Nadezhda Mokhireva

Parisa Khataei

Mahdi Esmaeelpour

Vladyslav Bordiug

a.k.a “group 11”

Software Requirements Analysis and Design

Topfit Application

Contents

[Introduction 3](#_Toc24141473)

[Purpose 3](#_Toc24141474)

[Scope 3](#_Toc24141475)

[System Overview 3](#_Toc24141476)

[Project Perspective 3](#_Toc24141477)

[System Context 4](#_Toc24141478)

[General Constraints 4](#_Toc24141479)

[Assumptions 5](#_Toc24141480)

[Dependencies 6](#_Toc24141481)

[Functional Requirements 7](#_Toc24141482)

[Functional Requirements for *Login* feature 7](#_Toc24141483)

[Functional Requirements for *Registration* feature 7](#_Toc24141484)

[Functional Requirements for *Password Recovery* feature 8](#_Toc24141485)

[Functional Requirements for *Account Deactivation* feature 8](#_Toc24141486)

[Functional Requirements for *Start Recording* feature 9](#_Toc24141487)

[Functional Requirements for *Saving Record* feature 9](#_Toc24141488)

[Functional Requirements for *Activity Deletion* feature 10](#_Toc24141489)

[Functional Requirements for *Schedule Workout* feature 10](#_Toc24141490)

[Functional Requirements for *Notifications* feature 11](#_Toc24141491)

[Use Cases 11](#_Toc24141492)

[Use Case #1: Select Workout 11](#_Toc24141493)

[Use Case #2: Delete Workout 12](#_Toc24141494)

[Use Case #3: Contact Tech Support 13](#_Toc24141495)

[Data Modelling and Analysis 14](#_Toc24141496)

[Normalized Data Model Diagram 14](#_Toc24141497)

[Activity Diagrams 15](#_Toc24141498)

[Sequence Diagrams 20](#_Toc24141499)

[UML Class Diagram 22](#_Toc24141500)

[Process Modelling 23](#_Toc24141501)

[Non-Functional Requirements 23](#_Toc24141502)

[Logical Database Requirements 25](#_Toc24141503)

[Approval 27](#_Toc24141504)

# Introduction

## Purpose

The primary purpose of this document is to present a detailed insight of the TopFit application. It will explain the system in details, covering the requirements, intended use and functionality, as well as the graphical representation (the interface). Details about the application’s limitations will also be provided. This report is intended for both the stakeholders’ and the developers’ use and acts as a support documentation of the application.

## Scope

The final product will be presented in a form of mobile fitness application. The application will be designed to provide users with exercise routines that can be completed from home. By following the guidelines, users will be able to save time and money on attending a regular fitness facility. In order to not miss a session, it will be possible to synchronize the workout schedule with the personal calendar. Upon doing this, the user can opt for receiving notifications about upcoming sessions. Moreover, users will be encouraged to move more throughout the day by the GPS tracking feature and tracking their progress in the dedicated section of the application. Should the user discover a technical issue, they can contact the development team and address their concern.

At this version of the system, the following functions will be out of scope:

* diet and nutrition information
* personal training
* communication within the application

# System Overview

## Project Perspective

TopFit application is a new system designed based on the market demand. The idea takes origin in other fitness applications that currently populate the market and contains the most popular exercise routines among different categories of people. The application is intended to use in place of attending a fitness facility: the content is free and can be used from any convenient place. The first release will be only available for use on Android mobile devices, but future development allows for expansion towards Android Wear and iOS.

## System Context

TopFit application seeks to improve the exercise experience by offering free content and user-friendly interface. By utilizing it, people with sedentary lifestyle and a lack of time / money to attend regular fitness facilities will be able to exercise at any convenient place. TopFit offers well-balanced routines developed by professionals and confirmed to be effective and safe to perform without supervision. By undertaking the development, the team hopes to advance the knowledge of mobile application development and design. Conducting the project is also supposed to promote the company and enhance its reputation among competitors. While at the current moment generating revenue is not the primary purpose, it will likely serve as a long-term goal.

## General Constraints

The TH application is restricted with the following constraints:

**Cost** – the project has no budget because open-source technologies are used. Thus, no costs should be involved in the development.

**Scope** – development of a native Android application

**Quality** - application must be of high performance and store user data securely despite the absence of cost and the type of technology chosen. All the testing will be performed on developers’ hardware, all of which has similar capacity.

**Customer Satisfaction** – the application should have sufficient functionality to satisfy the target audience and thus to withstand the market competition.

**Risk** – the team has very limited resources and time; thus, losing a team member will significantly impact the development process. Another critical risk is failing hardware: due to the absence of budget the team is unlikely to afford new equipment. Replacement parts can be purchased, but this counter-acts with another crucial resource: time.

**Resources** – the team members are consistent and will not change during the development. The work is conducted on the developers’ own hardware with the open-source software that can be accessed at no cost. As for the time, the project is to be completed until March 27, 2020.

