that structuring a system as single executable or deployable component.

b. Advantages

There are some advantages while applying monolithic software architecture in building the system including: simple to develop, easy to make radical changes to the application, straightforward to test, straightforward to deploy, easy to scale (Chris, 2019). That advantages will be discussed in detailed. Firstly, while building the system, the development team can focus on choosing one IDEs and tools so that the development team just need to focus on some technologies to build the entire system. In fact, some IDEs need to be pay if the development team want to use. The development team just need to use one or two IDEs to build the entire system. Therefore, the cost of buying tools would be reduced. In fact, the code of the system would be changed overtime. By using monolithic software architecture, the development can change the code easier. For example, if database schema need to be changed, database designers just change schema of database and redeploy the whole system. Following that, testing takes an important role in the development process. Risks could be reduced in production, if the system is tested carefully. Monolithic software architecture allows customers to test the system as single executable. Therefore, the developers just need to write test cases for the system. By the way of illustrations, REST API and user interfaces could be tested. In contrast, if the developers want to test REST API in microservices software architecture, the developers have to understand relationships between services. Aside that, because of the system just has an executable file. It means that the development team just ned to deploy the system by deploying that single to the server. In reality, scalable is very important when the system has multiple requests at the same time, the system that using monolithic software architecture could be easy to scale with helping of load balancing mechanism.

c. Disadvantages

Beside the advantages of monolithic software architecture, there are some disadvantages that will be discussed in detailed in this section. When the system become larger. It is not easy for a new developer can fully understand (Chris, 2019). Even experience developers will need a lot of time to understand the whole system because the system contains many complex features. Therefore, the development team have to train new members and it may take a lot of time. It will result in fixing bugs or implementing new features may become difficult and new task may not be finished before the deadline. As mentioned above, the development just need to use one or two IDEs to build the entire system. However, when the system become larger, it will slow down the IDE (Chris, 2019). Hence, the developers will need a computer with better hardware or need more time to build, edit, run and test the code. In reality, monolithic system will work as single executable file so that many modules will work together in a single system and it will result in confliction about resource requirements between

modules. For example, chat module is stored in a large database and will need a server with a lot of memory. Otherwise, image processing module need a lots of CPU. Hence, the developers have to configure the server to fulfil the requirements of each module. The last but not least, monolithic software architecture makes the development team feel difficult to apply or change new technologies or frameworks. It would be risky to write the entire application with new frameworks or technologies.

d. Usage and popularity

Nowadays, most of successful applications are still using monolithic software architecture in the systems because monolithic is easy to develop, test and deploy. Hence, if the application is too large, monolithic software architecture may be a good option to structure the system.

2. Microservices Architecture

a. Definition

Microservice is an architectural style that structures an application as a collection of small autonomous services around a business domain (Chris, 2019).

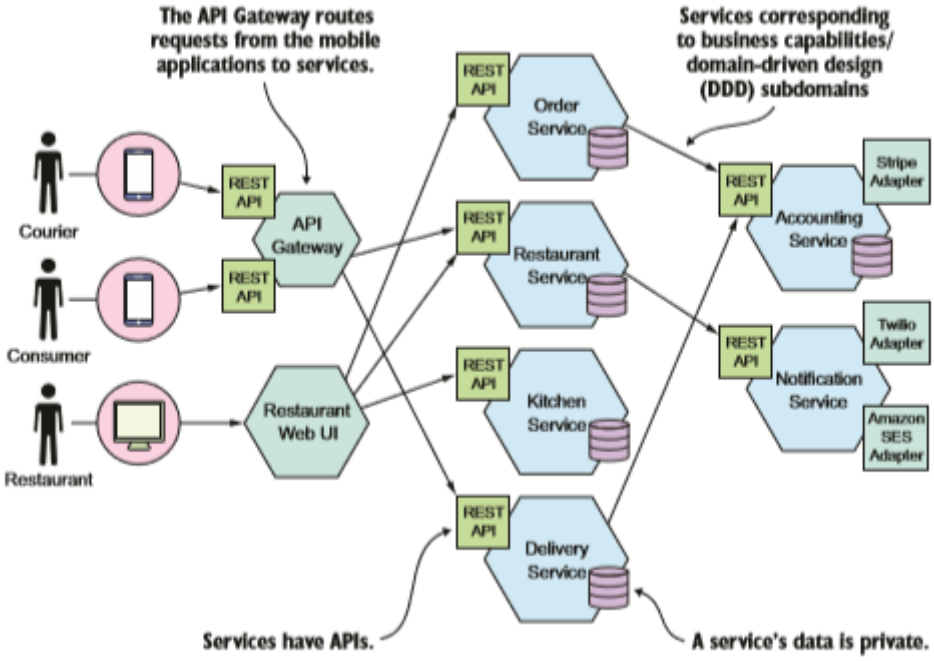


Figure 8. Microservices Architecture *(Chris, 2019)*.

Unlike what demonstrate at figure 7, the system will be broken into smaller parts. As readers could see at the image above, FTGO application will be divided into many small services including order

service, restaurant service, kitchen service, delivery service and so on. Each service is corresponding to each module that was described above.

b. Advantages

There are some advantages of microservices architecture. Complexity of the system will be reduced by creating multiple services for specific object, for example, user's service, product's service and so on (Chris, 2019). For this reason, when a new developer joins the development process. The new developer does not need to understand the whole system, the new member just need to understand the service that need to be built so that the development team does not have to train new member and time consuming and cost will be reduced. On the other hand, by using microservice architecture, the development can feel free to apply new technologies and frameworks such as the development team can use Spring Boot framework to create user's service and ExpressJS framework to create product's service and so on. Services are small and easy to maintain and deploy (Chris, 2019). Each service in microservices architecture could be understood that a subsystem. Therefore, the development team can deploy each service independently. It will result in each service could be independently scalable. The last but not least, many kinds of application can connect to the server via REST such as mobile applications, web applications and so on.

c. Disadvantages

There are some disadvantages of microservices that should be considered before choosing microsevices as architecture of the system. Structuring the system into smaller services is challenging (Chris, 2019). In fact, some services will be too small or some service may be quite large. In this case, the development team will need an experience solution architect that has ability to understand the whole system and design the system appropriately. In addition, the development will need to create many meetings with involving of many stakeholders in order to specify which services should be built to suit the requirements. In microservices architecture, some services are too small and it will result in maintaining that services will be difficult (Chris, 2019). Aside that, data consistency between services is not easy to achieve. In this situation, there are so many different communication methods that could solve the problem, for example, Saga pattern, Event Sourcing pattern and so on. Hence, the development team has to face with choosing the right communication method for the system. Unlike monolithic application, testing in microservice architecture is not easy because if the developers want to test a service, the developers may have to run another services. For example, if user want to test order's service and order's service may require data from user's service so that the developers need to run user's service in order to test order's service. The last but not least, deploying microservices is not

an easy task, the development team need to use some technologies to deploy microservices system such as Docker, Kubenetes and the microservices system is suitable for the team with many people.

d. Usage and Popularity

When the system scales up, the system will be hard to maintain. For this reason, microservices architecture should be applied. Microservices architecture could be used, when the development team does not want to depend on specific technologies. It means that the development team can use different technologies to create different services There are some famous enterprises that are using Microservices architecture such as Netflix, Amazon and so on (Richardson, 2019).

3. Choose the most suitable architecture for Fitness system

In conclusion, a fitness system need to be created. In fact, the developer will work alone and there are many services in the system, but not too much. For this reason, monolithic software architecture will be used for Fitness system. On the other hand, the developer can still use different technologies to create different modules and in this case, recommendation system need to be integrated to the system and recommendation system could be written by using Python programming language with Flask framework and other services can be created by using another technology such as Spring Boot framework and so on. The last but not least, monolithic software architecture is easy to test.

B. Frontend

As what discussed above, monolithic will be chosen as primary software architecture of the system. In fact, the online fitness system is a distributed system. It means that most of services of the system will be provided as REST APIs. Therefore, the developer needs to use frontend's technologies to consume data from REST APIs. There are some solutions for this case, the developer can use AJAX, but in fact, AJAX is not good for SEO (Search Engine Optimization) because of spiders of search engine does not execute of the Ajax calls. On the other hand, single page application takes an important role in modern frontend technology. Single page application is an app that works inside browser and does not require page reloading during use. Single page application could be SEO friendly with helping of server side rendering mechanism. There are some famous examples of single page application such as Google, Facebook, GitHub and so on.

There are some frameworks and libraries that have ability to create single page application, for example, Angular, ReactJs and so on. In this section, Angular and ReactJs will be compared and choose the most suitable technology to create frontend of the system.

1. Angular

a. Definition

According to Angular documentation official, Angular is known as a frontend framework backed by Google. Angular could be used to develop application on mobile, web and desktop. Angular supports dependency injection, end to end tooling and many other features that could be applied to develop the application easier.

b. Advantages

There are some advantages of Angular. In order to develop an Angular application, the developer will need to use Typescript programming language that supports OOPs concepts. In fact, many developers will feel familiar with OOPs concepts. Because of Angular backed by Google and Google is one of the biggest enterprises in the world. It means that Angular may be safe choice to build the frontend of any systems. In addition, documentation of Angular is very detailed. Therefore, the developer does not need to find information from other sources or asking other developers. Following that, dependency inject is a popular pattern that allow a class require dependencies from external resources. In fact, Angular comes with dependency injection framework that will resolve dependencies problem and make users feel develop the application easier. Aside that, Angular supports CLI (Command Line Interface). Hence, the developers can create components for the application by typing on the terminal. For example, if the developers want to create a class in Angular application, the developers just need to type ***"ng g class name of the class"*** on the terminal. In addition, Angular support language service, it means that the developers can write code with supporting of auto completion inside of component external HTML template files, readers could see the image below for more information.

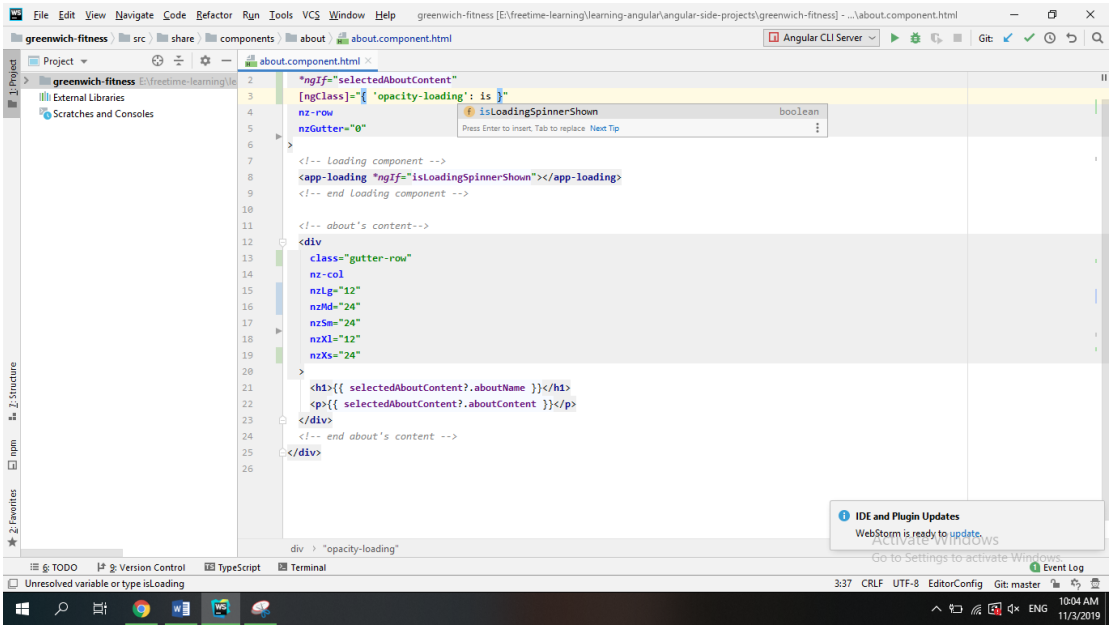


Figure 9. Auto completion in HTML files

The last but not least, Angular supports SEO (Search Engine Optimization) with server side rendering mechanism and Angular also provides two ways data bindings. For this reason, when the developers perform some events such as key up event, the properties of components will be changed and vice versa.

```
<!--month picker -->
<nz-month-picker
  (ngModelChange)="onMonthChanged($event)"
  [(ngModel)]="selectedDateToViewCoachPaymentHistories"
  nzPlaceHolder="Select Month"
></nz-month-picker>
<!-- end month picker -->
```

Figure 10. Two ways binding in Angular

c. Disadvantages

There are some disadvantages of Angular that will be discussed in this section. Angular contains many different concepts (injectables, components, pipes, modules etc.) that will make Angular is harder to learn in comparison with React and Vue.js, which have only concept of components. Therefore, a novice programmer will feel Angular is not easy to learn. Following that, because of Angular

applications are built by using Typescript programming language so that while compiling Typescript code will be converted to Javascript code and it will result in slower performance.

d. Usage and Popularity

Angular has more than 50.000 stars on GitHub (GitHub, 2019). There are some famous companies that are using angular in their application such as Google, Apple, Telegram, Microsoft and so on. Angular will be used if the development team want to create single page application. Especially, if the developers want to create enterprise applications.

2. React

a. Definition

According to Facebook, ReactJs is a JavaScript library for building UI components for web applications. React is maintained by Facebook and many leading tech brands are using React in their development environment.

b. Advantages

There are some advantages of ReactJs. ReactJs is easy to learn, if a novice programmer wants to learn about ReactJs, that programmer just needs to learn about concept of components. Following that, ReactJs brings JSX concept that is similar to HTML syntax to create template.

```
const name = 'Josh Perez';
const element = <h1>Hello, {name}</h1>;

ReactDOM.render(
  element,
  document.getElementById('root')
);
```

Figure 11. JSX syntax

Aside that, ReactJs provides detailed documentation with huge community. Hence, when the developer has any problems, the developer can find solution easily. In order to create ReactJs application, the developers just spend more time writing modern Javascript. The developers do not need to care too much about framework code. By applying React Virtual DOM implementation and various rendering optimizations, performance of React is very fast. Aside that, ReactJs supports SEO by using server side

rendering. Managing state is very important in building ReactJs application. Redux will come into play. Redux is the most popular framework for managing state in React.

c. Disadvantages

There are some disadvantages of ReactJs. ReactJs just supports “View” part. Angular supports MVVM (Model View - View Model) model. For this reason, ReactJs is not really compatible with enterprise applications. On the other hand, ReactJs is moving away from class-based components. Therefore, the developers, that familiar with Object Oriented Programming (OOP), the developers may not feel comfortable while developing ReactJs applications. The last but not least, there are many ways to write CSS in React applications, it will result in confusing between traditional stylesheets (CSS Modules) and CSS in JS (Emotion and Styled Components).

d. Usage and Popularity

ReactJs has more than 139.000 stars on GitHub (GitHub, 2019). There are some famous companies that are using ReactJs in the system such as Facebook, Instagram, Netflix, New York Times, Yahoo and so on ReactJs will be used if the development team wants to create single page application.

3. Choose the most suitable frontend technology for Fitness system

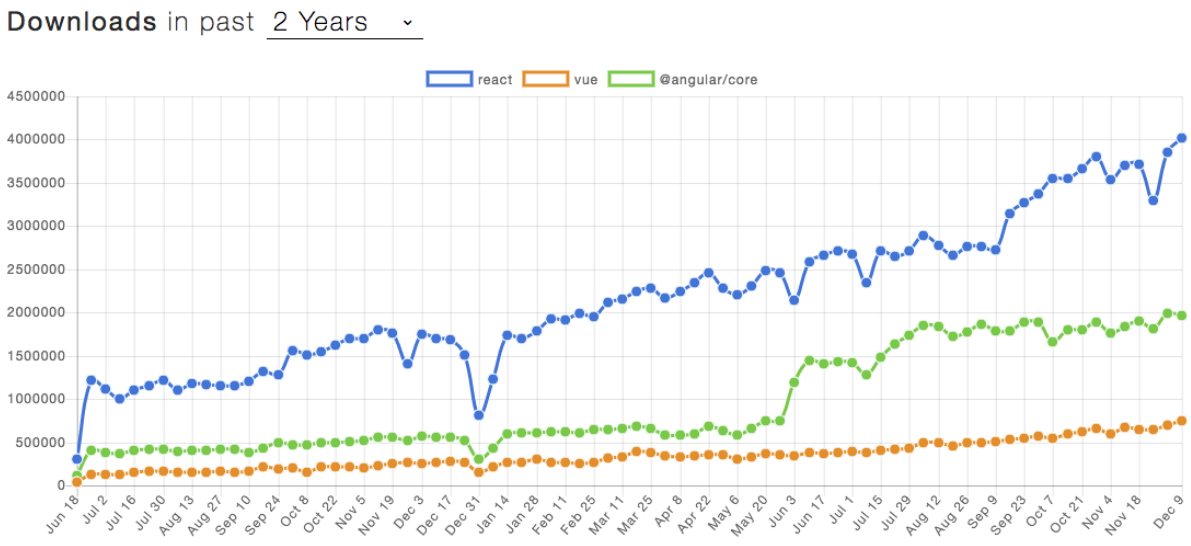


Figure 12. Comparison between Frontend frameworks and libraries

In conclusion, both Angular and ReactJS are used to create single page application. According to the image above, number of downloads of ReactJS is greater than number of download than Angular. However, Angular framework is quickly growing. Hence, the developer will choose Angular for some

of reasons. In order to create Angular application, the developer can use Typescript that has OOP concepts and the developer has some experiences in OOP (Object Oriented Programming). On the other hand, dependency injection of the features related to the components which modules and modularity in general. The last but not least, structure and architecture specifically created for great project scalability and enterprise application.

C. Recommendation System

Recommender systems are software tools and techniques providing suggestions for items to be of use to a user (Francescco Ricci, 2011). Netflix has ability to suggest movies to users or users bought a product on Amazon. The system would suggest “Frequently bought together” or the system based on users purchase history. Recommendation system is a wide part of machine learning and the age of recommendation system is smaller than the age of classification.

Readers need to understand why recommendation is important. According to Pareto principle, which was also known as 20/80 principle, determined that roughly eighty percent of the effects come from 20 percent of the causes, for example, the majority of words, which people are using day by day are just a small part of dictionary’s vocabulary. Therefore, when a person is doing business, he or she needs to recognize that the top sold products are just small part of all products that was contained in warehouse. For this reason, if the owner wants to increase revenue, he or she needs to show top sold products in the place where customers easy to see the products. Meanwhile, other products would be store in warehouse, but this approach has a limitation, the popular products were displayed, but the products may not be what a specific customer wants. In fact, a store could have the product which a specific custom needs, but the store did not show the product because the products was not popular. On the other hand, the online store could solve this issue because the online store could show everything the store wants. Following that, the online store could apply recommendation system, so the online store could recommend or predict the products for a specific customer.

In the fitness system, recommendation system need to integrated. Therefore, the system can suggest coaches to users. There are many algorithms that can help the developer create recommendation system. In this section, content-based and collaborative filtering will be compared to choose the most suitable algorithm for the system.

1. Content-based Recommendation System

a. Definition

Figure 13. Content based recommendation system

Content-based system evaluates the characteristics of items that are recommended, for example, users view a lot of adventure and super hero movies, so the system needs to recommend movies that have the same adventure or super hero characteristics. Readers can understand that content-based recommender engine based on characteristics of the items.

b. Advantages

In order to create content-based recommendation system, the developer just needs to create utility matrix of items because of content-based recommendation system based on characteristics of the items. In fact, matrix calculation will take a lot of time and matrix calculation of content-based algorithm will take less time than other recommendation algorithms. For this reason, if the server of the development team is not strong enough. Content-based algorithm may be a good choice for the situation. According to Google Developers (Developers, 2019), the model of content-based algorithm does not need to care about any data of other users, since the recommendations are specific to this user. Hence, it makes it easier to scale to a large number of users. Aside that, the model can capture the specific interests of a user and can recommend niche items that very few other users are interested in.

c. Disadvantages

There are some disadvantages of content-based recommender engine such as the system could not use information of other users for statistic. That information is so useful because user's behaviours could be arranged into some simple groups. Aside that, creating feature items is not easy in real life and asking users to add tags to items is also difficult because some of users are not feel free to do it. On the other hand, according to Google Developers (Developers, 2019), since the feature representation of the items are hand-engineered to some extent, this technique requires a lot of domain knowledge. For

example, if content-based recommendation system is integrated in ecommerce system, the researchers need to understand characteristics of the items in ecommerce system.

2. Collaborative Filtering Recommendation System

a. Definition

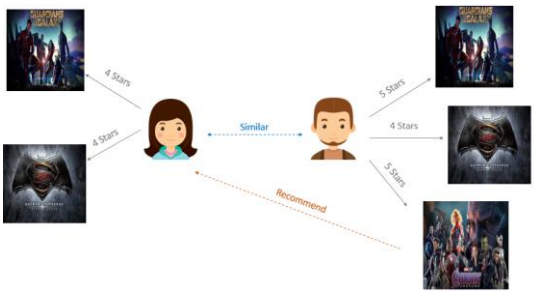


Figure 14. Collaborative filtering recommendation system

Collaborative filtering suggest items based on the similarity between users of items. For example, there are two users A and B, user A and B viewed “guardians of galaxy” movie and “Batman” movie. In addition, user A and B gave “guardians of galaxy” 4 stars and 5 stars, respectively. Following that, user A and B also gave “Batman” 4 stars. However, user B also viewed “Avenger” movie and gave it 5 stars. For this reason, the recommendation system should recommend “Avenger” movie to user A.

b. Advantages

There are some advantages of collaborative filtering recommender engine, for example, collaborative filtering analyses based on user’s behaviours. In real life system, user’s information is so useful for analysing. User’s behaviours could be arranged into simple groups. Users do not need to add tags to items. Following that, the model does not require the domain knowledge. The collaborative filtering is suitable for large scale system, the more users are using the system, the better the recommender engine will become. Aside that, when people are talking about recommendation system, accuracy will be the first priority.

c. Disadvantages

Collaborative filtering recommender engine also contains some disadvantages. Firstly, collaborative filtering would take more time to calculate because collaborative filtering needs more time to calculate matrix, find user similarities and predict the result. In fact, collaborative filtering algorithm works with matrix and in real life system, most matrixes are spare matrix. It means that number of zero values is

greater than number of nonzero values. For this reason, collaborative filtering will need more memory to store user's rating values

3. Choose the most suitable recommendation system for Fitness system

In conclusion, in the fitness social network system, collaborative filtering will be chosen to create recommendation system because the system could analyse and base on user's behaviours. In reality, user's information is so useful to reuse and analyse. Especially, in the fitness system, we want to suggest coaches to users. It means that it is important to reuse other user's information to suggest coaches for a new user. Therefore, the system could become more friendly and increase user's experiences. The last but not least, collaborative filtering system is not difficult to implement.

E. Methodologies

There are several methodologies to create a software system. Waterfall method and Agile method will be disused in this section then the developer will choose the most suitable one for Fitness system.

1. Waterfall

a. Definition

The waterfall model could be understood that a sequential, down-flow model and the waterfall model is often used in building software system (Dubey, et al., 2015). There are several steps in waterfall model including: gathering the document of requirement, design, coding, perform the system, testing, fix the bug and errors and finish the project and so on. There is a special point about waterfall model, the next step in waterfall model will be executed when the previous step is finished.

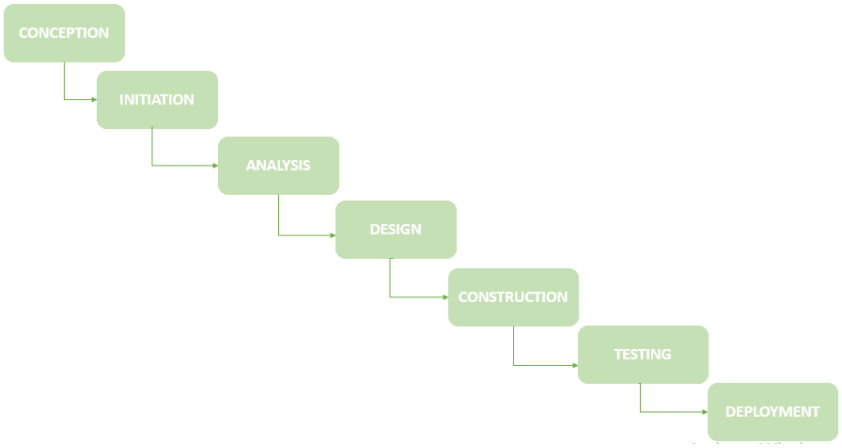


Figure 15. Waterfall model

e. Advantages

There are some advantages of waterfall model. Firstly, each stage in the waterfall model is clear, and the development team just need to focus on each stage in the development process. On the other hand, each phase is finished in specified period of time (Dubey, et al., 2015), the development team will do the next step when the previous step is finished. Following that, the development team can easy to manage their work for each stage and the customer and the development team can make the plan and design without problems. The appropriate documentation for each stage will be provided. For this reason, waterfall model is suitable for the long-term project.

f. Disadvantages

There are some disadvantages of waterfall model. The customer cannot view the next step until the previous step is finished. The development cannot go back to the previous step (Dubey, et al., 2015). In waterfall model, if customer want to change anything, it may be hard and waste more time to implement it. Aside that, gathering document of requirement is not easy because the customer may not know what they want.

2. Agile

a. Definition

Figure 16. Agile method

[Source: https://reqtest.com/wp-content/uploads/2019/02/Agile-methodology-_965_resized.jpeg]

Agile methodology is the development methodology that will be applies to the project team. By using Agile methodology, importance tasks will be delivered first. All the time of the development process will be divided into “sprint”. Each of sprint will have duration and tasks. The deliverables are sorting

by the priority of business value which had been determined by the customer. If a task is not done, that task will be reprioritized and will be developed in the future.

b. Advantages

There are some advantages of Agile methodology. The developers and customers can join the development process. Following that, the customers can change their requirements. The customers can view the progress of the development process. On the other hand, Agile methodology can increase relationship between the development team and the clients. The last but not least, the most importance tasks will be delivered first.

c. Disadvantages

There are some disadvantages of Agile methodology. The development process could be difficult if customers cannot visualize about the final product. Product backlog and prioritizing tasks are not easy in the big system. The method is suitable for the development team that contains many experience developers. A novice programmer could join the team if the team has enough time to train that programmer.

3. Choose the most suitable method for the Fitness system

The fitness system contains many functions and fitness domain, technologies have been researched and choose the most suitable technologies to build the fitness system. Aside that, the developer has some experience in building different kinds of software before. For this reason, Agile method could be more suitable method to build the fitness system, the developer could know about which tasks should be built and deliver first. Because this project is used for the university final project. For this reason, Mr. Doan Trung Tung – supervisor of the author will be product owner for the project and supervisor will give opinion and feedbacks about product backlog and sprint backlogs in order to make the system could be delivered before the deadline with the best performance.

F. Conclusion

In conclusion, this section demonstrated domain of Fitness system. For this reason, readers could understand that fitness takes an important role in human's health and lifestyle. On the other hand, many different technologies have been researched to find the most suitable technologies to build the system, for example, in order to build the system, the author could choose monolithic architecture or

microservices architecture. Following that, the author could use Angular or ReactJs to build the frontend and NodeJs or Spring Boot to build the backend. After researching about these technologies, the author decided to use monolithic software architecture for system architecture, Angular framework for frontend and Spring boot framework for backend, collaborative filtering algorithm for recommendation system. The reasons why the author wants to use these technologies to build the system that was described above.

3 REVIEW OF OTHER PRODUCTS

3.1 Introduction

Nowadays, there are so many applications out there that support people in doing exercises and increase health This section will review of other products to find advantages and disadvantages of each products. From that, the author could create requirements and functionalities for the fitness online system in order to bring benefits to users and increase user's experiences while user using the system.

3.2 Freeletics

Link CHplay: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi

Figure 17. Freeletics Screenshot

[Source: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi]

Freeletics is a leading brand in the online fitness. Freeletics is lauched since 2013. The mission is to help people increase health and lifestyle. Nowadays, 38 million users accros the world using freeletics. The average start of Freeletics on Google Play Store is 4.2 with more than 160.000 ratings. There are some advantages of Freeletics that could be learnt. Firstly, user interfaces of Freeltics is great with dark mode that is good for eyes and easy to use. Freeletics takes care a lot about user's experiences.

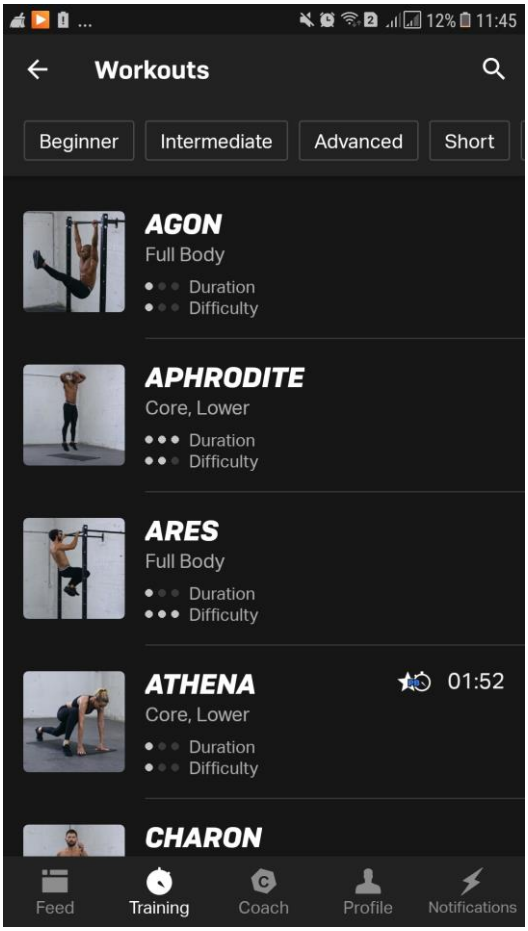


Figure 18. Freeletics Workouts

[Source: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi]

By using freeletics, users can do the exercises without equipment. It means users just need to use bodyweight to do the training. All exercises in Freeletics cover every muscle groups, for example, arms, legs, core and so on. Especially, users do not need to go to the gym or need any equipment. Users can do the exercises anytime and anywhere.

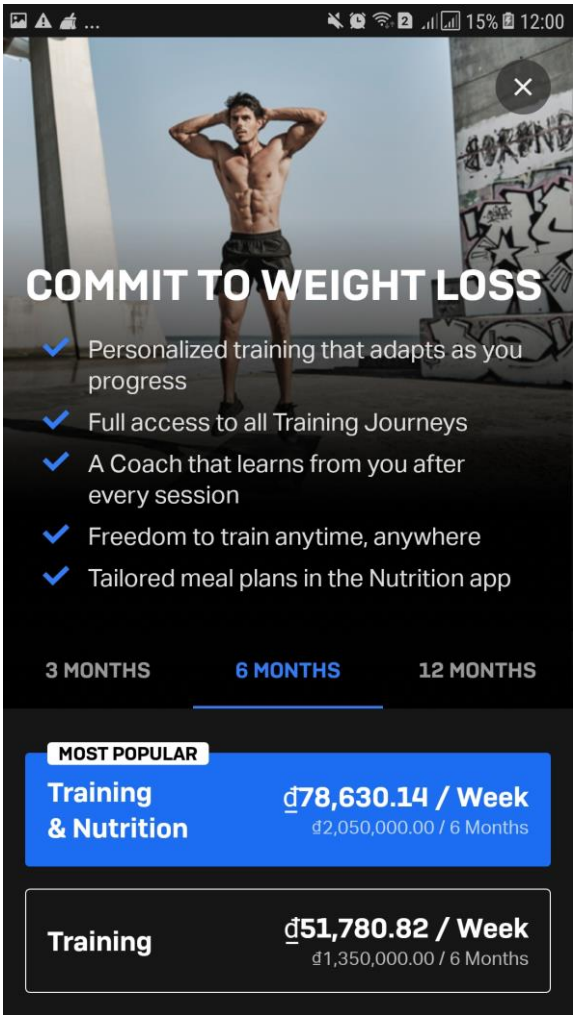


Figure 19. User can hire coach in Freeletics

[Source: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi]

On the other hand, Users can hire online coach. Freeletics will connect coaches and users across the world. The training plan will be made based on user's requirements. Following that, users can keep track the personal goals, schedule and levels. Aside that, users will get a detailed fitness test for a better coach experience and choose user's availability – 2,3,4,5 sessions per week. Give feedback to the coach to adjust the training plan.

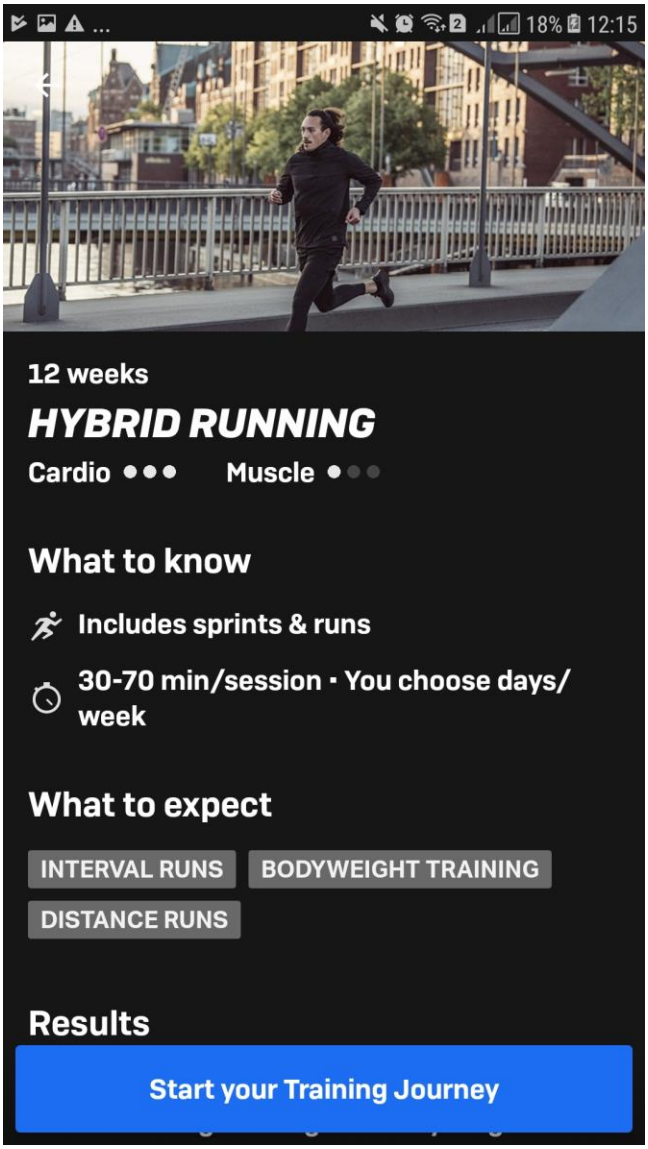


Figure 20. Freeletics support running

[Source: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi]

Freeletics also supports running for weight loss. Users could choose a training plan for your running goals whether that's to run faster, for endurance or running for weight loss. GPS tracks running distance and time. Readers could see that Freeletics supports many ways that could help people increase health and lifestyle.

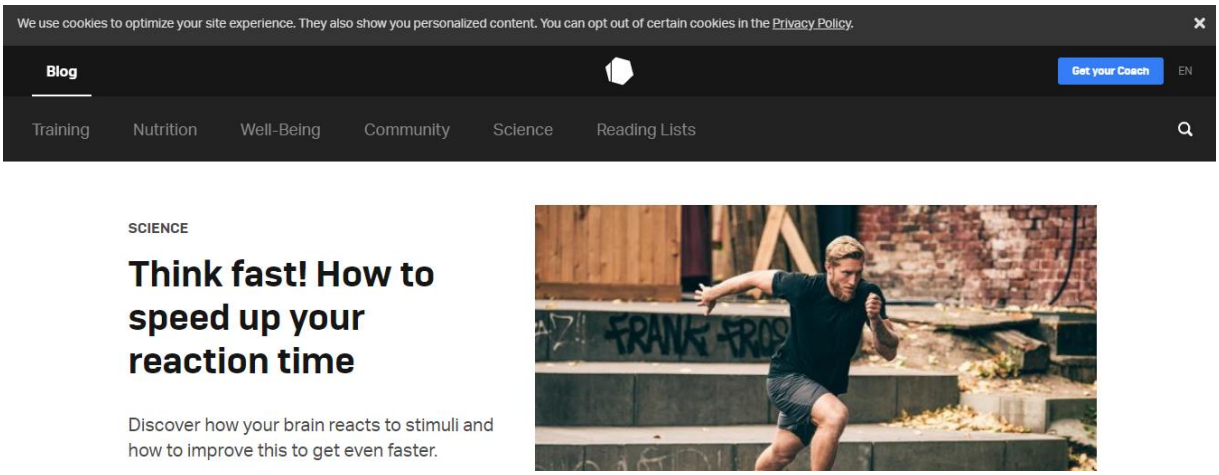


Figure 21. Freeletics blog

[Source: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi]

Freeletics contains many blogs where users could read about training, science, and nutrition. These articles will help people in dieting, losing weight and so on. These articles provide information that users may need to become healthier.

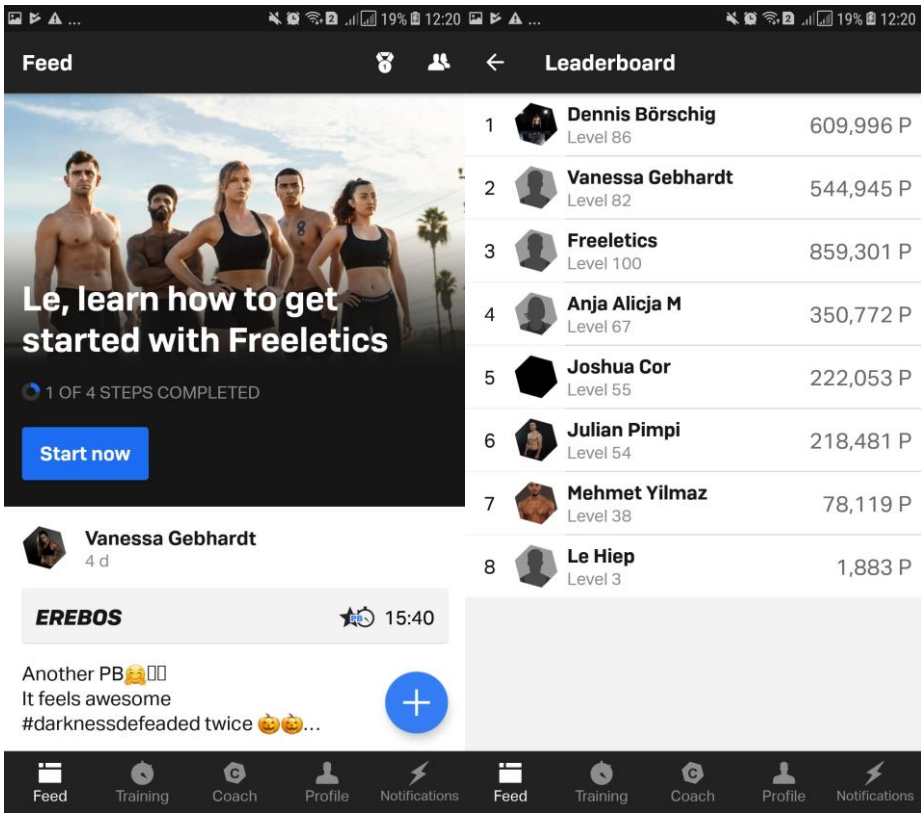


Figure 22. Freeletics provides new feed and leaderboard

[Source: https://play.google.com/store/apps/details?id=com.freeletics.lite&hl=vi]

The last but not least, Freeletics has “New Feed” feature where users can share emotion, feeling after finishing the exercises. In addition, users can give emoji, write comments to pass motivation to each other. Freeletics also contains “Leader Board” feature where people can view their ranking and do better the next time.

In conclusion, freeletics provide many ways that support people in training. Following that, freeletics also connect coaches and fitness which related to the main function of the online fitness system. However, there are some points that Freeletics should consider to bring more user’s experiences including users can listen to music while doing the exercises. Freeletics should provide products feature on their mobile applications. Aside that, users just can give feedbacks to their coaches. Coaches and users cannot communicate directly. Freeletics should provide functionalities that can help users and coaches communicate to each other such as chat function and video call function. The price to hire coach is not really competitive because people can go to the gym with that price. The last but not least, Freeletics should provide chart to help users keep track their body indexes such as weight, height, BMI and so on.

3.3 Home workout – No Equipment

Link CHPlay:

https://play.google.com/store/apps/details?id=homeworkout.homeworkouts.noequipment&hl=vi

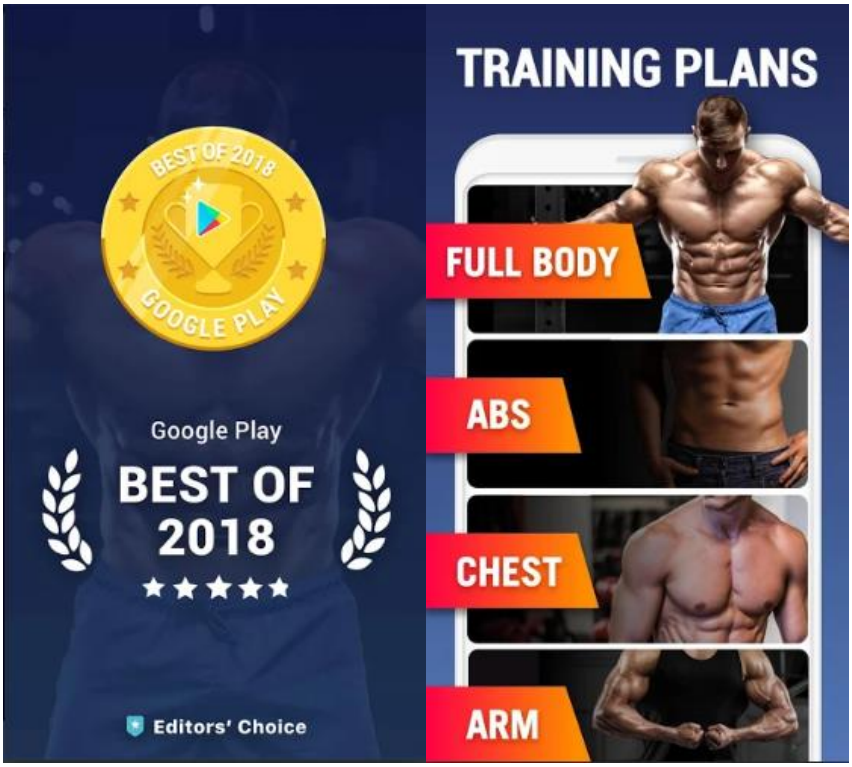


Figure 23. Home Workout - No Equipment Screenshot

[Source:
https://play.google.com/store/apps/details?id=homeworkout.homeworkouts.noequipment&hl=vi]

Home Workout – No Equipment is the best application 2018 on Google Play – editor's choice. The application provides training schedule every day. With few minutes per day, users can build muscles and keep shape at home without go to the gym. On the other hand, people do not need to use any training equipment or coaches. All exercises are suitable with user's bodyweight and help people build all muscles including arms, legs, abs and so on. The application has more than 50.000.000 installed, more than 800.000 ratings and the average rating is 4.8. There are some advantages of Home Workout – No Equipment that could be learnt. Firstly, user does not need to login to use the application. For this reason, users do not need to scare about losing sensitive information. User interfaces of the application is great. Users do not need to be trained before using the application.

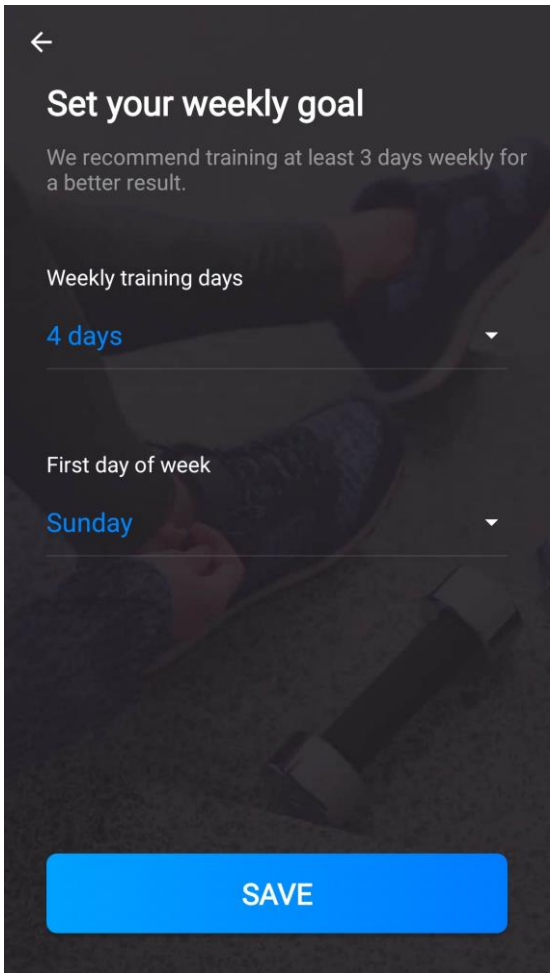


Figure 24. Set Goal - Home Workout - No Equipment

[Source:
https://play.google.com/store/apps/details?id=homeworkout.homeworkouts.noequipment&hl=vi]

The application can help users set goal based on user's requirements. For example, users can set weekly training days. First day of the week that users want to do the training. This is a really great features because the purposes of each user may be different such as burning fat, building muscles and increasing endurance.

