Software Requirements Specification

for

FitBud

Version 1.0 approved

Prepared by SAMMY

Nanyang Technological University

3rd November 2022

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 2

1.5 References 2

2. Overall Description 3

2.1 Product Perspective 3

2.2 Product Functions 4

2.3 User Classes and Characteristics 5

2.4 Operating Environment 5

2.5 User Documentation 6

2.6 Assumptions and Dependencies 7

3. External Interface Requirements 7

3.1 User Interfaces 7

3.2 Hardware Interfaces 12

3.3 Software Interfaces 13

3.4 Communications Interfaces 13

4. System Features 14

4.1 Register User 14

4.2 Authenticate User 16

4.3 Create Personalised Training Plan 17

4.4 Track Fitness Progress 19

4.5 Discover Nearby Users 20

5. Non-functional Requirements 20

5.1 Performance Requirements 20

5.2 Safety Requirements 21

5.3 Security Requirements 21

5.4 Software Quality Attributes 21

5.5 Business Rules 22

Appendix A: Glossary 23

Appendix B: Analysis Models 24

# Introduction

## Purpose

The software requirements specified in this document provides supportive information on FitBud. This document will explain the purpose, features and interface of the application. In addition, the document will present a detailed description about the operating environment, design and implementation constraint of FitBud.

## Document Conventions

The format of the software requirement specifications are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Font Type** | **Font Size** | **Bold** |
| Section Headers | Arial | 18 | Yes |
| Sub-Section Headers | Arial | 14 | Yes |
| Normal Text | Arial | 11 | No |

## Intended Audience and Reading Suggestions

This software requirement specifications document is intended for developers, product managers, project managers, users and testers.

Developers can review the application capabilities and understand which section of the application should be targeted to add new features or improve the capabilities of the application for further enhancement. Developers could read this document in the following order:

1. Overall Description
2. External Interface Requirements
3. System Features
4. Other Non-Functional Requirements

This documentation can help project managers in the planning and execution of their project. For project managers the suggested sequence for reading is:

1. Introduction
2. Overall Description
3. Other Non-Functional Requirements

This documentation can help product managers in setting a product vision and the development of the product. For project managers the suggested sequence for reading is:

1. Introduction
2. Overall Description
3. Other Non-Functional Requirements
4. External Interface Requirements
5. System Requirements

This documentation can help testers better identify bugs by understanding what the software requirements are. For testers the suggested sequence for reading is:

1. Overall Description
2. External Interface Requirements
3. System Requirements
4. Other Non-Functional Requirements

This documentation can help users understand the purpose and the functionality of the application. For users the suggested sequence for reading is:

1. Introduction
2. Overall Description
3. Other Non-Functional Requirements

## Product Scope

The product implementation is FitBud. FitBud is a mobile application which is primarily aimed to help full-time national servicemen (NSFs) and operationally ready national servicemen (NSMen) to pass their individual physical proficiency test (IPPT). IPPT is a standard physical test that NSFs and NSMen need to complete as part of their national service obligations. The test consists of 3 physical tests – push-ups, sit-ups and 2.4km run. This app could also be used by fitness enthusiasts to get fitter.

FitBud aims to provide fitness exercises for users to accomplish and thereafter, track their progress. Through this, FitBud aims to make its users fitter and healthier. Ultimately, improving their IPPT score.

## References

For SRS documentation, we utilized information from websites to design a suitable format and standard. The references are:

* Singapore Government Data - <https://data.gov.sg/>
* MongoDB - <https://www.mongodb.com/docs/>
* AWS - <https://docs.aws.amazon.com/>
* React Native - <https://reactnative.dev/docs/getting-started>
* Expo - <https://docs.expo.dev/>

# Overall Description

## Graphical user interface, application Description automatically generatedProduct Perspective

The above diagram represents the overall system architecture of FitBud.  
  
FitBud is a new, self-contained product. FitBud does access government database APIs to access data for IPPT training venues and exercises venue, and ExRx.net for exercises. Our app builds upon this data using our own algorithm to determine the optimal exercises and nearest venue for the user.