## Assumptions

In order for the project to move forward according to the plan, the following assumptions must be valid:

**Resources**

* Users will be able to test the Beta-version of the product at the agreed time
* Hardware used by developers will be properly configured (all the necessary software present and functional)
* Estimated development time and cost are not exceeded
* The necessary training for developers will be conducted independently with no additional expenses

**Solution**

* The first release of the application will be free of charge and available for download for Android OS phones
* All the defined functions will be implemented and working without interruption

**Scope**

* Project scope will remain consistent after the scope statement is signed by the involved parties

**Methodology**

* Project will follow agile methodology throughout the development cycle, with every sprint delivering all the defined outcomes

**Technology**

* The primary technology used for development will be the Java programming language.
* Android Studio will be used as the IDE.
* Alpha-testing will be conducted on developers’ own devices, while beta-testing will take place of testers’ Android phones.

## Dependencies

**Internal**

* Workouts are complete sets of exercises
* All workouts are safe to perform
* User accounts are well-protected from external access
* Application content receives positive feedback from users
* Content is periodically reviewed and renewed.

**External**

The success of the project is greatly dependent on the market demand for fitness applications.

* Opening of a big number of fitness facilities in a city will make them more accessible that might decrease the demand for the *TopFit* application
* If prices for gym membership decrease, customers are likely to choose a traditional way of exercising over using the application.

The project also depends on competitors’ success:

* If a similar application with more features is released by a competitor company, market share might change in favor of the competitors.
* On the opposite, if no significant activity is performed by other companies, *TopFit* might eventually dominate the market.

# Functional Requirements

This subsection provides the details about the core functionality of the application and the flow of information within the system.

## **Functional Requirements for *Login* feature**

|  |  |
| --- | --- |
| **Introduction** | All users will be able to login with their username and password upon launching the application. |
| **Inputs** | 1. User enters username. 2. User enters password. 3. User presses *Submit* button |
| **Processing** | On *Submit* button click in the form, authVerify() function will be called in order to validate the input data. It will check if the username and password combination match the one in the database. |
| **Outputs** | User in redirected to the landing page based on their permission level. They are now able to retrieve any data associated with their account (e.g. activity history, personal data, statistics) |

## **Functional Requirements for *Registration* feature**

|  |  |
| --- | --- |
| **Introduction** | All users must have accounts in order to get access to the application’s functionality. If a user doesn't have an account, they will be asked to create one. |
| **Inputs** | 1. User creates username. 2. User enters valid email address. 3. User creates password. 4. User press the *Register* button |
| **Processing** | On *Register* button click a registerUser() function will verify that all mandatory fields are filled out with the proper values, none are left empty. A confirmation email will be sent to the user. |
| **Outputs** | User can access the application functionality and utilize the features according to their permission level. |

## **Functional Requirements for *Password Recovery* feature**

|  |  |
| --- | --- |
| **Introduction** | If a user forgets their password, they can reset it by providing the email address associated with their account. |
| **Inputs** | 1. User enters email that was provided during registration 2. User presses the *Reset* button |
| **Processing** | On *Reset* button click the forgotPassword() function will check if the provided email exists in the database. If so, the user will be emailed instructions on how to create a new password. |
| **Outputs** | Upon creating the new password, the user can access the application again. |

## **Functional Requirements for *Account Deactivation* feature**

|  |  |
| --- | --- |
| **Introduction** | If a user no longer wants to use the application, they might have their account deactivated. |
| **Inputs** | 1. User provides the reason why they don’t want to use the application anymore 2. User enters their password. 3. User presses *Deactivate* button |
| **Processing** | On *Deactivate* button click the deactivateAccount() function will be called. It will delete the account related to the user from the database. |
| **Outputs** | User account and all information related to account will be deleted. |

## **Functional Requirements for *Start Recording* feature**

|  |  |
| --- | --- |
| **Introduction** | When a user presses the button while they are walking or running, the system starts to record the way that the user is going through and launches a timer to keep track of how long the user spends for the activity. |
| **Inputs** | 1. User is currently at the “Start Activity” screen 2. User presses the *Start* button |
| **Processing** | On *Start* button click the startRecord() function will be called. It will launch the process of tracking the user activities. |
| **Outputs** | The user will be able to save and later access their activity history. |

## **Functional Requirements for *Saving Record* feature**

|  |  |
| --- | --- |
| **Introduction** | The user presses *Save* button in order to save their activity record. |
| **Inputs** | 1. User previously pressed *Start Recording* 2. User presses *Save* button |
| **Processing** | On *Submit* button click the saveRecord() function will write a record to the database table. The record is essentially a set of coordinates that describe the user’s route. |
| **Outputs** | User saves their progress and is later able to review it. |

## **Functional Requirements for *Activity Deletion* feature**

|  |  |
| --- | --- |
| **Introduction** | The user can delete some activities recorded that they don’t want to be associated with their account. |
| **Inputs** | 1. User chooses *History* option from the menu. 2. User selects the record that they want to delete. 3. User presses the *Delete* button. |
| **Processing** | On *Delete* button click the deleteActivity() method will be called. It allows users to delete unused or old activities to make some space for new ones. This method actually deletes a database record, so there’s no way to restore the information. |
| **Outputs** | User deletes record that is no longer associated with their account. A success / failure message is displayed |