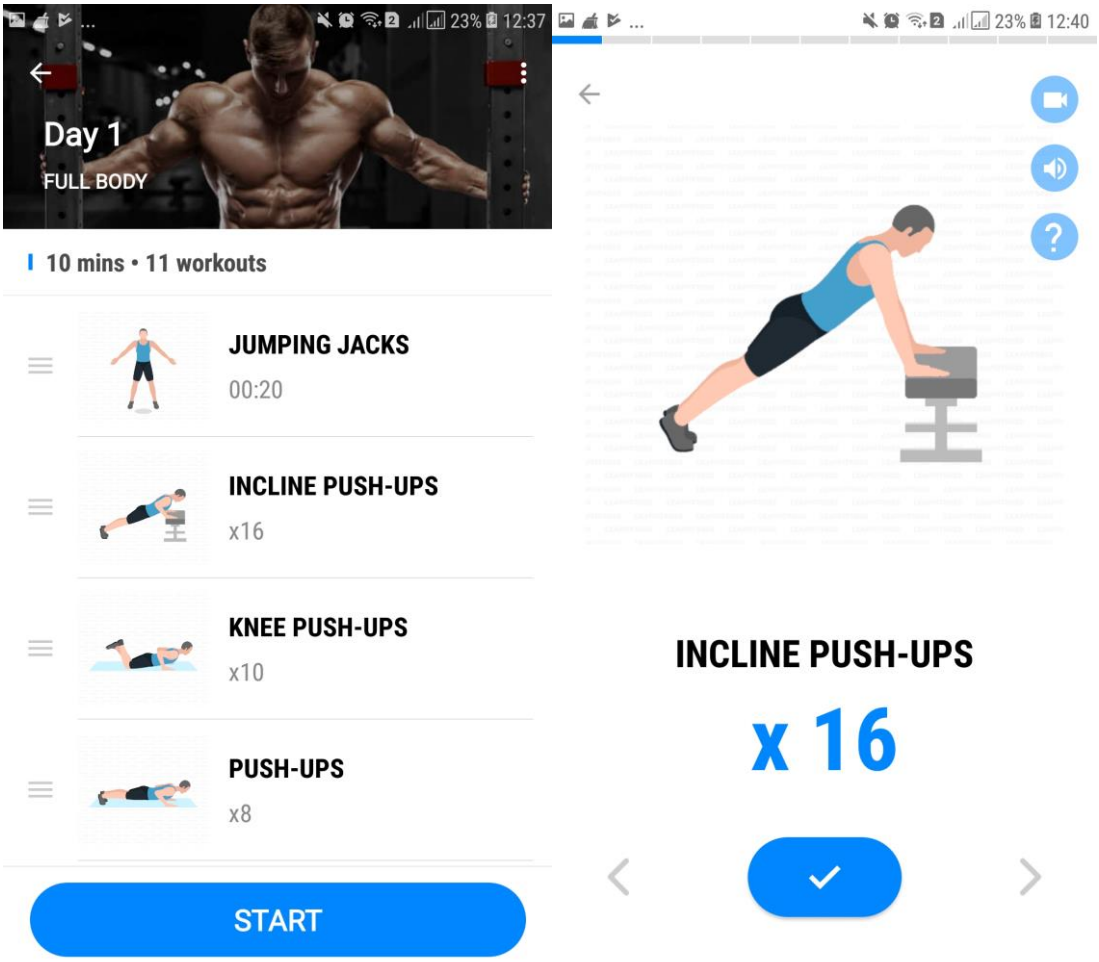


Figure 25. Users can do exercises without equipment

[Source: https://play.google.com/store/apps/details?id=homeworkout.homeworkouts.noequipment&hl=vi]

All exercises do not require any equipment. On the other hand, these exercises help people build all muscles including full body, lower body abs, chest and so on. As readers could see the image above, users could watch tutorial videos for each exercise. Therefore, users will know their performance is correct or not.

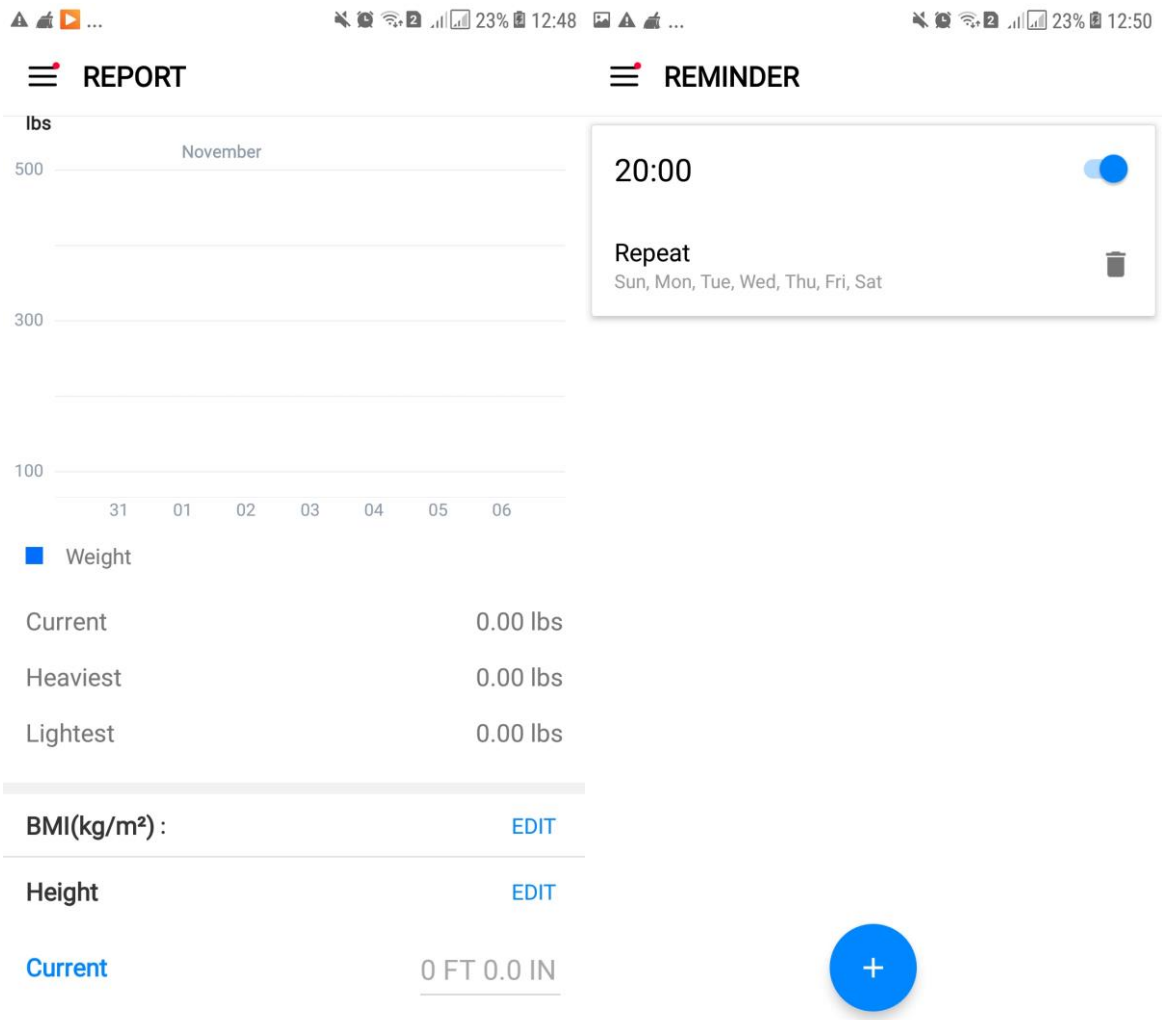


Figure 26. Report about body indexes and reminder

[Source: https://play.google.com/store/apps/details?id=homeworkout.homeworkouts.noequipment&hl=vi]

Unlike Freeletics, the application provides report function where users can review about how many workouts that users have done, how many calories was burned and time consuming. On the other hand, report will provide body indexes including weight, height, BMI and so on. Hence, users can keep track their body indexes in order to make right plans to increase health. In addition, the users could set reminder to remind users when users will need to do the training.

In conclusion, Home Workout – No Equipment provides many features to users, users do not need to log in to use the system. Aside that, all exercises in the application do not require any equipment. The application has great user interfaces and user's experiences. There are some points that the developer team of Home Workout – No Equipment should consider to increase user's experiences. The application should suggest exercises and workouts to users. For this reasons when new users using the application, new users cannot know about which exercises should do first. The application should provide coaches so that users could receive the right training plans and nutrition plans because

requirements of each user may be different. In order to increase user's experiences in doing exercises, the application should let user listen to music while doing the training.

3.4 Fitness and Bodybuilding

Link CHPlay: https://play.google.com/store/apps/details?id=softin.my.fast.fitness&hl=vi

Figure 27. Fitness and Bodybuilding application

[Source: https://play.google.com/store/apps/details?id=softin.my.fast.fitness&hl=vi]

This application with more than 10.000.000 installed, more than 130.000 ratings and the average rating value is 4.8. The application is belonging to VGFIT LLC. By using Fitness and Bodybuilding, users can achieve great results, users can increase any muscles including abs, legs and so on. Furthermore, the application provides workout plans and nutrition plans. There are some advantages that could be learnt. Firstly, users do not need to login or register to use the application. For this reason, users can feel free to use the application without scare about losing sensitive information.

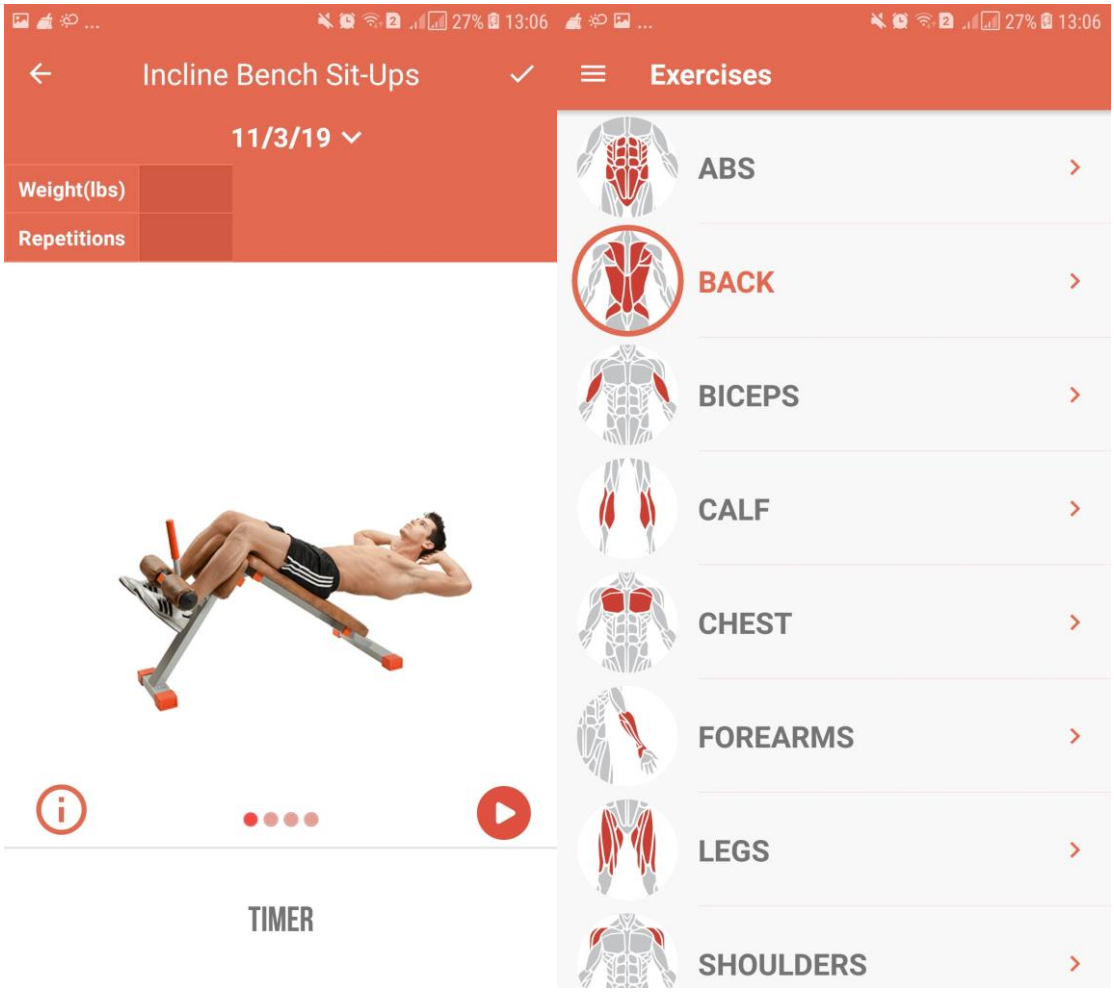


Figure 28. User can do exercises in the application

[Source: https://play.google.com/store/apps/details?id=softin.my.fast.fitness&hl=vi]

Users can do exercises to increase any muscle groups with video support. Therefore, users can know their performance is correct or not. On the other hand, the application includes text instruction with pictures for each exercise. Hence, this will increase user's experiences and users will know about their performance is correct or not.

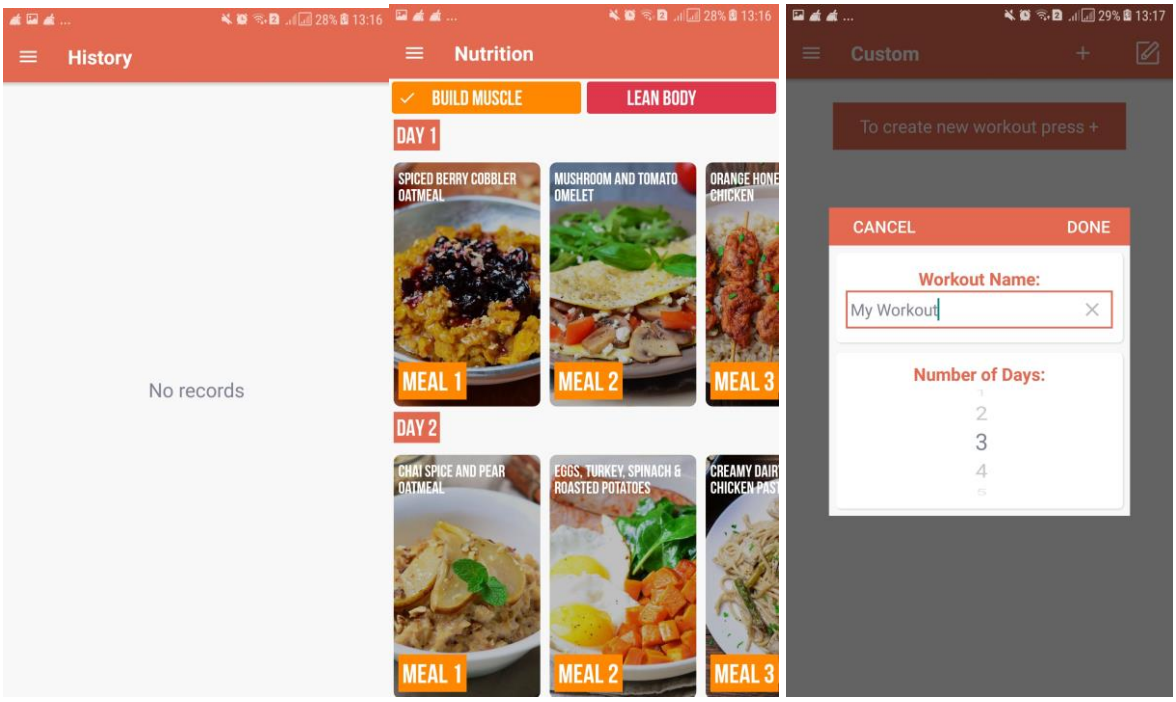


Figure 29. Users can view history, nutrition plans and customize workout plans

[Source: https://play.google.com/store/apps/details?id=softin.my.fast.fitness&hl=vi]

The application also provides "View History" feature where users can view list of workouts or exercises that users have done. The application contains nutrition plan for users. User can view calendar to mark workout days and create customize workout plans with photos. The last but not least, user interfaces of fitness and bodybuilding application is great, users can feel easy to use the application.

In conclusion, Fitness and Bodybuilding application provides features that could suit user's requirements. However, there are some points that the development team of Fitness and Bodybuilding application should consider to increase user's experiences. The application should suggest exercises and workouts to users. For this reason, when a new user is using the application, that user could know which exercises should be done first. Following that, the application should provide coaches to users. Hence, training plans and nutrition plans could be made for each user. The application should let user listen to music while doing exercises and it will result in increasing user's experiences. The last but not least, the application should provide reports to let users keep track body indexes such as weight, height, BMI and so on.

3.5 Conclusion

All products, that have been reviewed in this section, are very popular. That products have more than 10.000.000 installed on Google Play store with more than 100.000 ratings. For this reasons, that products will crease reliable for this report. After reviewing about that products, there are some

advantages should be learnt in order to make requirements for the online fitness system. Firstly, all products have great user interfaces and user experiences, users do not need to be trained before using these applications. Secondly, all products provide all exercises for all muscle groups with video and images so that users can easy follow the exercises and users can know their performance is correct or not. Thirdly, two of three applications (Freeletics – Fitness and Bodybuilding) contains nutrition plans for users. Hence, nutrition plan is an important feature that should be consider while building the online fitness system. In contrast, there are some disadvantages that should be taken care so the online fitness system could avoid these disadvantages in order to make the system better and increase user's experiences, for example, the fitness online system want to connect coaches and users across the world. Freeletics also do that but Freeletics does not provide any methods to make users and coaches communicate to each other. For this reason, the online fitness system should provide communication method, for example, chat function or video call function. Aside that, all products, that was mentioned above, do not have music function so that users cannot listen to music while doing the exercises. If users want to listen to music while doing the exercises, users have to install or use another system/application. As readers could see above, all products are mobile application, users have to install these products on their mobile devices and it will result in more RAM and memory have to be spent for these applications. In order to solve this problem, the online fitness system will be web-based solution, for this reason, users do not have to install anything to use the system. The system will be platform independent. It means that it does not matter if users are using Android operating system or IOS operating system. The last but not least, people are living in 4.0 industry with growing of artificial intelligence, internet of things, big data. All products above do not apply innovation of 4.0 industry but the online fitness system does. Recommendation system will be applied to suggest coaches to users and chat bot will be created to support users 24/24. The following part will describe requirements analysis for the online fitness system.

4 REQUIREMENTS ANALYSIS

4.1 Analyze Requirements for the Fitness System

According to reviewing products that have the same domain and researching about fitness domain and market. Readers could see that the fitness industry is blooming very quickly. There are some functionalities that should be made in order to make a success online fitness system.

- Users need to register to use the system so that users can have an account to interact with the system. On the other hand, the system can analyze user's information to bring more user's experiences such as recommend coaches and so on.
- Users need to login to the system so that users can use the system and the system could know which functions that will be provided to users.
- Users can view coach' list, view selected coach's information, filter coaches so that user can view information of coaches that users feel interests.
- Users can hire coaches so that users can receive training schedule from coach.
- Users can make payment so that users can make payment to buy products and hire coach.
- Users can do single exercises and workouts with videos support and images so that users can do the training (if user do not hire any coaches, user can still do single exercises and workouts in the system).
- Users chat text or make a video call with user's coaches so that user can ask any questions to user's coaches or receive some advice from user's coaches.
- Users can listen to music while users can listen to music while using the system. For this reason, user may not feel boring. For example, user can do the training and listen to music at the same time.
- Users can read blogs so that user can find useful information about some topics. For example, training, nutrition topic, science topic.
- Users can buy products so that users can get products that have ability to support user's training.
- Users can write new feeds and like, dislike and write, comments for selected new feed so that user can interact with other users in the system. This will increase user's experiences in the system.
- Users can view notifications so that user can receive notifications from the system or from other users. For example, when user do payment to hire a coach, the system will send a notification to users.
- User can view galleries so that user can view galleries of the system to feel motivated to the training even harder.

- User can convert points to gift so that user can exchange points to gift. In this case, gifts may be vouchers or something useful for users.

As what described in literature search, Agile methodology was chosen for making plans. For this reason, prioritizing tasks takes an important role in developing the online fitness system, the developer could know which functions should be delivered first. The image below will demonstrate product backlog of the system. The author created product backlog that demonstrate priorities of each requirement, the range of priorities values is from 1 to 5. 1 is the most importance requirements and 5 is less important requirements there are some importance features that should be delivered first including user can view coaches, view coach and make payment. That functions related to the main function because the main purpose of the system is to connect coaches and users across the world. For this reason, that functions will developed for the first sprint. The fitness online system will be divided into 7 sprints, time for each spring is 2 weeks and detail of each sprint will be discussed later in this document.

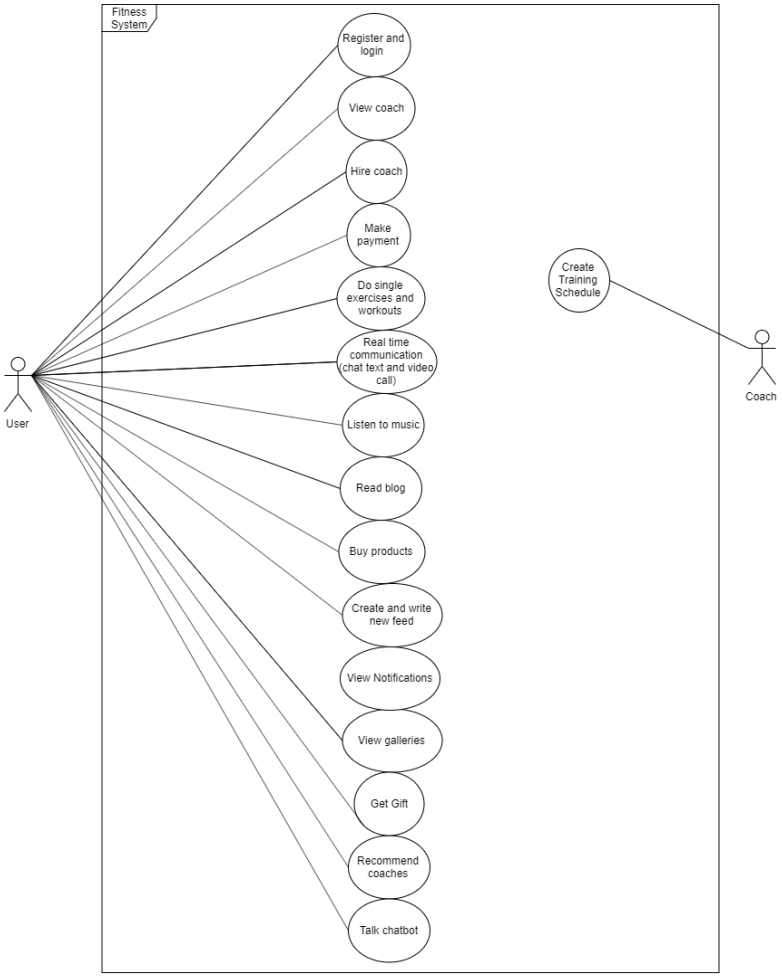


Figure 30. Use case diagram for Fitness system

According to the use case diagram above, in the online fitness system, there are two user's roles including user's role and coach's role. Users can do almost functionalities that was described above.

Following that, coaches can do what normal user can do. Moreover, coach can create training schedule for their memberships and manage their revenue of each month.

4.2 Conclusion

In conclusion, all requirements are based on literature search and reviewing of similar products. All requirements, that was described above, inherits disadvantages of similar products including all exercises in the fitness online system have ability to help people build all muscle groups. On the other hand, the system also provides blogs about many topics such as nutrition, science and so on, these articles would help users in dieting, training.

The online fitness system also provides many solutions to solve problems which similar products have to face with. By using the system, users can hire coaches. It means that the system connects coaches and users across the world and there are many ways to help users and coaches communicate to each other, for example, chat text or make a video call. In addition, there are some functionalities that the online fitness system has and other products do not. While doing the exercises, users can listen to music or download galleries in order to set as wallpaper. For this reason, user's experiences will be increase and users will feel motivated to work even harder.

The last but not least, if users want to use similar products, users have to install the applications to their mobile devices and it will result in lack of RAM and memory. In contrast, the online fitness system will be a web-based solution. It means that users do not need to install the system. Aside that, the system is platform-independent. Therefore, it does not matter if users are using Android operating system or IOS operating system.

5 DESIGN OF FITNESS SYSTEM

5.1 Architecture Diagram

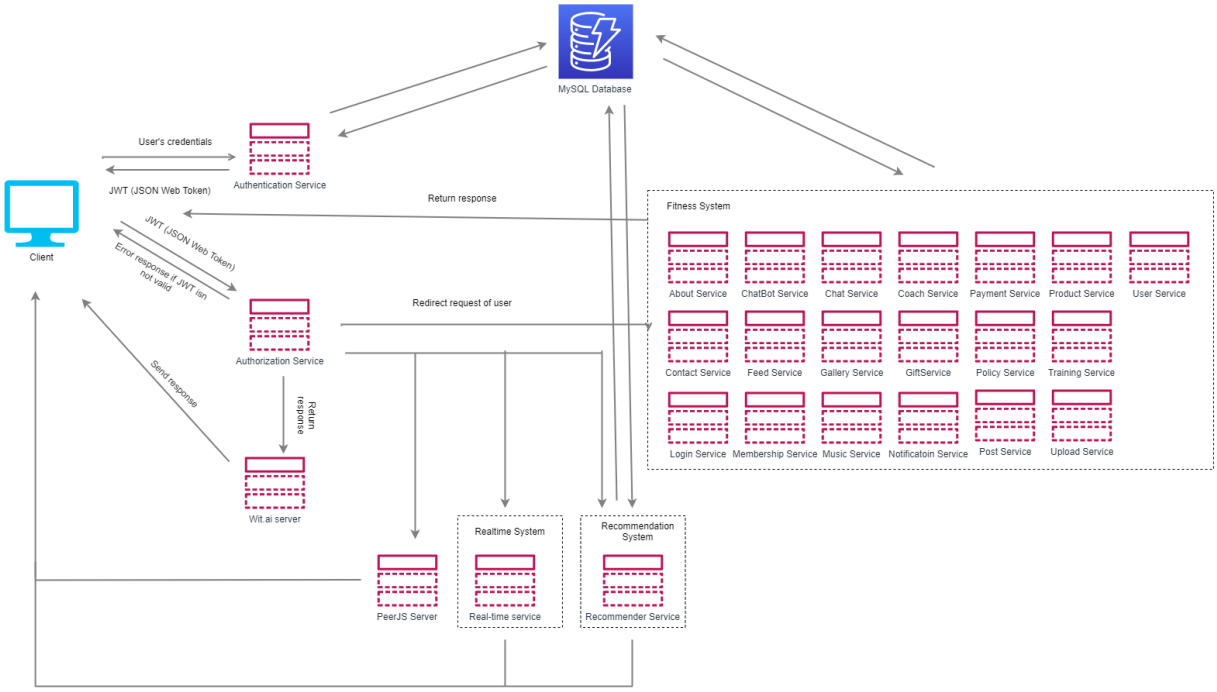


Figure 31. Architecture Diagram

The online fitness system is a web-based solution. According to the diagram above, when user logged in the system, the authenticated service will check user's credentials in the database, if user's credentials are correct, the authentication service will generate a token (JSON Web Token) and return to user and when user send another requests, the token will be added to header. The server will use authorization service to get token of user from header and then verify the token if the user is allowed to access the backend service, or not.

Recommendation system will take responsibilities to suggest coaches to users. Collaborative filtering algorithm will be applied to build recommendation system. The recommendation system will be based on ratings of each user to suggest coaches for selected user. Recommendation system will interact with MySQL database in order to analyze data and recommend result to users. Recommendation system will be built by using Python programming language and Flask framework.

Fitness system will contain almost backend services of the online fitness system including handling transactions, getting data and so on. Following that, the developer will get something important. Fitness system will be built by using Spring Boot framework.

According what mentioned above, the author wants to provide methods to help users and coaches communicate to each other such as users and coaches and chat text or make a video call. On the other hand, socket service will take responsibilities to handle real time tasks. Socket service will be built by

using NodeJS and Socket.IO. Because of socket service handle real time tasks. For this reason, Socket service does not need to interact with the database. Storing chat's messages will be handled by using Fitness system.

The last but not least, the developer wants to create chat bot to support users 24/24 and increase user's experiences in the system. Chat bot will be created by using Wit.ai of Facebook. When users interact with the chatbot, Wit.ai server will handle the requests of users.

In conclusion, the architecture of the online fitness system can fulfill the requirements. On the other hand, appropriate technologies are applied for each service of the system. It will result in better performance with great support from community.

5.2 Entity Relationship Diagram

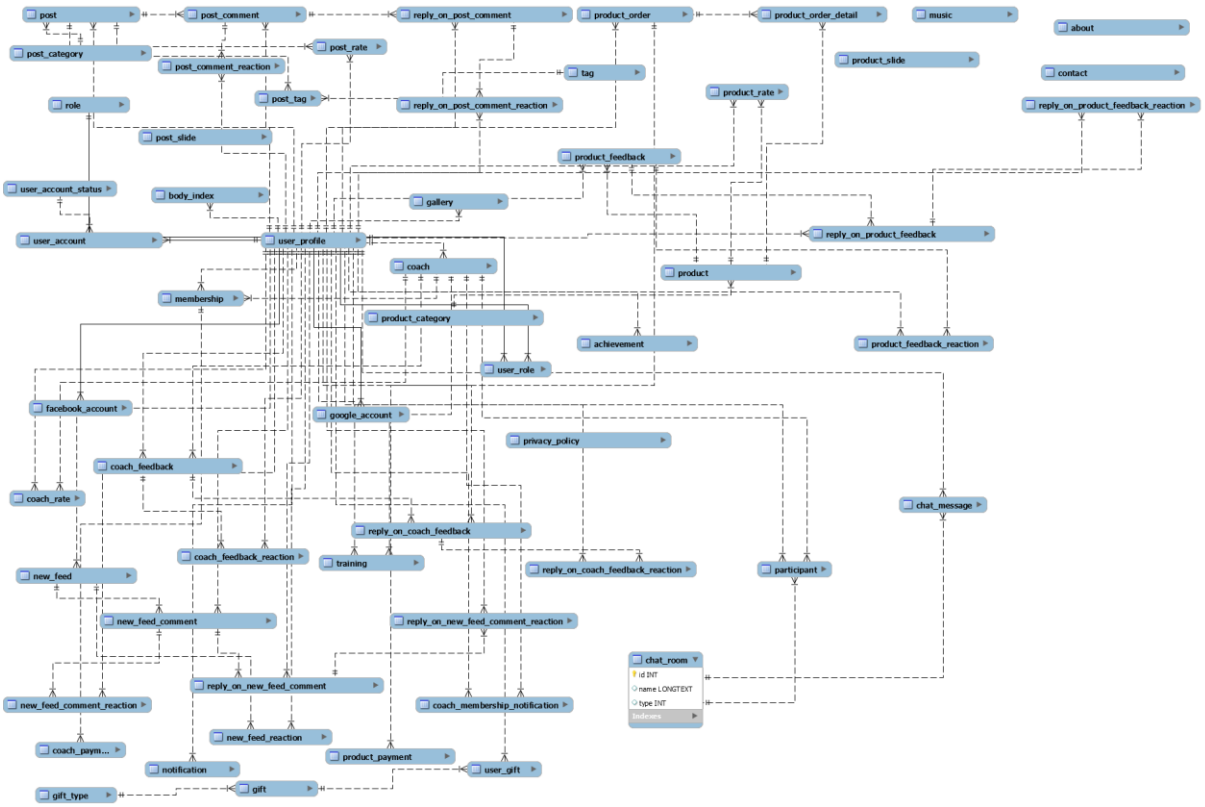


Figure 32. ERD of the fitness system

In order to create database for the online Fitness system, the developer has to create ERD (Entity Relationship Diagram). The image above determined, the purpose of each table and relationship between tables will be discussed below.

On the other hand, many tables were created in order to build the online fitness system. Firstly, user's information takes an important role in every system. Therefore, the developer created some tables to manage user's information.

“user_profile”: store user’s information (user’s avatar, name, etc.).
“user_account”: store user’s credentials (user’s name, password, password reminder token, etc.).
“google_account”: store user’s google’s account if user logged in by using google’s account.
“facebook_account”: store user’s facebook’s account if user logged in by using facebook’s account.
“body_index”: store user’s body index (weight, height, BMI, etc).
“user_account_status”: store user’s account’s status, for example, EMAIL_CONFIRMED – user confirmed email. EMAIL_NOT_CONFIRMED – user did not confirm email and so on.
“role”: store role of user in the system (ADMIN_ROLE, USER_ROLE, COACH_ROLE) and so on.
“user_role”: because of a user can have many roles and a role can be assign for many users. For this reason, “user_account” table and “role” table has many-to-many relationship. In fact, many-to-many relationship should be divided into two one-two-many relationships. Therefore, “user_role” table is created to store roles of each user.

Following that, the author wants to make user feel motivation to do the training even harder. Hence, the developer has created **“gallery”** table, this table will be store galleries of the online fitness system, users can view galleries and download any gallery that users may like and can download it and set it as desktop wallpaper.

Aside that, after reviewing other products and according to requirement analysis, the developer wants to increase user’s experiences in the system. For this reason, the author created listening to music feature. It means that when user using the system, users can listen to music, for example, users can listen to music while users are doing the training and so on. Therefore, **“music”** table is created store all music in the systems.

In addition, users can read blogs in the system, blogs may include different topics such as science, nutrition, training and so on. In order to create blog system in the online fitness system, some tables were created that will be described below.
“post_category”: store post’s category including (science, training, nutrition and so on).
“post”: post will be used to store post in the system.
“post_comment”: because in the system, the author wants user can write comments for any post. For this reason, “post_comment” will be used to stored comment of each post.
“post_comment_reaction’: users can like or dislike any comment that users want. Hence, “post_commet_reaction” is used to store user’s reaction of each post’s comment.
“post_rate”: users can rate any posts that users want, for example, users can rate 5 stars or four stars and so on. “post_rate” is created to store user’s rate for each post.
“tag”: store post’s tags.

“post_tag”: because of a post can have many tags and a tag can be assigned to many posts. It means that “post” table and “tag” table has many-to-many relationship. relationship. In fact, many-to-many relationship should be divided into two one-two-many relationships. Therefore, “post_table” table is created to tags of each post.

“reply_on_post_comment”: store replies on each post’s comment.

“reply_on_post_comment_reaction”: users can like and dislike any reply on each post’s comment. Hence, “reply_on_post_comment_reaction” is used to store user’s reaction of replies on each post’s comment.

Otherwise, users can buy products in the system, in order to integrate ecommerce system to the online fitness system, some tables were created. These tables will be discussed below.

“product_category”: store product’s categories such as products for men, women and so on.

“product”: store product’s information.

“product_feedback”: the author wants users can write feedbacks for any product. Therefore, “product_feedback” will be user’s feedback of any products.

“product_feedback_reaction”: users can like or dislike any feedback that users want. Hence, “product_feedback_reaction” is used to store user’s reaction of each product’s feedback.

“product_rate”: users can rate any products that users want, for example, users can rate 5 stars or four stars and so on. “product_rate” is created to store user’s rate for each post.

“product_slide”: store slides of the ecommerce system. These slides will be used to create slideshow.

“reply_on_product_feedback”: store replies on each product’s feedback.

“reply_on_product_feedback_reaction”: users can like and dislike any reply on each product’s feedback. Hence, “reply_on_product_feedback_reaction” is used to store user’s reaction of replies on each post’s comment.

“product_order”: store product’s order.

“product_order_detail”: store detail of each product’s order.

The main purpose of the online fitness system is to connect coaches and users across the world. Many tables were created to achieve the goal of the system.

“coach”: store coach’s information.

“coach_rate”: users can rate any coach that users want, for example, users can rate 5 stars or four stars and so on. “coach_rate” is created to store user’s rate of each coach.

“coach_feedback”: the author wants users can write feedbacks for any coach. Therefore, “coach_feedback” will be used to store user’s feedback of any coach.

“coach_feedback_reaction”: users can like or dislike any feedback that users want. Hence, “coach_feedback_reaction” is used to store user’s reaction of each coach’s feedback.

“reply_on_coach_feedback”: store replies on each coach’s feedback.

“reply_on_coach_feedback_reaction”: users can like and dislike any reply on each coach’s feedback. Hence, “reply_on_coach_feedback_reaction” is used to store user’s reaction of replies on each coach’s feedback.
“membership”: store relationship between users and coaches. It means that if a user hired a coach, coach’s id and user’s id will be store in **“**membership**”** table.
“training”: store training schedule between users and coaches.

The developer wants to increase user’s experiences. Hence, the author created “new feed” feature in order to connect all users in the online fitness system. “new feed” feature is the place where users can upload what they are feeling after they finished the exercises and other users can like and write comment. Readers could understand that “new feed” feature of the online fitness system will be similar to “new feed” feature of Facebook or Twitter. Many tables were created to achieve “new feed” feature.
“new_feed”: store user’s status after users finished the exercises.
“new_feed_reaction”: In the online fitness system, users can like or dislike any status of other users. Hence, “new_feed_reaction” is used to store user’s reaction of each new feed.
“new_feed_comment”: store user’s comments of each new feed.
“new_feed_comment_reaction”: users can like or dislike any comments of each new feed. Therefore, “new_feed_comment_reaction” is used to store user’s reaction of comments of each new feed.
“reply_on_new_feed_comment”: store replies on each comment of selected new feed.
“**reply_on_new_feed_comment_reaction**”: store user’s reaction of replies on each comment of selected new feed.

There is some common information that need to be stored in the system, for example, introduction about the system, contact’s information, privacy policy. Hence, the author created some tables that will be used to store this information.
“about”: store introduction about the online fitness system.
“contact”: store contact’s information of the system
“privacy policy”: every system needs to contain information about privacy policy to make sure that the system will not sell user’s information or use user’s information for bad purposes.

In the online fitness system, users can buy fitness products and hire coaches. For this reason, users will need a way to make payment. The author integrated PayPal in the system. Aside that, the developer created some tables to store information of payment transactions.
“product_payment”: store information of payment transactions while user bought products.
“coach_payment”: store information of payment transactions while user hired coaches.

In fact, when users hired coaches, the system will give users some points and users can convert points to some gifts, for example, vouchers and so on. Hence, some tables were created to store gifts for each user.

"gift_type": store type of gifts for example, voucher, cash, etc.

"gift": store gift's information. For example, name of voucher, value of voucher and so on.

"user_gift": because of a user can have many gifts and a gift can be assigned to many users. It means that "gift" table and "user_profile" table has many-to-many relationship. relationship. In fact, many-to-many relationship should be divided into two one-two-many relationships. Therefore, "user_gift" table is created to store gifts of each user.

Sometimes, the online fitness system need to send notification to users in the system. Therefore, the author created some tables to store notifications of the system.

"notification": store common notifications in the system.

"coach_membership_notification": store notifications between coaches and users in the system.

The last but not least, the system provides some methods to make users and coaches can communicate to each other, for example, users and coaches can chat text and make a video call. In order to achieve these functions, the developer created some tables.

"chat_room": store chat room in the system

"participant": store participants for each chat room.

"chat_message": store chat messages.

In conclusion, all requirements were achieved, according to the ERD, the database was created and contains 58 tables, all tables were designed based on normalization principles and requirements of the system. All tables are followed 3NF. On the other hand, the database was created for scalable purpose. It means that if the author wants to add new features, the database can suit the requirements.

5.3 Write Frames of High Level Requirements

This section will demonstrate different wireframes of the system. Following that, the author will base on the wireframes to create the

5.3.1 Login and Register Wireframes

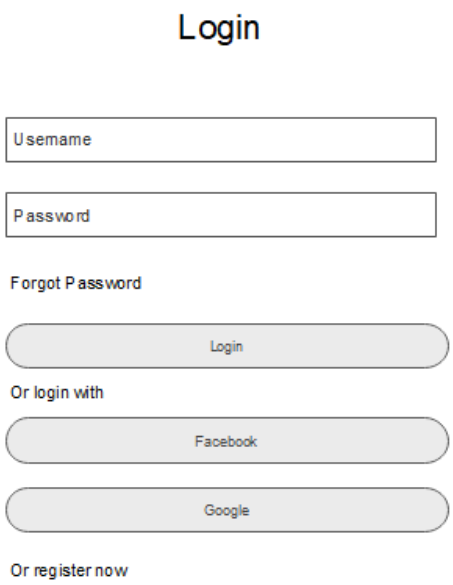


Figure 33. Login wireframe

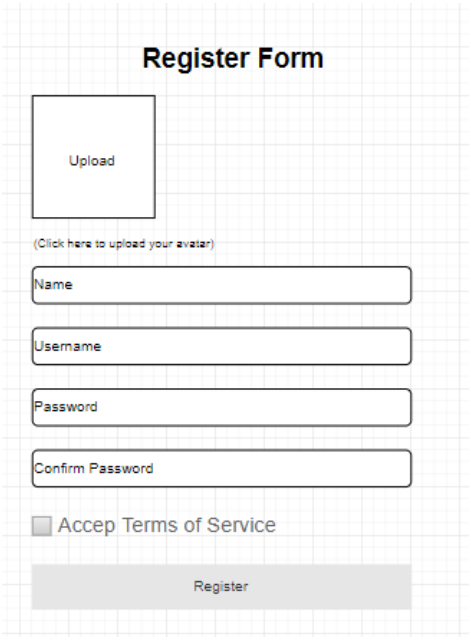


Figure 34. Register form wireframe

According to requirement analysis, users will need to register a new account and login in order to use the system.