## Diagram Description automatically generatedProduct Functions

The diagram above represents the use case diagram of FitBud. A high-level summary of the use case is as follows:

* **Register User**
  + To prompt for user information during registration.
* **Authenticate User**
  + To prompt for username and password.
* **Create Personalized Training Plan**
  + To create a personalized training plan for the user based on their current and target fitness.
* **Track Fitness Progress**
  + To track user’s push-ups, sit-ups and run timings.
* **Discover Nearby Users**
  + To connect nearby users with similar IPPT scores and arrange meet-ups for users.

## User Classes and Characteristics

We anticipate that there would be two category of users:

1. **NSFs and NSMen** – users that want to improve their IPPT score
2. **Fitness Enthusiasts** – users that are looking to get fit

|  |  |  |  |
| --- | --- | --- | --- |
| **User Type** | **Purpose** | **Frequency of Use** | **Product Functions Used** |
| **NSFs/NSMen** | To achieve their IPPT goals | Everyday | * Customized Fitness Training Plan * Tracking of Fitness Progress * Discover Nearby Users |
| **Fitness Enthusiasts** | To get fitter | Weekly | * Customized Fitness Training Plan * Tracking of Fitness Progress * Discover Nearby Users |

Based on our product vision and the user profile, our primary target audience are NSFs and NSMen.

## Operating Environment

* React-Native Dependencies
  + react-native-async-storage/async-storage: 1.17.10
  + react-native-community/checkbox: 0.5.12
  + react-native-community/datetimepicker: 6.2.0
  + react-navigation/bottom-tabs: 6.4.0
  + react-navigation/native: 6.0.13
  + react-navigation/native-stack: 6.9.0
  + tensorflow-models/posenet: 2.2.2
  + tensorflow/tfjs: 4.0.0
  + tensorflow/tfjs-converter: 4.0.0
  + tensorflow/tfjs-react-native: 0.8.0
  + types/react: 18.0.0
  + types/react-native: 0.69.1
  + expo: 46.0.13
  + expo-asset: 8.6.1
  + expo-auth-session: 3.7.1
  + expo-camera: 12.3.0
  + expo-checkbox: 2.2.0
  + expo-font: 10.2.0
  + expo-gl: 11.4.0
  + expo-gl-cpp: 11.4.0
  + expo-permissions: 13.2.0
  + expo-random: 12.3.0
  + expo-splash-screen: 0.16.2
  + expo-status-bar: 1.4.0
  + expo-web-browser: 11.0.0
  + in: 0.19.0
  + react:18.0.0
  + react-native: 0.69.6
  + react-native-canvas: 0.1.38
  + react-native-css-transformer: 1.2.4
  + react-native-fs: 2.20.0
  + react-native-gifted-chat: 1.0.4
  + react-native-modal-datetime-picker: 14.0.0
  + react-native-safe-area-context: 4.3.1
  + react-native-screens: 3.15.0
  + react-native-status-bar-height: 2.6.0
  + react-native-webview: 11.23.0
  + terminal: 0.1.4
  + typescript: 4.6.3
  + babel/core: 7.12.9
* API Server
  + googlemaps/google-maps-services-js": "^3.3.16
  + axios: 1.0.0
  + bcrypt: 5.1.0
  + cors: 2.8.5
  + dotenv: 16.0.3,
  + express: 4.18.1
  + mongoose: 6.6.4
  + node-fetch-commonjs: 3.2.4
  + strava-v3: 2.1.0
  + nodemon: 2.0.20

## User Documentation

FitBud’s user documentation would consists of a user manual, system reference, online help, tutorial and a demonstration of FitBud.

Since most of our users are unlikely to be technically proficient, it is imperative for us to adopt the following standards in our documentations:

1. Simple words and short sentences
2. Shorter verbs and nouns to show operations instead of explaining a long process

Therefore, user documentation has to be clear and readable. Format requirements are as follows:

1. Proper indentation
2. Headings, page styles and section titles
3. Consistent text fonts and styles
4. Insert pictures for key steps and buttons
5. Using bold text to highlight dangerous or invalid operations

## Assumptions and Dependencies

Assumptions:

1. Users are physically capable and able to perform the exercises
2. Users are aware of how to perform the exercises
3. Users have the integrity and key in valid information for the optimal experience
4. The data from the third-party providers are accurate and true
5. The computer vision model is accurately trained

Dependencies:

1. Government database APIs to access IPPT venues and training locations
2. Database of exercises derived from ExRx.net
3. IPPT Score calculator from the internet
4. TensorFlow model for CV