## **Functional Requirements for *Schedule Workout* feature**

|  |  |
| --- | --- |
| **Introduction** | User has an option to schedule a workout and add it to their personal calendar. |
| **Inputs** | 1. User navigates to the *Calendar* tab of the menu. 2. User presses the *New Event* button. 3. User selects the workout that they want to schedule and time and date as well. 4. User presses *Save* button. |
| **Processing** | On *Save* button click the saveSchedule() function will be called. It writes a record to the database table and adds the event to calendar. |
| **Outputs** | User can see all their scheduled workouts in the calendar and can choose to be reminded about upcoming events. |

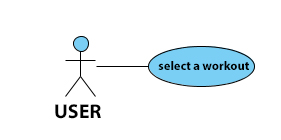
## **Functional Requirements for *Notifications* feature**

|  |  |
| --- | --- |
| **Introduction** | User will be automatically reminded by the app about upcoming events. |
| **Inputs** | Note: User must give permission to the application “To Send Notifications” in order to receive them.   1. User must have a scheduled workout. 2. User navigates to the application settings. 3. User selects *Notifications*. 4. User sets the time frame when they will be reminded about the upcoming event. 5. User presses *Turn On*. |
| **Processing** | On *Turn On* button click the OnNotific() function will be called. It adds a record to the database table and a reminder to the system. |
| **Outputs** | User will receive notifications about upcoming workout before certain period of time (based on their selection). |

**Use Cases**

### **Use Case #1: Select Workout**

Diagram



Brief Description

The user selects from list of available workouts the desirable one.

Initial Step-By-Step Description

Before this use case can be initiated, the user has already logged in the TopFit app.

1. The user selected Workout dropdown menu button.

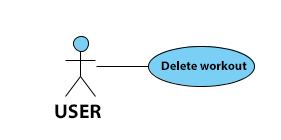
2. The system uses the Button\_ListWorkouts function to bring up the list of available workouts.

3. The user chose workouts that are available for them.

4. The system generates the list of exercises needed for this workout.

### **Use Case #2: Delete Workout**

Diagram



Brief Description

The user deletes a previously scheduled workout from a calendar.

Initial Step-By-Step Description

Before this use case can be initiated, the user has already logged in the TopFit app.

1. The user selects Calendar option from the side menu.

2. The system uses the Button\_Calendar to bring up the screen with calendar.

3. The user click on the workout in the calendar that they want to delete.

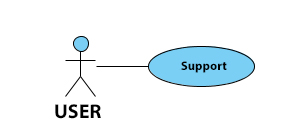
4. The system generates a warning message to user to confirm the deletion.

5. The system uses the Button\_Delete\_Workout to delete workout from the user’s calendar, if “YES” was selected.

6. Otherwise, no action is taken, and the user is brought back to the calendar screen

### **Use Case #3: Contact Tech Support**

Diagram



Brief Description

The user contacts technical support when facing a problem.

Initial Step-By-Step Description

Before this use case can be initiated, the user has already logged in the TopFit app.

1. The user selected Support option from the side menu.

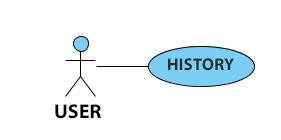
2. The system uses the Button\_Support to bring up the Contact Technical Support screen.

3. The user provides information about the problem in text box and presses the Submit button.

4. The system uses the Button\_Support\_Submit to generate a ticket with a problem.

**Use Case #4: Activity History**

Diagram



Brief Description

The user receives tracks of their activity

Initial Step-By-Step Description

Before this use case can be initiated, the user has already logged in the TopFit app.

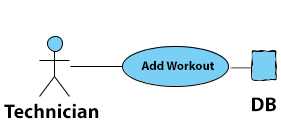
1. The user selected History option from the side menu.

2. The system uses the Button\_History to bring up the History screen.

3. The system generates history of user's daily activities, retrieving the information from the database table.

**Use Case #5: Add New Workout**

Diagram



Brief Description

The technician adds new workout routine to the system.

Initial Step-By-Step Description

Before this use case can be initiated, the technician has already logged in the TopFit app.