5.3.2 View Coach and Hire Coach Wireframes

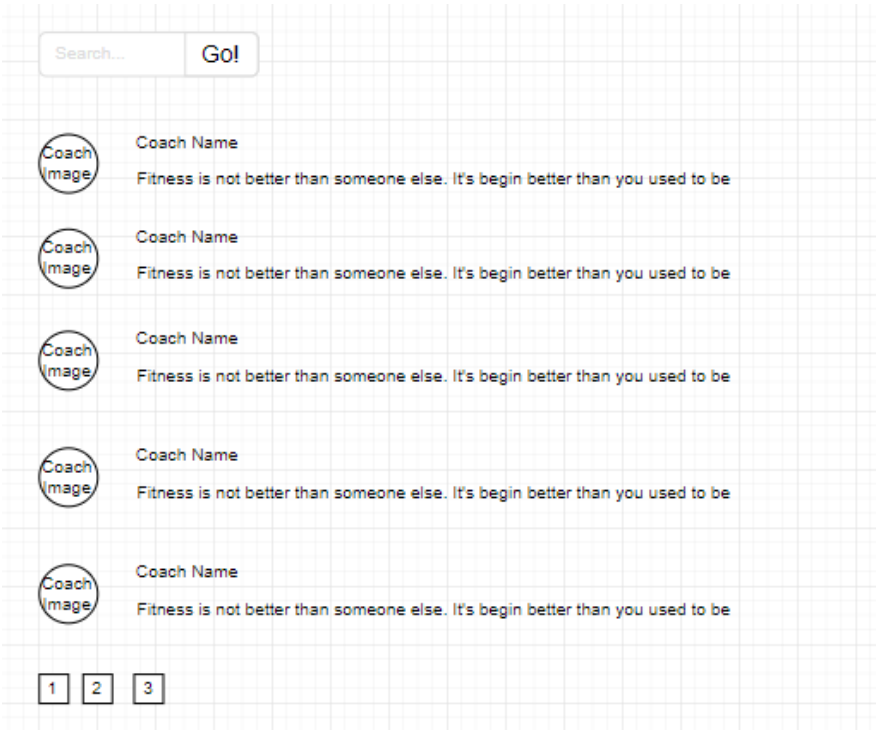


Figure 35. View Coaches

Figure 36. Hire coach wireframe

5.3.3 Do Single Exercise and Workout Wireframe

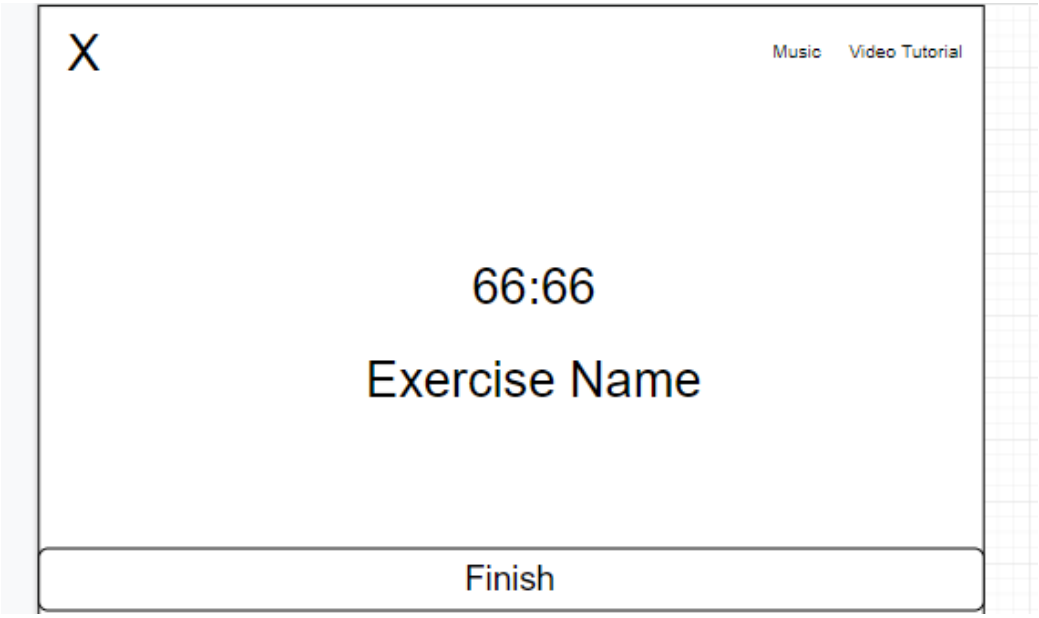


Figure 37. Do Single Exercise and Workout Wireframe

5.3.4 View List of Song Wireframe

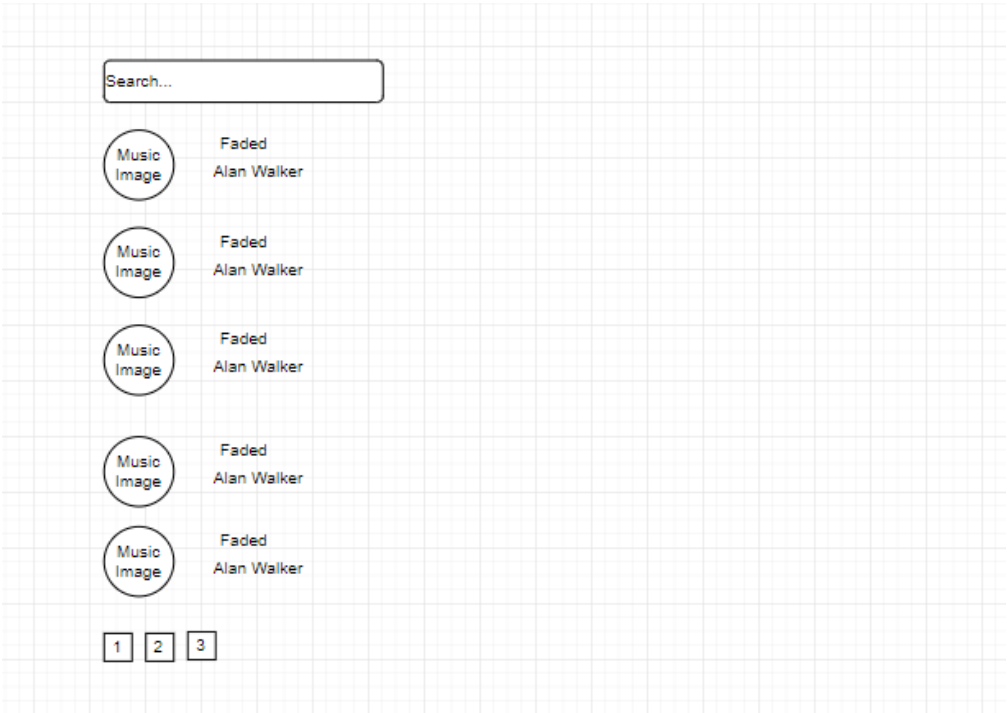


Figure 38. List of Song Wireframe

5.3.5 View List of Galleries Wireframe

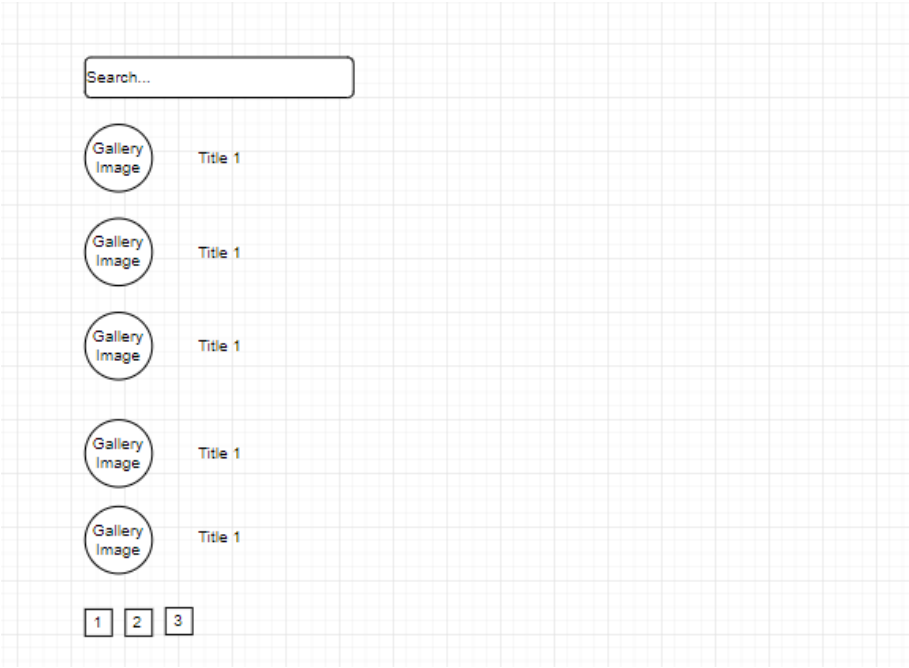


Figure 39. List of Galleries Wireframe

5.3.6 Coach Payment Confirmation Wireframe

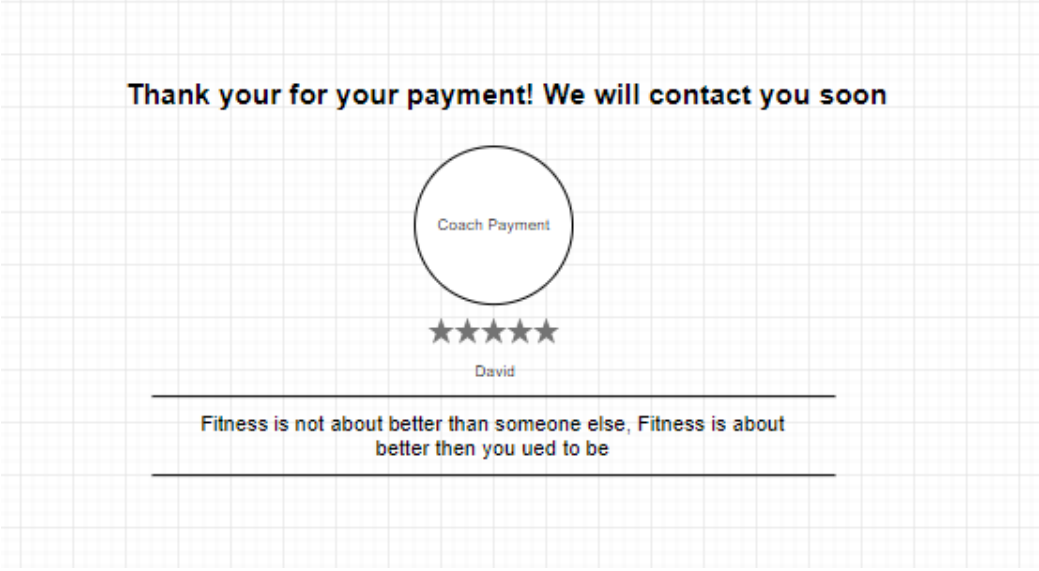


Figure 40. Coach Payment Confirmation Wireframe

5.3.7 Product Payment Confirmation Wireframe

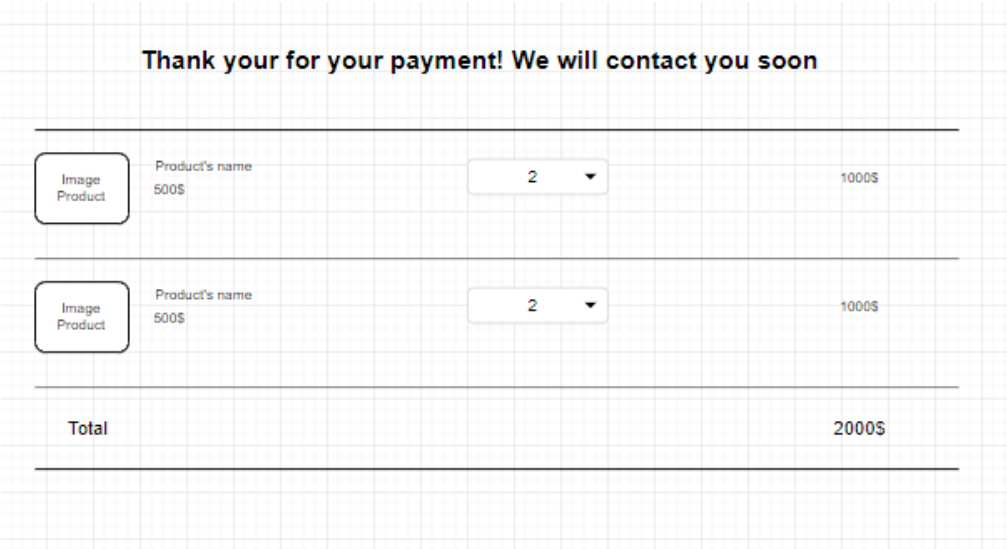


Figure 41. Product Payment Confirmation Wireframe

5.3.8 NewFeed Wireframe

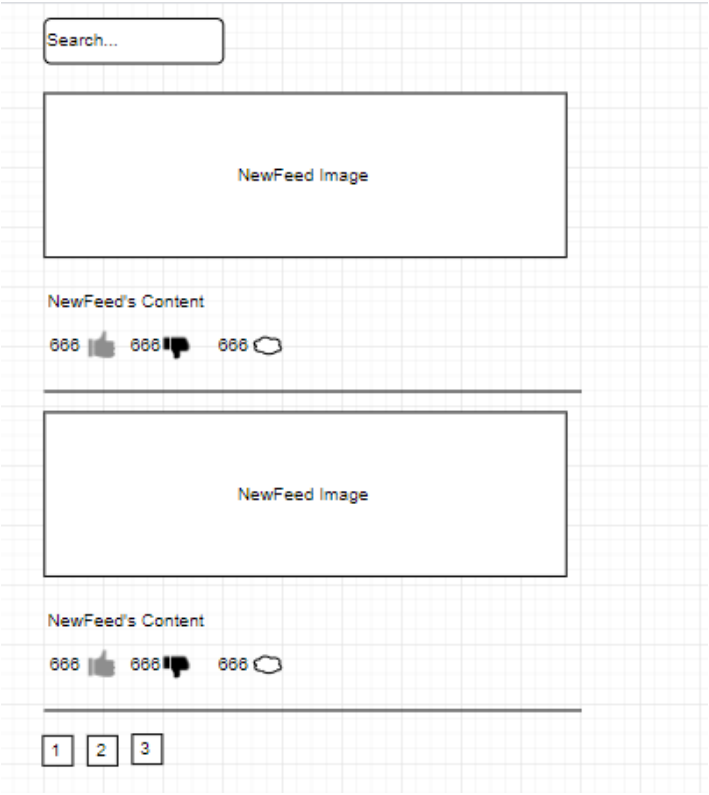


Figure 42. NewFeed Wireframe

5.3.9 Add Training Schedule Wireframe

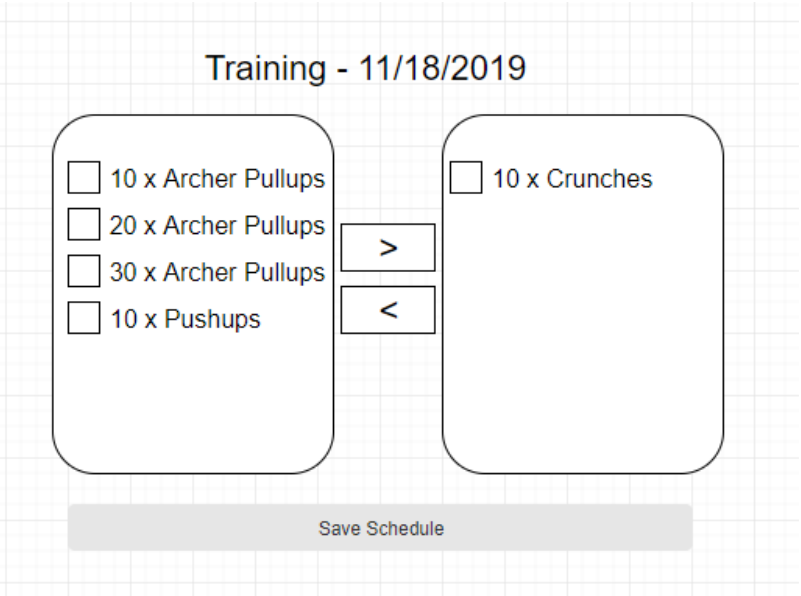


Figure 43. Save Training Schedule Wireframe

5.3.10 View List of Training Schedule Wireframe

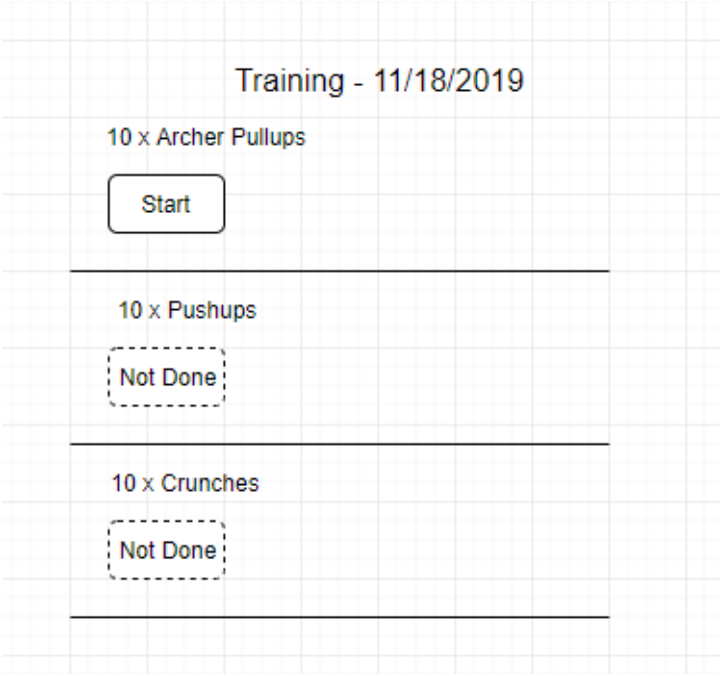


Figure 44. View List of Training Schedule Wireframe

The author created some wireframes of high level requirements. These wireframes were created base on some factors including attractive, user's experiences and so on. The images above demonstrated that main functionalities of the online fitness system are easy to use. Users do not need to be trained before using the system. For more information about the real user's interfaces, users could see the appendix of this document.

6 DEVELOPMENT OF FITNESS SYSTEM

6.1 Frontend

According to literature search, the system need to consume REST APIs from backend services. There are many ways to consume data from backend services including AJAX, fetch API with supporting of ES6 (ECMAScript 2015). For this reason, the author decided to use Single Page Application (SPA). There are many technologies that support ReactJs, Angular, VueJs and Angular is chosen in order to create frontend for the online fitness system.

By using Angular, the developer could increase user's experiences and reduce cost for frontend cloud server because the Single Page Application will not reload the entire page and it just specify which parts on the page should be reloaded based on user's interaction.

6.1.1 Structure of frontend application

There are so many ways to structure a frontend application. For the small project, structuring frontend project is not really important. The developer team can write everything in one or two files. In fact, structuring frontend application for the fitness online system takes an important role in the development process because there are many features in the system and the developer wants to structure the project appropriately. Therefore, the frontend application could be scalable when the developer wants to add more features to the system and easy to test functionalities in the system.

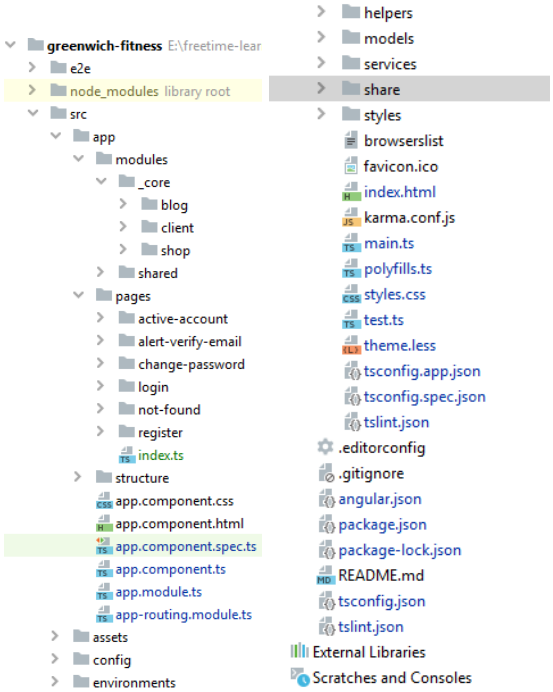


Figure 45. Structure of Fitness frontend application

1. Modules

Firstly, according to requirements analysis part, users could buy products and read blogs and do many things. For this reason, the frontend fitness online system could be divided into three modules. Dividing the system into different modules will increase maintainability and scalability. The first module is **client**, the second module is **shop**, the third module will be **blog**.

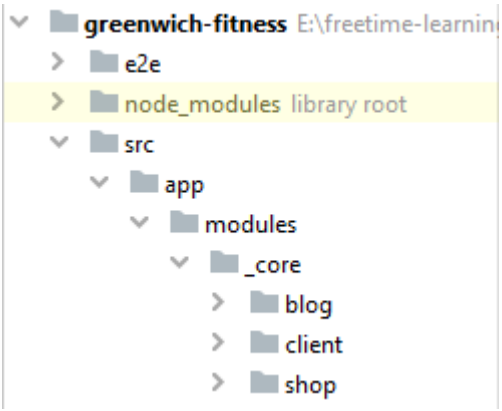


Figure 46. The frontend application will be divided into different modules

The client module will contain many different components that supporting many features, for example, user can do the exercises, hire coach, listen to music, view galleries and so on. The shop module will allow users to buy products. Following that, the blog modules will help users to read articles about different topics including nutrition, science, training.

According to Angular documentation, modules is a set of components, services, directive, etc. An angular application will be divided into logical pieces and each piece of code will be called as "module" which perform a single task. In order to create module for the fitness system the developer use the statement

ng g module name-of-module (ng g m name-of-module)

For each module, the developer will import dependent modules and declares necessary components as readers could see at the appendix A of this document. All module will import some similar modules, for example:

```
import {FormsModule, ReactiveFormsModule} from '@angular/forms';
```

Figure 47. Import modules in Angular to interact with forms

"FormsModule" and "ReactiveFormsModule" will help the developer interact with forms in Angular application.

2. Share Module

Especially, there are some common components that will be used in different modules in the application. For this reason, the developer created a module that will be called “shared”.

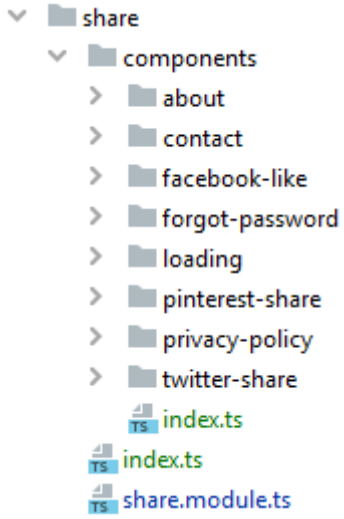


Figure 48. Share Module

As readers could see at the image above, there are some common components that will be used in different places in the application, for example, loading component that has ability to show loading indicator when user perform API calls or about and contact components will be used in blog module and shop module to demonstrate information of the system and so on. The advantage of creating share module is to create common components that will be used in other modules. For this reason, the developer just need to create once and use anywhere.

3. Pages Folder

There are some components that are not belonging to any modules. For example, login page, register page and so on are not belonging to any modules. For this reason, the developer created a folder that called “pages”.

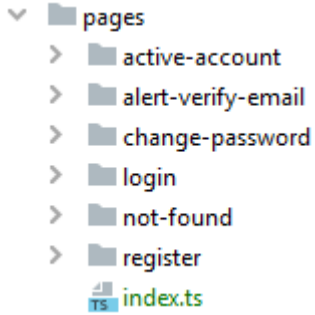


Figure 49. Pages folder

This folder will contain any pages that are not belonging to any modules. As readers could see at the image above. There are some pages that are using in the system including login page, not found page, active account page and so on.

4. Styles Folder

In Angular, global styles will be written in styles.css file. However, in order to make the system become more scalable. The developer created a folder that is called “styles”.

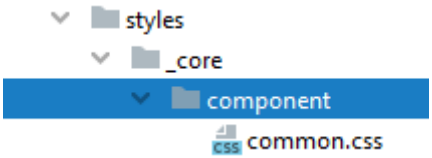


Figure 50. Styles folder

Common.css file will store common styles that could be used in any places of the system such as padding, margin, colors, weight, height and so on.

```
.height-100 {
  height: 100%;
}

.pd-left-0 {
  padding-left: 0;
}

.pd-left-72 {
  padding-left: 72px;
}

.pd-left-24 {
  padding-left: 24px;
}

.mg-bottom-24 {
  margin-bottom: 24px;
}

.none-border {
  border: none;
}

.height-auto {
  height: auto;
}
```

Figure 51. Common.css file

The developer has to face with a problem. By creating common.css file, css properties in that file will not work. The developer has to include relative path to that file to styles.css file as readers could see the image below.

```
/* You can add global styles to this file, and also import other style files */
@import "styles/_core/component/common.css";

body {
  box-sizing: border-box !important;
  background-color: #fafafa !important;
}

::-webkit-scrollbar {
  display: none;
}
```

Figure 52. Include relative path of common.css file to styles.css file

5. Model Folder

The data, that will be returned from the backend services, will have JSON format and Angular application will be created by using Typescript programming language with supporting of OOP concepts. For this reason, the developer need to male JSON data format to normal objects. In order to do that, the developer need to create classes in Angular application.

ng g class name-of-class

The statement above will be used to create a class in Angular application. Because of these classes will be used in different modules, the developer will not create classes for each module, the developer will create a folder that will be called “model” that could be used in any places in the application.

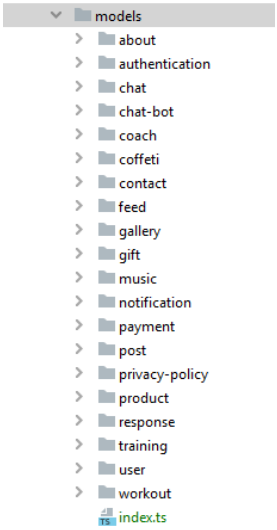


Figure 53. Model folder

6. Helpers Folder

There are common functionalities that could be needed for different places in the system. These functionalities could be stored in a folder that will be called “helper”.

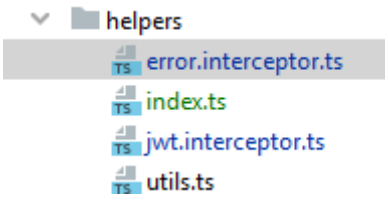


Figure 54. Helper folder

In this system, there are some files in this folder including “error.interceptor.ts” file will be use to intercept any errors that could be happened while interacting with the backend services. Following that, in order to increase security of the system, JSON Web Token will be used, description about JSON Web Token will be discussed in the backend part. In the frontend application, “jwt.interceptor.ts” will be used to add JSON Web Token to headers before sending API requests to the server. Aside that, if the developer wants to get current date, the developer need to create a function that return current date and what if many places need to know about current date. Hence, that function need to be available for many places in the system, “Util.ts” file will be created to solve the problem. “Util.ts” file will contain common function such as a function that return current date and so on. For this reason, any place which needs to know about current date, just need to import “Util.ts” file.

7. Services Folder

As mentioned above, the frontend application need to consume data from REST APIs from the backend services. If the developer wants to send requests to the backend service, “HttpClientModule” will be imported to the system.

```
import {HttpClientModule} from '@angular/common/http';
```

Figure 55. Import module to interact with the backend services

There are some concepts that readers need to understand first before moving on. Angular supports Dependency Injection (DI) via Dependency Injection Framework. Dependency Injection is an important application design pattern where a class asks for dependencies to external sources rather than creating them itself. On the other hand, a service in Angular is used when a common functionality needs to be provided to various modules. For this reason, the best practice of connecting to the backend services is to create a service and then inject “HttpClient” to constructor of that service and inject that service to any places that need it as users could see at the image below

```
import {Injectable} from '@angular/core';
import {HttpClient, HttpHeaders} from '@angular/common/http';
import {Observable} from 'rxjs';
import {About} from '@gw-models';

const httpOptions = {
  headers: new HttpHeaders( headers: {'Content-Type': 'application/json'})
};


@Injectable({
  providedIn: 'root'
})
export class AboutService {

  constructor(private http: HttpClient) {
  }

  /** GET: get about by id */
  public getAbout(url): Observable<About> {
    return this.http.get<About>(url, httpOptions);
  }
}
```

Figure 56. Create a service to connect to backend service to get about information

```
export class AboutComponent implements OnInit {
  selectedAboutContent: About;
  isLoadingSpinnerShown: boolean;


  /**
   *
   * @param aboutService - inject aboutService
   */
  constructor(private aboutService: AboutService) {
  }
```

Figure 57. Inject AboutService to selected component via constructor

Because of that services will be used in different places. Hence, a “service” folder will be created to store all service files as readers could see the image below.

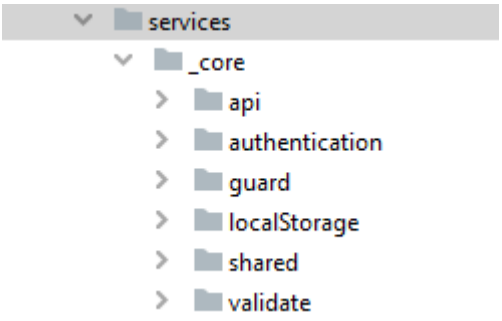


Figure 58. Service folder

“api” folder will be used to store all services file that have ability to interact with the backend services. On the other hand, “authentication” folder will be used to store services that authenticate user’s

credentials. “guard” folder is created to authorize users for each route. “localStorage” folder stores services to handle local storage in the application. “shared” folder stores common services and “validate” stores services for validation.

8. Config folder

Figure 59. Config folder

Because of REST API URLs could be changed over time. For this reason, the developer need to create a file that can store API URLs so that when API URLs are changed, the developer just need to change API URLs in that file without changing other files.

```
// api about management
apiAboutManagementPrefix = 'about-management',
apiAbouts = 'abouts',
```

Figure 60. About API Urls that are stored in config.ts file

```
/**
 * get about's content
 */
private getAboutContent(): void {
  this.isLoadingSpinnerShown = true;
  const aboutId = 1;
  const getAboutUrl = `${Config.apiBaseUrl}/${Config.apiAboutManagementPrefix}/${Config.apiAbouts}/${aboutId}`;
  this.aboutService.getAbout(getAboutUrl)
    .subscribe( next: (selectedAboutContent: About) => {
      if (selectedAboutContent) {
        this.selectedAboutContent = selectedAboutContent;
      }
      this.isLoadingSpinnerShown = false;
    });
  }
}
```

Figure 61. About API Url

As readers could see above, “getAboutUrl” variable storing API Url that will be used to get data. However, whenever about API Url is changed, the developer just need to change value in the “config.ts” file without changing “getAboutUrl” variable.

9. Conclusion

In conclusion, the fitness frontend application is structured for enterprise application and could be scaled in the future when the developer wants to add more features to the system. Each folder will have each mission. For this reason, it will be easier when the developer needs to fix bug or change anything. On the other hand, if new developers join the project, they could understand the structure the project. Therefore, Training new developers will not take much time.

6.1.2 Applying Angular in Fitness System

All requirements are achieved by using Angular framework. However, in this section, there are some special cases will be discussed to clarify how the developer applies Angular to solve specific problems.

1. Interact with the backend services

Firstly, in order to interact with the backend services in Angular, the developer needs to import "HttpClientModule" to the module which needs to perform API calls.

```
import {HttpClientModule} from '@angular/common/http';
```

Figure 62. import module to interact with backend services

When performing API calls, "HttpClient" needs to be injected to constructor via Dependency Injectiom Framework. Concepts of Dependency Injection Framework was described above. Following that, "HttpClient" needs to be injected to constructor where the developer wants to interact with the backend services.

```
constructor(private http: HttpClient) {
}
```

Figure 63. Inject HttpClient to constructor in order to interact with backend services

However, the developer realize that many components can call the same APIs. Therefore, the developer needs a way to define API interaction functions once and can be used anywhere in the frontend application so it is time service comes into play. Like Dependency Inject Framework, the concept of service was described above.

```
@Injectable({
  providedIn: 'root'
})
export class AboutService {

  constructor(private http: HttpClient) {
  }

  /** GET: get about by id */
  public getAbout(url): Observable<About> {
    return this.http.get<About>(url, httpOptions);
  }
}
```

Figure 64. Create services so that it can be injected anywhere in the application

As readers could see above, the developer create a service called “About” service that can interact with About APIs. The developer can use About in any places by injecting it via constructor without redefine “getAbout” function as readers could see the code below.

```
export class AboutComponent implements OnInit {
  selectedAboutContent: About;
  isLoadingSpinnerShown: boolean;

  /**
   *
   * @param aboutService - inject aboutService
   */
  constructor(private aboutService: AboutService) {
  }
```

Figure 65. inject AboutService any places that need it

Like “About” service, other services will be done in the same way.

2. Authentication and Authorization

Every big system will need authentication and authorization and the online fitness system is not an exception. In order to achieve authentication and authorization in Angular framework, the developer followed some steps below.

```
@Injectable({providedIn: 'root'})
export class AuthenticationService {

  public currentUser: Observable<AuthenticationUser>;
  private currentUserSubject: BehaviorSubject<AuthenticationUser>;

  constructor(private http: HttpClient) {
    this.currentUserSubject = new BehaviorSubject<AuthenticationUser>(JSON.parse(localStorage.getItem( key: 'currentUser')));
    this.currentUser = this.currentUserSubject.asObservable();
  }

  /**
   *
   * @param url - url that will be used to login
   * @param username - user's name that will be used to login
   * @param password - user's password that will be used to login
   */
  login(url: string, username: string, password: string) {
    return this.http.post<any>(url,  body: {userName: username, password: password})
      .pipe(map( project: response => {
        const responseJson = JSON.parse(response);
        if (responseJson != null && responseJson.userName != null && responseJson.token != null) {
          this.handleLoginSuccessfully(response);
        }
        return response;
      }));
  }
```

Figure 66. Create AuthenticationService to authenticate user's credentials

Firstly, the developer created “Authentication” service that has ability to authenticate user.

When user clicked on login button on the login page, “Authentication” service will be called to authenticate user’s credentials.

```
localStorage.setItem('currentUser', JSON.stringify(authenticationUser));
```

Figure 67. save user's information to local storage.

After user logged in successfully, JSON Web Token and user’s information will be returned from the backend services. that information will be stored in the local storage. For this reason, user does not need to logged in again the next time unless user logged out.

```
canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot) {
  const currentUser = this.authenticationService.currentUserValue;
  if (currentUser) {
    // check if route is restricted by role
    if (route.data.roles) {
      // role not authorised so redirect to home page
      // check role of user-account
      let isValidRole = false;
      for (const eachRole of route.data.roles) {
        if (currentUser.roles.indexOf(eachRole) !== -1) {
          isValidRole = true;
          break;
        }
      }
      if (!isValidRole) {
        this.router.navigate( commands: ['/']);
        return false;
      }
    }
    // logged in and authorised so return true
    return true;
  }

  // not logged in so redirect to login page with the return url
  this.router.navigate( commands: ['/login'],  extras: {queryParams: {returnUrl: state.url}});
  return false;
}
```

Figure 68. AuthGuard file is created to authorize user

For authorization, the developer created "auth.guard.ts" file. This file will get current user's information from local storage and compare roles of current user to roles of each route. If roles of current user matched to roles of each route, users will be allowed to access that route.

```
const routes: Routes = [
  {
    path: 'client',
    loadChildren: '@gw-client-module/client.module#ClientModule',
    canActivate: [AuthGuard],
    data: {
      roles: [Role.Admin, Role.Coach, Role.User]
    }
  },
  {
    path: 'shop',
    loadChildren: '@gw-shop-module/shop.module#ShopModule',
    canActivate: [AuthGuard],
    data: {
      roles: [Role.Admin, Role.Coach, Role.User]
    }
  },
```

Figure 69. Roles will be added to each route for authorization

Each route will contain roles that are allowed to access as readers could see the example code below

```
@Injectable()
export class JwtInterceptor implements HttpInterceptor {
  constructor(private authenticationService: AuthenticationService) {
  }

  intercept(request: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {
    // add authorization header with jwt token if available
    const currentUser = this.authenticationService.currentUserValue;
    if (currentUser && currentUser.token) {
      request = request.clone( update: {
        setHeaders: {
          Authorization: `Bearer ${currentUser.token}`
        }
      });
    }

    return next.handle(request);
  }
}
```

Figure 70. JWT will be added to header before sending requests to the server

Because of the backend services are just available based on user's roles. Hence, the frontend application needs to add JSON Web Token before sending requests to the server so that the server can recognize role of current user and decide to allow current user access to that backend service or not. The example code will demonstrate how to add JSON Web Token to header before sending requests to the sever in Angular framework.

3. Exercises and Workouts.


```
{
  "workouts": [
    {
      "base_name": "achilles",
      "slug": "achilles-none-5",
      "title": "Achilles",
      "full_title": "Achilles",
      "subtitle": null,
      "subtitle_heading": null,
      "title_suffix": null,
      "rounds_count": 5,
      "base_rounds_count": 5,
      "category_slug": "regular",
      "volume_description": "x1",
      "free": false,
      "focus": ["full_body"],
      "pace": null,
      "tags": [],
      "body_regions": [],
      "points": "450.0",
      "points_for_star": "90.0",
      "points_for_personal_best": "90.0",
      "label": { "text": "NEW", "type": "brand_new" },
      "description": null,
```


Figure 71. Local JSON data

Allowing users do exercises and workouts in the system is one of the main requirements. Instead of creating backend services that return data of exercises and workouts, the developer decided to create local JSON data to store data of exercises and workouts. There are more than 200 exercises and 100 workouts in the system. By doing this, the system does not need to call to backend services while loading data of exercises and workouts. Hence, the performance will be increased and the cost of cloud server will be reduced.

```
@Injectable({
  providedIn: 'root'
})
export class ReadLocalJsonService {

  constructor(private http: HttpClient) {
  }

  /**
   *
   * @param localJsonUrl - url of local json
   */
  public getJSON(localJsonUrl): Observable<any> {
    return this.http.get(localJsonUrl);
  }
}
```

Figure 72. Create ReadLocalJsonService to read data from local JSON file

The developer created “ReadLocalJson” service to read exercises and workouts data from local JSON file as readers could see the example code below

```
/**
 * save single exercise's state
 */
private saveSingleExerciseState(): void {
  localStorage.setItem(Config.currentExerciseTime, this.currentExerciseTime);
  localStorage.setItem(Config.currentSecondExerciseTime, String(this.currentSecondsExerciseTime));
  localStorage.setItem(Config.isCountDownSingleExerciseShown, 'true');
}
```

Figure 73. Save state of exercises

There is a problem that the developer has to face with. The system need to keep state while user doing the exercises, for example, when the user doing the exercise, the user goes to view video about exercise tutorial and then come back to the training. In this case, the system need to keep and restore state of current exercise including current time, current progress. In order to solve the problem, the developer decided to store current time and current progress to local storage. For this reason, when the user come back to the training, the system can retrieve state of current exercise from local storage. The example code below will demonstrate how to solve the problem.

4. Payment to Hire Coach and Buy Products

The main purpose of the system is to connect coaches and users across the world. Users will need a way to pay for coach. For this reason, Paypal will be integrated to the system.

```
@Injectable({
  providedIn: 'root'
})
export class PaymentService {

  constructor(private http: HttpClient) {
  }

  /**
   *
   * @param url - url that will be used to make payment
   */
  public makePayment(url: string) {
    return this.http.post<Response>(url, body: {}, httpOptions);
  }

  /**
   *
   * @param url - url that will be used to complete payment
   */
  public completePayment(url: string) {
    return this.http.post<Response>(url, httpOptions);
  }
}
```

Figure 74. Create payment service to let user make payment

The developer created a service is called “Payment” service that will make payment when users want to buy products or hire coach. The example code below will show how “Payment” service was created in the system. The rest of payment process will be handled by the backend services and will be discussed later in this document.

```
  /**
   * call complete coach payment
   */
  private callCompleteCoachPayment(): void {
    this.isLoadingSpinnerShown = true;
    const completePaymentUrl = `${Config.apiBaseUrl}/
${Config.apiPaypalManagementPrefix}/
${Config.apiCompletePayment}?
${Config.paymentIdParameter}=${this.paymentId}&
${Config.payerIdParameter}=${this.payerId}`;
    this.paymentService.completePayment(completePaymentUrl)
      .subscribe( next: (response: any) => {
        if (response && response.status.localeCompare( that: 'success') === 0) {
          this.addMembership();
        } else {
          this.isPaymentSuccessfully = false;
          this.removeDataFromLocalStorage();
        }
        this.isLoadingSpinnerShown = false;
      });
  }
```

Figure 75. make payment

"callCompleteCoachPayment" is created to complete payment when user wants to hire coach.

5. Chat text and Video Call in Real Time

The system will connect coaches and users. Moreover, the system also provides some methods that allow coaches and users communicate to each other including chat text and video call in real time.

```
@Injectable({
  providedIn: 'root'
})
export class SocketService {
  socket: any;

  constructor() {
  }

  /**
   *
   * @param serverUrl - server socket that will be connected
   */
  connect(serverUrl: string): void {
    this.socket = io(serverUrl);
  }

  /**
   *
   * @param chatRoom - chat's room that will be created
   */
  joinChatRoom(chatRoom: string): void {
    this.socket.emit('createRoom', chatRoom);
  }
```

Figure 76. Socket service is created to handle real time tasks

In fact, real time tasks are not easy to achieve. The developer has to create "Socket" service to handle real time problems. The "Socket" service will provide necessary methods to handle chat text and make a video call between users and coaches. There are some libraries need to be installed in this case including "socket.io-client" that could be installed by using

npm i socket.io-client.

In order to make chat text between two users, the developer created some functions that will be described below.

```
/**
 * connect to socket server
 */
private connectToSocketServer(): void {
  this.socketService.connect(Config.serverSocketUrl);
}
```

Figure 77. Connect to socket server

Users should connect to the socket server. After connecting successfully, users can chat text together or make video calls.

```
    /**
     * get chat messages
     */
    private getChatMessages() {
      const selectedChatRoomId = this.selectedParticipant.chatRoom.id;
      const getChatMessagesUrl = `${Config.apiBaseUrl}/
    {Config.apiChatManagementPrefix}/
    {Config.apiChatMessages}?
    {Config.chatRoomIdParameter}=${selectedChatRoomId}`;
      this.isLoadingSpinnerShown = true;
      this.chatMessageService.getChatMessages(getChatMessagesUrl)
        .subscribe( next: (chatMessages: ChatMessage[]) => {
          if (chatMessages && chatMessages.length) {
            this.showChatMessages(chatMessages);
          }
          this.joinChatRoomBetweenUserAndCoach();
          this.isLoadingSpinnerShown = false;
        });
    }

  /**
   *
   * @param chatMessages - chat's messages that will be shown
   */
  private showChatMessages(chatMessages: ChatMessage[]) {
    chatMessages.map((eachChatMessage: ChatMessage) => {
      if (eachChatMessage.userProfile.id === this.selectedUserProfile.id) {
        this.currentChatContent = this.currentChatContent +
          `<div class="message-right-container"><div class="message-right">${eachChatMessage.message}</div></div>`;
      } else {
        this.currentChatContent = this.currentChatContent +
          `<div class="message-left-container"><div class="message-left">${eachChatMessage.message}</div></div>`;
      }
    });
  }
```

Figure 78. get chat messages from the server

List of chat messages, that two users have made in the past, need to be loaded from the server. For this reason, “getChatMessages” and “showChatMessages” functions are created to send request to the backend service to load list of chat messages.

```
  /**
   * join chat room between user and coach
   */
  private joinChatRoomBetweenUserAndCoach(): void {
    const selectedChatRoom = `${this.selectedParticipant.chatRoom.id} - ${this.selectedParticipant.chatRoom.name}`;
    this.socketService.joinChatRoom(selectedChatRoom);
    this.socket = this.socketService.getSocket();
    this.listenChatMessagesFromServer();
    this.listenPeerIdsFromServer();
    this.listenStopVideoCallEvent();
  }
```

Figure 79. join chat room

After chat messages were loaded, coach and user need to be joined to the chat room so that coach and user can send message to each other.

```
/**
 * send chat message
 */
public sendChatMessage(): void {
  const chatMessage = {
    'sender': `${this.selectedUserProfile.id}`,
    'message': `${this.currentUserChatMessage}`
  };
  this.socketService.sendChatMessage(chatMessage);
  this.saveChatMessageToServer(chatMessage);
  this.currentUserChatMessage = '';
}
```

Figure 80. send chat messages

"sendChatMessage" will be created to let user send messages to other users.

```
/**
 *
 * @param chatMessage - chat's message that will be saved to the database
 */
private saveChatMessageToServer(chatMessage: any) {
  const chatMessageObj = new ChatMessage();
  chatMessageObj.chatRoom = this.selectedParticipant.chatRoom;
  chatMessageObj.message = chatMessage.message;
  chatMessageObj.userProfile = this.selectedUserProfile;
  const addChatMessageUrl = `${Config.apiBaseUrl}/${Config.apiChatManagementPrefix}/${Config.apiChatMessages}`;
  this.chatMessageService.addChatMessage(addChatMessageUrl, chatMessageObj)
    .subscribe();
}
```

Figure 81. Chat message will be saved to the server

"saveChatMessageToServer" will take responsibility in save chat messages to the server so that chat message could be loaded from the server as needed.

On the other hand, coaches and video can make a video call.

```
<script src="https://cdn.jsdelivr.net/npm/peerjs@0.3.20/dist/peer.min.js"></script>
```

Figure 82. Integrate peer-to-peer library to create video call feature

According to "Peer Js" documentation (JS, 2019), in order to achieve video call function, the developer need to integrate the library to the index.html file. The rest of communication process will be handled by the backend services that will be discussed later in this document. In order to achieve video call requirement, the developer need to create some functions.

```
/**
 * peer connect for video call
 */
private peerConnect(): void {
  this.peer = new Peer({key: 'lwjd5qra8257b9'});
  this.onPeerOpened();
}
```

Figure 83. Connect to Peer server

The developer need to connect to “Peer Js” server. “Peer Js” will generate a key which the developer can use to connect to Peer server.

```
/**
 * on peer opended
 */
private onPeerOpened() {
  this.peer.on('open', id => {
    console.log(id);
    this.peerId = id;
    this.socketService.sendPeerId(this.peerId);
  });
}
```

Figure 84. Send peer ID to other users through socket server

The next function is “onPeerOpened”, this function will send peerId of current user to other users through socket server. Hence, other users can know about peer’s id of other users and then can connect to each other.

```
/**
 * open stream
 */
private openStream(): any {
  const config = {audio: false, video: true};
  return navigator.mediaDevices.getUserMedia(config);
}
```

Figure 85. Ask users for permission to access webcam and microphone

Before making a video call, the system will ask users for permission to access webcam and microphone. Therefore, users can see each other via webcam and communicate by using microphone.

```
/**
 *
 * @param stream - stream
 */
private playRemoteStream(stream): void {
  this.remoteStreamNativeElement.srcObject = stream;
  this.remoteStreamNativeElement.play();
}
```

Figure 86. function to open remote stream while two users connected

When two users A and B make a video call. In reality, user A should the user B via webcam and vice versa. For this reason, the developer created “playRemoteStream” function to show remote stream when two users connected.

```
/**
 * show confirm video call modal
 */
private showConfirmVideoCallModal(): void {
  const that = this;
  this.modalService.confirm( options: {
    nzTitle: 'Video Call',
    nzContent: `${this.selectedCoach.userProfile.fullName} is calling you`,
    nzOnOk: () => {
      this.isLoadingRemoteStream = true;
      this.openStream()
        .then( onfulfilled: stream => {
          that.videoCallModalNativeElement.style.display = 'block';
          that.videoCallModalContentNativeElement.style.background = '#fefefe';
          that.localStream = stream;
          that.callObj.answer(stream);
          that.callObj.on('stream', remoteStream => {
            that.isLoadingRemoteStream = false;
            that.playRemoteStream(remoteStream);
            that.videoCallModalContentNativeElement.style.background = 'transparent';
          });
        });
    },
    nzOnCancel: () => {
      this.socketService.stopVideoCall();
    }
  });
}
```

Figure 87. Show confirmation dialog when someone is calling current user

The developer wants when other user is calling current user, a confirmation dialog will be shown to make sure that current user wants to answer that user, or not. Therefore “showConfirmVideoModal” will be created to achieve that requirement.

```
/**
 * make video call
 */
public makeVideoCall(): void {
  this.isLoadingRemoteStream = true;
  this.openStream()
    .then( onfulfilled: stream => {
      this.videoCallModalNativeElement.style.display = 'block';
      this.videoCallModalContentNativeElement.style.background = '#fefefe';
      this.localStream = stream;
      const call = this.peer.call(this.remoteId, stream);
      call.on('stream', remoteStream => {
        this.playRemoteStream(remoteStream);
        this.isLoadingRemoteStream = false;
        this.videoCallModalContentNativeElement.style.background = 'transparent';
      });
    });
}
```

Figure 88. the function that let user to make a video call

When the user clicked on “call” button. The system should make video call between two users. Therefore, “makeVideoCall” function is created to handle the request.

6. NgZorro

According to NgZorro (NgZorro, 2019), NgZorro following the Ant Design specification and provides many components which will help building rich, interactive user interfaces. There are some features of NgZorro including a set of high quality Angular components and written in Typescript, that components are suitable for enterprise applications, high performance and there are some famous companies that are using NgZorro in their system, for example, Alibaba, Apache Flink and so on. For this reason, the author decided to use NgZorro to build user interfaces for the online fitness system. In order to integrate NgZorro to an Angular application, the developer follows some steps that will be described below.

```
$ cd PROJECT-NAME
$ ng add ng-zorro-antd
```

```
$ ng new my-project
$ cd my-project
$ ng add ng-zorro-antd

? Add icon assets: (y/N)
? Set up custom theme file: (y/N)
? Choose your locale code: (en_US, ca_ES, de_DE, zh_CN ... )
? Choose template to create project: (Use arrow keys)
  blank
❯ sidemenu
```

Figure 89. Add NgZorro to Angular application

[Source: https://ng.ant.design/docs/getting-started/en]

In order to integrate NgZorro to current Angular application, the author follow these steps above. However, in the online fitness system, the author wants to customize workflow. Hence, the author followed some steps below.

```
$ npm install ng-zorro-antd --save
```

Figure 90. install NgZorro

[Source: https://ng.ant.design/docs/getting-started/en]

```
import { BrowserModule } from '@angular/platform-browser';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
import { NgZorroAntdModule, NZ_I18N, en_US } from 'ng-zorro-antd';
import { AppComponent } from './app.component';

/** config angular i18n **/
import { registerLocaleData } from '@angular/common';
import en from '@angular/common/locales/en';
registerLocaleData(en);

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    FormsModule,
    HttpClientModule,
    BrowserAnimationsModule,
    /** import ng-zorro-antd root module, you should import NgZorroAntdModule and avoid importing sub mo
    NgZorroAntdModule
  ],
  bootstrap: [ AppComponent ],
  /** config ng-zorro-antd i18n (language && date) **/
  providers    : [
    { provide: NZ_I18N, useValue: en_US }
  ]
})
export class AppModule { }
```

Figure 91. Import necessary module

[Source: https://ng.ant.design/docs/getting-started/en]

```
{
  "assets": [
    ...
    {
      "glob": "**/*",
      "input": "./node_modules/@ant-design/icons-angular/src/inline-svg/",
      "output": "/assets/"
    }
  ],
  "styles": [
    ...
    "node_modules/ng-zorro-antd/ng-zorro-antd.min.css"
  ]
}
```

Figure 92. Add Styles and SVG assets

7. Conclusion

In conclusion, all requirements, that was mentions in requirement analysis, were achieved. Angular takes an important in success of the system. Angular helps the developer solve all problems that occurred in the development process. By using Angular, the developer can interact with the backend services without any problems. Angular also supports the author in building some advance features such as chat text, video call and integrate PayPal to the system. Angular contains rich libraries from the community to build the user interfaces, for example, Angular Material, NgZorro and so on. Therefore, instead of writing pure CSS files to make the system become more attractive, the author just need to use libraries that supports many components. For this reason, time of creating user

interfaces for the system will be reduced. The last but not least, structuring of the Angular application is easy to maintain, scale up in the future. The structure is suitable for enterprise applications.

6.2 Backend

In order to create backend for the online fitness system, the author decided to choose three frameworks Spring Boot framework – using Java programming language, ExpressJs – using Javascript programming language, and Flask framework - using Python programming language. Spring Boot framework will take responsibility to handle almost backend services of the online fitness system. Aside that, ExpressJS framework will be used to create real time services and recommendation model was deployed to production by using Flask framework. For this reasons, each backend services will take different responsibilities to make the online fitness system work well in production. This section will demonstrate how the author created backend services to solve specific problems.

6.2.1 Structure of backend project

1. Structure of Spring Boot project

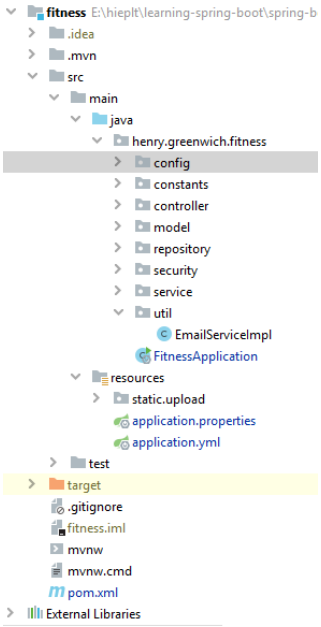


Figure 93. Structure of Spring Boot project

The online fitness system contains many features and Spring Boot framework was applied to handle almost backend services. It means that Spring Boot project should have a good structure so that the developer can more functionalities in the future. This section will demonstrate how the author structure the Spring Boot project for the online fitness system.

a. Config Folder

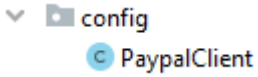


Figure 94. Config folder

While developing backend services for the fitness system, the author need to store configuration. Hence, "config" folder is created to store configuration. As readers could see the image above, "PaypalClient" file is creatd to configure PayPal in order to let users hire coaches and buy products via PayPal so "PaypalClient" file will be stored in "config" folder. The detail about "PaypalClient" file will be discussed in detail later.

b. Constants Folder

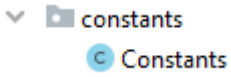


Figure 95. Constants folder

The backend services need to store some information such as name of database, name of tables, fields and so on. This information will not be changed in the entire application so that this information should be defined as constants. Hence, the author created a folder that called "constants" to store all constants variables in the Spring Boot project and "Constants" file will contain all constant variables and this file will be included in "constants" folder.

c. Controller Folder

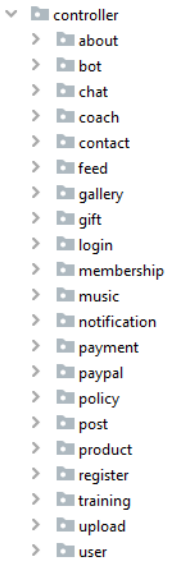


Figure 96. Controller folder

When users send request to the backend services. "controller" will receive requests and return responses to users. As mentioned above, Spring Boot project handles almost backend services of the

online fitness system. Hence, there are so many “controller” files in the Spring Boot project so that the author will create “controller” folder that will be used to store all controller files in the project.

d. Service Folder

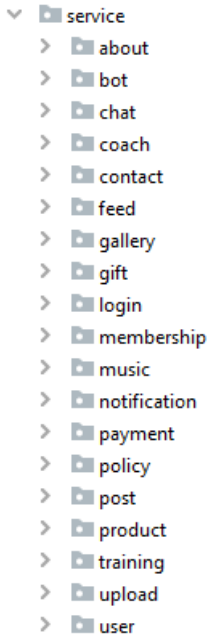


Figure 97. Service folder

“Controller” will receive user’s requests and return the responses. However, controller will not handle business logic directly. For handling business logic, “service” files were created and will be store in “service” folder. By using service folder, the Spring Boot project is structured clearly. It means that when the developer has to fix bugs about business logic or add more business logic to the system, the developer can think about “service” folder immediately.

e. Repository Folder

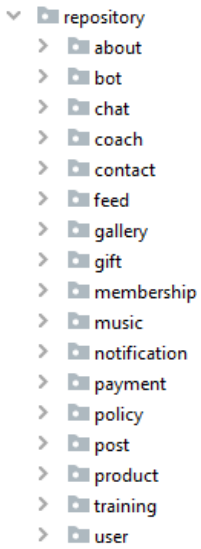


Figure 98. Repository Folder

Sometimes, in order to handle business logic, “service” files will need to interact with the database. In the Spring Boot project, “service” files will not call to the database directly. On the other hand, “service” files will base on “repository” files. “repository” files will take responsibility to interact with the database as needed. Therefore, the author decided to create “repository” folder to store all “repository” files in the project.

f. Model Folder

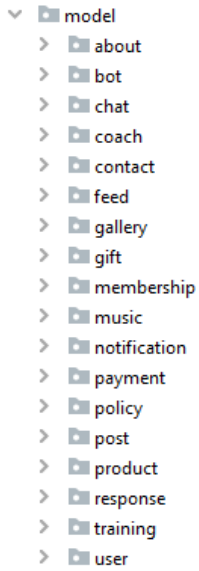


Figure 99. Model folder

Because of Spring Boot project is created by using Java programming language. Java programming language comes with OOP (Object Oriented Programming) concepts so when people is talking about Java. It means that people want to mention about objects. For example, the developer need to create “Coach” class file to store state and behaviors of a coach or “User” class file to store state and behaviors of a user and so on.

```java
@Setter
@Getter
@AllArgsConstructor
@NoArgsConstructor
@Entity
@Table(name = Constants.ABOUT_TABLE)
public class About {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = Constants.ABOUT_ID)
    private Long id;

    @Column(name = Constants.ABOUT_NAME)
    private String aboutName;

    @Column(name = Constants.ABOUT_CONTENT)
    private String aboutContent;

    @Column(name = Constants.ABOUT_META_TITLE)
    private String aboutMetaTitle;
```

Figure 100. Mapping properties of classes to corresponding fields in the tables of the database

On the other hand, the author wants to ease the development so that the developer used JPA (Java Persistence API), it kinds of ORM (Object Relation Mapping) that will be used to map properties of classes to corresponding fields in the tables of the database. Hence, the author created “model” folder to store all “model” class files in the project.

g. Security Folder

Figure 101. Security folder

Security takes an important role in every system. In order to increase security of the online fitness system. The author applied JWT (JSON Web Token) concepts. It means that when users logged in to the system, users will receive a token and that token will be added to headers before users sending requests to the backend services, if the token is valid, users will be allowed to access and vice versa. Therefore, users created “security” folder to store all files that will be necessary for achieving security of the system. The detailed of JWT concepts will be discussed later in this document.

h. Util Folder

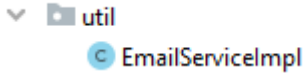


Figure 102. Util folder

There are some common functionalities that will be used in different places in the Spring Boot project so that it requires to create that functions once and can be used anywhere in the project. By understanding that problem, the author created "util" folder to store all common functionalities. For example, "EmailSerivceImpl" file is created to send email to users and that file could be used in different places in the Spring Boot project such as when users register a new account, the system will send an email for confirmation and so on.

i. application.yml

```yaml
spring:
  application:
    name: greenwich_fitness_api
  datasource:
    url: jdbc:mysql://localhost:3306/greenwich_fitness
    username: root
    password: 12345
  servlet:
    multipart:
      max-file-size: 100MB
      max-request-size: 100MB
  jackson:
    serialization:
      FAIL_ON_EMPTY_BEANS: false
  mail:
    host: smtp.gmail.com
    port: 587
    username: hieplt.developer@gmail.com
    password: Greenwich31121995
    properties:
      mail:
        smtp:
          auth: true
          starttls:
            enable: true
  devtools:
    restart:
      enabled: false
```

Figure 103. application.yml file

The Spring Boot project will need to be configured. Hence, "application.yml" file comes into play. The author can do something in this file including connecting to the database, configuring email server and upload server or changing prefix of REST APIs so readers could see that "application.yml" takes an important role in the Spring Boot project.

2. Structuring ExpressJS project

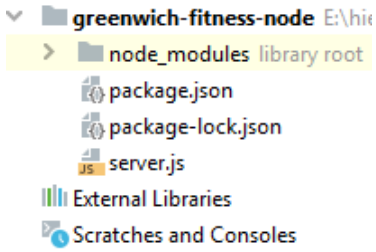


Figure 104. structure of ExpressJs project

In order to achieve real time tasks, the author decided to use NodeJS/ExpressJS framework with Socket.io library. The image above demonstrated how structure the ExpressJS project. The first folder is “node_modules”. This folder will store all libraries that are necessary for the project.

```
{
  "name": "greenwich-fitness-node",
  "version": "1.0.0",
  "description": "greenwich-fitness-node",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && ex
  },
  "keywords": [
    "greenwich-fitness-node"
  ],
  "author": "Henry",
  "license": "MIT",
  "dependencies": {
    "express": "^4.17.1",
    "nodemon": "^1.19.2",
    "socket.io": "^2.2.0"
  }
}
```

Figure 105. Package.json file

“package.json” file includes all libraries that are used to build the system. In this case, “package.json” file contained three libraries including “express” library to use ExpressJs framework in the project, “nodemon” library will be reloaded the server whenever the code was changed. Hence, the developer does not need to reload the server by hand. “socket.io” library will be integrated to achieve real time tasks.

```
const express = require('express');
const app = express();
const PORT = 3000;
const server = require('http').Server(app);
const io = require('socket.io')(server);
server.listen(PORT, () => {
  console.log(`App is running on port ${PORT}`);
});

let peerIdArr = [];

io.on('connection', socket => {
  socket.on( event: 'createRoom', handler: chatRoom => {
    socket.join(chatRoom);
    socket.room = chatRoom;
  });
  socket.on( event: 'sendMessageToRoom', handler: chatMes
    io.sockets.in(socket.room).emit('serverSendMessageT
  });
  socket.on( event: 'sendPeerIdToRoom', handler: peerId =
    socket.peerId = peerId;
    peerIdArr.push(peerId);
    io.sockets.in(socket.room).emit('serverSendPeerToRo
  });
  socket.on( event: 'stopVideoCall', handler: () => {
    io.sockets.in(socket.room).emit('serverStopVideoCal
  });
  socket.on( event: 'disconnectServer', handler: () => {
    peerIdArr.splice(peerIdArr.indexOf(socket.peerId),
```


Figure 106. Server.js file

"server.js" file is used to create socket server that handle real time tasks, for example, chat text between users and coaches or user and coach can make a video call.

3. Structuring Flask Project

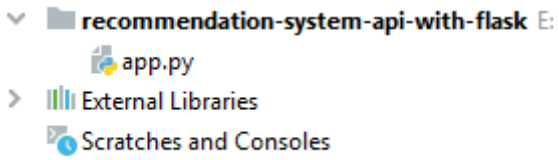
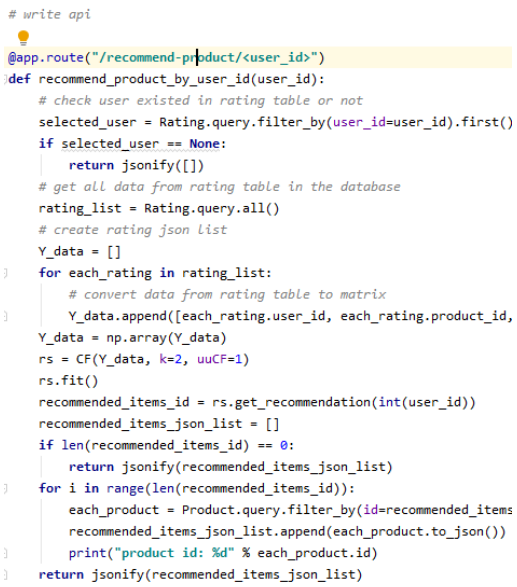


Figure 107. Structure of Flask project

As mentioned above, recommendation system will be integrated to recommend coaches to user. In order to implement recommender engine, the author will use Python programming language. However, Python programming language just could be used to create recommendation model. In fact, the model should be deployed to production via REST APIS so that Angular application can consume recommended results and suggest to users. For this reason, Flask framework will be chosen to create REST APIs and provide recommendation model to production.



```python
# write api
@app.route("/recommend-product/<user_id>")
def recommend_product_by_user_id(user_id):
    # check user existed in rating table or not
    selected_user = Rating.query.filter_by(user_id=user_id).first()
    if selected_user == None:
        return jsonify([])
    # get all data from rating table in the database
    rating_list = Rating.query.all()
    # create rating json list
    Y_data = []
    for each_rating in rating_list:
        # convert data from rating table to matrix
        Y_data.append([each_rating.user_id, each_rating.product_id,
    Y_data = np.array(Y_data)
    rs = CF(Y_data, k=2, uuCF=1)
    rs.fit()
    recommended_items_id = rs.get_recommendation(int(user_id))
    recommended_items_json_list = []
    if len(recommended_items_id) == 0:
        return jsonify(recommended_items_json_list)
    for i in range(len(recommended_items_id)):
        each_product = Product.query.filter_by(id=recommended_items
        recommended_items_json_list.append(each_product.to_json())
        print("product id: %d" % each_product.id)
    return jsonify(recommended_items_json_list)
```

Figure 108. app.py file

The image above demonstrated how structure the Flask project. There is only one file in the project folder. “app.py” will be created to build recommendation model and provide that model via REST APIs.

6.2.2 Apply Backend Technologies in Fitness System

1. Create REST API with different technologies

a. Create REST API with Spring Boot framework

In order to create REST API by using Spring Boot framework, the developer followed some steps that will be described below.

```java
@Setter
@Getter
@AllArgsConstructor
@NoArgsConstructor
@Entity
@Table(name = Constants.ABOUT_TABLE)
public class About {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = Constants.ABOUT_ID)
    private Long id;

    @Column(name = Constants.ABOUT_NAME)
    private String aboutName;

    @Column(name = Constants.ABOUT_CONTENT)
    private String aboutContent;

    @Column(name = Constants.ABOUT_META_TITLE)
    private String aboutMetaTitle;

    @Column(name = Constants.ABOUT_META_KEYWORDS)
    private String aboutMetaKeywords;

    @Column(name = Constants.ABOUT_META_DESCRIPTION)
    private String aboutMetaDescription;
```

Figure 109. About model class file

Firstly, the author need to create a model class file and with supporting of JPA (Java Persistence API), properties of the model class file to fields in the tables of the database. For example, the image above shown that “About” class file contains some properties that will be mapped to field in the “about” table of the database so that the developer can perform some common tasks such as create, read, update, delete data without writing raw query.

```
@Repository
public interface AboutRepository extends JpaRepository<About, Long> {
    /**
     * @param id - about's id that user want to get
     * @return selected about
     */
    About findAboutById(Long id);
}
```

Figure 110. “AboutRepository” file

Secondly, after creating model class file, the author need to interact with the database such as insert data, create data and so on. As mentioned above, by using JPA, the developer can do some common task without using raw queries. The image above demonstrated “AboutRepository” file was created to interact with “about” table, that file extends “JpaRepository” to get support of JPA. Following that, the author just need to use “findAboutById” function to get about content by id from the database without creating selecting data statement.

```
@Service
public class AboutService {
    private AboutRepository aboutRepository;

    /**
     * @param aboutRepository - inject aboutRepository
     */
    public AboutService(AboutRepository aboutRepository) { this.aboutRepo

    /**
     * @param id - about's id
     * @return selected about
     */
    public About getAboutById(Long id) { return this.aboutRepository.find
}
```

Figure 111. AboutService file

Thirdly, “service” files need to be produced after “repository” files were created. The role of “service” files is to handle business logic of the backend services. However, while handling business logic, “service” files may need to interact with the database. Hence, “service” files will need supporting from “repository” files. Spring Boot framework support dependency injection so that “repository” files will be injected to “service” files via constructor injection, setter injection, field injection. The image above shown that “About” service will require “About” repository. Hence, “About” service can get data from “about” table via “AboutRepository” object.

```java
@Controller
@RequestMapping("about-management")
public class AboutController {
    private AboutService aboutService;

    /**
     * @param aboutService - inject aboutService
     */
    public AboutController(AboutService aboutService) { this.aboutService = abc

    /**
     * @param id - about's id that user want to get
     * @return selected about
     */
    @GetMapping(value = "/abouts/{id}", produces = {MediaType.APPLICATION_JSON_
    @ResponseBody
    public About getAbout(@PathVariable Long id) { return this.aboutService.get

}
```

Figure 112. AboutController file

As what described above, all requests of users will be handled by "controller" files. "controller" files will receive requests from users and then call appropriate "service" file to handle the business logic through dependency injection. After business logic was handle by "service" files, "controller" will return responses to users based on user's requests.

Note: this section takes "About" REST APIs to show the process of creating REST APIs by using Spring Boot framework. The process of creating other REST APIs will be the same.

b. Creating REST APIs with Flask Framework

In order to deploy recommendation model to production, Flask framework will be chosen to create REST APIs by using Python programming language. The author followed some steps to create REST APIs in Flask framework.

```python
from flask import Flask, jsonify
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config["SQLALCHEMY_DATABASE_URI"] = "mysql+pymysql://root:12345@loca
app.config["SQLALCHEMY_TRACK_MODIFICATIONS"] = False
db = SQLAlchemy(app)
```

Figure 113. Import necessary libraries

Firstly, the developer imported some necessary libraries to using Flask framework and connect to the database via "SQLAlchemy" library as readers could see at the image above.

```python
class Rating(db.Model):
    user_id = db.Column(db.Integer, primary_key=True)
    product_id = db.Column(db.Integer, primary_key=True)
    rate = db.Column(db.Integer, unique=False)

    def __init__(self, user_id, product_id, rate):
        self.user_id = user_id
        self.product_id = product_id
        self.rate = rate

    def to_json(self):
        return {
            "user_id": self.user_id,
            "product_id": self.product_id,
            "rate": self.rate
        }
```

Figure 114. Rating class

Similar to what the developer did in the Spring Boot project, the developer used "SQLAlchemy" to map properties of classes to fields in the tables of the database. Hence, the author can perform some common tasks without using raw queries.

```python
@app.route("/recommend-product/<user_id>")
def recommend_product_by_user_id(user_id):
    # check user existed in rating table or not
    selected_user = Rating.query.filter_by(user_id=user
    if selected_user == None:
        return jsonify([])
    # get all data from rating table in the database
    rating_list = Rating.query.all()
    # create rating json list
```

Figure 115. Define route in Flask framework

The last but not least, the developer needs to provide API endpoints to users and the image above determined how the author achieved that. "@app.route("name_of_route")" will be used to specify the API endpoint so that whenever users access to that endpoint, Flask framework will perform corresponding function.

2. Authentication and Authorization

Authentication and Authorization play an important role in backend services. The system will know the current user is allowed to access the system, or not. If the current user is allowed to access the system, the system could know about which functionalities that the current user can do.

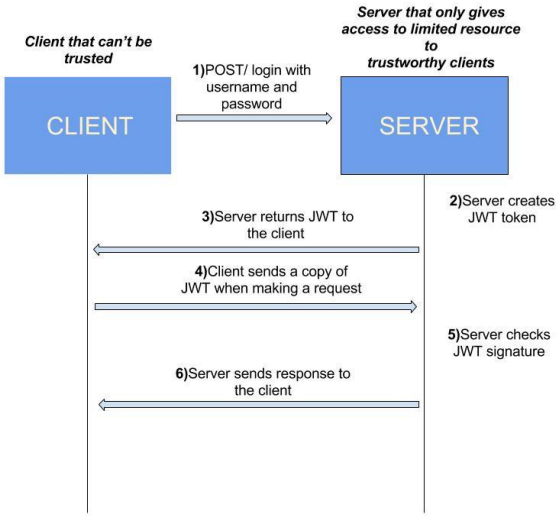


Figure 116. JWT flow diagram

[Source: https://miro.medium.com/max/1200/1*l-FS80RhxUgjZOKGgOXnTQ.jpeg]

The online fitness system using JWT (JSON Web Token) to authenticate user's credentials and authorize users. The image above demonstrated how JWT works in the system. Firstly, client will login to the system and the backend service will check user's information in the database, if this information is correct, the backend service will allow client to login to the system and return a token. Whenever the client sends a request to the backend service, that token will be added to header of the request. The server will get that token from header of the request and then check JWT signature. If the request is valid, the server will return responses back to the client.

In order to achieve JWT in the online fitness system to authenticate and authorize user, the author followed some steps that will be described in this section.

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/com.auth0/java-jwt --
<dependency>
    <groupId>com.auth0</groupId>
    <artifactId>java-jwt</artifactId>
    <version>3.4.0</version>
</dependency>
```

Figure 117. Add necessary dependencies for security

Firstly, the author added two dependencies to create security for the system, "spring-boot-start-security" to achieve security in Spring Boot framework and "java-jwt" to interact with JWT.

```java
@EnableWebSecurity
public class WebSecurity extends WebSecurityConfigurerAdapter {
    private UserDetailsServiceImpl userDetailsService;
    private BCryptPasswordEncoder bCryptPasswordEncoder;
    private FacebookAccountService facebookAccountService;
    private UserRoleService userRoleService;
    private GoogleAccountService googleAccountService;

    /**
     * @param userDetailsService     - inject userDetailsService
     * @param bCryptPasswordEncoder  - inject bCryptPasswordEncoder
     * @param facebookAccountService - inject facebookAccountService
     * @param userRoleService        - inject userRoleService
     * @param googleAccountService   - inject googleAccountService
     */
    public WebSecurity(UserDetailsServiceImpl userDetailsService,
                       BCryptPasswordEncoder bCryptPasswordEncoder,
                       FacebookAccountService facebookAccountService,
                       UserRoleService userRoleService,
                       GoogleAccountService googleAccountService) {
        this.userDetailsService = userDetailsService;
        this.bCryptPasswordEncoder = bCryptPasswordEncoder;
        this.facebookAccountService = facebookAccountService;
        this.userRoleService = userRoleService;
```

```java
/**
 * @param http - http
 * @throws Exception - throw Exception
 */
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.cors().and().csrf().disable().authorizeRequests().antMatchers(HttpMethod.POST, Constants.LOGIN_URL) ExpressionUrlAuthorizationConfigurer<H>.Authorize
            .permitAll().antMatchers(HttpMethod.POST).permitAll().antMatchers(HttpMethod.PUT).permitAll() ExpressionUrlAuthorizationConfigurer<H>.ExpressionInt
            .antMatchers(HttpMethod.GET, ...antPatterns: "/resources/**").permitAll().anyRequest().authenticated() ExpressionUrlAuthorizationConfigurer<H>.Expres
            .antMatchers( ...antPatterns: "/users/**").hasRole("ADMIN").and() HttpSecurity
            .addFilter(new JWTAuthenticationFilter(authenticationManager())) HttpSecurity
            .addFilter(new JWTAuthorizationFilter(authenticationManager(), this.userDetailsService,
                    facebookAccountService, userRoleService, googleAccountService)) HttpSecurity
            // this disables session creation on Spring Security
            .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);
}
```

Figure 118. “WebSecurity” file

“WebSecurity” file is used to configure security of the system. This file will specify which routes are allowed to access and which routes need to be authorized.

```java
public class JWTAuthenticationFilter extends UsernamePasswordAuthenticationFilter {
    private AuthenticationManager authenticationManager;

    /**
     * @param authenticationManager - inject authenticationManager
     */
    public JWTAuthenticationFilter(AuthenticationManager authenticationManager) {
        this.authenticationManager = authenticationManager;
    }
```

```java
/**
 * @param userDetails - user's details that will be used to generate jwt
 * @return jwt
 */
private String generateJWT(User userDetails) {
    return JWT.create().withSubject(userDetails.getUsername())
            .withExpiresAt(new Date(System.currentTimeMillis() + Constants.EXPIRATION_TIME))
            .sign(Algorithm.HMAC512(Constants.SECRET.getBytes()));
}
```

Figure 119. "JWTAuthenticationFilter" file

Following that, whenever users send credentials to the server, user’s information will be checked by “JWTAuthenticationFilter” file. If user’s credentials are valid, JWT token will be generated and send back to the user so that the user can use that JWT for the next requests.

```java
public class JWTAuthorizationFilter extends BasicAuthenticationFilter {
    private UserDetailsServiceImpl userDetailsService;
    private FacebookAccountService facebookAccountService;
    private UserRoleService userRoleService;
    private GoogleAccountService googleAccountService;

    /**
     * @param authManager            - inject authManager
     * @param userDetailsService     - inject userDetailsService
     * @param facebookAccountService - inject facebookAccountService
     * @param userRoleService        - inject userRoleService
     * @param googleAccountService   - inject googleAccountService
     */
    public JWTAuthorizationFilter(AuthenticationManager authManager, UserDetailsServiceImpl userDetailsService,
                                  FacebookAccountService facebookAccountService, UserRoleService userRoleService,
                                  GoogleAccountService googleAccountService) {
        super(authManager);
        this.userDetailsService = userDetailsService;
        this.facebookAccountService = facebookAccountService;
        this.userRoleService = userRoleService;
        this.googleAccountService = googleAccountService;
    }
```

Figure 120. "JWTAuthorizationFilter" file

The last but not least, after the user receive JWT from the backed service, that token will be add to header whenever the user send request to the backend service and then, the server will get that token and verify user's information to make sure that the user is allowed to access that backend service, or not. All of these things will be handled by "JWTAuthorizationFilter" file as readers could see at the image above.

3. Payment

The main purpose of the online fitness system is to connect coaches and users across the world. It means that users will need a way to pay for coaches. For this reason, the author decided to integrate PayPal to the system and this section will discuss about how to integrate PayPal to the backend of the system.

```xml
<dependency>
    <groupId>com.paypal.sdk</groupId>
    <artifactId>rest-api-sdk</artifactId>
    <version>1.13.1</version>
</dependency>
```

Figure 121. Integrate PayPal dependency

Firstly, the author need to add required dependency to use PayPal in Spring Boot framework. PayPal support different technologies and in this case, PayPal was integrated to the Spring Boot project.

```java
// paypal client
public static final String PAYPAL_CLIENT_ID = "AZO0eJ86SfloUJjP5C_B9uzZaTsiX_VJVtTijSGaEVr16WYGFpS-YdIEvH2oYDo1hk_mPJ79jlVcMT0K";

// paypal secret
public static final String PAYPAL_SECRET = "EMgFSH4yBhc74CRcSZ0yWvznsNYN6x-BZfsqPLE_rMSAoPVLxKYQFOT36R-9ghBV9HzUIfI1B14OsEk4";
```

Figure 122. PayPal Client and PayPal Id

Following that, the developer need to login to PayPal Developer page, and get PayPal client id and PayPal secret key, the image above determined that, the author store two values in two constant variables because readers could see that these values will not be changed.

```java
@Component
public class PaypalClient {

    public PaypalClient() {

    }

    /**
     * @param sum - total payment
     * @return status and redirect url
     */
    public Map<String, Object> createPayment(String sum) {
        Map<String, Object> response = new HashMap<>();
        Amount amount = new Amount();
        amount.setCurrency("USD");
        amount.setTotal(sum);
        Transaction transaction = new Transaction();
        transaction.setAmount(amount);
        List<Transaction> transactions = new ArrayList<>();
        transactions.add(transaction);

        Payer payer = new Payer();
        payer.setPaymentMethod("paypal");
```

Figure 123. "PaypalClient" file

Aside that, the author created “PaypalClient” file, this file will get PayPal client id and PayPal secret id to make payment and complete payment, readers could understand the role if this file is to handle business logic of online payment in the system.

```java
@Controller
@RequestMapping("paypal-management")
public class PaypalController {
    /**
     * payPlalClient - payPalClient
     */
    private PaypalClient payPalClient;

    /**
     * @param payPalClient - inject payPalClient
     */
    public PaypalController(PaypalClient payPalClient) { this.payPalClient = payPalClient; }

    /**
     * @param sum - sum
     * @return created payment
     */
    @PostMapping(value = "/paypal/make/payment", produces = {MediaType.APPLICATION_JSON_VALUE})
    @ResponseBody
    public Map<String, Object> makePayment(@RequestParam("sum") String sum) { return payPalClient.createPayment(sum); }

    /**
     * @param paymentId - paymentId
     * @param payerId   - payerId
```

Figure 124. "PaypalController" file

Whenever users want to make payment, requests of users will be handled by “PaypalController” file. This file will require “PaypalClient” object as dependency and then handle business logic of online payment. If the payment process was handled successfully, or not, the server will send responses to the users.

4. Chat text and video call

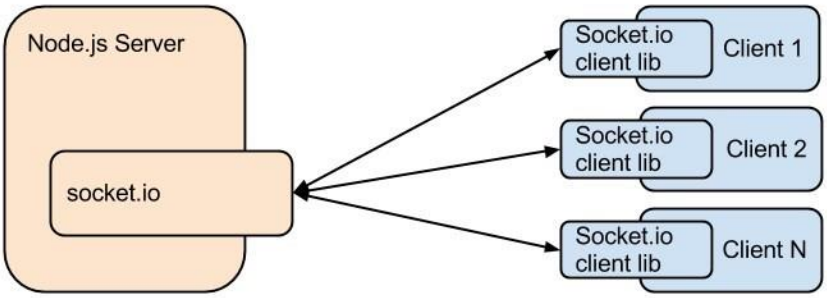


Figure 125. Socket.io server diagram

[Source: https://i2.wp.com/softwareengineeringdaily.com/wp-content/uploads/2016/02/socket-io.jpg?resize=720%2C268&ssl=1]

As what discussed above, in order to achieve real time tasks in the system, the author decided to use NodeJs/ExpressJs framework with supporting of Socket.IO library. Socket.IO enables real-time tasks and can work on every platform, browser and device (Socket.IO, 2019).

```
npm install --save socket.io
```

Figure 126. Install Socket.IO

[Source: https://socket.io/get-started/chat/]

The statement above should be executed on the terminal to install Socket.IO.

```
const server = require('http').Server(app);
const io = require('socket.io')(server);
server.listen(PORT, () => {
  console.log(`App is running on port ${PORT}`);
});
```

Figure 127. Create socket server

After Socket.IO was installed, the author followed the code above to create socket server and now the socket server was created successfully. It is time to write some code to hand business logic of chat text and video call. Firstly, chat text function will be described.

```
io.on('connection', socket => {
  socket.on( event: 'createRoom',  handler: chatRoom => {
    socket.join(chatRoom);
    socket.room = chatRoom;
  });
  socket.on( event: 'sendMessageToRoom',  handler: chatMessage => {
    io.sockets.in(socket.room).emit('serverSendMessageToRoom', chatM
  });
  socket.on( event: 'sendPeerIdToRoom',  handler: peerId => {
    socket.peerId = peerId;
    peerIdArr.push(peerId);
    io.sockets.in(socket.room).emit('serverSendPeerToRoom', peerIdAr
  });
  socket.on( event: 'stopVideoCall',  handler: () => {
    io.sockets.in(socket.room).emit('serverStopVideoCall');
  });
  socket.on( event: 'disconnectServer',  handler: () => {
    peerIdArr.splice(peerIdArr.indexOf(socket.peerId),  deleteCount: 1
    socket.disconnect();
  });
});
```

Figure 128. Business logic of socket server

The code above will perform some tasks including allowing users to connect to the socket server. The socket server will create chat room between user and coach. On the other hand, the socket server will handle event when a user send message to the chat room. Aside that, whenever the user logged out the system, the user will be disconnected to the socket server. Therefore, the system can save resources and reduce cost of cloud server.

As what described in the frontend part, the author used “Peer Js” library to make a video call between two users. Users should know about peer’s id of each other. For this reason, when a cliet connected to the socket server, the socket server will send peer’s id of the client to other users so that users can know about peer’s id of each other and then the client can make a video call to any user that the client wants.

```
@Controller
@RequestMapping("chat-management")
public class ChatMessageController {
    private ChatMessageService chatMessageService;

    /**
     * @param chatMessageService - inject chatMessageService
     */
    public ChatMessageController(ChatMessageService chatMessageService) {
        this.chatMessageService = chatMessageService;
    }

    /**
     * @param chatRoomId - chat's room's id that will be used to get chat's messages
     * @return chat's messages
     */
    @GetMapping(value = "/messages", produces = {MediaType.APPLICATION_JSON_VALUE})
    @ResponseBody
    public List<ChatMessage> getChatMessages(@RequestParam(required = false) Long chatRoomId) {
        return this.chatMessageService.getChatMessages(chatRoomId);
    }