1. The admin user (technician) selects Add Workout button from the side menu.

2. The system uses the Button\_Tech\_Add\_Workout to bring up the Add Workout screen.

3. The admin user (technician) provides information about the new workout, uploads the data and presses Submit button.

4. The System uses the Button\_Tech\_Add\_Workout\_Submit to persist the new workout in database.

## **Data Modelling and Analysis**

### **Normalized Data Model Diagram**

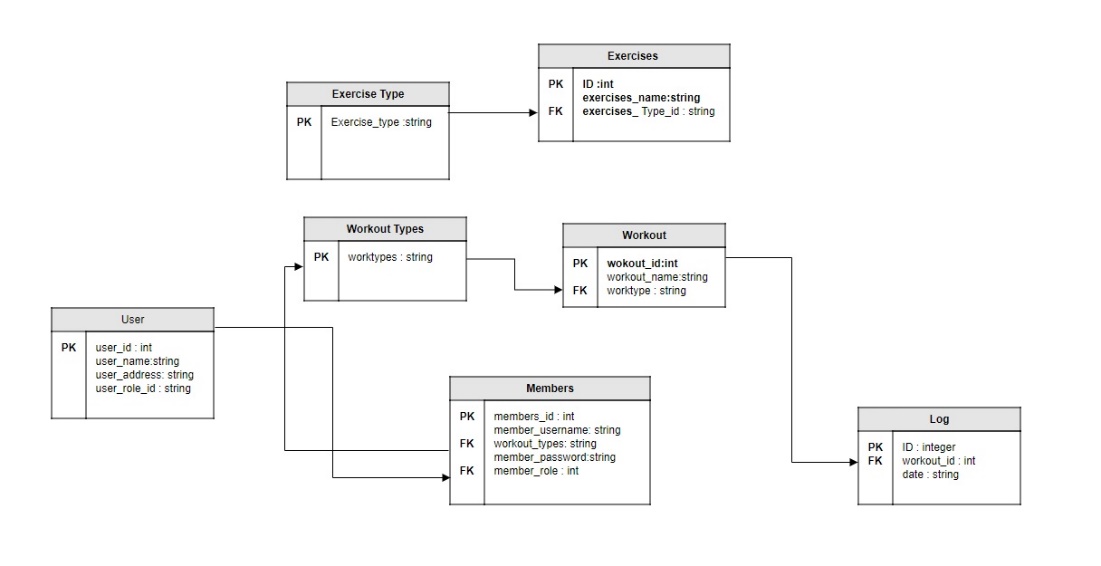


Figure 1.Normalized Data Model Diagram

### **Activity Diagrams**

Figure 2. Select Workout

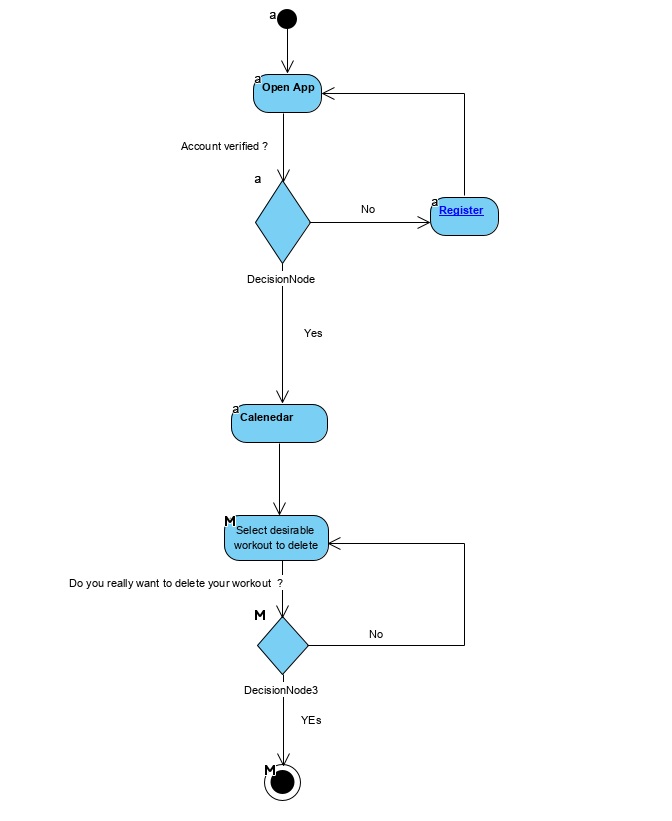


Figure 3. Delete Workout

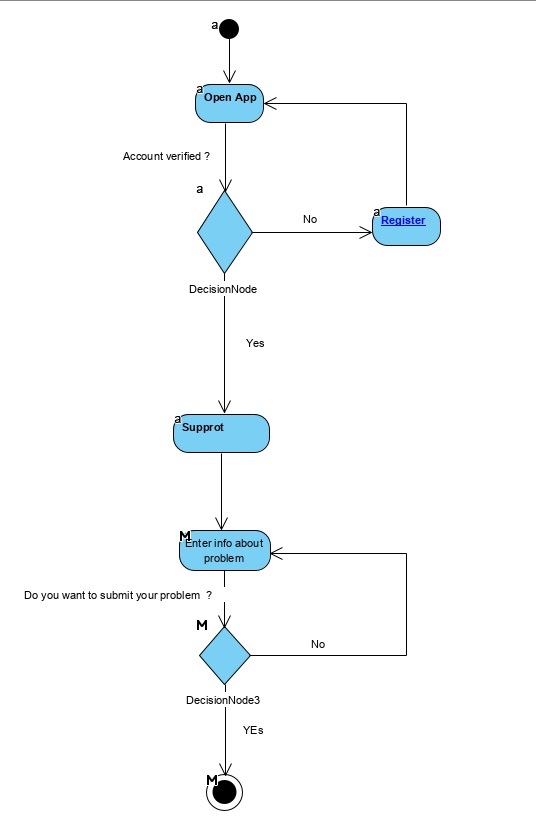


Figure 4. Contact Tech Support

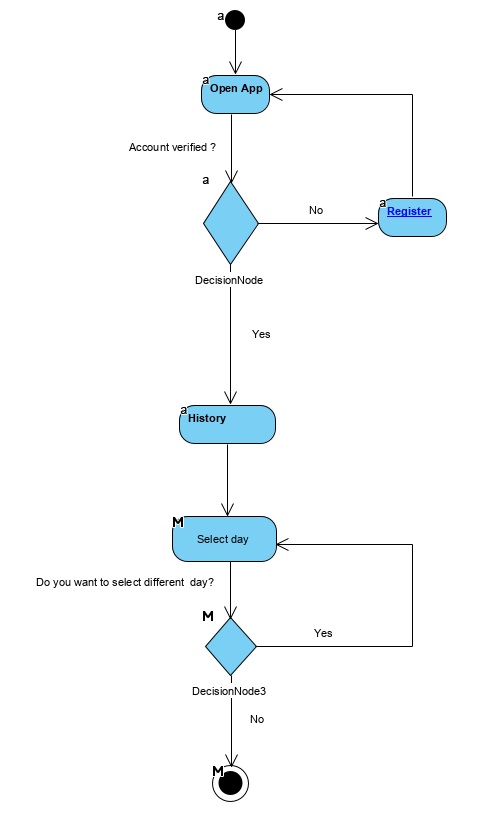


Figure 5. Workout History

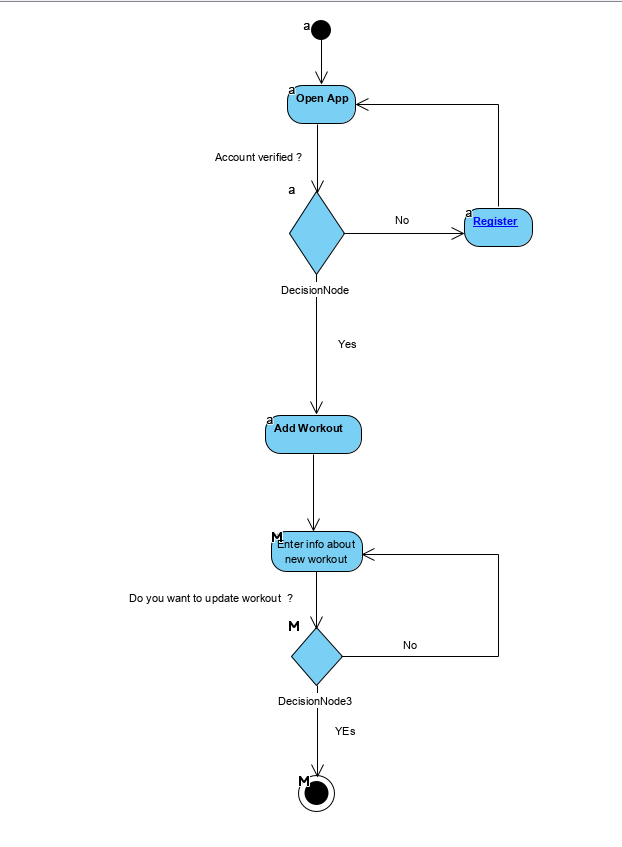


Figure 6. Add New Workout

### **Sequence Diagrams**

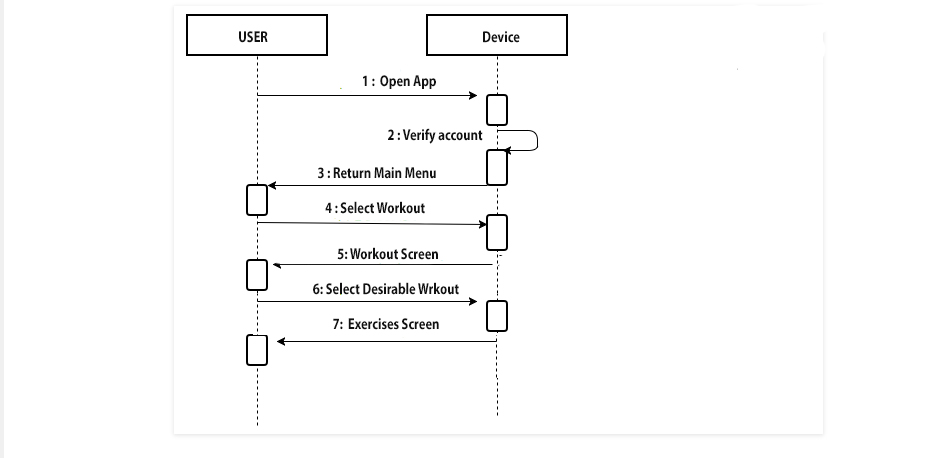


Figure 7. Select Workout

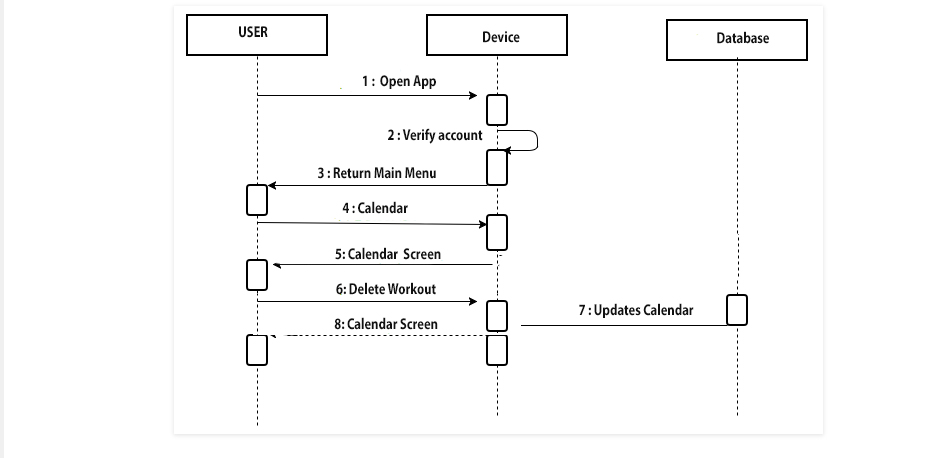


Figure 8. Delete Workout