    /**
     * @param chatMessage - chat's message that will be added to the database
     * @return inserted chat's message
     */
    @PostMapping(value = "/messages", produces = {MediaType.APPLICATION_JSON_VALUE})
    @ResponseBody
```

Figure 129. "ChatMessageController" file

The socket server can connect users in real time, but the socket will not insert chat messages of user to the database. Storing chat message in the database will be handled by the Spring Boot project. The developer created “ChatMessageController” file to save all chat messages to the database.

6.2.3 Recommendation System

Recommendation system need to be integrated to recommend coaches to users. This section will demonstrate how the author implement recommendation algorithm in the online fitness system. According to the literature search, collaborative filtering will be applied to the system. There are two kinds of collaborative filter: user-user collaborative filtering and item-item collaborative filtering. Firstly, the author need to create utility matrix

	u0	u1	u2	u3	u4	u5	u6
c0	5	5	2	0	1	?	?
c1	3	?	?	0	?	?	?
c2	?	4	1	?	?	1	2
c3	2	2	3	4	4	?	4
c4	2	0	4	?	?	?	5

Figure 130. Sample of utility matrix

As readers could see at figure above, there are 6 users (from u0 to u6) and 4 coaches (from c0 to c4). Following that, each cell contains value of rating of users to each coach, for example, user u0 rated 5 stars to coach c0 and so on. “?” character is a value which the system has to predict. In this document, sim(ui, uj) is the similarity between user ui and user uj. At figure 1, user u0 and user u1 like c0, c1, c2 and u0, u1 do not like c3 and c4. Therefore, a similarity function is good if the function could ensure:

sim(u0, u1) > sim(u0, ui) with i > 1.

In this scenario, the question is what is the good similarity function and in order to find similarity between two users, you need to create feature vector for each user and apply a function which has ability to find similarity between that vectors. That vectors should be created based on utility matrix. However, the problem is there are so many missing ratings in the utility matrix, so the system need to find temporary values for missing ratings and that values should not effect to similarity between vectors. The values are used to find similarity between users and would not be used to predict the final results. The question is which value should be used to replace “?” in the utility matrix. There is an option that should be considered is 0, 0 is not really good because 0 is the lowest score that a user could rate to a coach and a better option is 2.5 because 2.5 is an average value between 0 (lowest rating value) and 5 (highest rating value). However, 2.5 is not compatible with easy-going users and fastidious users because easy-going users may want to rate 3, 4 or 5 points to a coach and vice versa

with fastidious users. For this reason, in this situation, the best option is average of values which a user has rated. For more information, you could see figure below.

	u0	u1	u2	u3	u4	u5	u6
c0	5	5	2	0	1	?	?
c1	3	?	?	0	?	?	?
c2	?	4	1	?	?	1	2
c3	2	2	3	4	4	?	4
c4	2	0	4	?	?	?	5
	3.25	2.75	2.5	1.35	2.5	1.5	3.33

Figure 131. Find average rating values of each user

The last row at figure above is the average value of all rating values of each column in the utility matrix. As you could see above, easy-going user has high average value and vice versa. If rating values of each user was got to minus to the average rating values of that user and "?" was replaced by 0 value, normalized utility matrix would be created as you could see at figure below

	u0	u1	u2	u3	u4	u5	u6
c0	1.75	2.2.5	-0.5	-1.33	-1.5	0	0
c1	0.75	0	0	-1.33	0	0.5	0
c2	0	1.25	-1.5	0	0	-0.5	-2.33
c3	-1.25	-0.75	0.5	2.67	1.5	0	0.67
c4	-1.25	-2.75	1.5	0	0	0	1.67

Figure 132. Normalized utility matrix Y

The importance of normalized utility matrix should be explained. Subtracting the mean of each column results in positive and negative values in each column. Positive values correspond to user like the coaches and vice versa. The value of 0 corresponds to the undefined whether the users like the item or not. Technically, the dimension of the utility matrix is huge with millions of users and coaches, memory is may not enough to store all of these values. On the other hand, the number of preceding ratings is usually a small. For this reason, spare matrix should be used for this situation so memory would be used to store non-zero values and positions of them.

After normalized utility matrix was created, there are some similarity functions would be used such as cosine similarity and person correlation and so on. Cosine similarity would be explained in detail in this document.

$$cosine\_similarity(u1,u2)=cos(u1,u2)=uT1u2/||u1||2.||u2||2$$

As the formula above, u1 and u2 are vectors which correspond to user 1 and user 2 in the normalized utility matrix. The similarity between two vectors in [-1, 1]. The value of 1 determines that two vectors is absolutely similar and vice versa. You could see figure 4 for more information about user similarity matrix

	u0	u1	u2	u3	u4	u5	u6
u0	1	0.83	-0.58	-0.79	-0.82	0.2	-0.38
u1	0.83	1	0	-1.33	0	0.5	0
u2	0	1.25	1	0	0	-0.5	-2.33
u3	-1.25	-0.75	0.5	1	1.5	0	0.67
u4	-1.25	-2.75	1.5	0	1	0	1.67
u5	0.2	-0.23	0.47	-0.29	0	1	0.56
u6	-0.38	-0.71	0.96	0.18	0.16	0.56	1

Figure 133. Similarity matrix

At figure 4, u0 is similar to u1 and u5 because u0 and u1 also cared about c0, c1, c2 and u0, u5 also cared about c1. Following that, u1 is similar to u0 and u2 is similar to u3, u4, u5, u6. There is a point which you should notice. When number of users is huge so similarity matrix is also big. Hence, memory is may not enough to store all of these values.

In order to find the level of interests of a user to an item, neighbor users would be based on. When working with large-scale problems, you could see that K-nearest neighbors (KNN) would be used frequently because of KNN is easier than other solutions. In neighbor-hood collaborative filtering, missing ratings would be specified based on the information of k neighbor users and you just need to focus on users who rated items. Following that, predicted ratings are usually defined as the weighted average of the normalized ratings and the weights are determined based on the similarity between the two users.

The common formula used to predict the rating of user u for item i was:

$$\hat{y}_{i,u}=\sum_{uj\in N(u,i)}\bar{y}_{i,uj}\,sim(u,uj)\Big/\sum_{uj\in N(u,i)}|sim(u,uj)|$$

After applying the formula above, the result would be got as you could see at figure below

	u0	u1	u2	u3	u4	u5	u6
c0	1.75	2.2.5	-0.5	-1.33	-1.5	0.18	-0.63
c1	0.75	0.48	-0.17	-1.33	-1.33	0.5	0.05
c2	0.91	1.25	-1.5	-1.84	-1.78	-0.5	-2.33
c3	-1.25	-0.75	0.5	2.67	1.5	0.59	0.67
c4	-1.25	-2.75	1.5	1.57	1.56	1.59	1.67

Figure 134. Predic values for each user

By way of illustration, you could calculate normalized rating of u1 to i1 with number of nearest neighbors is 2. Firstly, you need to determine users who rated i1 (u0, u3 and u5). Secondly, similarities between u1 and u0, u3, u5 should be determined (0.83, -0.4, -0.23). The two largest values are 0.83 and -0.23 which correspond to u0 and u5. Thirdly, by finding normalized ratings of u0, u5 to i1, you would get 0.75 and 0.5 respectively. Finally, you could predict the result by using the formula above

$$\hat{y}_{i1,u1}=0.83\times 0.75+(-0.23)\times 0.50.83+|-0.23|\approx 0.48$$

If you want to convert rating values to the 5^{th} values, you could add values in each column of the matrix with average rating value of each user. The ways, which the system recommends items to users,

could be different, for example, predicted coaches could be sorted or the system just need to recommend coaches that have positive predicted rating values.

There are some disadvantages of user-user collaborative filtering, for instance, number of users are always larger than number of coaches. For this reason, similarity matrix is usually very huge. As mentioned above, memory is may not enough to store all of the values of the matrix. On the other hand, utility matrix is usually sparse with many columns are often sparse because users do not often rate items. There is another solution for this situation, item-item collaborative filtering could be used. This approach is more efficient for some of reasons, number of items is usually smaller than number of users. Therefore, similarity matrix is also smaller than similarity matrix of user-user collaborative filtering. On the other hand, numbers of known values is equal but number of columns is greater than number of rows. For this reason, in each row, number of known values is greater than number of known values in each column.

The process of finding missing ratings in item-item collaborative filtering is the same as the process of user-user collaborative filtering, so you could see at these figures below for more information.

	u0	u1	u2	u3	u4	u5	u6		
c0	5	5	2	0	1	?	?		2.6
c1	3	?	?	0	?	?	?		2
c2	?	4	1	?	?	1	2		1.75
c3	2	2	3	4	4	?	4		3.17
c4	2	0	4	?	?	?	5		2.75

Figure 135. Utility matrix and average item rating values

	u0	u1	u2	u3	u4	u5	u6
c0	2.4	2.4	-6	-2.6	-1.6	0	0
c1	2	0	0	-2	0	0	0
c2	0	2.25	-0.75	0	0	-0.75	-0.75
c3	-1.17	-1.17	-0.17	0.83	0.83	0	0.83
c4	-0.75	-2.75	1.25	0	0	0	2.25

Figure 136. Normaized utility matrix Y before predicting

	c0	c1	c2	c3	c4
c0	1	0.77	0.49	-0.89	-0.52
c1	0.7	1	0	-0.64	-0.14
c2	0.49	0	1	-0.55	-0.88
c3	-0.89	-0.64	-0.55	1	0.68
c4	-0.52	-0.14	-0.88	0.68	1

Figure 137. Similarity Matrix

	u0	u1	u2	u3	u4	u5	u6
c0	2.4	2.4	-6	-2.6	-1.6	-0.29	-1.52
c1	2	2.4	-0.6	-2	-1.25	0	-0.25
c2	2.4	2.25	-0.75	-2.6	-1.2	-0.75	-0.75
c3	-1.17	-1.17	-0.17	0.83	0.83	0.34	0.83
c4	-0.75	-2.75	1.25	1.03	1.16	0.65	2.25

Figure 138. Normalized utility matrix Y

In fact, item-item collaborative filtering could be got by transposing the matrix of user-user collaborative filtering and you could assume that items are rating users. After getting the final result, you could transpose the matrix again to get the result of user-user collaborative filtering.

7 EVALUATION

This section will evaluate the online fitness system. There are some criteria that will be used to evaluate the system including the design, testing, functionalities and methodology and future work.

7.1 Human Interaction

The online fitness system was designed based on some design principles (Nielsen's 10 heuristics, Norman's 7 principles, Shneiderman's 8 golden rules). For more information about these principles, readers can see at:

https://www.nngroup.com/articles/ten-usability-heuristics/

https://sites.google.com/a/nu.edu.pk/hci-060129/lectures-1/norman-s-7-principles

https://faculty.washington.edu/jtenenbg/courses/360/f04/sessions/schneidermanGoldenRules.html

There are some principles will be applied to evaluate the application.

7.1.1 Provide Feedback

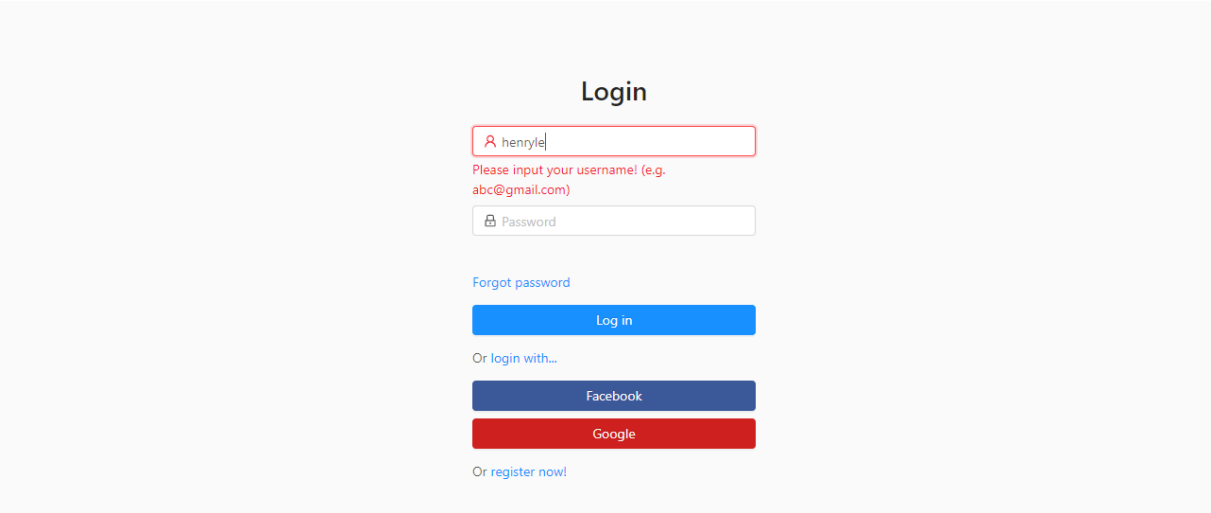


Figure 139. The system will inform users if user's credentials are not correct

In this principle, users need to know what they are dealing with. For example, when user logged in to the system, the system should inform users that users logged in successfully, or not. On the other hand, when users hired coaches, the system should let users know that hiring coaches is successfully, or not.

7.1.2. Consistency

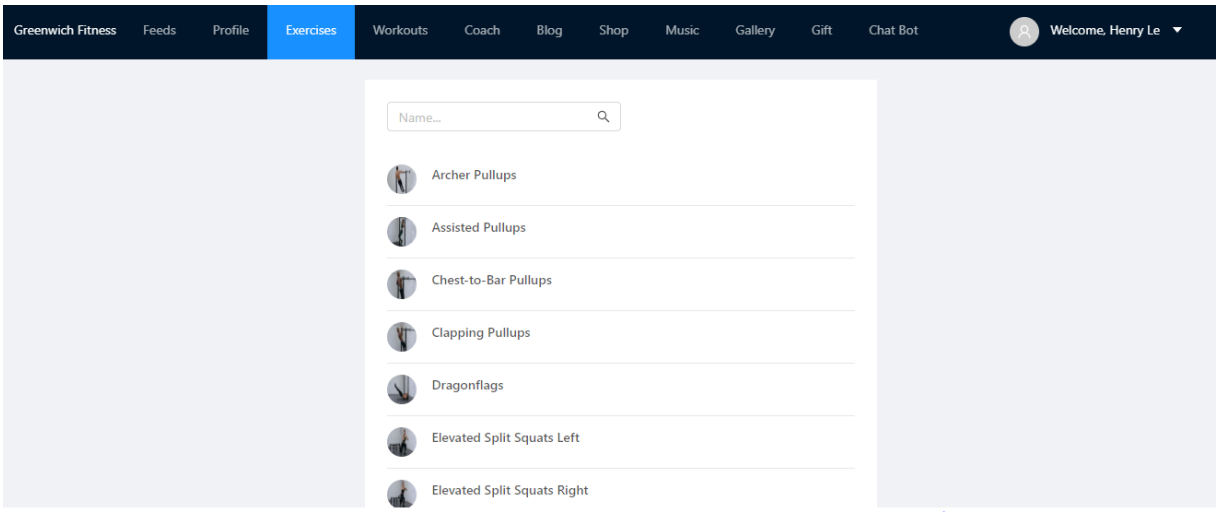


Figure 140. User's interfaces of the system are consistency (colors, fonts, ...)

The online fitness system should be designed with consistency. As mentioned above, frontend of the system was applied by using NgZorro libraries in order to create components of the system. For this reason, user's interfaces of the system are consistency with the same colors, fonts and icons. Readers could see these images below for more information. As readers could see the image above, colors, fonts and user's interfaces are consistency. Each exercises in the list have the same user's interface.

7.1.3. Easy reversal of actions

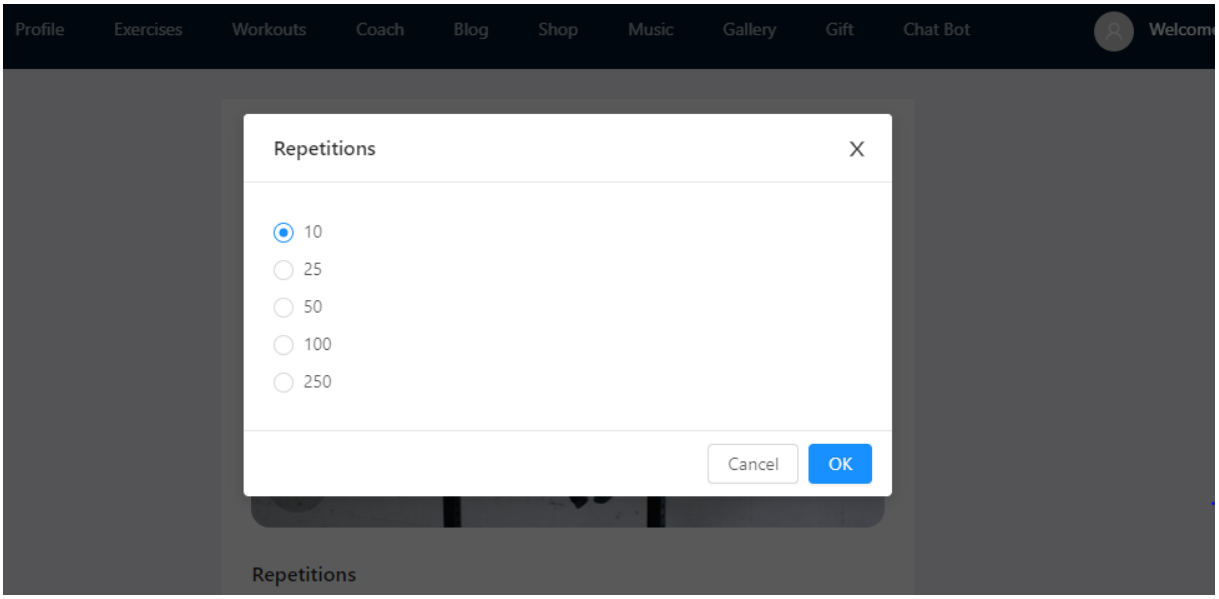


Figure 141. Users can reverse actions as needed

In some cases, users need to reverse previous actions. The online fitness system always shows confirm dialog to ensure that users can return back if users want. The image above demonstrated that users can

change repetitions of any exercise or workout that users want. Following that, if users want to reverse action, users just need to click on “Cancel” button.

7.1.4. Recognize rather than recall

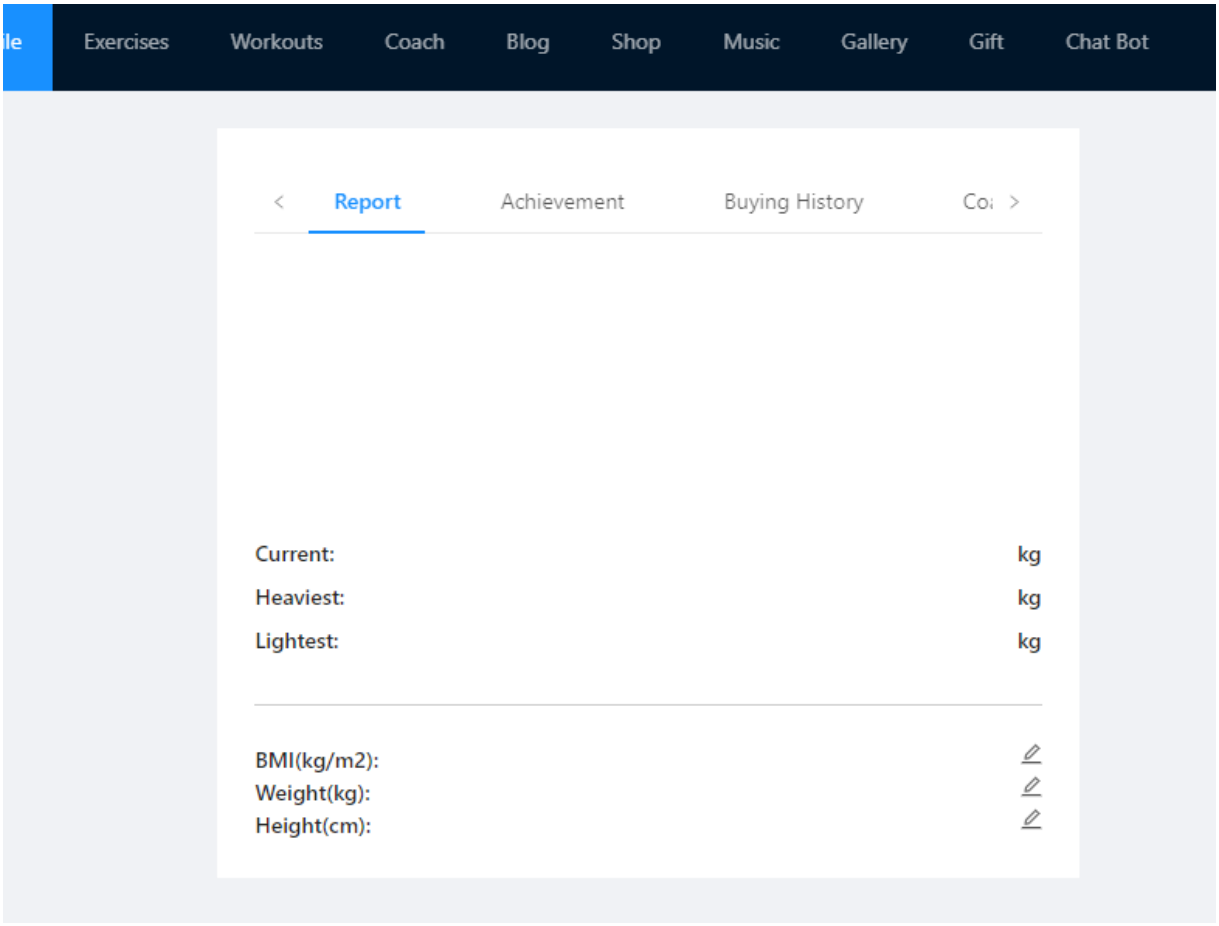


Figure 142. Users can see previous weight and height without remember the result

Users do not need to remember the information from previous part that relating to current action because in some cases, there are so many information that users have to deal with. The image above shown that users can view previous weight, height or body indexes result without remember them.

7.2 Security

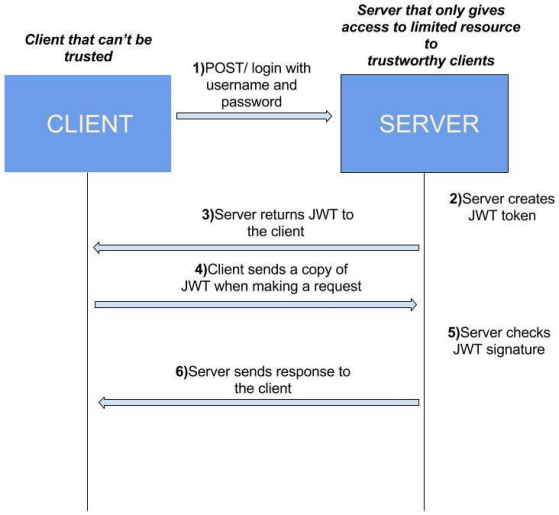


Figure 143. JWT flow diagram

There are some security methods that was applied to the system. For the frontend application, the system use "guard" files to authorize users. On the other hand, Spring Boot framework use Spring security to authenticate and authorize users.

When users logged in the system, the system will check user's credentials. If user's credentials are valid, a token will be created and return back to users and then if users want to send other requests to the backend services, this token will be added to headers, and the token will be verified by the server, if the token is valid, requests of users can access to the backend services.

On the other hand, the author also integrated security to the frontend of the online fitness system. As mentioned above, when users logged in successfully. The server will return roles of users so that when users want to access any route of the frontend. The frontend application will check roles of users to specify that users are allowed to access that route, or not.

The last but not least, in the database of the online fitness system, user's password is encrypted by using SHA algorithm. SHA algorithm is one of the best security encryption algorithm at the moment.

In conclusion, security was considered on both backend and frontend of the online fitness system. For this reason, the system can protect sensitive information of users from hackers. There are more security methods will be considered to make security of the system be better. For example, the developer can integrate captcha or security questions and so on.

7.3 Testing

The developer wants to make sure that the system can work well without any error before deploying the system to production. The author user Agile methodology to manage the development process and each spring will contain a test phase to make sure that functionalities of each spring can work as expected.

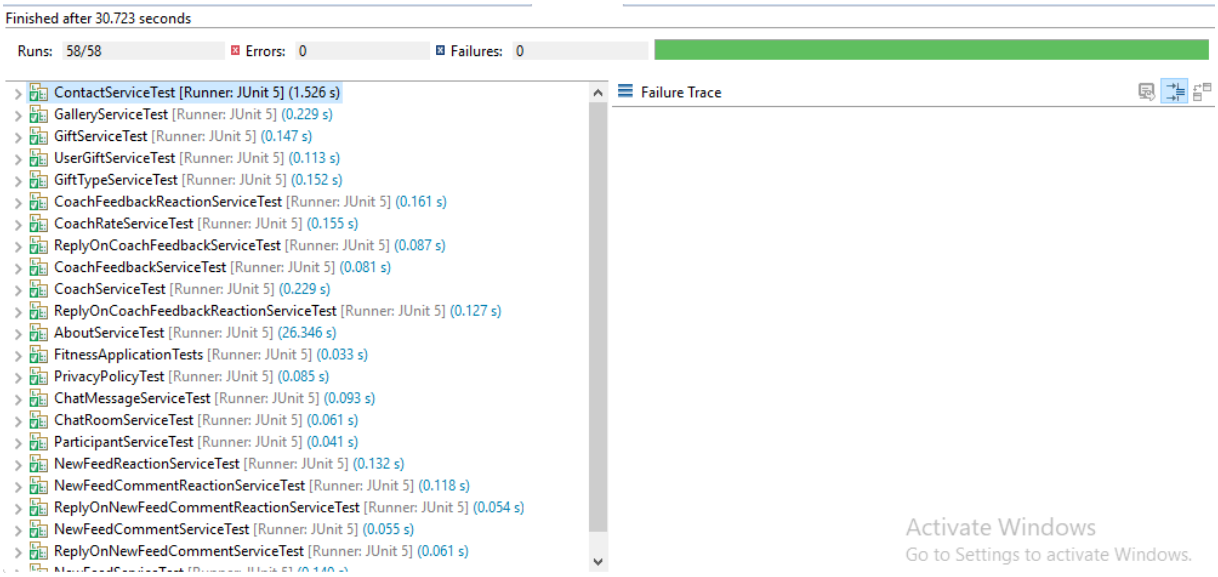


Figure 144. Unit Tests

In the online fitness system, services are the most important part because services contain all business logic of the system. For this reason, the author write unit tests in order to test all services of the system. In order to test all services, the developer used Junit 5 – the newest version of Junit and Mockito framework to mock necessary dependencies for each service.

As readers could see above, 58 test cases were created and they are all passed. More tests cases will be created in the future. For example, the developer can write test cases to test frontend of the application or test controllers and so on.

For more information about the source code of unit tests, readers could access GitHub link at the Appendix A in this document.

7.3.1 Testing of About Service, Contact Service and Privacy Policy Service

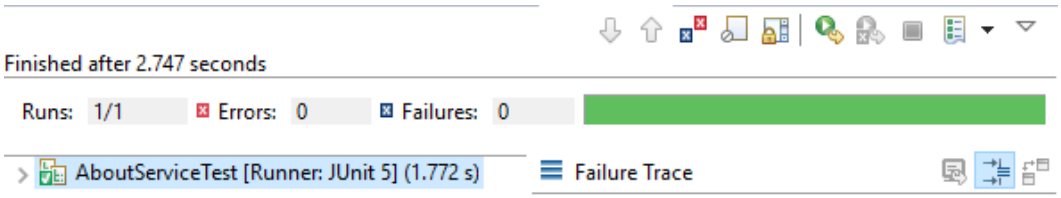


Figure 145. Result of Testing About Service

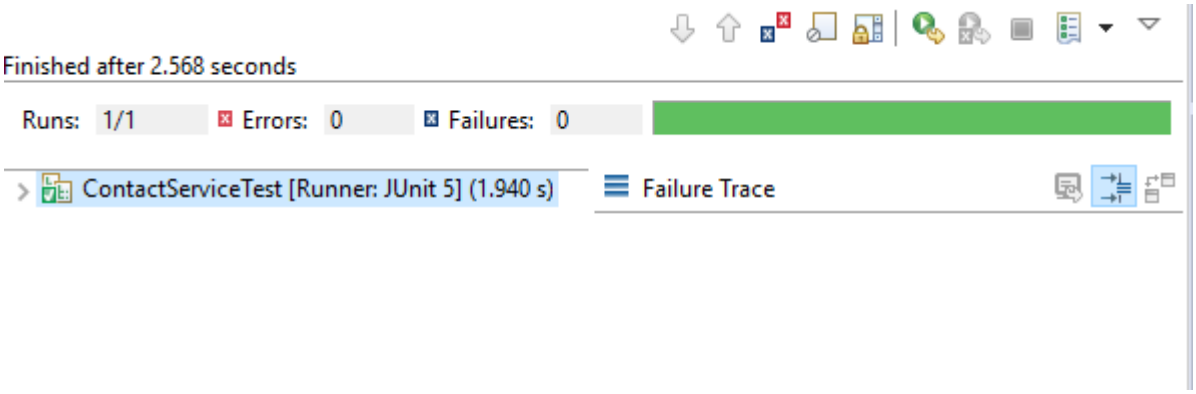


Figure 146. Result of Testing Contact Service

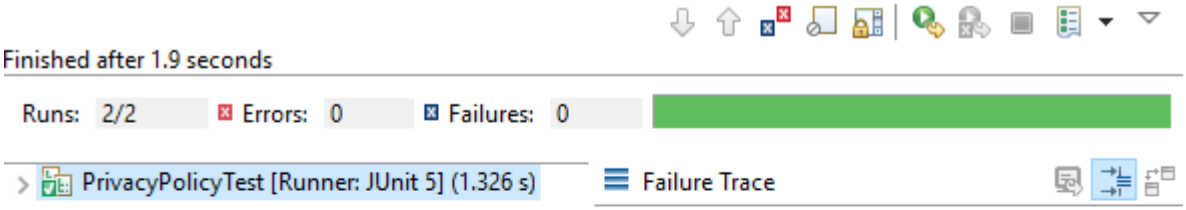


Figure 147. Result of Testing Privacy Policy Service

Service Name	Service Function	Status
About Service	Get About Content By Id	Pass
Contact Service	Get Contact Content By Id	Pass
Privacy Policy Service	Get Privacy Policy By Id	Pass
	Update Privacy Policy	Pass

Table 1. Table of Testing About Service, Contact Service, Privacy Policy Service

The developer tests all functionalities of three services “About” service, “Contact” service, “Privacy Policy” service. All functionalities were passed.

7.3.2 Testing of Chat Feature

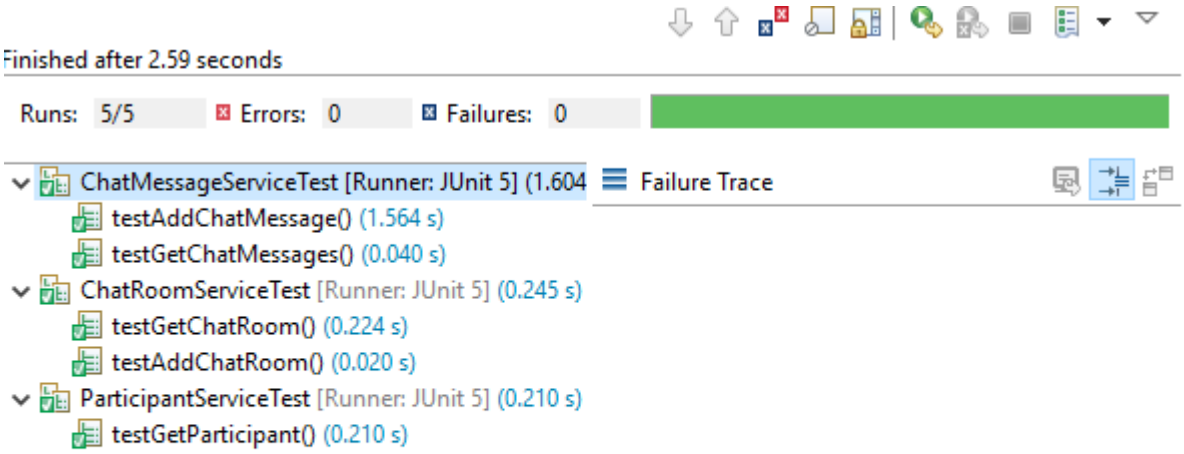


Figure 148. Result of Testing Feature

Service Name	Service Function	Status
Chat Message Service	Add Chat Message	Pass
	Get Chat Messages	Pass
Chat Room Service	Add Chat Room	Pass
	Get Chat Room	Pass
Participant Service	Get Participant	Pass