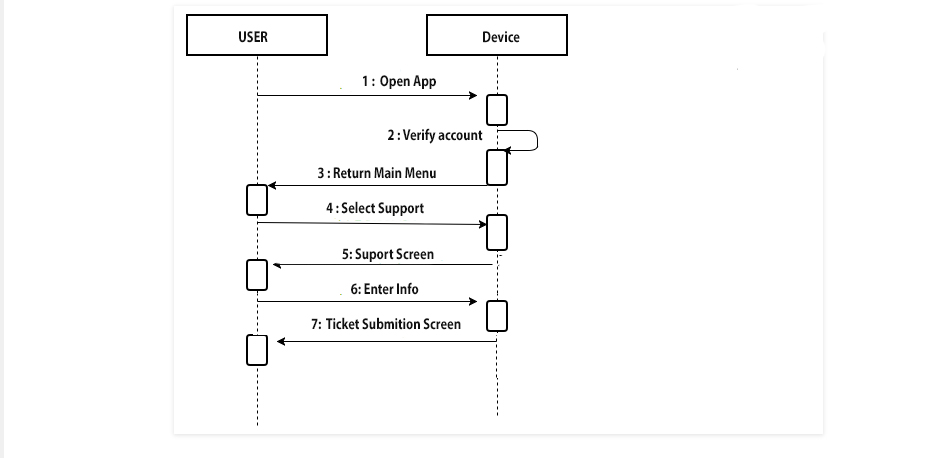


Figure 9. Contact Tech Support

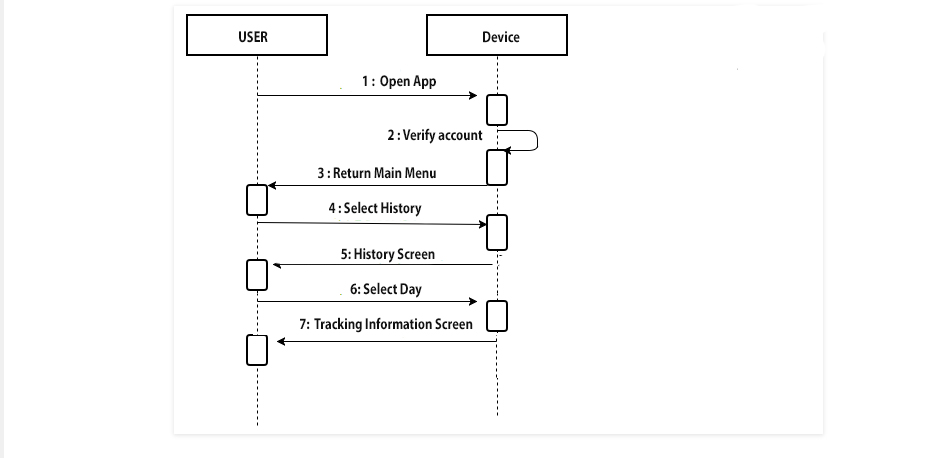


Figure 10. Activity History

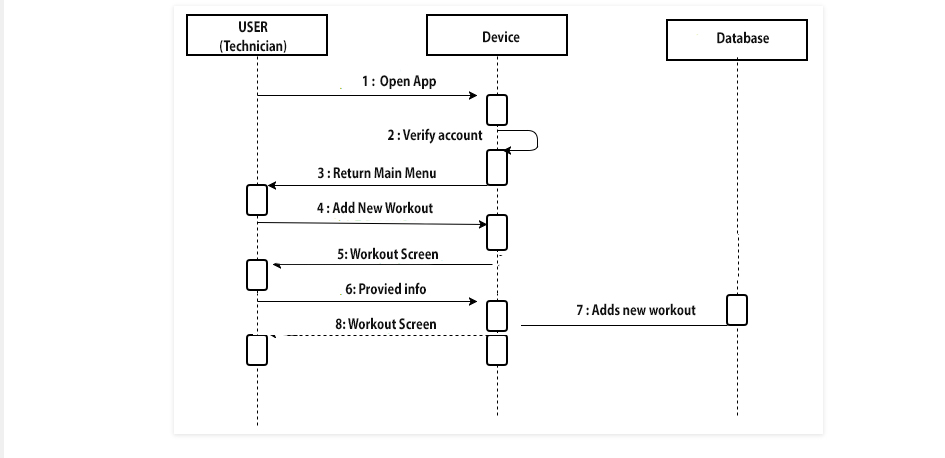


Figure 11. Add Workout

### **UML Class Diagram**

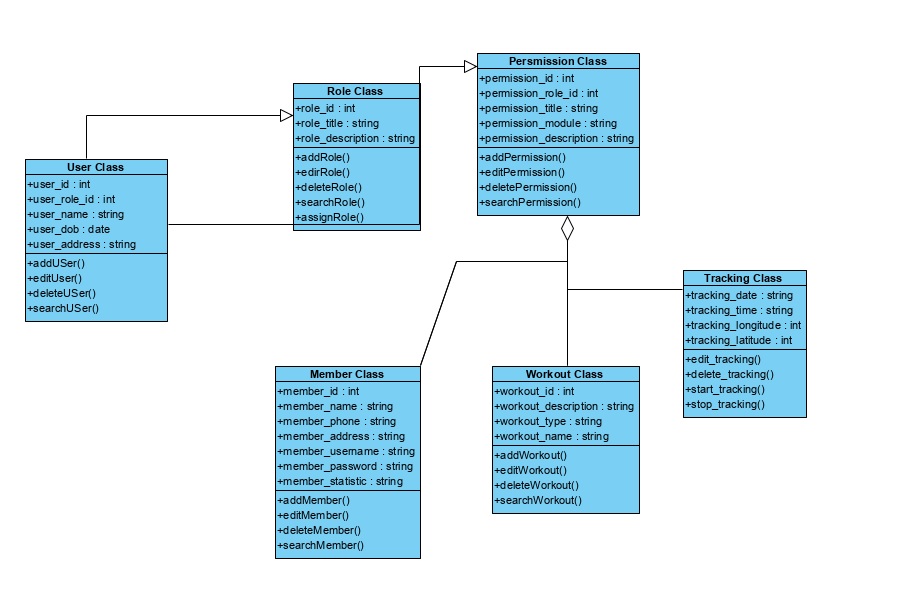


Figure 12. Class Diagram

## **Process Modelling**

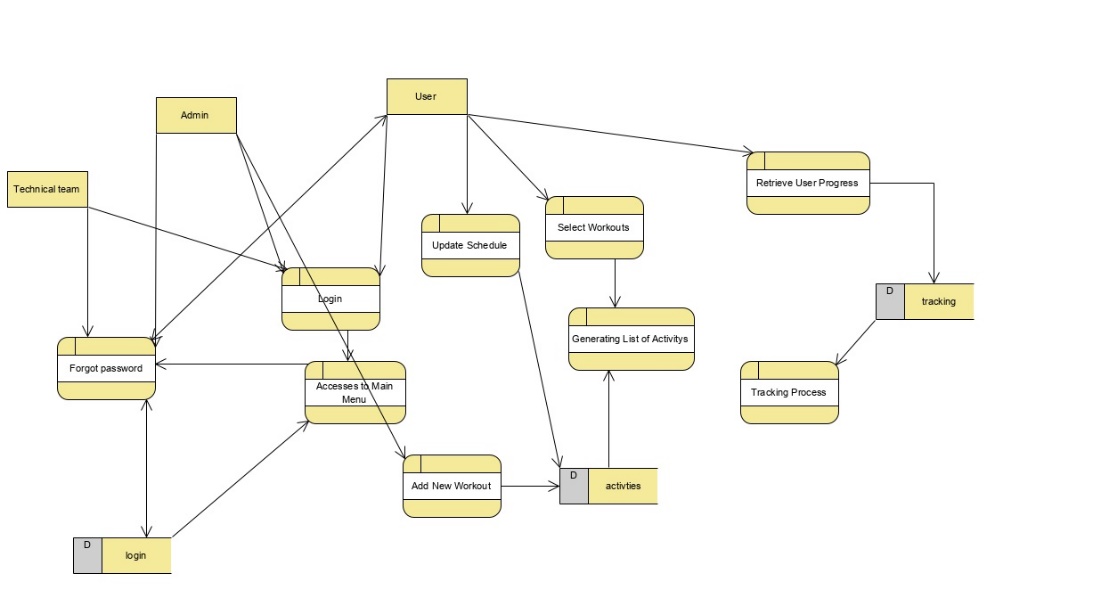


Figure 13. Data Flow Diagram

# Non-Functional Requirements

Non-Functional Requirements (NFR) of the system define its quality attribute. They represent a set of standards used to judge the specific operation of a system disregarding the functionality. For example, how fast does TopFit application load? Properly identifying non-functional requirements in the project is essential for the usability and effectiveness of the entire application. Failing to meet NFR can result in producing a system that fails to satisfy the users.

The development team has come up with the list of NFR for the TopFit application:

* **Performance**:
  + the start up time for TopFit shouldn’t exceed 3 seconds
  + the tasks and services should run in the background and not block the UI
  + the application must not significantly affect battery life or slow the entire device down
  + the system must be fully operational at least 85% of the time.
  + down time after a failure shall not exceed 4 hours.
* **Scalability**:
  + app should able to adjust to increased usage rates and handle more data as time progress.
  + when the user adds their workout schedule, the system should be capable of storing and manipulating the data without delays
* **Usability**:
  + the user should be able to navigate the application without any training from experts/support documentation
  + the technician must not take more than 2 days to familiarize themselves with the application source code
* **Security**:
  + the application data should be securely stored
  + the data must be inaccessible from outside environment and invulnerable for hacker attacks
  + an authentication token will be saved on local devices to be used with a pair of keys (public and private)
* **Availability**:
  + the first release of the application will be available at Google Play Market and via Direct Installs in the APK format.
  + version 1.0 of TopFit will only be intended to use on Android devices, with a potential of future expansion
* **Maintainability**:
  + the team chose to use Git VCS to conduct the project. Should a bug be found, it will be easy to roll back to a stable version.
  + automatic and manual testing will be performed prior to releasing the application to the market
  + if a user happens to discover an inconsistency, they have an option to contact the development team.
  + upcoming versions of the application will target to address user concerns, improve application performance and develop new features.
* **Portability:** 
  + the first release of the application will be developed on such a version of Android that it will run on all the mobile phones.
  + in the later releases, newer Android versions will be supported, and the interface will be extended for use on other devices, such as Smart Watch, tablets etc.