Table 2. Table of Chat Feature

Chat Feature will handle chatting between two users in the system. The developer tests all functionalities of chat features. All test cases of chat feature were passed.

7.3.3 Testing Coach Feature

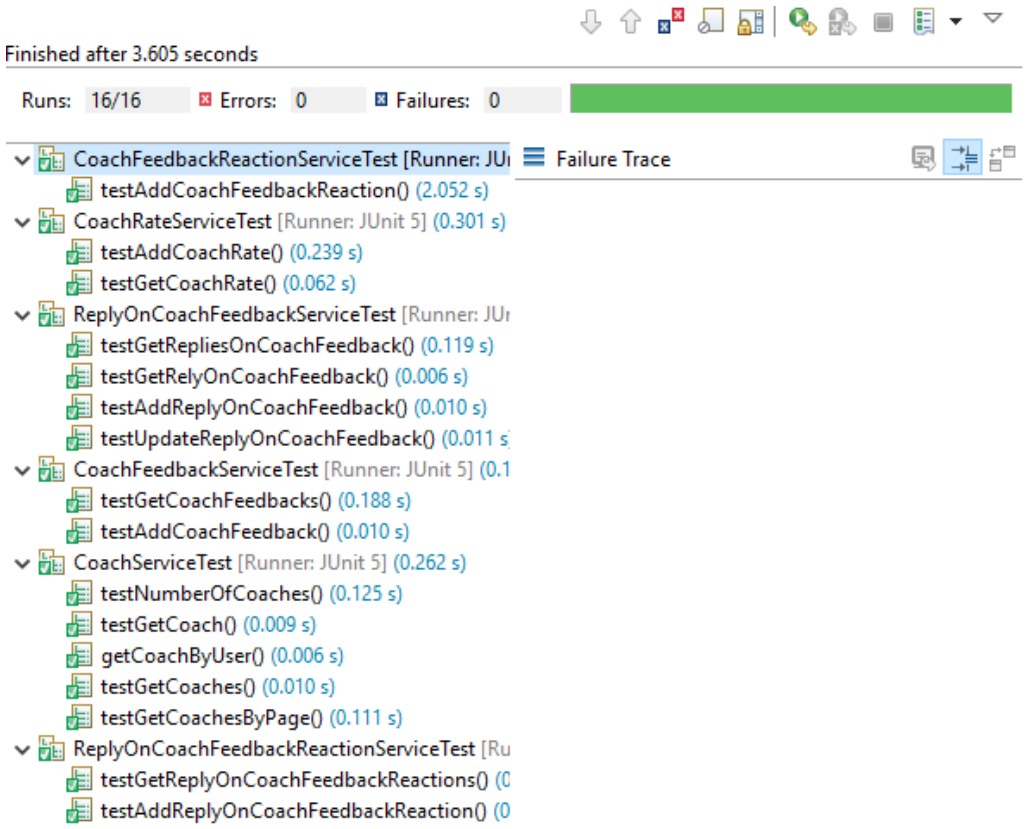


Figure 149. Result of Testing Coach Feature

Service Name	Service Function	Status
Coach Feedback Reaction Service	Add Coach Feedback Reaction	Pass
Coach Rate Service	Add Coach Rate	Pass
	Get Coach Rate	Pass
Reply On Coach Feedback Service	Get Replies On Coach Feedback	Pass
	Get Reply On Coach Feedback	Pass
	Add Reply On Coach Feedback	Pass
	Update Reply On Coach Feedback	Pass
Coach Feedback Service	Get Coach Feedbacks	Pass
	Add Coach Feedback	Pass
Coach Service	Get Number of Coaches	Pass
	Get Coach	Pass
	Get Coach By User	Pass
	Get Coaches	Pass

	Get Coaches By Page	Pass
Reply On Coach Feedback Reaction Service	Get Reply On Coach Feedback Reactions	Pass
	Add Reply On Coach Feedback	Pass

Table 3. Table of Testing Coach Feature

Coach feature will contain all functionalities that relate to coaches in the system. There are 16 functionalities of coach feature and all test cases were passed.

7.3.4 Testing of New Feed

7.3.5 Service

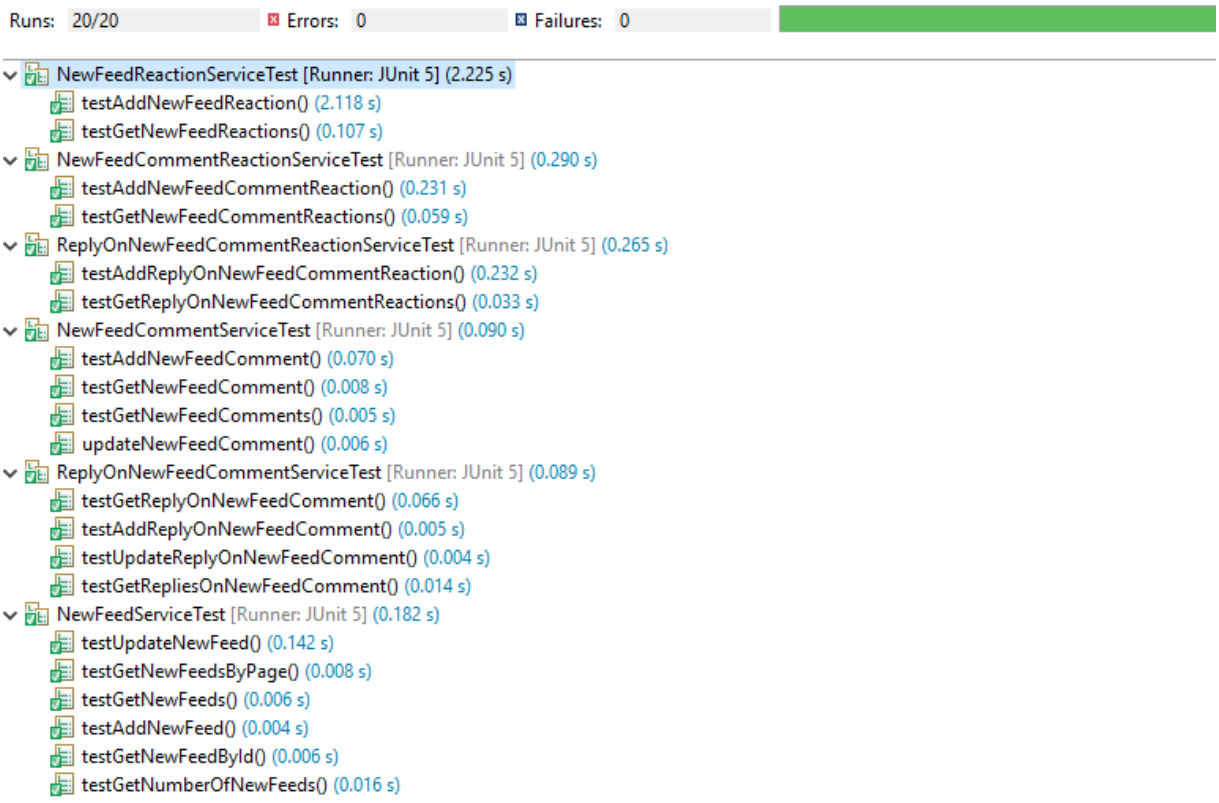


Figure 150. Result of Testing New Feed Feature

Service Name	Service Function	Status
New Feed Reaction Service	Add New Feed Reaction	Pass
	Get New Feed Reactions	Pass
New Feed Comment Reaction Service	Add New Feed Comment Reaction	Pass
	Get New Feed Comment Reactions	Pass
Reply On New Feed Comment	Add Reply On New Feed	Pass

Reaction Service	Comment Reaction	
	Get Reply On New Feed Comment Reactions	Pass
New Feed Comment Service	Add New Feed Comment	Pass
	Get New Feed Comment	Pass
	Get New Feed Comments	Pass
	Add New Feed Comment	Pass
Reply On New Feed Comment Service	Get Replies On New Feed Comment	Pass
	Get Reply On New Feed Comment	Pass
	Add Reply On New Feed Comment	Pass
	Update Reply On New Feed Comment	Pass
New Feed Service	Update New Feed	Pass
	Get New Feeds By Page	Pass
	Get New Feeds	Pass
	Get New Feed By Id	Pass
	Get Number of New Feeds	Pass

Table 4. Result of Testing New Feed Feature

The New Feed feature will allow users to upload their status and image after users finished the exercise or workout. For this reason, the online fitness system will connect all users in the system. In order to make sure the new feed feature can work well in production. The developer tested all services of new feed feature and all tests case were passed.

For more information about other test cases, readers can view the source code at the Appendix A in this document.

7.4 Product Review

According to requirement analysis part, all requirements were achieved with great user's interfaces and user's experiences. The developer takes care a lot about user's experiences while users using the online fitness system. There are two kinds of users in the system.

1. **User**

In the online fitness system, users can login and create new account in order to use the system. On the other hand, users can change password or get password if users forgot. The system can support users do exercises and workouts with video tutorial, number of repetitions and so on. User's experiences take an important role in every system. For this reason, while users using the fitness system, users can read blogs about many topics including science, nutrition, training and so on. User can buy fitness products in the system. After reviewing other products, the author understood that most of fitness products do not allow users listen to music while users do the training. Therefore, the developer created a function that allows users listen to music while training so that users can see the system become more friendly and users do not need to use other systems. Following that, users can view galleries and download gallery and set as wallpaper. For this reason, users will not feel motivation to do the training even harder. In fact, the system let users keep trach their body index including weight, height, BMI and so on. The main purpose of the system is to connect coaches and users. In the system, users can hire coach and make payment via PayPal. After that, users can receive training schedule and nutrition plan from their coach.

2. Coach

The second role of users is coach. In fact, a coach can do everything that a normal user can do. On the other hand, coach can manage their revenue and receive payment from their users. The coach has to make training plan and nutrition plan for their users.

3. Machine Learning

In order to increase user's experiences, the author decided to integrate recommendation system and chat bot. Recommendation system was integrated successfully to recommend coaches to users. Therefore, users will know which coaches that users may want to hire. Following that, chat bot was also created so that the fitness online system does not need to hire consulters. Chat bot can have ability to support users 24/24.

4. Conclusion

In conclusion, all requirements were achieved in order to help people increase health and lifestyle. In addition, all requirements based on research of similar products and user's expectations. The project could be considered as success.

7.5 Development methodology review

In order to manage the development process, the author decided to use Agile method. Because of this is graduate project. It means that supervisor's opinions are very important. The supervisor will check the progress of the project after every two weeks. The developer will receive opinions of the supervisor to make the online fitness system better.

By applying Agile method, the author will know priority of each task and can know about which tasks should be delivered first. Each sprint was completed with testing phase and high quality assurance. After each sprint, some functions could be deployed to product. For this reason, the developer can receive feedbacks from users and the supervisor.

In fact, the online fitness system was divided into 7 sprints. For more information about each sprint, readers could see the Appendix B in this document.

Sprint 1: The goal of the first sprint is completing hiring coach progress. Therefore, users can view list of coaches and hire coaches and make payment or their requests. Because of the main purpose of the online fitness system is to connect coaches and users across the world. For this reason, these functions will be delivered in the first sprint.

Sprint 2: The goal of the second sprint is creating login function and create functions that help users increase health and lifestyle. In order to use the system, users will need an account to use the system. For this reason, login function should be considered in this sprint so that the system will know which functions that users are allowed to use. One of the objectives of the system is to help people increase health. Hence, this sprint should deliver some functions in order to achieve the purpose including doing single exercises and workouts, users can manage their profile to keep track body indexes (weight, height, BMI, etc.).

Sprint 3: The goal of the third sprint is to create functions that let users read blogs about many topics including nutrition, science, training. Blog system could be considered as a big module in the system so that the developer just spends time and effort to create blog system in this sprint.

Sprint 4: The goal of the fourth sprint is to create functions that let users buy fitness products. Ecommerce system could be considered as a big module in the system so that the developer just spends time and effort to create ecommerce system in this sprint.

Sprint 5: The goals of the fifth sprint is to create functionalities that increase user's experiences in the system. For example, users can listen to music while doing the exercises or users can view galleries and download any gallery and set it as wallpaper to make users feel motivated to do the training even harder. Aside that, user can view notifications from the system or other users. Users can also write new feeds and like/dislike, give comments for any new feed that users want. All functionalities have the same purpose is to increase user's experiences in the system. For this reason, these functions will be put in this sprint.

Sprint 6: People are living in 4.0 industry with growing quickly of artificial intelligence, big data and internet of things. Hence, the author wants to integrate recommendation system and chat bot in the online fitness system so that the goal is this sprint is to apply machine learning in the system. Therefore, the system can suggest coaches to users and chat bot can support users 24/24.
Sprint 7: The goal of this sprint is to provide some communication methods between coaches and users such as chat function and making video call function.

In conclusion, all requirements of the online fitness system were achieved. Product backlog contains proper tasks with appropriate priority. Each sprint will contain corresponding tasks to achieve sprint's goal. On the other hand, each sprint will have testing phase to test all functions of the sprint before deploying to production. The last but not least, all sprint are finished before the deadline with the guidance and helping of the supervisor – PhD. Doan Trung Tung.

7.6 Future Work

As mentioned above, all requirements are fully achieved. However, in order to make the online fitness system be better in the future. Some future works should be discussed in this section.
Firstly, the developer should research more about recommender algorithms to increase accuracy and performance of the recommendation system.
Secondly, the author should take care more about security because the system allows users to hire coaches via online payment. For this reason, security is very important. Sensitive information of users could be stolen from hackers.
Thirdly, performance of backend services should be considered because of the online fitness system is a graduate project. It means that there are too much users who using the system. Hence, the developer should have a solution when handling multiple requests, for example, million requests or even billion requests at the same time.
Fourthly, the author should let users keep tracks their running. At the moment, the online fitness system just includes exercises and workouts. Providing functions, that help users keep track their running distance, heart beat and so on, should be considered in the future.
The last but not least, the online fitness system should apply computer vision. Hence, when users doing the training, the system can detect user's performance automatically via webcam or camera of user's phone so that when performance of users is not correct, the system will inform users immediately.

8 CONCLUSION

The process of creating the online fitness system including research, design process and development process. There are many lessons could be learnt after finishing the project.

1. Knowledge

According to the literature review, the author has learnt about different technologies and software architecture and choose the most suitable technologies to create the online fitness system. Following the author have a knowledge about different methodology and choose the most suitable method to manage the development process of the system.

2. Development Skills

By doing the project, the author has learnt a lot about Spring Boot framework, MySQL database, Angular framework, how to handle real time tasks and so on. With the helping and guidance from the supervisor – PhD. Doan Trung Tung. The author improved programming skills, code cleaning and management. Following that, the author has ability to plan the project with Agile method and finish tasks before the deadline.

3. Report Writing Skills

The author has learnt a lot about how to write a report. With the guidance of the supervisor, the author knows to outline, structure a report and how to find proper academic sources. These skills is very important for the real life project and write research papers in the future.

9 REFERENCES

[1]. Caspersen, C., Powell , K. & Christenson , G., 1985. Physical activity, exercise, and physical fitness. *Health Rep,* 100(2), p. 126.

[2]. Chris, R., 2019. Escaping monolithic hell. In: M. Marina, M. Christian, Aleksandar Dragosavljevic & W. Lori, eds. *Microservices Patterns with examples in Java.* New York: Manning Publication Co, p. 3.

[3]. Chris, R., 2019. Escaping monolithic hell. In: M. Marina, M. Christian, A. Dragosavljevic & W. Lori, eds. *Microservices Patterns with examples in Java.* New York: Manning Publication Co, p. 4.

[4]. Developers, G., 2019. *Content-based Filtering Advantages & Disadvantages.* [Online]
Available at: https://developers.google.com/machine-learning/recommendation/content-based/summary
[Accessed 4 11 2019].

[5]. Dubey, A., Jain, A. & Mantri, A., 2015. COMPARATIVE STUDY: WATERFALL V/S AGILE MODEL. *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCHTECHNOLOGY,* 4(3), p. 70.

[6]. Francescco Ricci, L. R. B. S. P. B. K., 2011. Introduction. In: L. R. B. S. P. B. K. Francescco Ricci, ed. *Recommender Systems Handbook.* London: Springer, p. 1.

[7]. GitHub, 2019. *angular/angular.* [Online]
Available at: https://github.com/angular/angular
[Accessed 3 11 2019].

[8]. GitHub, 2019. *facebook/react.* [Online]
Available at: https://github.com/facebook/react
[Accessed 3 11 2019].

[9]. Ipek Ensari, E., Brian M. Sandroff, M. & Robert W. Motl, P., 2016. Effects of Single Bouts of Walking Exercise and Yoga on Acute Mood Symptoms in People with Multiple Sclerosis.. *International Journal of MS Care,* 18(1), p. 1.

[10]. JS, P., 2019. *Peer JS.* [Online]
Available at: https://peerjs.com/
[Accessed 3 11 2019].

[11]. Mi-Na Gim, P. M. J.-H. C. P. P., 2016. The effects of weekly exercise time on VO2max and resting metabolic rate in normal adults. *The Journal of Physical Therapy Science,* 28(4), p. 1.

[12]. NgZorro, 2019. *Ant Design of Angular.* [Online]
Available at: https://ng.ant.design/docs/introduce/en#companies-using-ng-zorro-antd
[Accessed 3 11 2019].

[13]. Organization, W. H., 2018. *Obesity and overweight.* [Online]
Available at: https://www.who.int/news-room/fact-sheets/detail/obesity-and-overweight
[Accessed 2 11 2019].

[14]. Richardson, C., 2019. *Microservice Architecture.* [Online]
Available at: https://microservices.io/articles/whoisusingmicroservices.html
[Accessed 22 11 2019].

[15]. Socket.IO, 2019. *Socket.IO.* [Online]
Available at: https://socket.io/#examples
[Accessed 5 11 2019].

[16]. Warburton , D., Nicol , C. & Bredin , S., 2006. Health benefits of physical activity: the evidence. *CMAJ*, 174(6), p. 801.

APPENDIX A - SOURCE CODE

Angular source code: https://github.com/henryle-3112/greenwich-fitness

Backend source code: https://github.com/henryle-3112/fitness

Recommendation system source code: https://github.com/henryle-3112/recommendation-system-fitness

Real-time service source code: https://github.com/henryle-3112/greenwich-fitness-node

Database: https://github.com/henryle-3112/fitness-database

APPENDIX B - PROJECT PROPOSAL

1. Overview

1.1. Introduction

A fitness network system including a web-based system. On the other hand, recommendation system will be integrated in this application. There are some functionalities in this application.

❖Web application:

- User:

- Sign up.
- Login (use user's account or Google account or Facebook account)
- Find coach (the application will base on user's needs and recommend coach to student by using recommendation system).
- Do workouts set.
- Do single exercise.
- Read blogs (e.g. nutrition, tutorial)
- Listen to music (e.g. student could listen music while doing exercises or listen music anytime they want).
- Write status.
- View other student's status.
- Like other student's status.
- Write comment to other user's status.
- View leaderboard

- Follow other users
- Change personal information (such as weight, height, …).
- View personal's level (personal's progress).
- View number of workouts that have been finished.
- View list of workouts that have been finished.
- View number of followers.
- Change settings of the application (audio, video, logout, rate app, feedback app).
- View notifications.
- Chat module
- Rate coach
- Video call

- Coach:

In this application, users, that have coach role, will have all functionalities of normal users as described above. Otherwise, coach can have some functionalities that normal users do not have.

- Each coach will have points that will be sent from their students. For this reasons, coach can convert points to cash or other gifts such as voucher and so on.

1.2. Why the system need to be created?