# Logical Database Requirements

The Logical Database Requirements for the first TopFit version is defined in MySQL database that includes 4 base tables: LOGIN, CLIENT\_INFO, ACTIVITY and LOG. The database will reside on a cloud server. Considering the budget of project (or, to be more exact, the absence of such) free hosting / free services will be utilized. The logical database model is designed based on the Conceptual Data Model. The latter considers all Business requirements for the current version of the application.

The **LOGIN** table is designed to authenticate the client and route the data to the appropriate path.

The **CLIENT\_INFO** table is responsible for secure personal information.

The **ACTIVITY** will include all activity that users undertake, recorded by the GPS tracker.

The **LOG** carries all statistical information that will be used for composing user profile statistics, making estimations and predictions. //fav workout, suggestions

As mentioned before, the ER model was done by fulfilling the requirements. All documentation, including ER models, Conceptual Data Model and the necessary research reflects the user needs. This information is forward to the next level - Physical Data Model.

LOGIN ER model id displayed in figure 1.

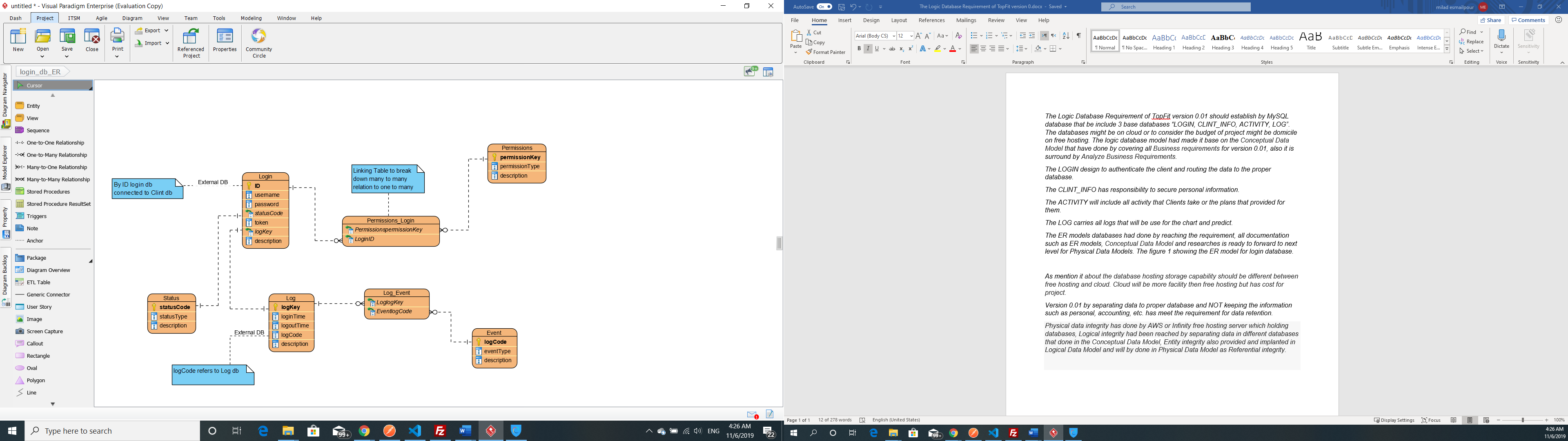


Figure 14. ER model for LOGIN database

Regarding the database hosting storage, the capacity is different depending on the storage type. Free hosting provides less functionality, but cloud server with all its benefits will implies expenses that the team is unable to afford.

Version 1.0 of the application includes separating data among appropriate tables and by no means storing the sensitive data such as personal information and accounting details insecurely.

**Physical data integrity** will be maintained by AWS or Infinity free hosting servers that allow for sufficient storage. As for the **Logical integrity**, it was reached by separating data in different database tables that was reflected in the Conceptual Data Model. **Entity integrity** is also crucial for the project success. It was provided and implanted in Logical Data Model and will be completed in Physical Data Model as **Referential integrity**.

The database requires a Windows, Mac OS or any other UNIX-based environment to function correctly. The estimated resources, keeping in mind the database growth, are as follows:

* 150 GB hard drive storage
* 1 GB RAM
* MySQL installed and properly configured

Should the implemented model be successful, the user will be able to see a coherent list of information from a central location that is retrieved from many file sources tied into the storage system.

# Approval

The signatures below indicate the approval of this document.

|  |  |  |
| --- | --- | --- |
| Project Role | Name | Date |
| Back-end Java Developer, project leader | Nadezhda Mokhireva | November 6, 2019 |
| Front-end Developer, UI Designer | Vladyslav Bordiug | November 6, 2019 |
| Back-end Java Developer, MySQL Database administrator | Mahdi Esmaeelpour | November 6, 2019 |
| Front-End Developer, Tester (for Alpha-version) | Parisa Khataei | November 6, 2019 |