According to Statista, revenue in the fitness amounts to US$16,857m in 2019. Following that, Revenue is expected to show an annual growth rate (CAGR 2019-2023) of 5.0%, resulting in a market volume of US$20,499m by 2023. The market's largest segment is Wearables with a market volume of US$14,528m in 2019. In global comparison, most revenue is generated in China (US$5,060m in 2019).

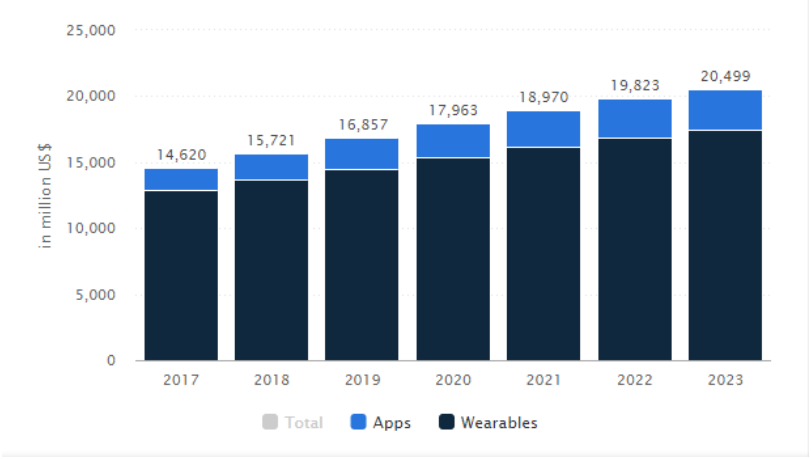


Figure 151. Revenue in the fitness

[Source: https://www.statista.com/outlook/313/100/fitness/worldwide#market-globalRevenue]

For this reason, this is a great time to create a fitness social network system and recommendation system will be integrated. Therefore, the system could increase user's experiences and revenue.

1.3. Technologies

There are some technologies will be used to create fitness social network system, for example, RESTful API, recommendation system, mobile application development.

Recommendation systems are software tools and techniques providing suggestions for items to be of use to a user. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read (Francescco Ricci, 2011). Content-based and collaborative filtering are two main types of recommendation systems. Content-based systems evaluate the characteristics of items that are recommended, for example, users view a lot of romantic movies, so the system needs to recommend movies that have the same romantic characteristics. However, items should be arranged into groups, but there are items that do not have specific groups. Aside that, determining the group or characteristics of each item is sometimes impossible. Collaborative filtering suggest items based on the similarity between users or items. In this approach, items were recommendation to users based on users with similar behaviours, by way of illustration, there are three users A, B and C and A, B, C liked Adele's songs. On the other hand, B and C also liked Marron5's songs. In addition, the system does not have any information about A likes Marron5's songs or not, but the system could predict that A would like Marron5's songs because of similar behaviours of B and C. The reasons of choosing recommendation systems in this application are recommendation could make the application become more intelligence and could increase revenue, user's experiences. Following that, there are some famous organizations that are using recommendation in their applications such as Google, Netflix, Facebook and so on.

The second technology, which will be used, is RESTful API. As mentioned above, a fitness social network mobile application will be created. In fact, mobile applications cannot interact directly with the database server. For this reason, RESTful API need to be used so the fitness social network mobile application can post data to the server or get data from the server via RESTful API. On the other hand, machine learning models of recommendation system could be deployed to the server via RESTful API. For this reason, RESTful API takes an important role in the system.

2. Aim

The project will include investigation in recommendation systems, RESTful API and mobile application development technologies and evaluation about the suitable recommendation system, RESTful API and mobile application technology. Following that, the project also include investigation about fitness domain and the suitable fitness domain for the system. For this reason, the fitness social network system will be created including a mobile application, an admin page and a recommendation system.

3. Objectives

I. Objective 1. Research about recommendation systems

Activities:

1. Read papers/books about recommendation systems
2. Research about algorithms for recommendation systems
3. Select an appropriate algorithm for recommendation systems

Deliverables:

1. A report (500 words) about recommendation systems
2. A report (500 words) about algorithms for recommendation systems
3. A report about (100 words) about selecting algorithm

II. Objective 2. Research about RESTful API

Activities:

1. Read papers/books about RESTful API
2. Research about how to implement RESTful API
3. Select an appropriate technology to implement RESTful API for the scenario

Deliverables:

1. A report (500 words) about RESTful API
2. A report (100 words) about selecting technology to implement RESTful API for the scenario.

III. Objective 3. Research about web application development

Activities:

1. Read papers/books about web application development
2. Research about different technologies to create web application
3. Select an appropriate technology to create web application for the scenario

Deliverables:

1 A report (500 words) about web application development
2 A report (100 words) about selecting technology to create web application for the scenario

IV. Objective 4. Gather requirements for the system (Research about domain)

Activities:

1. Research similar applications
2. Read books/papers about the product
3. Interview potential customers/users
4. Create survey

Deliverables:

1. A report (500 words) about the process of gathering requirements for the system
2. Requirement specification documentation

V. Objective 5. Analyse the system

Activities:

1. Create use case diagram
2. Describe use case diagrams

Deliverables:

1. Use case diagrams
2. Use case descriptions

VI. Objective 6. Design the system

Activities:

1. Create Entity Relationship Diagram (ERD)
2. Create Architecture Diagram
3. Create Graphical User Interfaces

Deliverables:

1. Entity Relationship Diagram
2. Class Diagram
3. Pseudo Codes / Flow Charts
4. Graphical User Interfaces

VII. Objective 7. Implement the system

Activities:

1. Select programming languages and frameworks
2. Select database management system
3. Select Integrated Development Environment (IDE)
4. Write Code

Deliverables:

1. A report (500 words) about selecting programming languages and frameworks
2. A report (100 words) about selecting database management system
3. A report (200 words) about selecting IDE
4. The software

VIII. Objective 8. Test the system

Activities:

1. Write test cases
2. Implement test cases
3. Write evaluations about each test case

Deliverables:

1. A list of test cases
2. A list of test results
3. A report (200 words) about test evaluations

IX. Objective 9. Evaluate the system

Activities:

1. Describe about the product built (screenshots and documentation)
2. Analyse strengths and weaknesses of the product
3. Analyse strengths and weaknesses of the selected technologies and algorithms

Deliverables:

1. A report (1000 words) about screenshots and documentation
2. A report (500 words) about strengths and weaknesses of the product
3. A report (500 words) about strengths and weaknesses of the selected technologies and algorithms

X. Objective 10. Conclusion and future works

Activities:

1. Write the conclusion
2. Write the future works

Deliverables:

1. A report (200 words) about the conclusion
2. A report (200 words) about the future works

XI. Objective 11. Write the final report

Activities:

1. Write the draft of the final report
2. Write the final report

Deliverables:

1. The first draft of the final report
2. The final report

4. Legal, Social, Ethical and Professional

The developer wants to create a fitness system. Therefore, there are some legal, social, ethical, professional issues which the project has to face.

1. The application may get some images and data information from other systems. Although, that data would be used for demo purposes but the application could face with copyright issue. There are some solutions for the issue. The developer could get images from free resources or the developer could design application's images. Hence, copyright issue would be solved.
2. On the other hand, the application has some functions such as login with social account (Facebook, Google) or make user send feedback to the application or let user rate the application. Therefore, the application could get some personal information, for example, user's image, user's name, user's email and so on. That information would be used for analyzing purposes.
3. However, data protection, which is one of the most legal issues, should be considered. Hackers could attack the system to stole sensitive information. For this reason, there are some techniques and solutions could be used to protect system's information. The developer could encrypt sensitive information by using some algorithms such as AES algorithm, MD5 algorithm and so on. The mobile application would use REST to get data and post data to the server so the server could use JWT (JSON Web Token) for authentication.

4. Spam should also be considered, by way of illustration, the mobile application has sign up function so user could create new account, some user could create fake account for bad purposes. For this reason, the developer could use email verification. By using this method, when users want to create new account, users must verify in user's emails.
5. Plagiarism is one of the ethical issues which should be considered. The developer may get some information from other systems such as blog's content or product's description and so on. For avoiding this issue, the developer could create new content, the developer should not get content from other systems.
6. On the other hand, delaying deadline is also one of the professional issues. If the developer does not want to delay deadline, the developer should create plan for the entire project. Hence, the author could follow the plan and deadline would not be delayed.

That's all about legal, social, ethical, professional issues which ecommerce mobile application has to face and the solutions for each issue.

5. Planning

ID	As a/an	I want to...	So that...	Priority	Status	ceptance Crite	Added in Sprint	Story Point
1	Coach / User / Administrator	Log in the system	Everyone can interact with the system	4	Done	The syste can check user's credentialr	2	20
2	User	View coaches	View coach's information	1	Done	Users can view list of coaches	1	20
3	User	Hire coach	User can received training schedule from coach.	1	Done	Users can hire any coach that users want	1	40
4	User	Make payment	User can make payment to hire coach and buy products	1	Done	Users can make paymet via PayPal	1	40
5	User	Do Single Exericses or Workouts	User can do the training (If user do not hire any coaches, user can still do single exercises and	2	Done	Users can doing single exercises and workouts without pay fee	2	20
6	User	Chat text or make a video call with user's coaches	User can ask any questions to user's coaches or receive some advice from user's coaches	3	Done	Users can chat text and make video call with their coaches	7	40
7	User	Listen to music	User can listen to music while using the system. For this reason, user may not feel boring. For example, user can do the training and listen to music at the same	4	Done	Users can listen to music while doing other stuffs in the system. For example, users can listen to music while doing the training	5	20
8	User	Read Blogs	User can find useful information about some topics. For example, training, nutrition topic,	5	Done	Users can read blogs, rate selected blog, give comment or like/dislike selected blog, answer replies and like/dislike replies	3	40

9	User	Buy Products	User can buy products that could support user's training.	5	Done	Users can buy view, search products, add products to cart and make payment by paying by cash or PayPal	4	40
10	User	Can view and write newfeeds and like, dislike and write comments for selected newfeed	User can interact with other users in the system	6	Done	Users can upload their status after finishing the exercise and like/dirlike other status. Users can give comments for any	5	30
11	User	View notifications	User can receive notifications from the system or from other users	4	Done	Users can view notifications from the system or from other users	5	20
12	User	View Galleries	User can view galleries of the system to feel motivated to do the	4	Done	Users can view galleries and daunload any gallery	5	20
13	User	convert points to gift	User can exchange points to gift. Gifts may be vouchers or something useful for users	4	Done	Users can convert points to get gifts (voucher)	5	20
14	User	View and update profile	User can view personal profile, and update information.	4	Done	Users can update and change their profile	2	20
15	Coach	Add and update training schedule for coach's	Coach's membership can receive training's	2	Done	Coach can add training schedule for their users	2	20
16	Coach	View coach's membership's achievements, logs and progress	Coach can base on membership's progress to update training schedule	2	Done	Coach can view progress of their users	2	20
17	User	Be recommended coaches, blogs and products	User can know about coaches, blogs and products that are suitable with user's requirements	7	Done	The system should recommend coaches to different users	6	40
18	User	Talk with chatbot	User can be supported by chatbot automatically	7	Done	Users can talk with chatbot, the chat bot can support users 24/24	6	20

Figure 152. Product Backlog

In order to manage the development process, the author decided to use Agile method. Because of this is graduate project. It means that supervisor's opinions are very important. The supervisor will check the progress of the project after every two weeks. The developer will receive opinions of the supervisor to make the online fitness system better.

By applying Agile method, the author will know priority of each task and can know about which tasks should be delivered first. Each sprint was completed with testing phase and high quality assurance. After each sprint, some functions could be deployed to product. For this reason, the developer can receive feedbacks from users and the supervisor.

In fact, the online fitness system was divided into 7 sprints. For more information about each sprint, readers could see the Appendix B in this document.

Sprint 1: The goal of the first sprint is completing hiring coach progress. Therefore, users can view list of coaches and hire coaches and make payment or their requests. Because of the main purpose of the online fitness system is to connect coaches and users across the world. For this reason, these functions will be delivered in the first sprint.

Sprint 2: The goal of the second sprint is creating login function and create functions that help users increase health and lifestyle. In order to use the system, users will need an account to use the system.

For this reason, login function should be considered in this sprint so that the system will know which functions that users are allowed to use. One of the objectives of the system is to help people increase health. Hence, this sprint should deliver some functions in order to achieve the purpose including doing single exercises and workouts, users can manage their profile to keep track body indexes (weight, height, BMI, etc.).

Sprint 3: The goal of the third sprint is to create functions that let users read blogs about many topics including nutrition, science, training. Blog system could be considered as a big module in the system so that the developer just spends time and effort to create blog system in this sprint.

Sprint 4: The goal of the fourth sprint is to create functions that let users buy fitness products. Ecommerce system could be considered as a big module in the system so that the developer just spends time and effort to create ecommerce system in this sprint.

Sprint 5: The goals of the fifth sprint is to create functionalities that increase user's experiences in the system. For example, users can listen to music while doing the exercises or users can view galleries and download any gallery and set it as wallpaper to make users feel motivated to do the training even harder. Aside that, user can view notifications from the system or other users. Users can also write new feeds and like/dislike, give comments for any new feed that users want. All functionalities have the same purpose is to increase user's experiences in the system. For this reason, these functions will be put in this sprint.

Sprint 6: People are living in 4.0 industry with growing quickly of artificial intelligence, big data and internet of things. Hence, the author wants to integrate recommendation system and chat bot in the online fitness system so that the goal is this sprint is to apply machine learning in the system. Therefore, the system can suggest coaches to users and chat bot can support users 24/24.

Sprint 7: The goal of this sprint is to provide some communication methods between coaches and users such as chat function and making video call function.

In conclusion, all requirements of the online fitness system were achieved. Product backlog contains proper tasks with appropriate priority. Each sprint will contain corresponding tasks to achieve sprint's goal. On the other hand, each sprint will have testing phase to test all functions of the sprint before deploying to production. The last but not least, all sprint are finished before the deadline with the guidance and helping of the supervisor – PhD. Doan Trung Tung.

6. Initial References

1. Book: Pro REST API Development with Node.js, Fernando Doglio published by Apress, 2015, Chapter 2
2. Book: Recommender Systems Handbook, Francesco Ricci, Lior Rokach, Bracha Shapira, Paul B.Kantor published by Springer, 2011, Chapter 4.

APPENDIX C - PROJECT SCHEDULE

ID	As a/an	I want to...	So that...	Priority	Status	ceptance Crite	Added in Sprint	Story Point
1	Coach / User / Administrator	Log in the system	Everyone can interact with the system	4	Done	The syste can check user's credentials	2	20
2	User	View coaches	View coach's information	1	Done	Users can view list of coaches	1	20
3	User	Hire coach	User can received training schedule from coach.	1	Done	Users can hire any coach that users want	1	40
4	User	Make payment	User can make payment to hire coach and buy products	1	Done	Users can make paymet via PayPal	1	40
5	User	Do Single Exericses or Workouts	User can do the training (If user do not hire any coaches, user can still do single exercises and	2	Done	Users can do single exercises and workouts without pay fee	2	20
6	User	Chat text or make a video call with user's coaches	User can ask any questions to user's coaches or receive some advice from user's coaches	3	Done	Users can chat text and make video call with their coaches	7	40
7	User	Listen to music	User can listen to music while using the system. For this reason, user may not feel boring. For example, user can do the training and listen to music at the same	4	Done	Users can listen to music while doing other stuffs in the system. For example, users can listen to music while doing the training	5	20
8	User	Read Blogs	User can find useful information about some topics. For example, training, nutrition topic,	5	Done	Users can read blogs, rate selected blog, give comment or like/dislike selected blog, answer replies and like/dislike replies	3	40
9	User	Buy Products	User can buy products that could support user's training.	5	Done	Users can buy view, search products, add products to cart and make payment by paying by cash or PayPal	4	40
10	User	Can view and write newfeeds and like, dislike and write comments for selected newfeed	User can interact with other users in the system	6	Done	Users can upload their status after finishing the exercise and like/dislike other status. Users can give comments for any	5	30
11	User	View notifications	User can receive notifications from the system or from other users	4	Done	Users can view notifications from the system or from other users	5	20
12	User	View Galleries	User can view galleries of the system to feel motivated to do the	4	Done	Users can view galleries and download any gallery	5	20
13	User	convert points to gift	User can exchange points to gift. Gifts may be vouchers or something useful for users	4	Done	Users can convert points to get gifts (voucher)	5	20
14	User	View and update profile	User can view personal profile, and update information.	4	Done	Users can update and change their profile	2	20
15	Coach	Add and update training schedule for coach's	Coach's membership can receive training's	2	Done	Coach can add training schedule for their users	2	20
16	Coach	View coach's membership's achievements, logs and progress	Coach can base on membership's progress to update training schedule	2	Done	Coach can view progress of their users	2	20
17	User	Be recommended coaches, blogs and products	User can know about coaches, blogs and products that are suitable with user's requirements	7	Done	The system should recommend coaches to different users	6	40

Figure 153. Product backlog

Sprint Goal	Complete hiring coach progress. Therefore, user can view list of coaches, hire coaches and make payment for their requests											
roduct Backlog Ite	Task	Volunteer	Status	riginal Estimat	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Sprint Review
User can view coaches's information	Create user table in the database	HiepIt	Done	2	0	0	0	0	0	0	0	0
	Create coach table in the database		Done	1	0	0	0	0	0	0	0	0
	Write api to manage user's information		Done	2	0	0	0	0	0	0	0	0
	Write api to manage coach's information		Done	2	2	0	0	0	0	0	0	0
	Create frontend to show list of coaches		Done	3	3	0	0	0	0	0	0	0
	Testing		Done	3	3	3	0	0	0	0	0	0
User can hire coach so that user can receive training schedule from coach	Create membership table in the datbase	HiepIt	Done	2	2	2	0	0	0	0	0	0
	Write api to manage membership's		Done	2	2	2	0	0	0	0	0	0
	Create frontend where user can make hire coach request		Done	3	3	3	0	0	0	0	0	0
	Testing		Done	3	3	3	3	0	0	0	0	0
User can hire coach so that user can receive training schedule from coach	Create membership table in the datbase	HiepIt	Done	2	2	2	0	0	0	0	0	0
	Write api to manage membership's		Done	2	2	2	0	0	0	0	0	0
	Create frontend where user can make hire coach request		Done	3	3	3	0	0	0	0	0	0
	Testing		Done	3	3	3	3	0	0	0	0	0
User can make payment, so that, user can hire coach	Integrate payment method in the system (e.g. Paypal)	HiepIt	Done	4	4	4	4	0	0	0	0	0
	Write api to handle payment		Done	3	3	3	3	0	0	0	0	0
	Create front end to handle payment request		Done	3	3	3	3	4	0	0	0	0
	Testing		Done	3	3	3	3	3	0	0	0	0

Figure 154. Sprint 1

Sprint Goal	Create login function and create functions that help users increase healthy and lifestyle											
oduct Backlog It	Task	Volunteer	Status	riginal Estima	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	print Revie
User can login to the system so that user can interact with the system	Write api for login function	HiepIt	Done	2	0	0	0	0	0	0	0	0
	Create frontend of login		Done	3	0	0	0	0	0	0	0	0
	Testing		Done	3	3	0	0	0	0	0	0	0
Do Single Exericses or Workouts so that User can do the training (If user do not hire any coaches, user can still do single	Create data for single exercises and workouts	HiepIt	Done	3	0	0	0	0	0	0	0	0
	Create front end to let users do single exercises or		Done	3	3	0	0	0	0	0	0	0
	Testing		Done	3	3	3	0	0	0	0	0	0
Users can update user profile so that User can view personal profile, and update information.	Write api update user's profile	HiepIt	Done	3	3	0	0	0	0	0	0	0
	Create frontend to let users update personal profile		Done	3	3	3	0	0	0	0	0	0
	Testing		Done	3	3	3	3	0	0	0	0	0

Figure 155. Sprint 2

Sprint Goal	Create functions that let users read blogs about many topics inlcuding nutrition, science and training											
roduct Backlog Ite	Task	Volunteer	Status	ginal Estim	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	print Revie
User can view list of blogs User can find useful information about some topics. For example, training, nutrition topic, science topic	Create database for blog system	HiepIt	Done	10	7	4	1	0	0	0	0	0
	Create api for blog system		Done	10	7	5	3	0	0	0	0	0
	Create front end for blog system		Done	10	8	7	4	1	0	0	0	0
	Testing		Done	3	3	3	3	3	0	0	0	0

Figure 156. Sprint 3

Sprint Goal	Create functions that let user to buy fitness products in the system											
oduct Backlog It	Task	Volunteer	Status	iginal Estim	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	print Revie
User can buy products so that users can buy products that could support user's training.	Create database for ecommerce	HiepIt	Done	10	7	4	1	0	0	0	0	0
	Create api for ecommerce		Done	10	7	5	3	0	0	0	0	0
	Create frontend for ecommer		Done	10	8	7	4	1	0	0	0	0
	Testing		Done	5	3	2	0	0	0	0	0	0

Figure 157. Sprint 4

Sprint Goal	Create functionalities that increase user's experiences in the system											
Product Backlog Item	Task	Volunteer	Status	Original Estimate	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Sprint Review
User can listen to music so that user can listen to music while using the system. For this reason, user may not feel boring. For example, user can do the training and listen to music at the same time	Create music table in the database	HiepIt	Done	3	0	0	0	0	0	0	0	0
	Write backend api		Done	4	0	0	0	0	0	0	0	0
	Create frontend		Done	3	3	0	0	0	0	0	0	0
	Testing		Done	3	3	0	0	0	0	0	0	0
Users can view and write newfeeds and like, dislike and write comments for selected newfeed so that users can interact with other users in the system	Create tables for the function	HiepIt	Done	3	3	0	0	0	0	0	0	0
	Write backend api		Done	3	3	3	0	0	0	0	0	0
	Create front end		Done	4	4	4	0	0	0	0	0	0
	Testing		Done	3	3	3	0	0	0	0	0	0
Users can view notifications so that users can receive notifications from the system or from other users	Create tables for notification feature in the database	HiepIt	Done	4	4	4	4	0	0	0	0	0
	Write backend		Done	3	3	3	3	0	0	0	0	0
	Create frontend		Done	3	3	3	3	0	0	0	0	0
	Testing		Done	3	3	3	3	0	0	0	0	0
Users can view gallleries so that users can view galleries of the system to feel motivated to do the training even harder	Create gallery table in the database	HiepIt	Done	2	2	2	2	2	0	0	0	0
	Write backend		Done	3	3	3	3	3	0	0	0	0
	Create frontend		Done	3	3	3	3	3	0	0	0	0
	Testing		Done	3	3	3	3	3	0	0	0	0

Figure 158. Sprint 5

Sprint Goal	Appply machine learning in the system including recommendation system and chat bot											
Product Backlog Item	Task	Volunteer	Status	iginal Estim	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Sprint Review
Recommend coaches to users so that user can know about coaches, blogs and products that are suitable with user's requirements	Write backend api to apply recommendation model	HiepIt	Done	3	0	0	0	0	0	0	0	0
	Create frontend		Done	4	0	0	0	0	0	0	0	0
Users can talk with the chat bot so that users can be supported by chatbot automatically	Create chat bot api	HiepIt	Done	3	3	0	0	0	0	0	0	0
	Create frontend		Done	4	4	4	0	0	0	0	0	0
	Testing		Done	3	3	3	0	0	0	0	0	0

Figure 159. Sprint 6

Sprint Goal	Providing communication methods between coaches and users											
Product Backlog Item	Task	Volunteer	Status	Original Estimate	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Sprint Review
Users can chat text or make video call so that users can ask any questions to user's coaches or receive some	Write backend api	HiepIt	Done	10	8	5	3	0	0	0	0	0
	Create		Done	10	8	6	2	0	0	0	0	0
	Testing		Done	3	3	3	3	0	0	0	0	0

Figure 160. Sprint 7

APPENDIX D - SCREEN CAPTURES

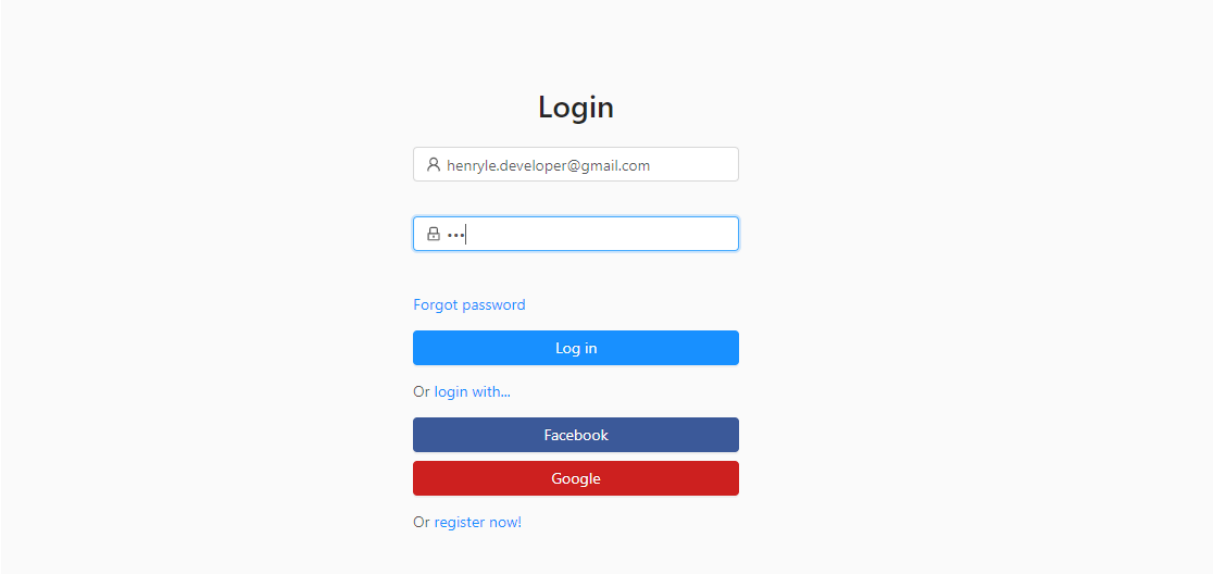


Figure 161. Login Screen

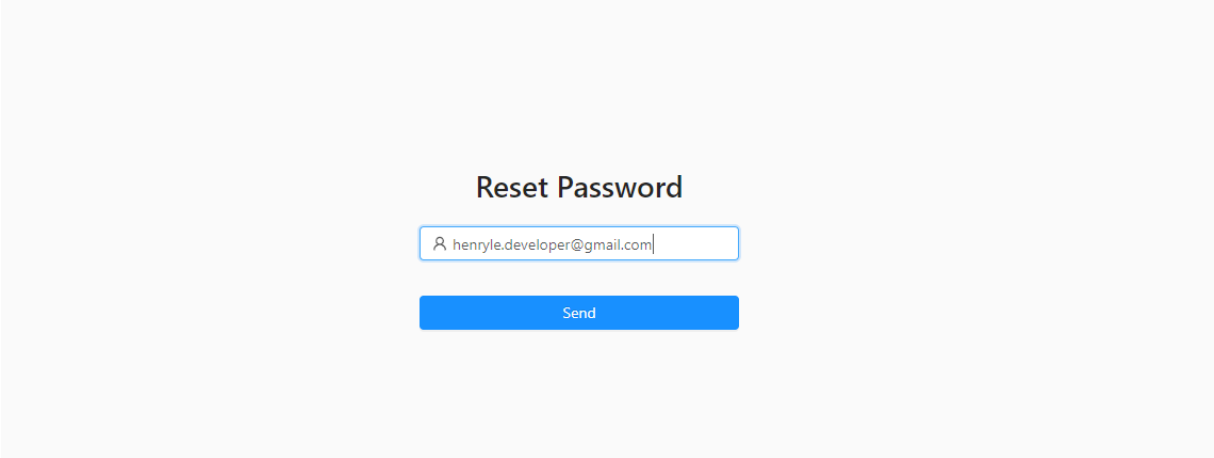


Figure 162. Forgot Password Screen

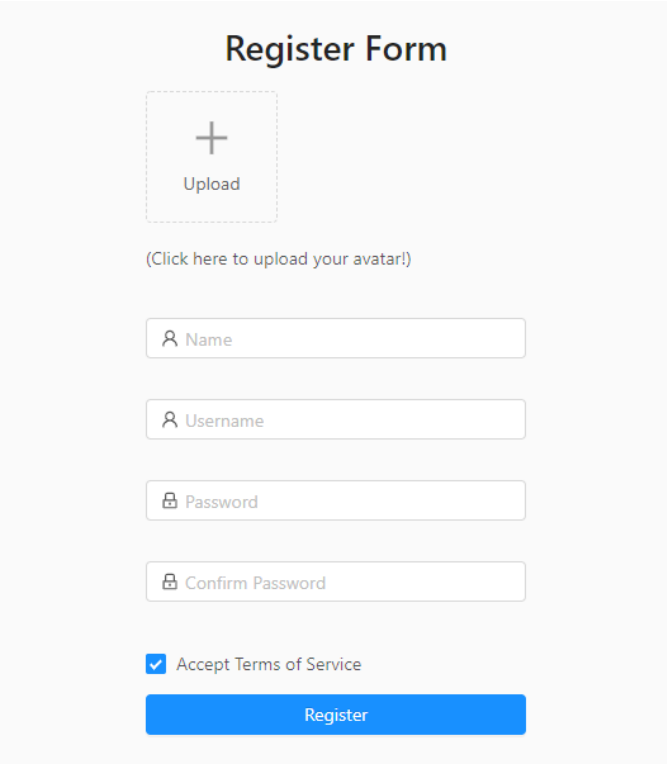


Figure 163. Register Screen

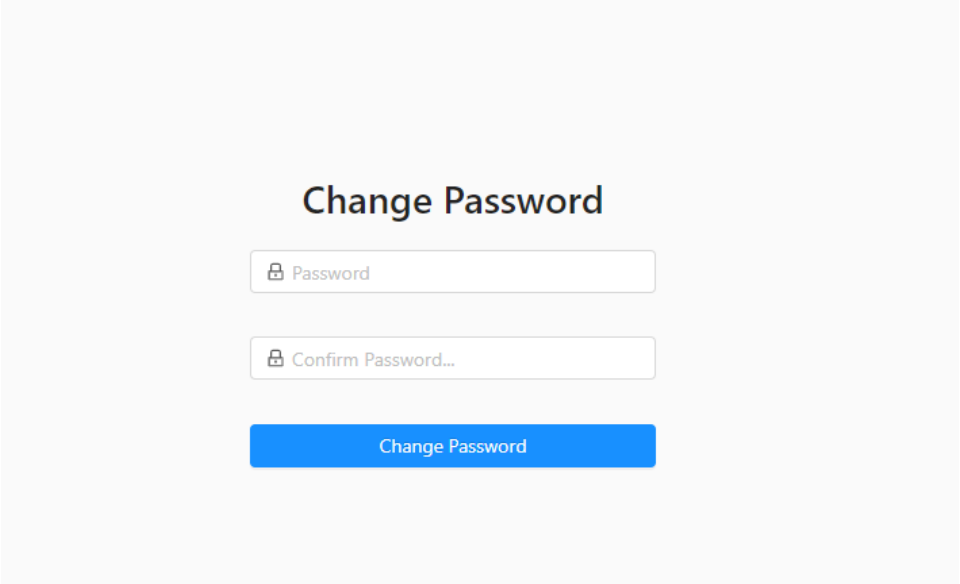


Figure 164. Change Password Screen

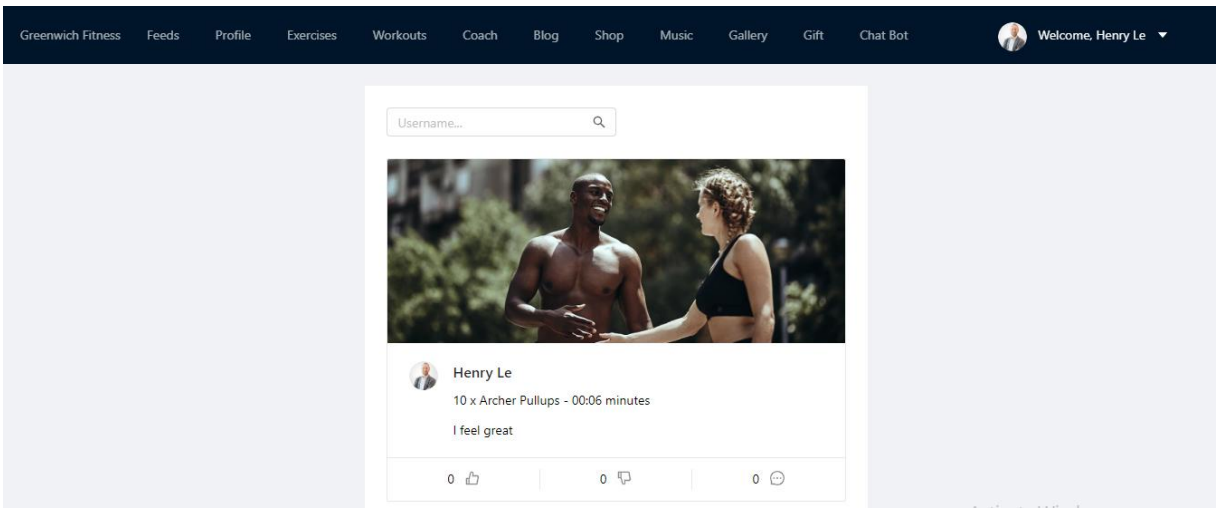


Figure 165. New Feed Screen

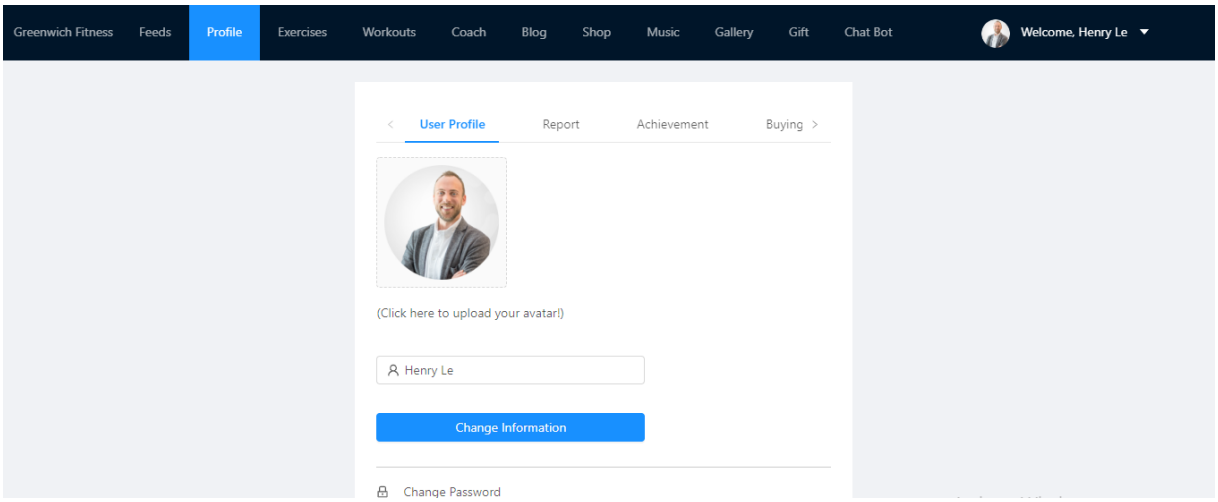


Figure 166. Profile Screen - Tab User Profile

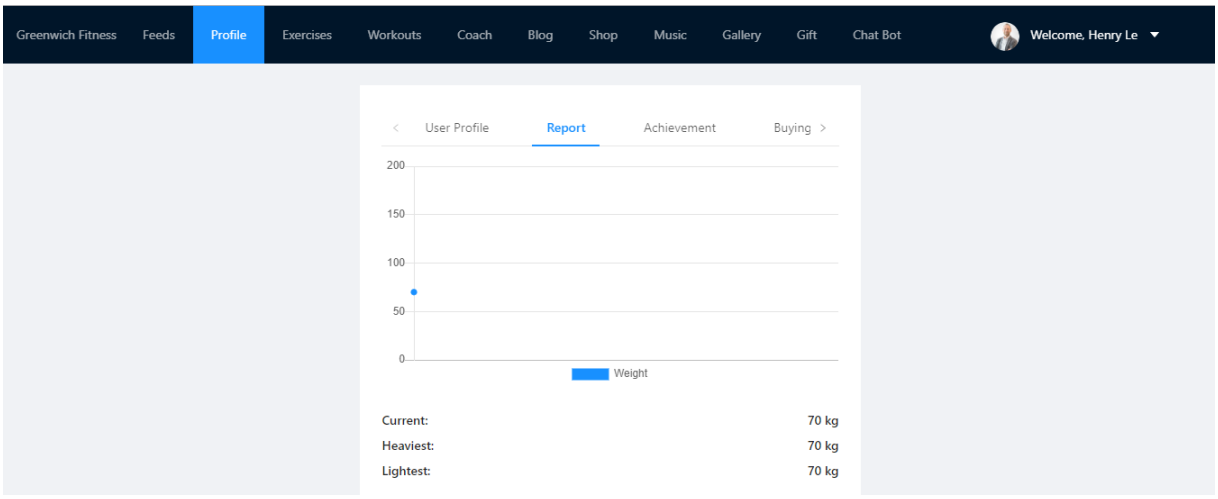


Figure 167. Profile Screen - Tab Report

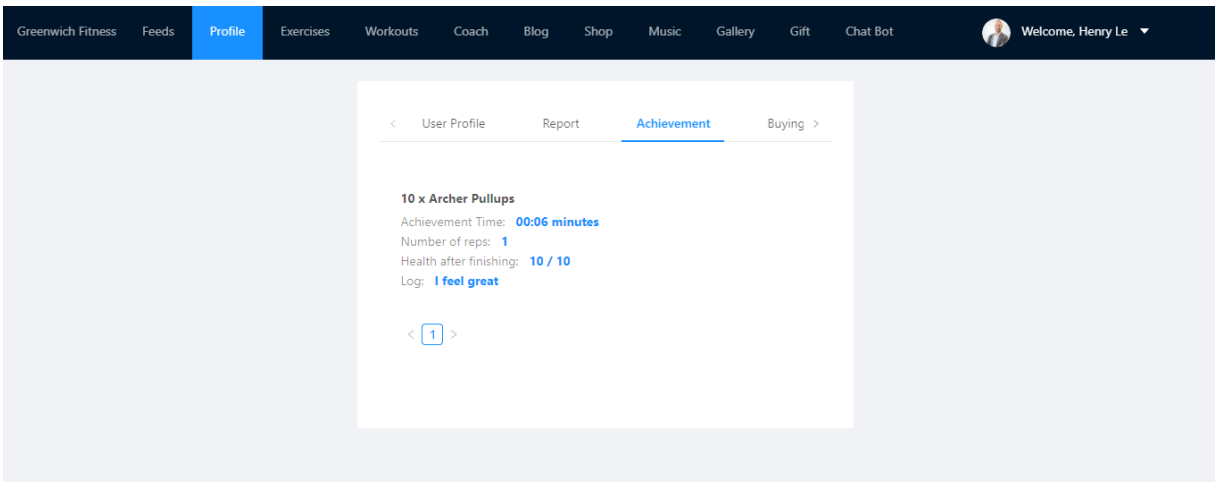


Figure 168. Profile Screen - Tab Achievements

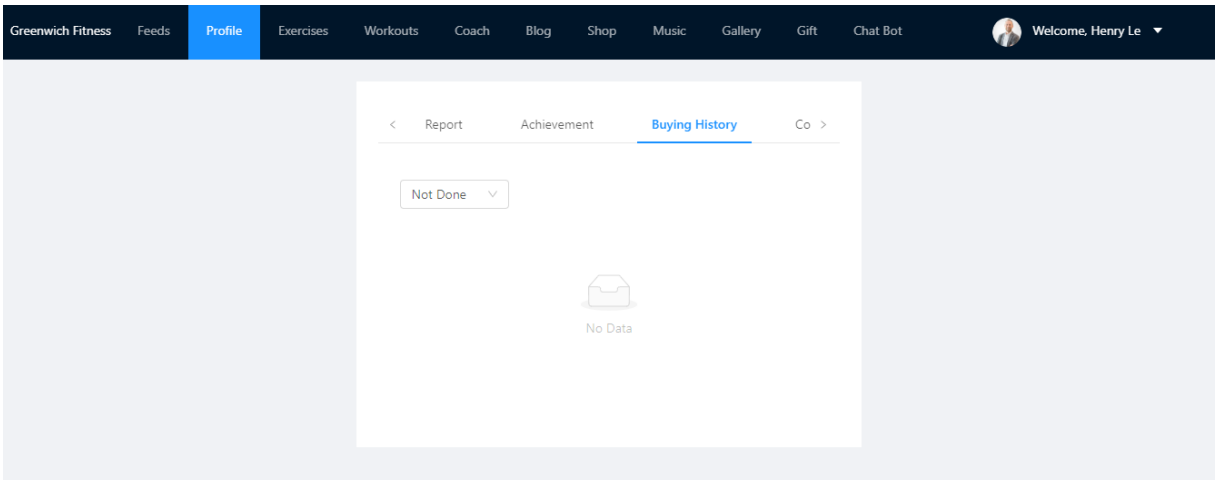


Figure 169. Profile Screen - Tab Buying History

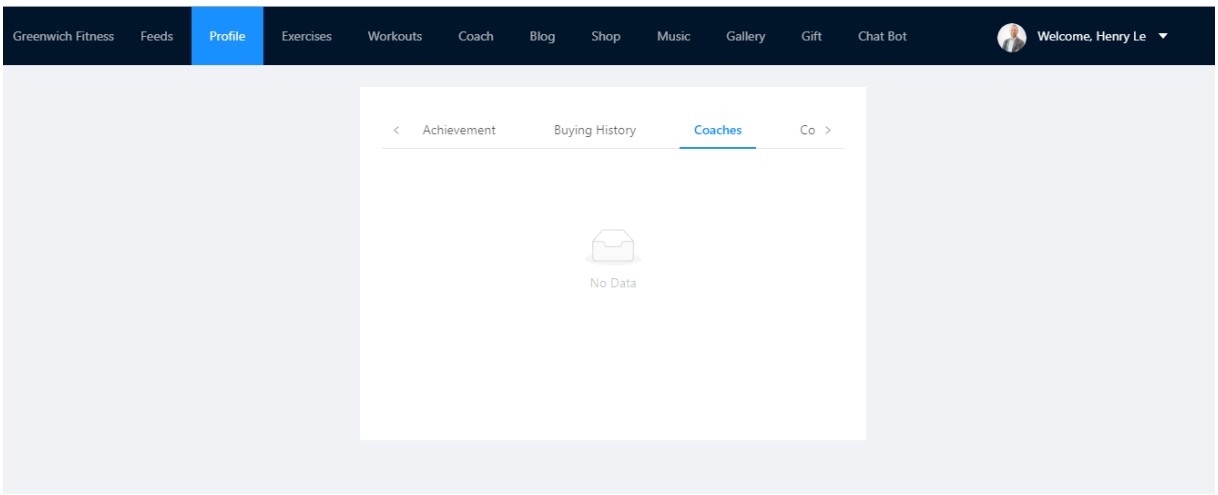


Figure 170. Profile Screen - Tab List of Coaches

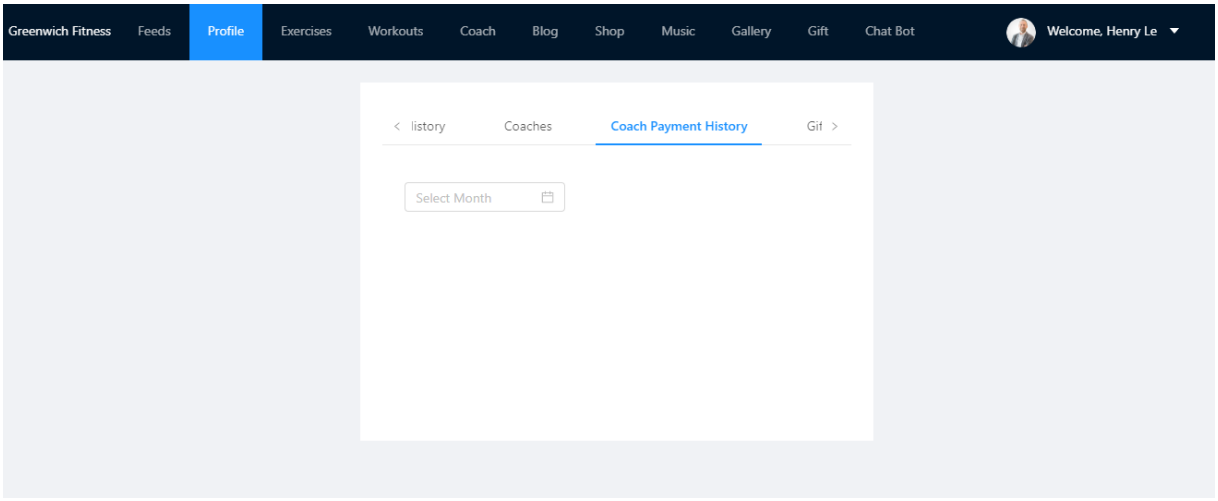


Figure 171. Profile Screen - Tab Coach Payment History

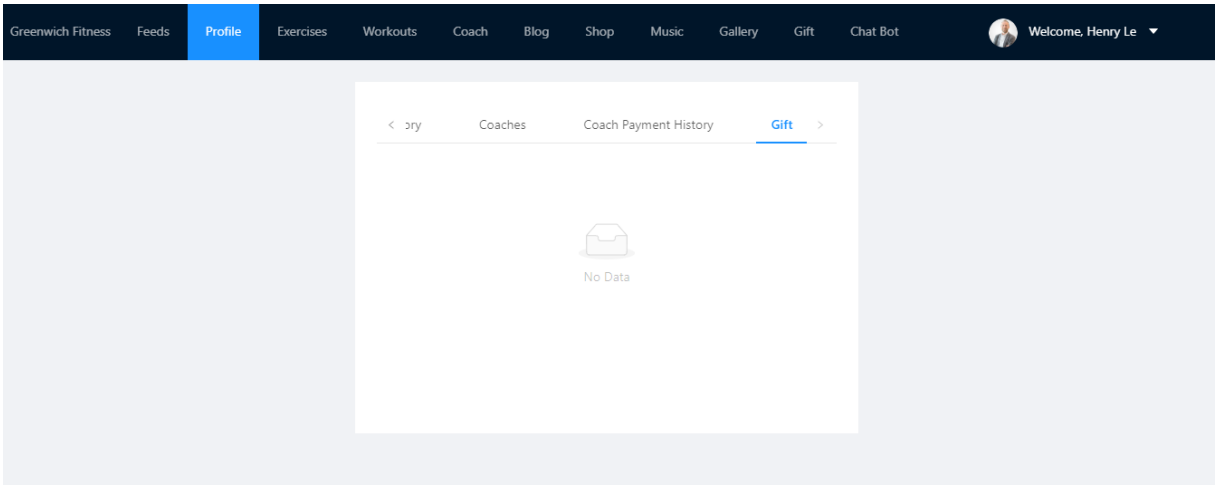


Figure 172. Profile Screen - Tab Gift

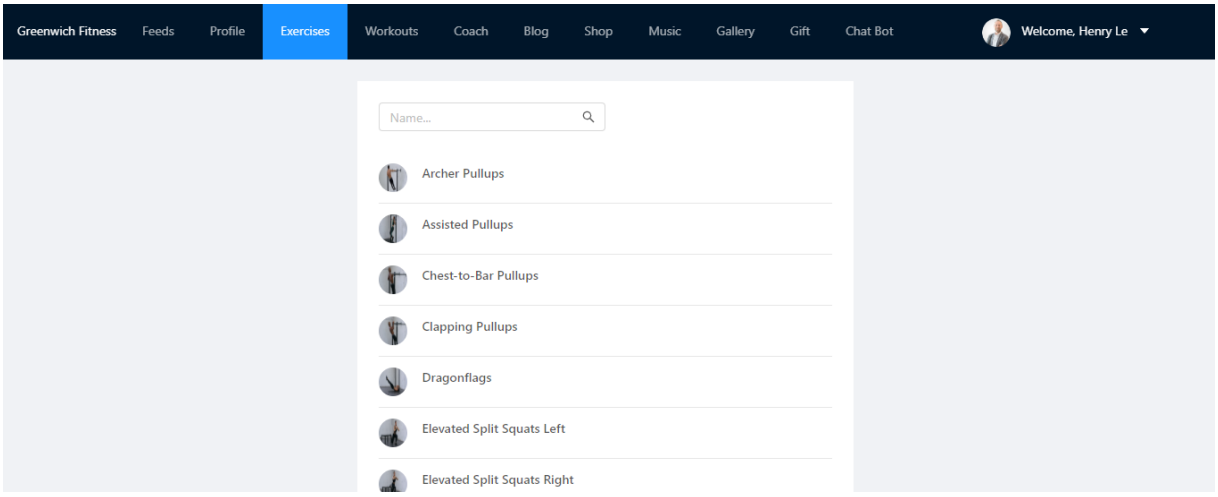


Figure 173. List of Exercises Screen

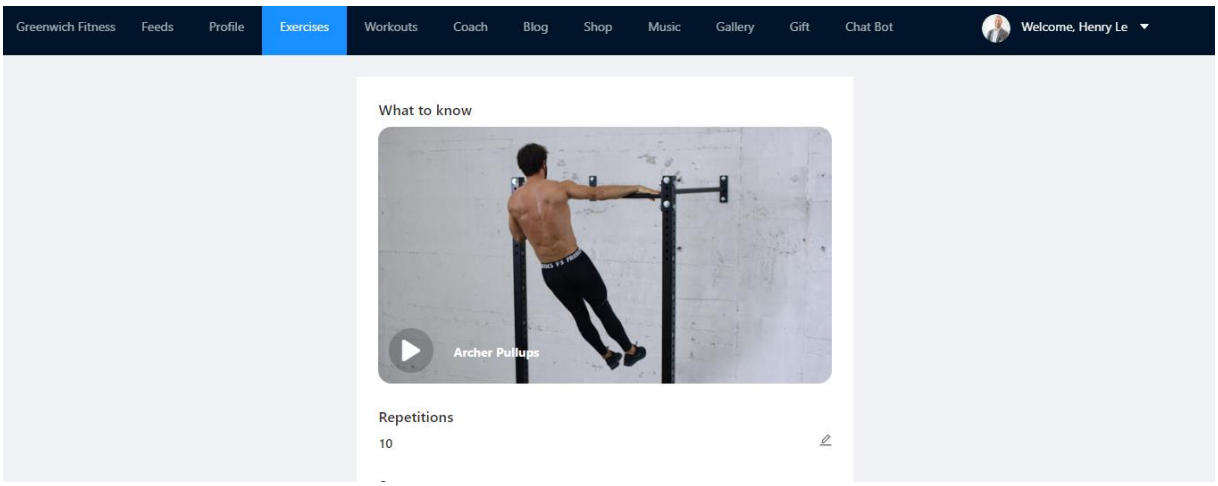


Figure 174. Detail of Exercise Screen

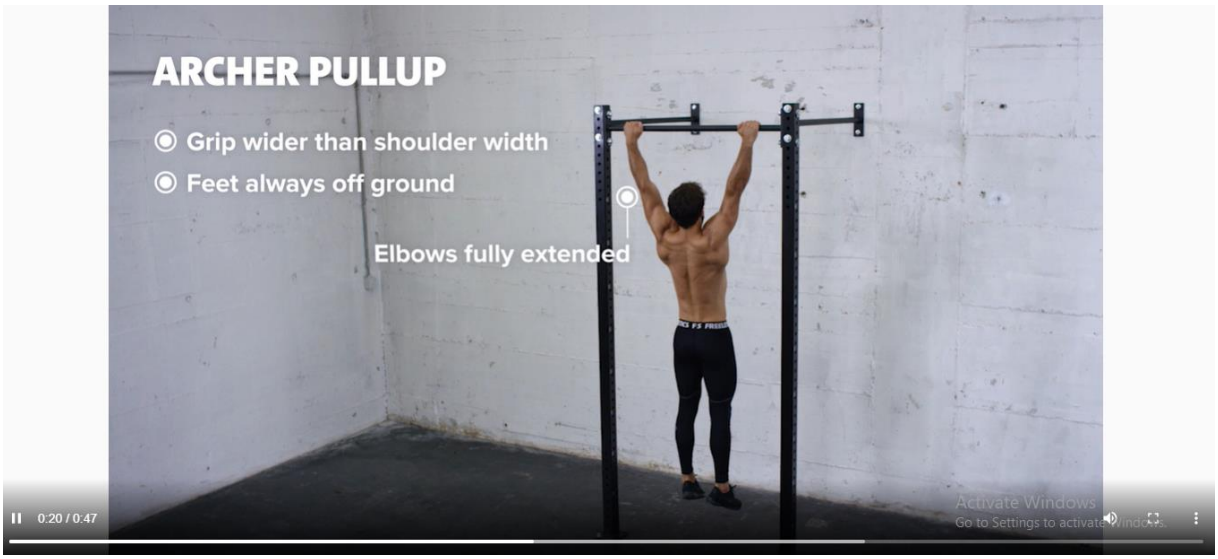


Figure 175. Exercise Video Tutorial Screen

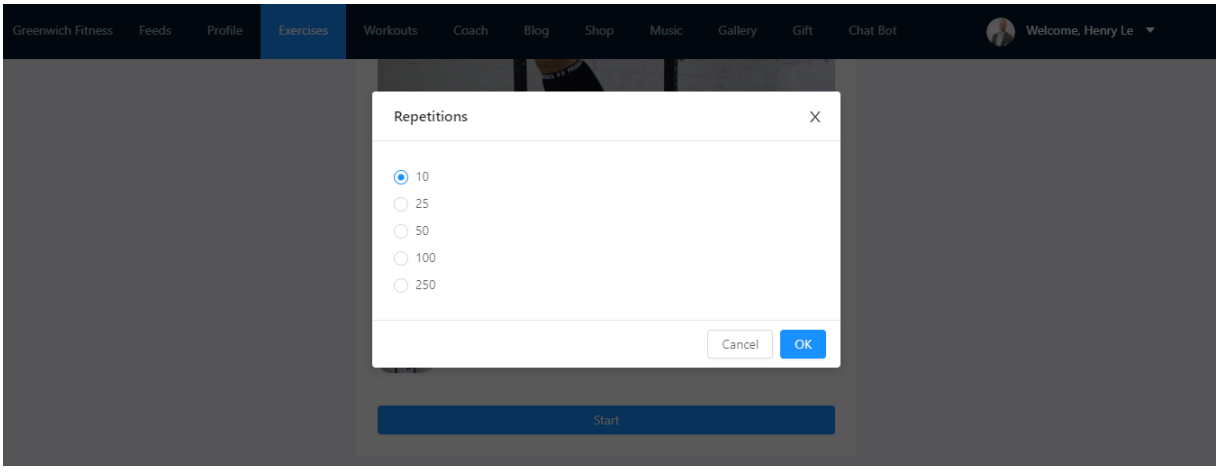


Figure 176. Change Exercise Repetition Screen

Figure 177. Countdown Screen

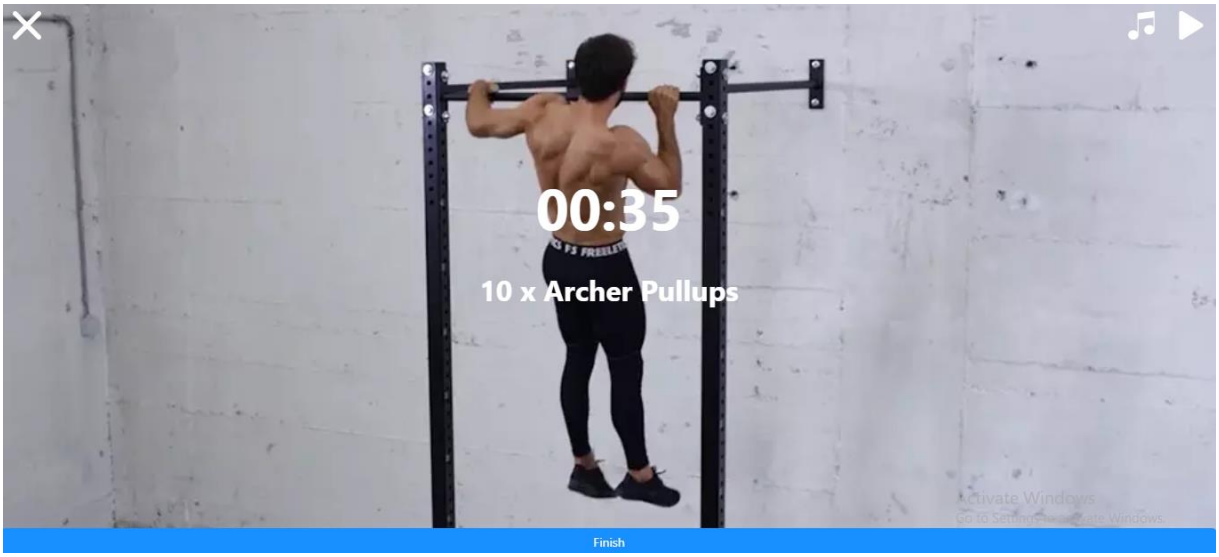


Figure 178. Exercise Training Screen

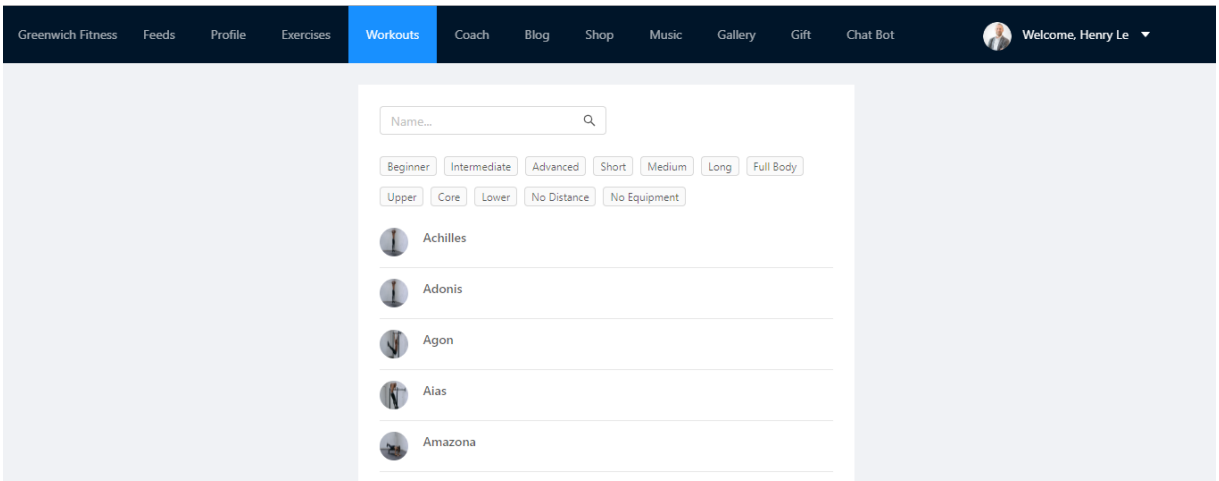


Figure 179. Workout List Screen

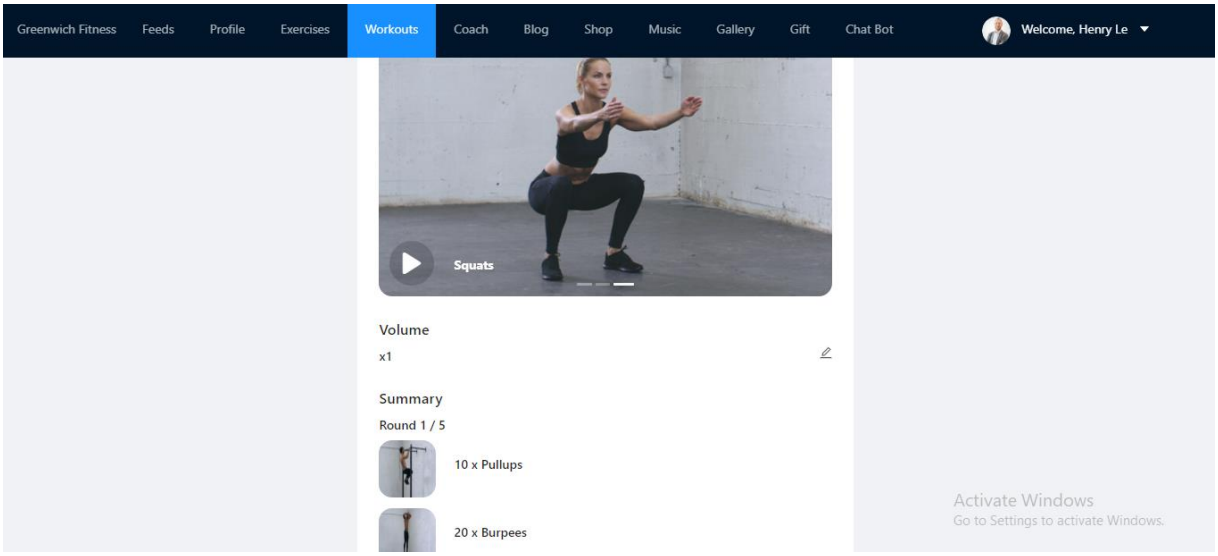


Figure 180. Detail of Workout Screen

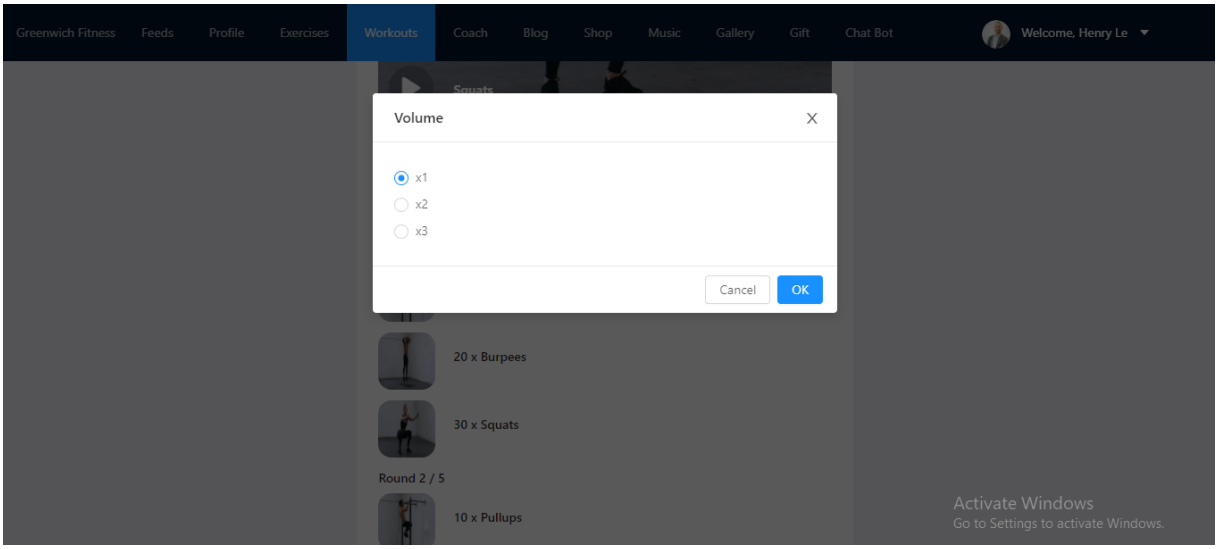


Figure 181. Change Workout Volume Screen

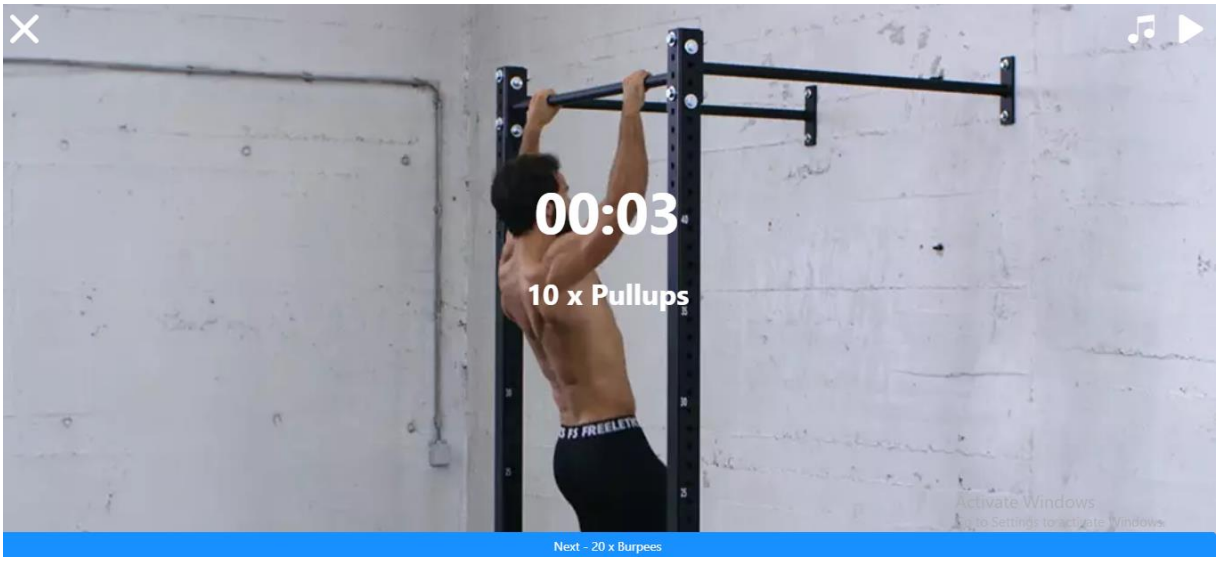


Figure 182. Workout Training Screen

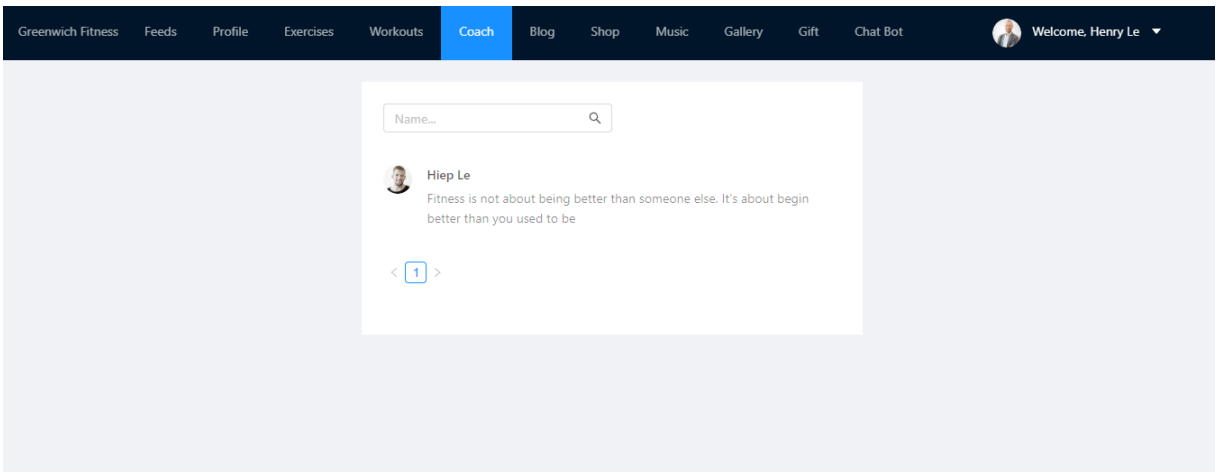


Figure 183. List of Coaches Screen

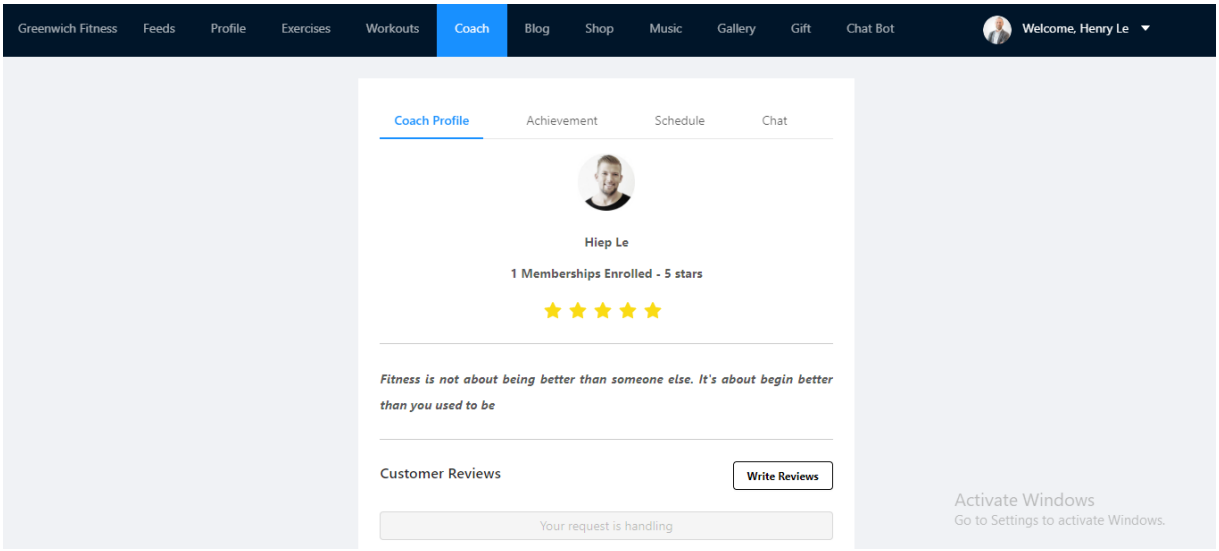


Figure 184. Coach Detail Screen - Tab Coach Profile

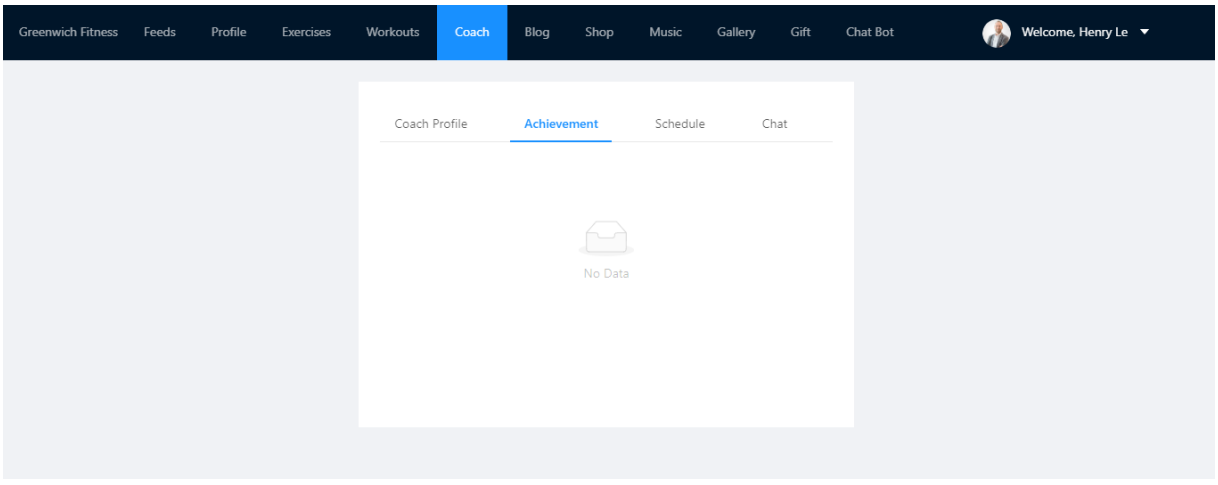


Figure 185. Coach Detail Screen - Tab Achievements

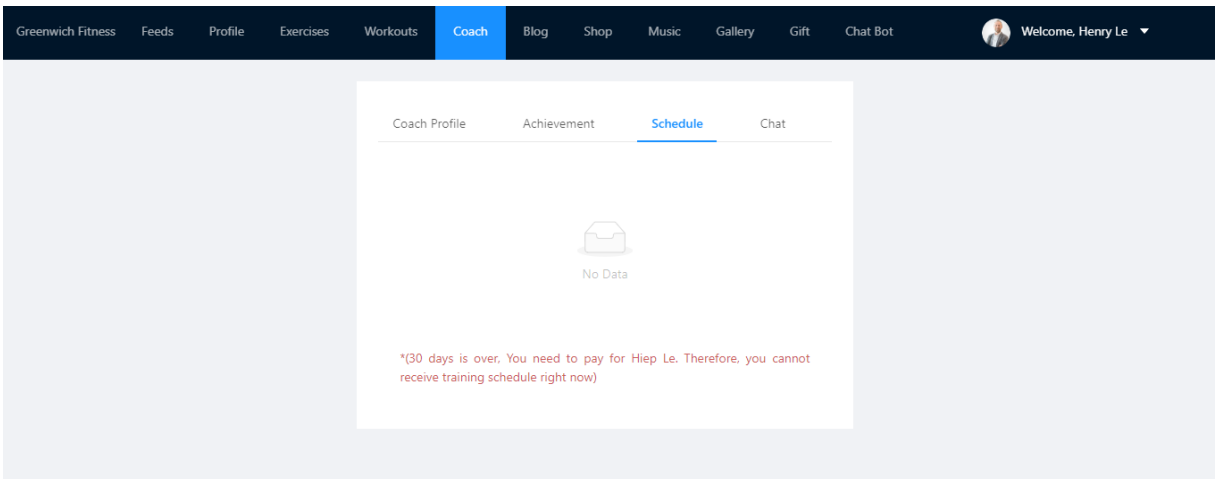


Figure 186. Coach Detail Screen - Tab Coach Schedule

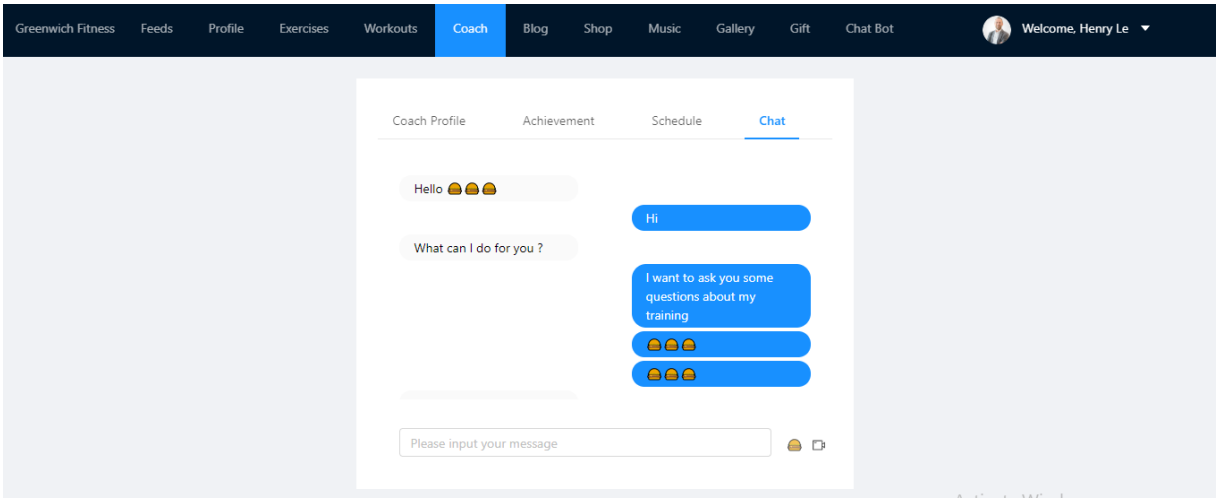


Figure 187. Coach Detail Screen - Tab Chat between Coaches and Users

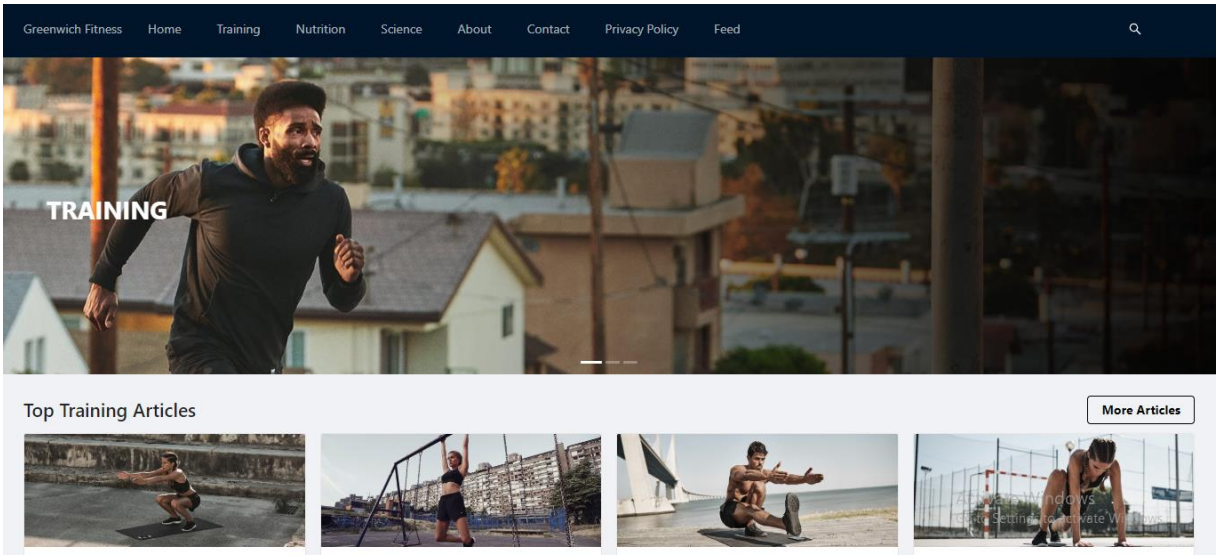


Figure 188. Blog System - Home Page

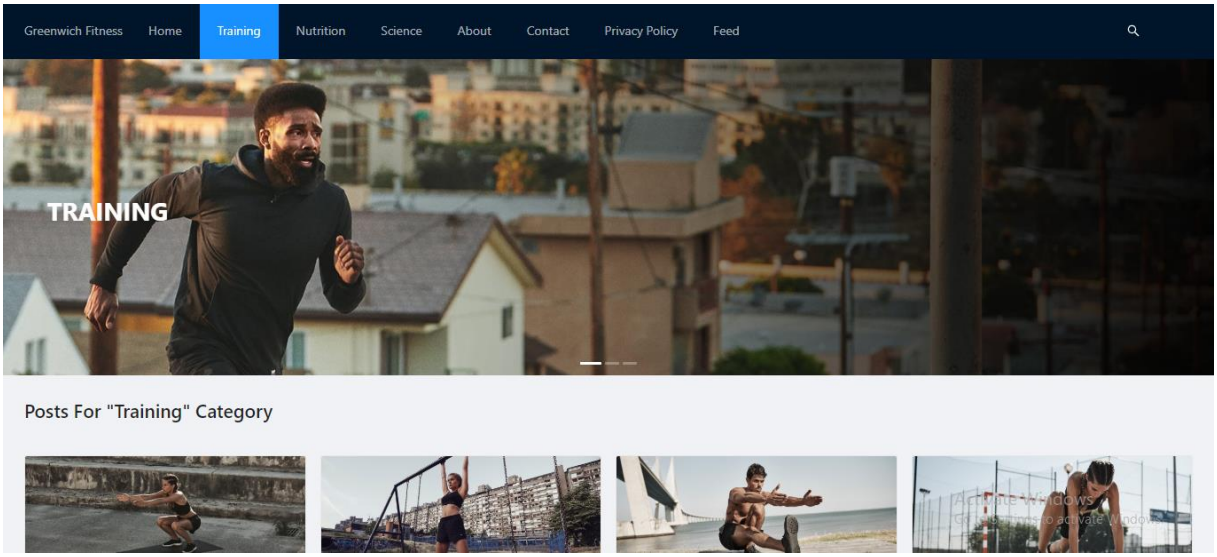


Figure 189. View Posts by Category Screen

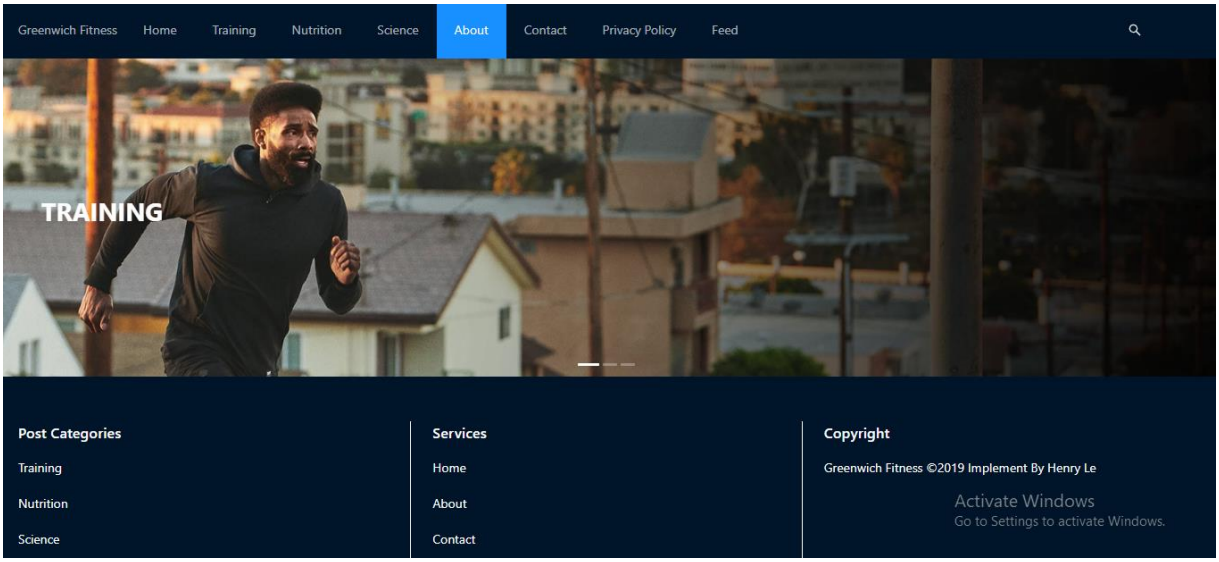


Figure 190. About Screen

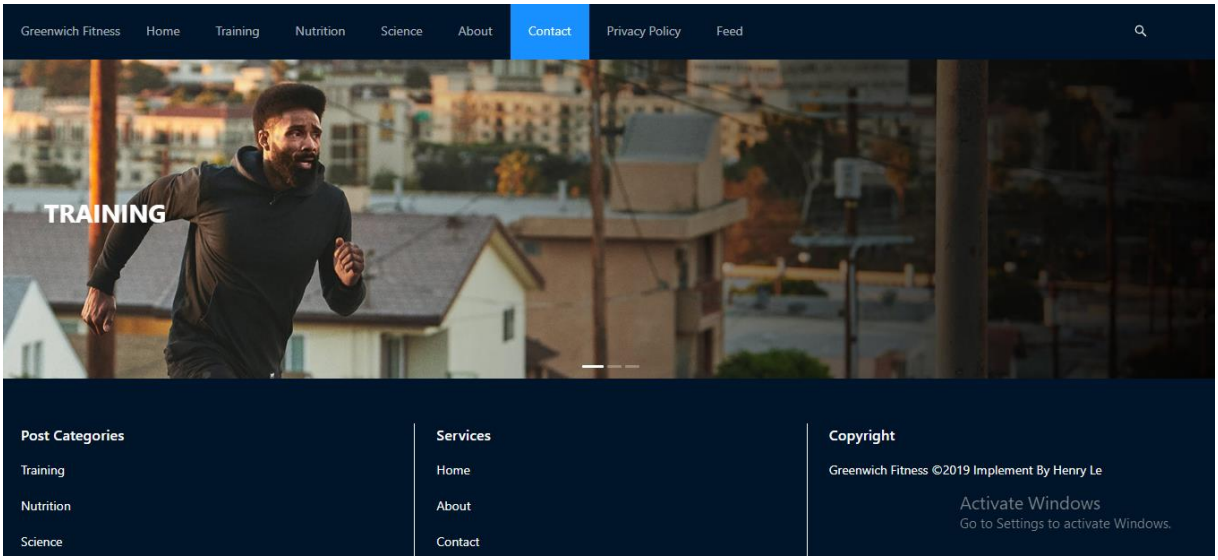


Figure 191. Contact Screen

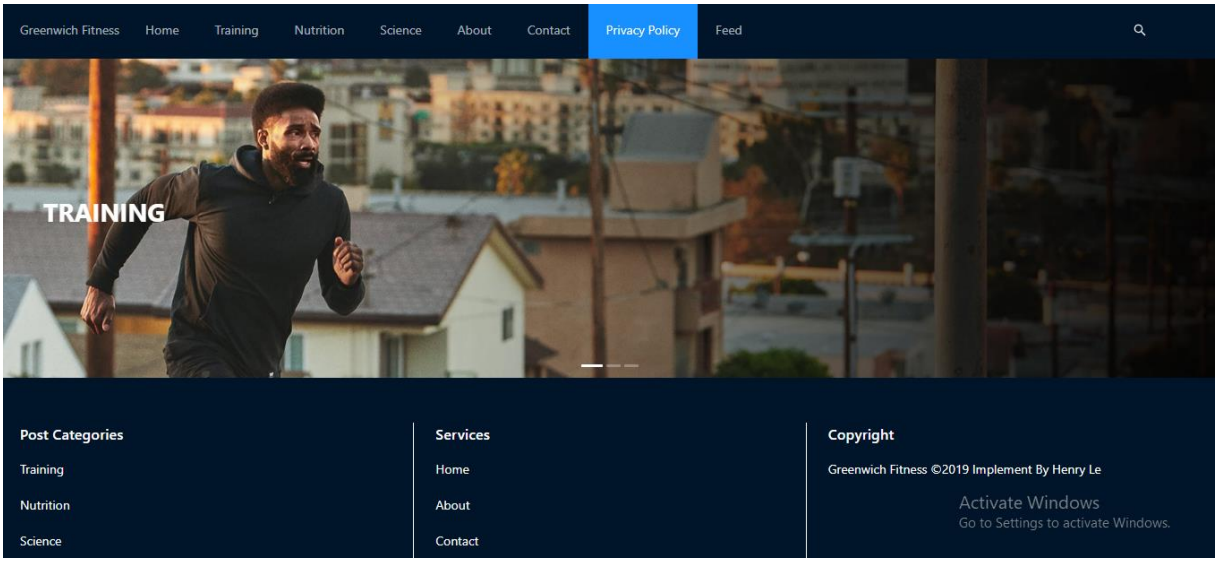


Figure 192. Privacy Policy Screen

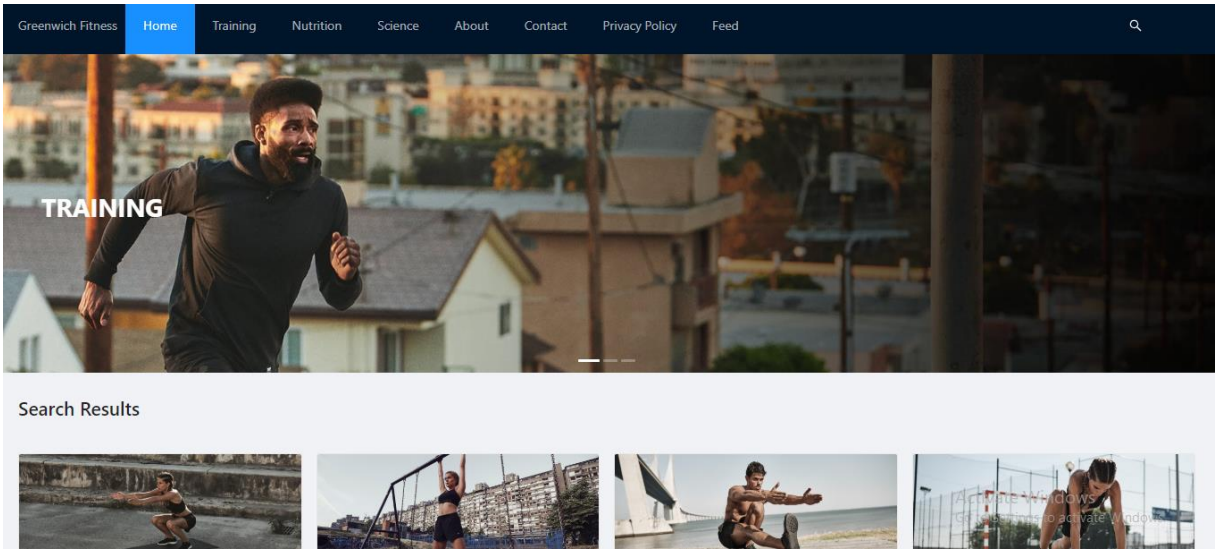


Figure 193. Search Posts Screen

What are Squats?

To squat is effectively to lower yourself down into a "seated" position with your quads parallel to the floor.

Begin upright with the knees gently flexed. Drive the hips backwards, flex the knees and lower yourself until your hips are at knee level or slightly lower. Your weight should be distributed evenly across your feet; resist the temptation to let your heels leave the ground. Ensure that your knees remain in line with your toes and that they're not turning inwards or outwards; this puts unnecessary pressure on the joints.

You don't need to focus on maintaining an entirely upright spine, but try to avoid bending over too much. When it comes to posture, keeping your arms out in front of you will help you to maintain a balanced position and engaging your core will prevent arching or rounding of the back.

What muscles are being used during Squats?

Squats are well known as an effective exercise to firm the gluteal muscles. And with good reason! Few exercises challenge your glutes to such an extent. This is because the main force is generated from the muscles in the glutes. Specifically, you are using the quadriceps located on the front of the thigh, the hamstrings and the associated hip muscles, all three gluteal muscles and the two calf muscles. But did you know that Squats are also beneficial to your core muscles, including both your abdominals and lower back? Extensor muscles as well as lateral and straight abdominal muscles stabilize the movement and are challenged with each Squat. This is

Figure 194. View Post Detail Screen

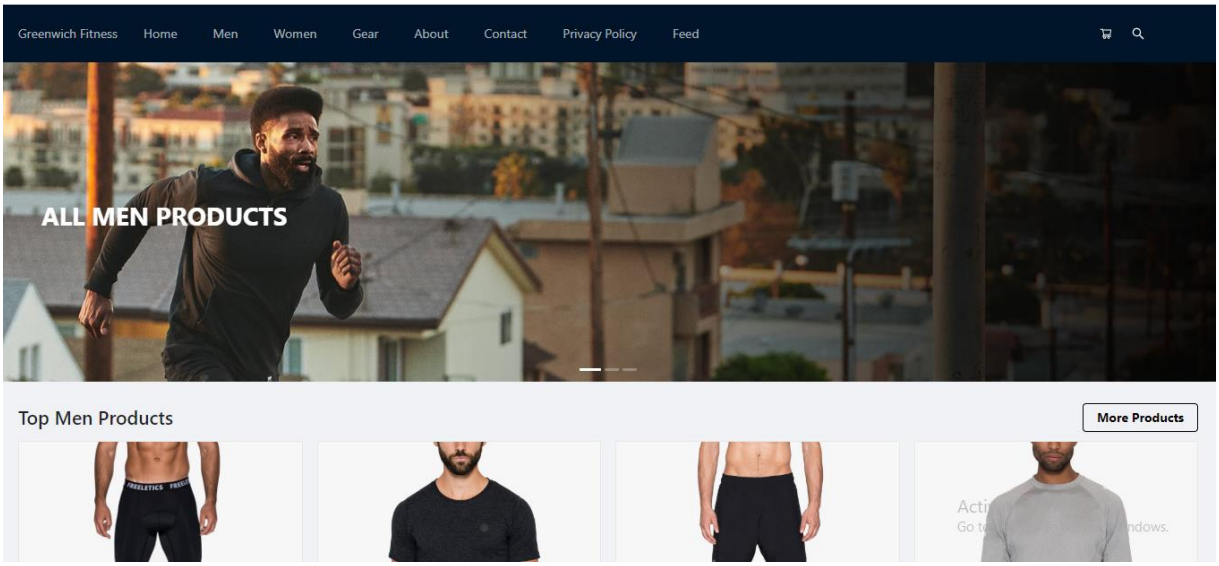


Figure 195. Ecommerce System - Home Page

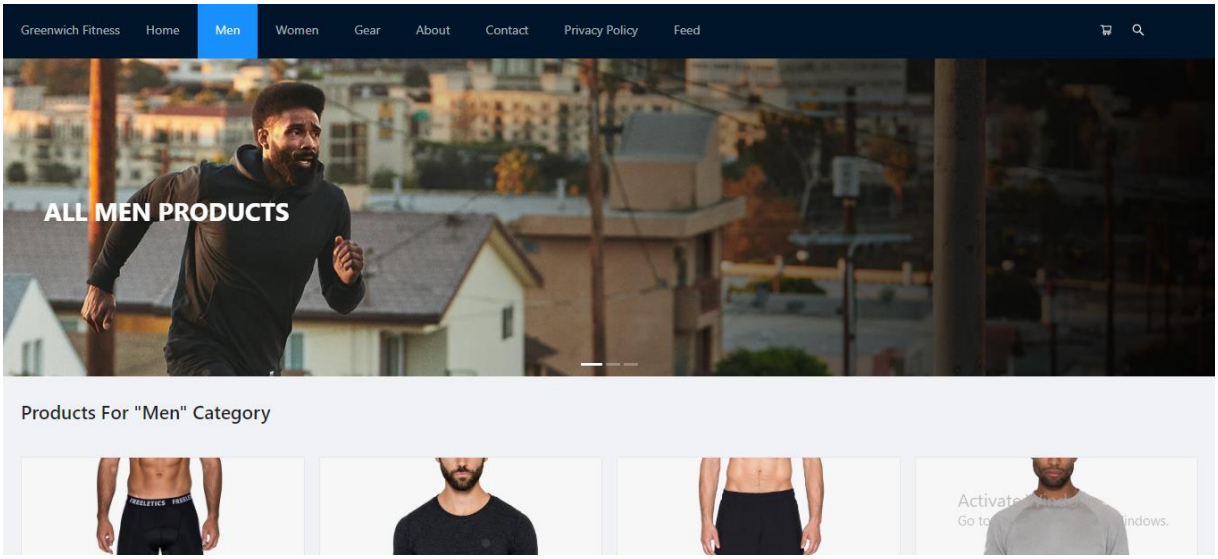


Figure 196. View Products by Category Screen

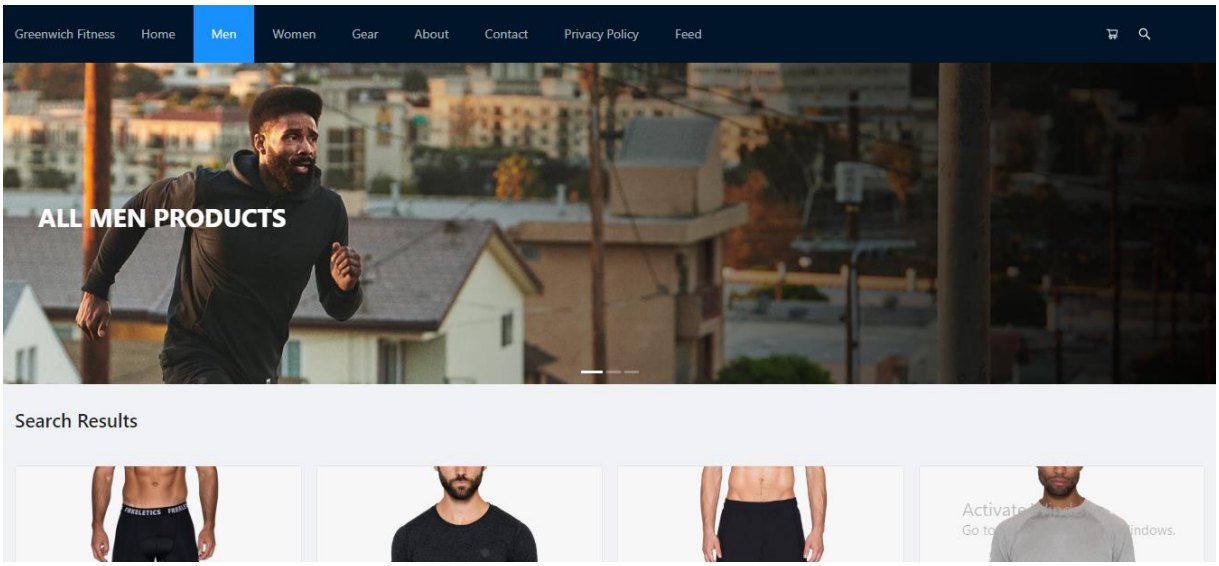


Figure 197. Searching Product Results Screen

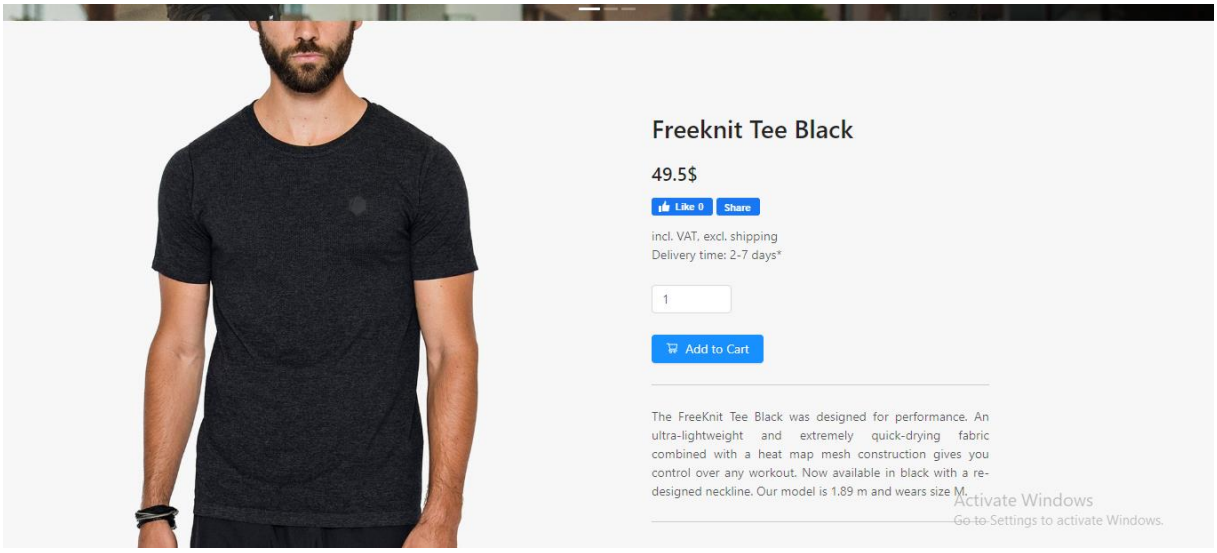


Figure 198. Product Detail Screen

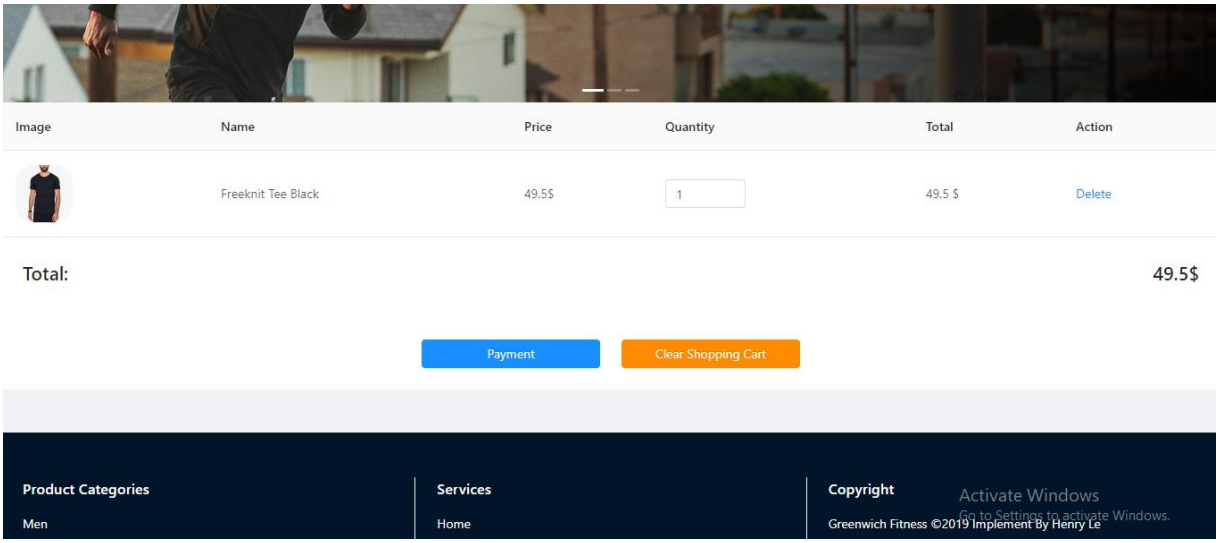


Figure 199. Shopping Cart Screen

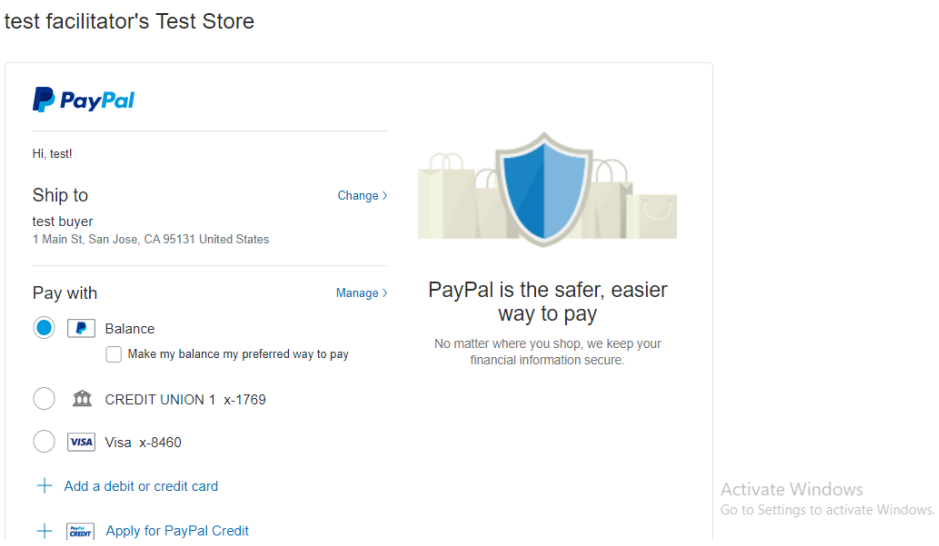


Figure 200. Payment by PayPal Screen

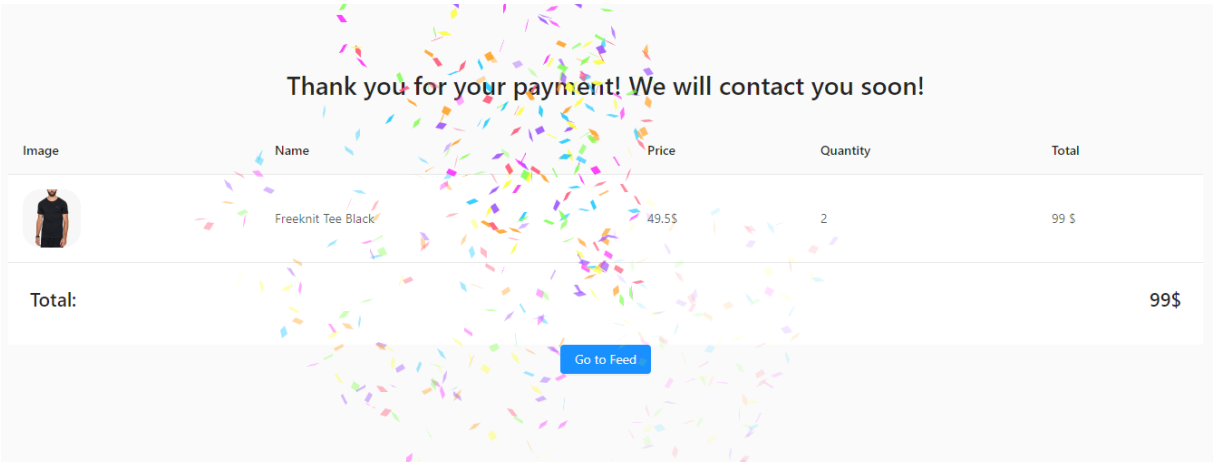


Figure 201. Buying Products Confirmation Screen

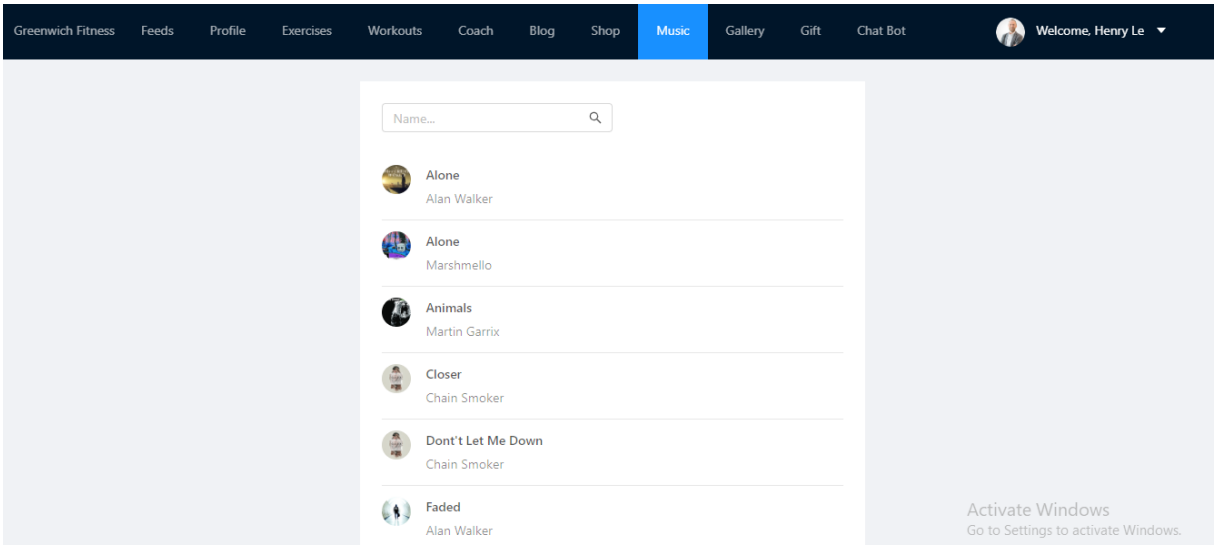


Figure 202. List of Songs Screen

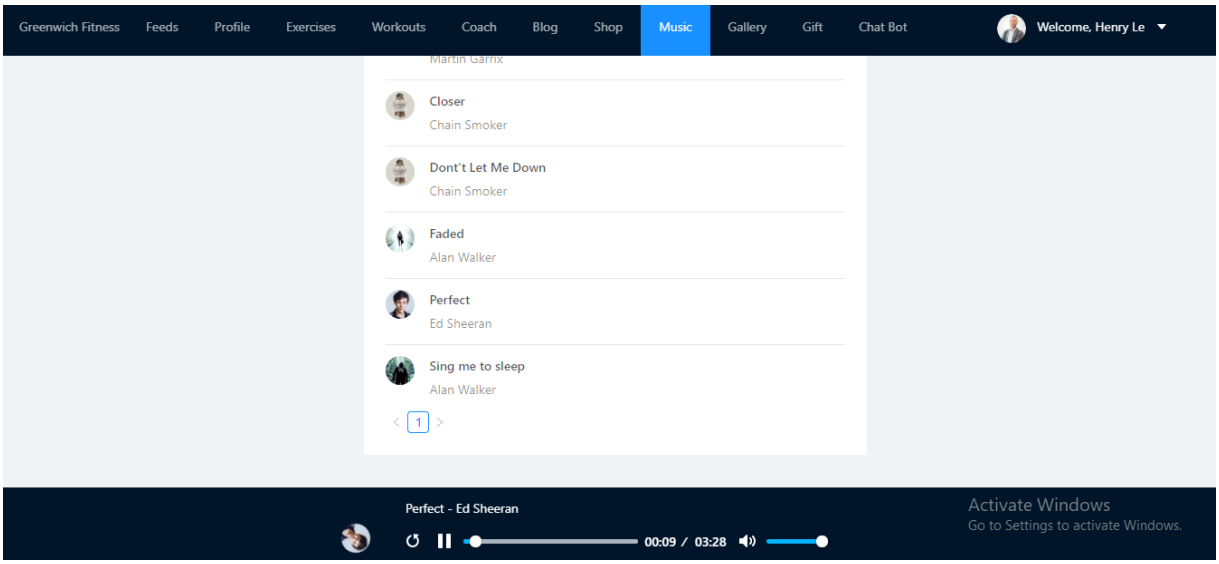


Figure 203. Playing Song Screen

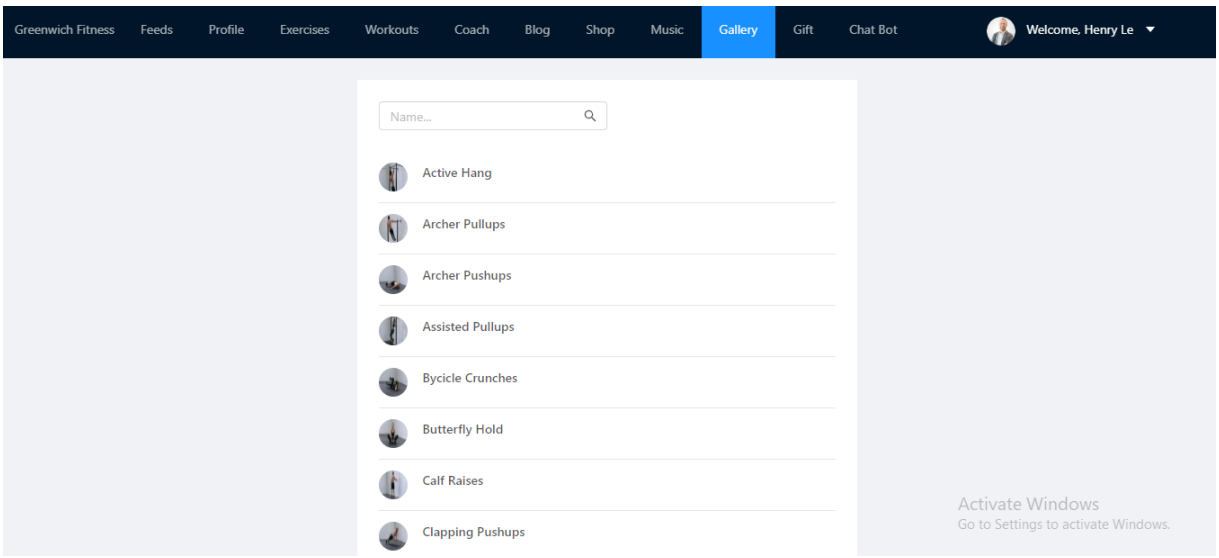


Figure 204. List of Galleries Screen

Figure 205. Detail of Gallery Screen

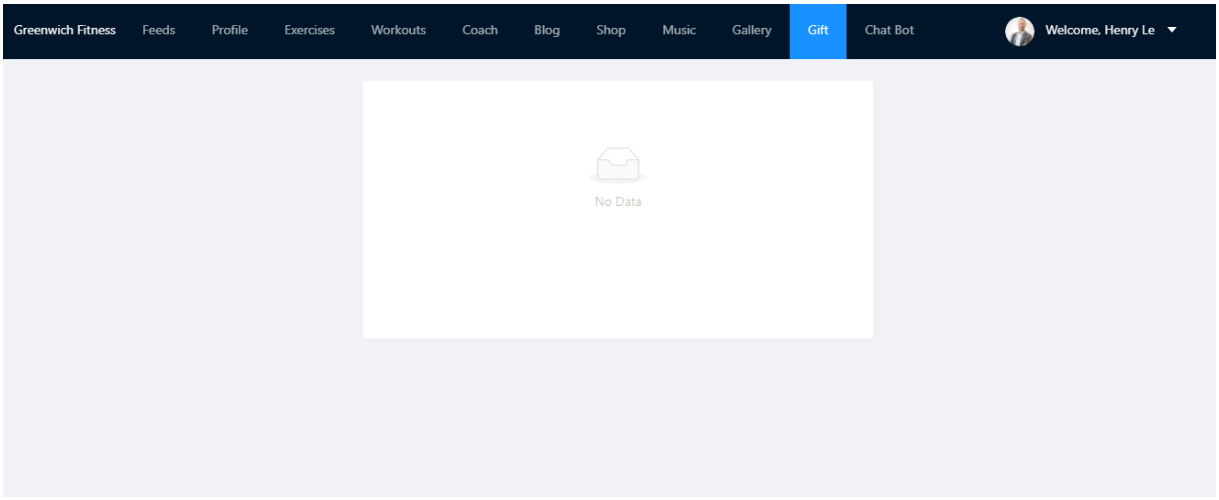


Figure 206. List of Gifts Screen

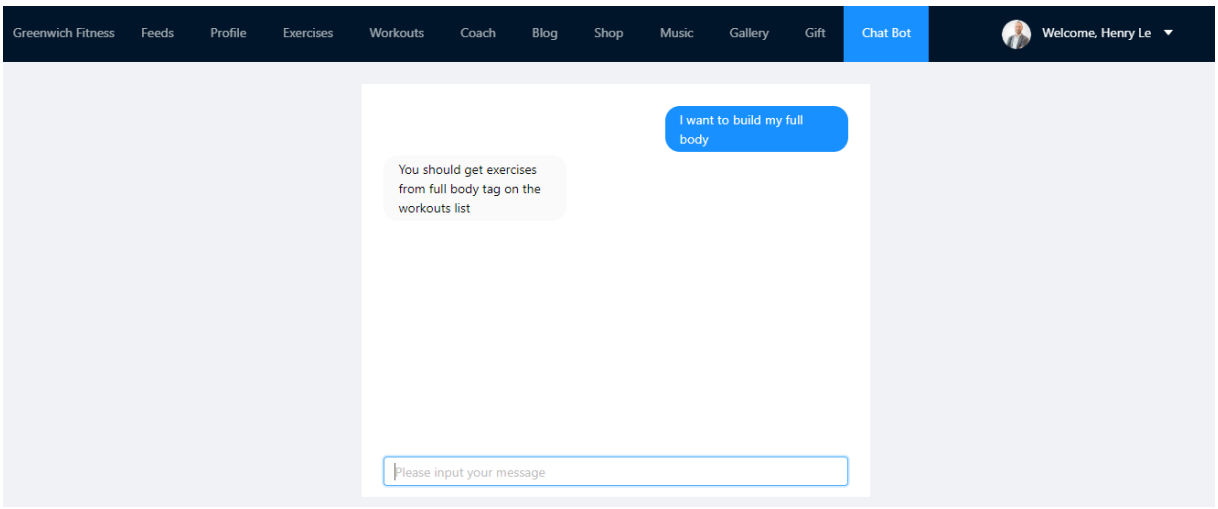


Figure 207. Talking with chat bot Screen

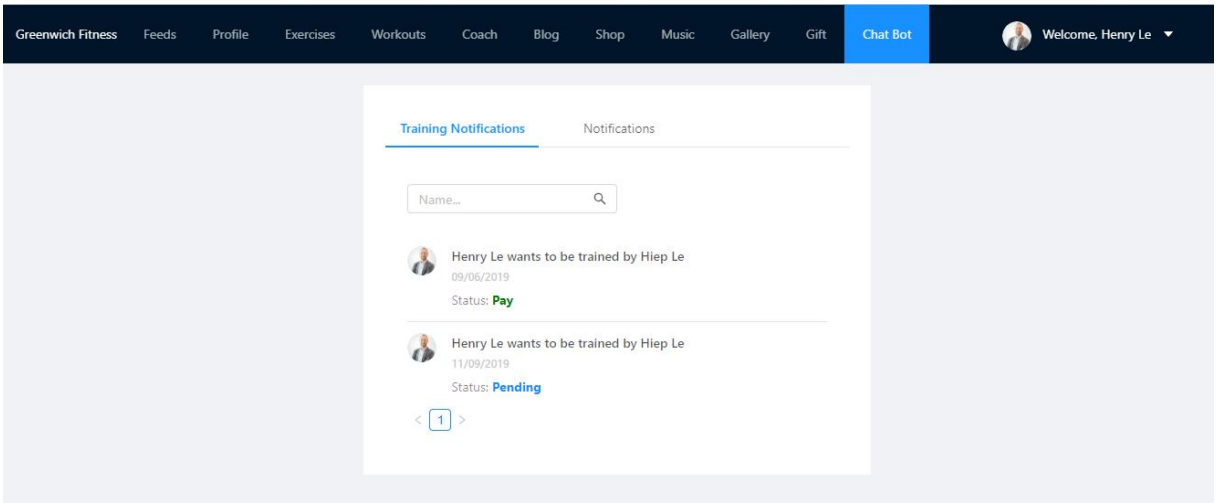


Figure 208. Notification Screen - Tab Training Notifications

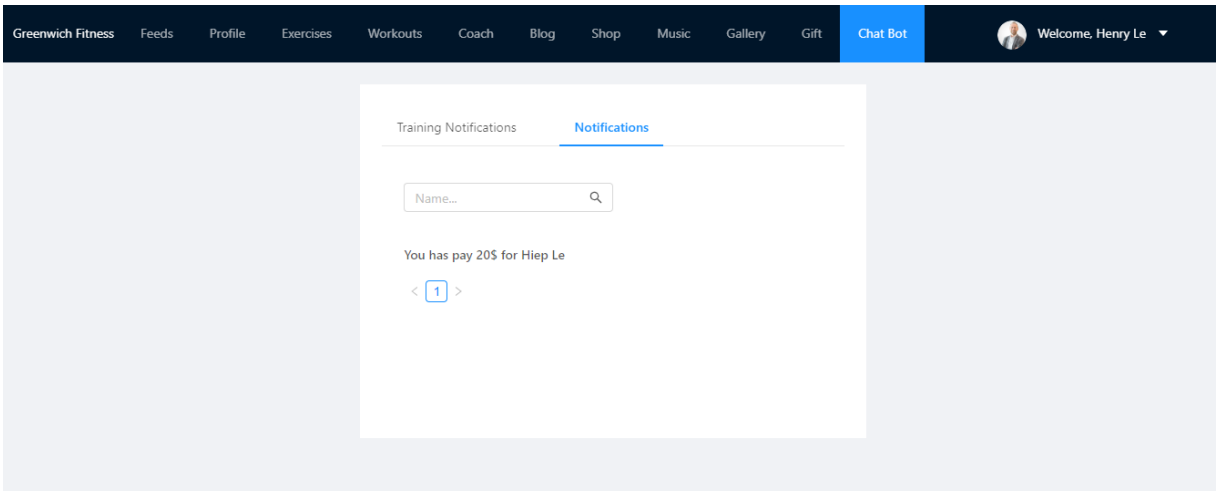


Figure 209. Notification Screen - Tab Notifications

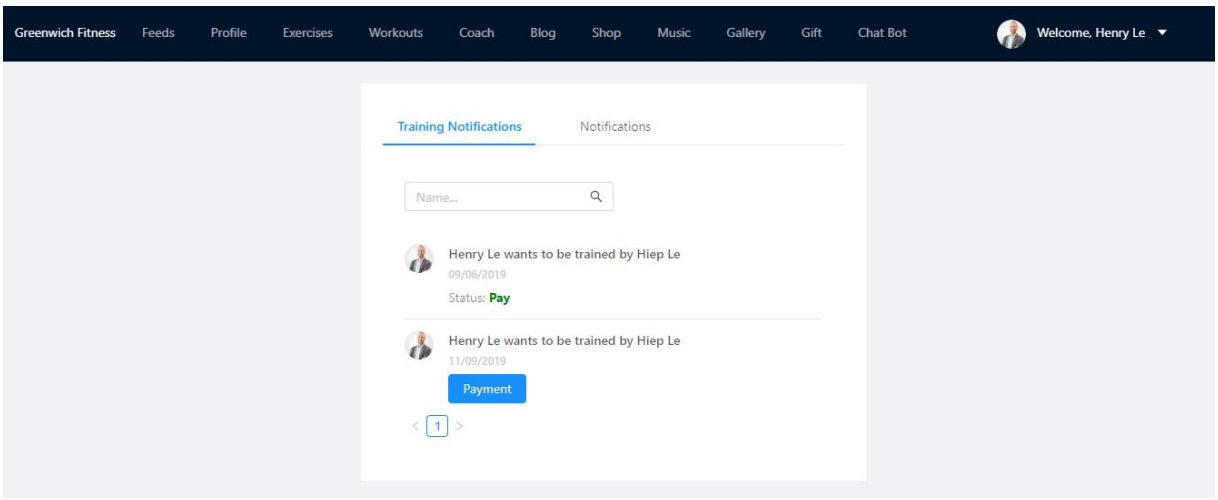


Figure 210. Coach Payment Screen

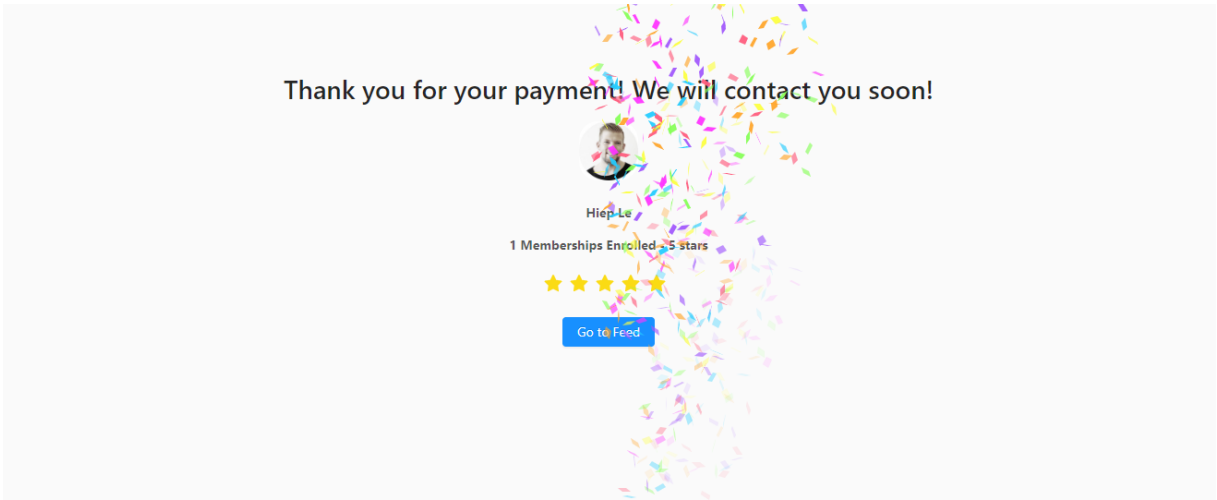


Figure 211. Coach Payment Confirmation Screen

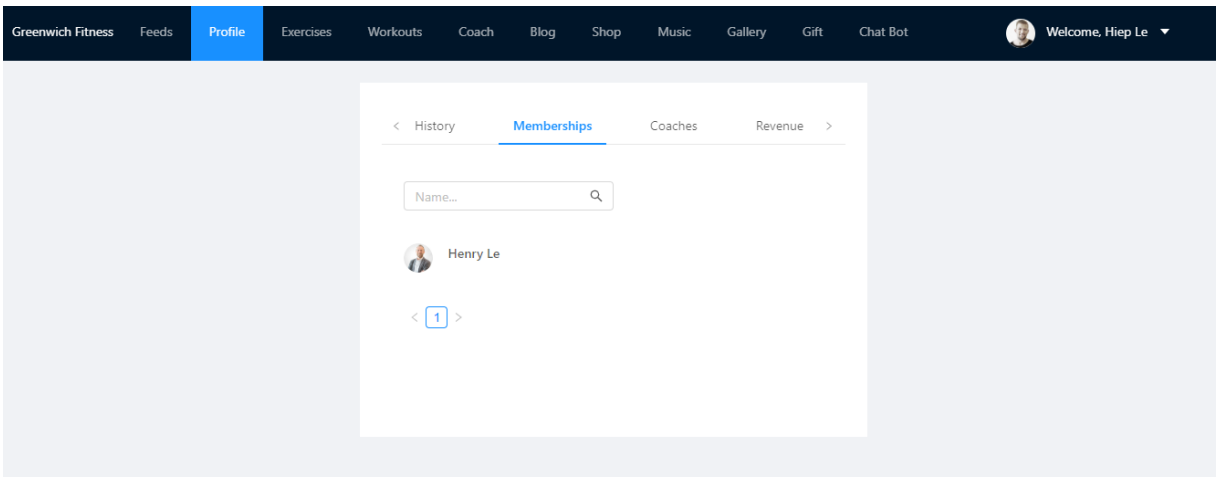


Figure 212. List of Memberships Screen

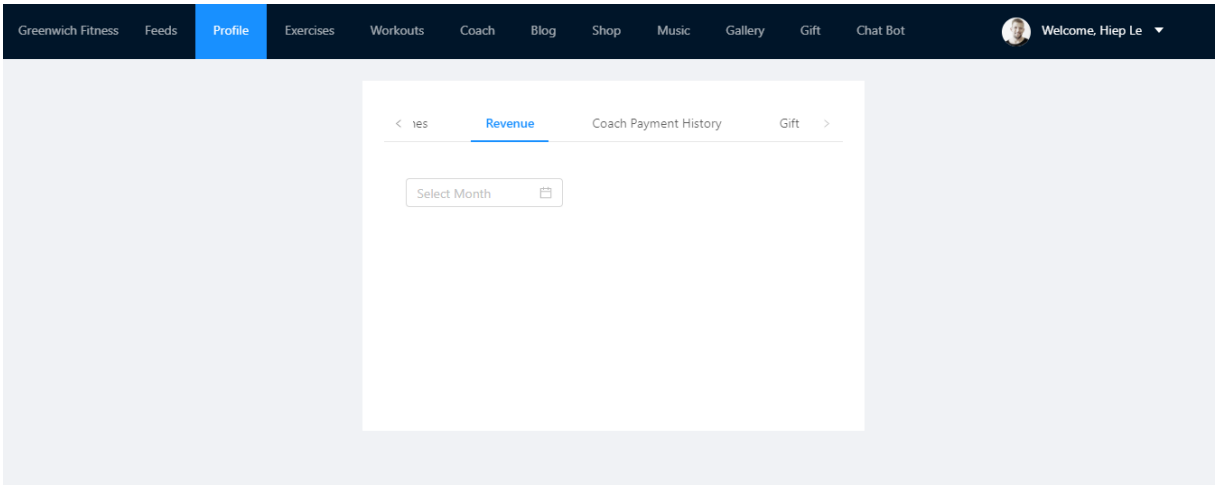


Figure 213. Coach's Revenue Screen

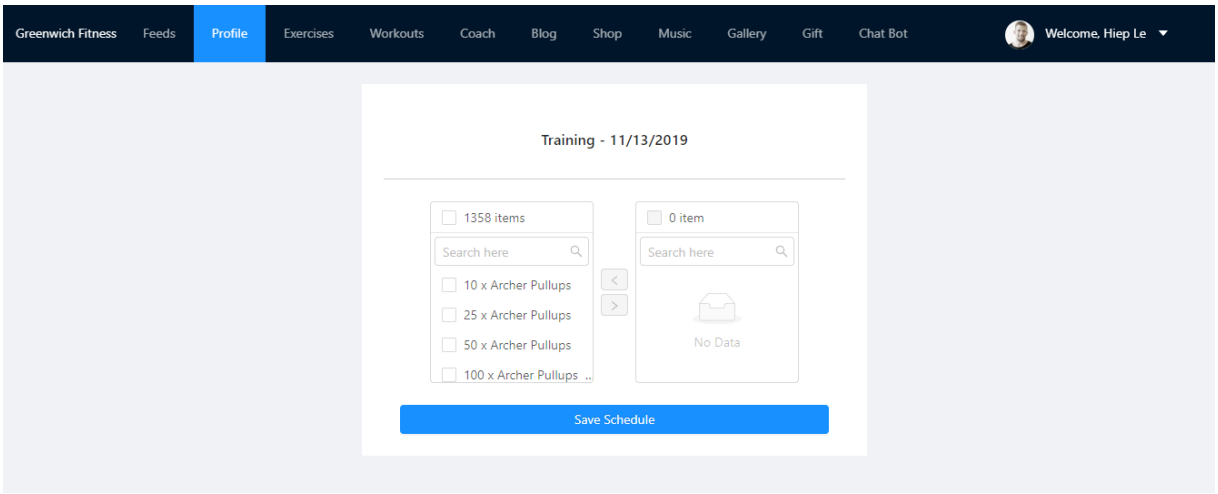


Figure 214. Add Training Schedule Screen

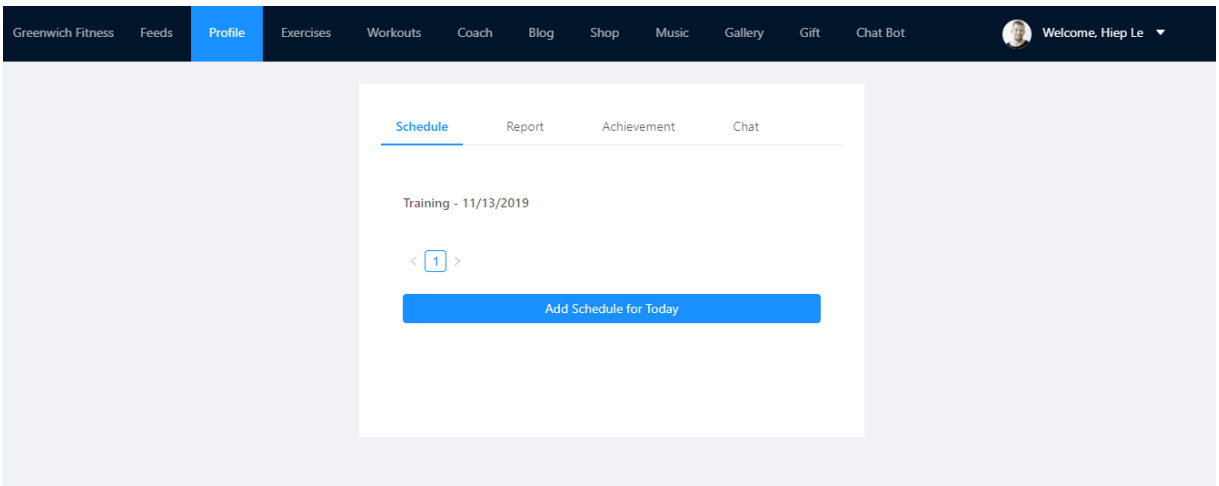


Figure 215. Membership Screen – Schedule Tab

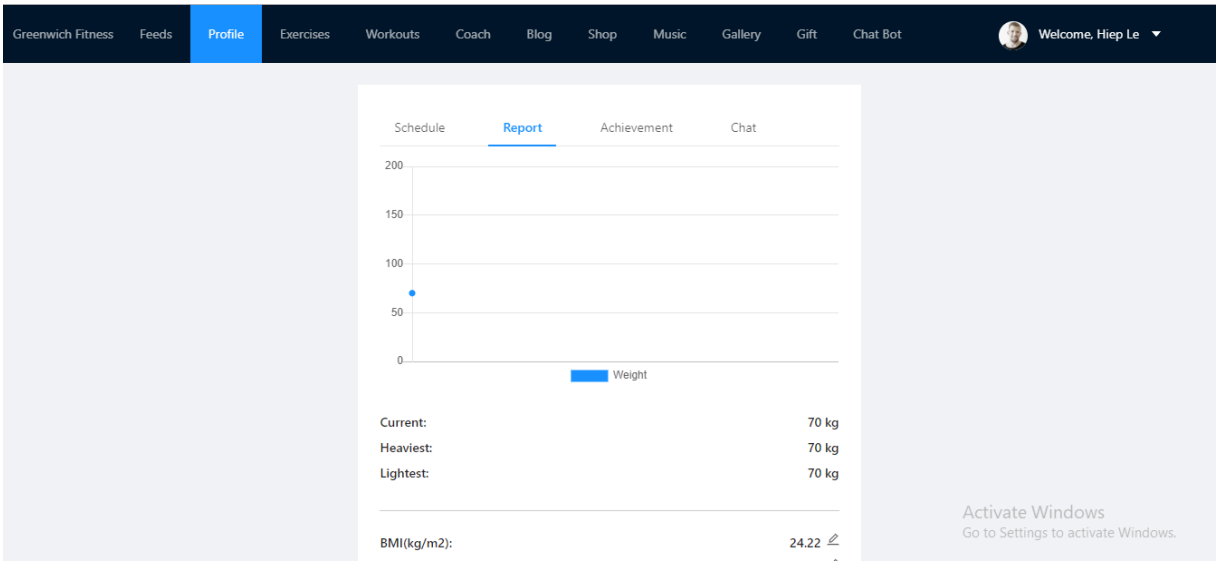


Figure 216. Membership Screen - Tab Report

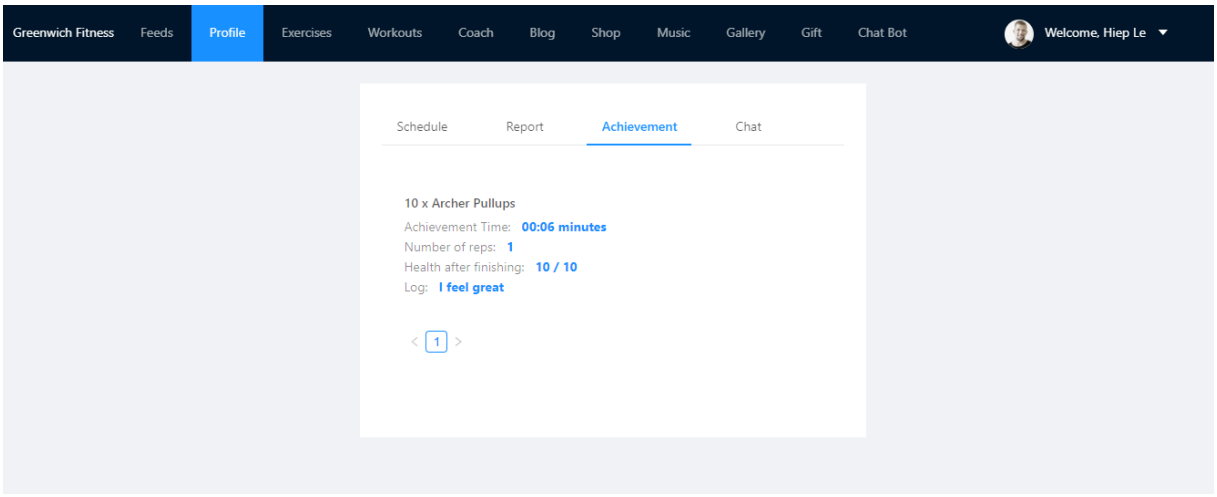


Figure 217. Membership Screen - Tab Achievement

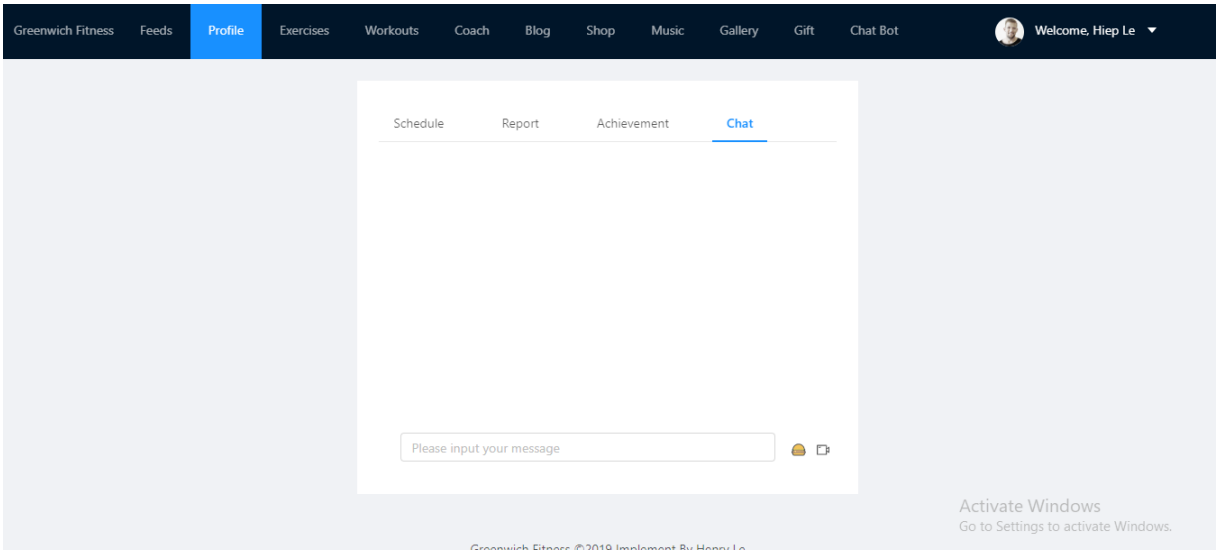


Figure 218. Chat Screen

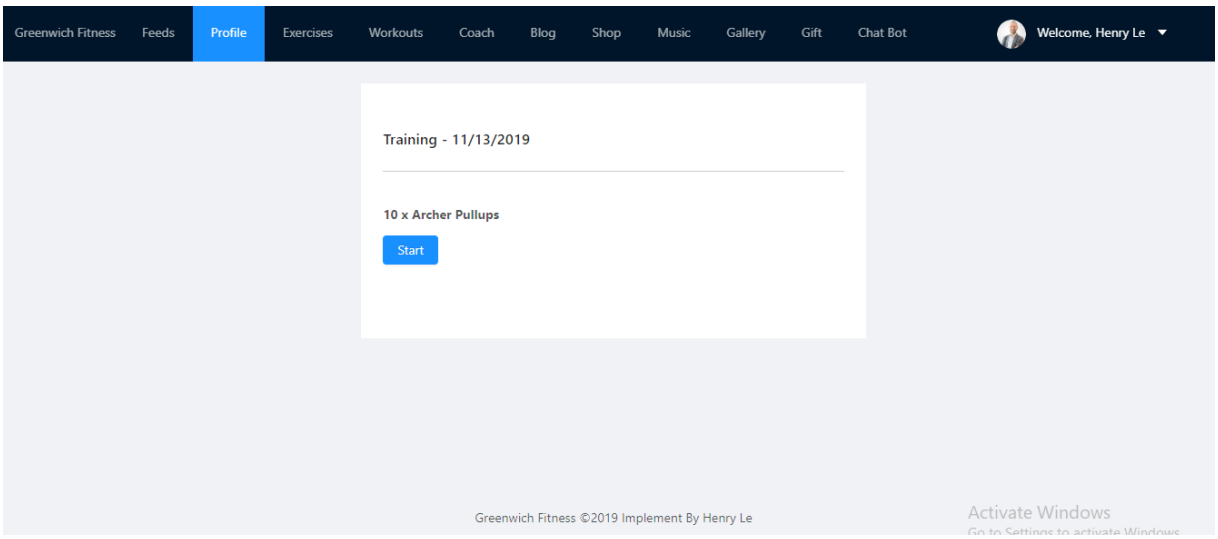


Figure 219. Membership Schedule Detail Screen