# External Interface Requirements

## User Interfaces

**Starting Page** – A landing page for users who have installed FitBud. Through this, new users can register for an account while existing users can login into their account.

| **Page** | **Description** | **Image** |
| --- | --- | --- |
| **Starting Page** | A landing page for users who have installed FitBud. Through this, new users can register for an account while existing users can login into their accounts. | A picture containing text  Description automatically generated |
| **Registration Page** | Registration Page for users to key in their personal information. | Graphical user interface, text, application, chat or text message  Description automatically generated |
| Graphical user interface, text, application, chat or text message  Description automatically generatedGraphical user interface, text, application, chat or text message  Description automatically generated |
| **Main Landing Page** | The user is greeted with the main landing page when he logins into the app.  In this page, the user can see the number of days to their IPPT test and their IPPT location.  Below the test location, the user can see the recommended exercises.   This layout was chosen to remind the user of their upcoming IPPT and motivate them to get fitter. | Graphical user interface  Description automatically generated with medium confidenceGraphical user interface  Description automatically generated with medium confidence |
| **Bonus Exercise Page** | Upon completion of the exercises, the user will be able to see bonus exercises below.  This serves as additional exercises for the user to perform in order to get fitter. In addition, it displays the nearest venue the user can carry out the bonus exercises. | Graphical user interface, text, application  Description automatically generated |
| **Social Page** | In this page, it shows nearby users with similar fitness level.  The purpose of the page is for the user to connect with other users of similar fitness level. This way users can chat and meet-up one another in the hopes that it will make them grounded and motivated to get fitter. | Teams, company name  Description automatically generated with medium confidence |
| **Train Page** | In the train page, users can have their push-ups, sit-ups and run tracked.   The push-ups and sit-ups are tracked through FitBud’s computer vision model. The model will feedback to the user on their form and repetitions.  The run is tracked through pulling data from Strava. | Text  Description automatically generated |
| **Profile Page** | The profile page displays the user’s age, previous IPPT result and their upcoming IPPT date. | Graphical user interface, text, application, chat or text message  Description automatically generated |

## Hardware Interfaces

The hardware interface components are as follows:

* Smartphone (Either iPhone or Android) – Presentation Layer in Software Architecture
* Server API (AWS) – Business App Logic Layer in Software Architecture
* MongoDB Database Server – Persistent Data in Software Architecture

## Software Interfaces

* Smartphone (iPhone/Android)
  + Expo platform for developing the app using React Native
  + Expo camera (12.3.0) to obtain live video feed of user doing exercises
  + Expo auth (3.7.1) for authenticating user with Strava
  + Tensorflowjs (4.0.0) posenet model for detecting correct pose in pushup and situp
  + Fetch API to send requests to the Server API endpoints through POST, PUT and PATCH requests
  + React Context API to manage global state in the application
* Server API
  + Amazon Linux 2 as the backend operating system
  + NodeJS (18.12.1) as the web server
  + MongoDB (6.0) as the database to store all user profile details, messages, and exercises
  + ExpressJS (4.18.1) to set up a router
  + Google Maps API (google-maps-services-js: ^3.3.16) to calculate distances between points and for geocoding
  + Strava-v3 (2.1.0) SDK for retrieving run data
  + Mongoose (6.6.4) for interfacing between NodeJS and MongoDB
  + Bcrypt (5.1.0) to hash passwords so they are encrypted at rest

## Communications Interfaces

* Secure Hypertext Transfer Protocol Communication (Mobile Application to Server API)
  + HTTP encrypted by TLS which provides authentication of the website and associated web server with which one it is communicating with, which protects against man-in-the-middle attacks.
  + Additionally, it provides bi-directional encryption of communications between a client and a server, which protects against eavesdropping and tampering with or forging the contents of communication.
  + Primarily, HTTP put request is to register user, HTTP post function is used to implement login functions, HTTP patch function is to update user info (such as after completing exercises)
* Secure Shell Protocol to connect to backend server hosted on cloud to make edits to API source code and the database
  + Cryptographic network protocol for operating network services
  + Enabled remote login and command-line execution
  + Connected local SSH client instance with cloud SSH server.
* MongoDB database connection
  + When NodeJS receives a post, patch or put request, there is a need to read or write data from the MongoDB database. A connection to the MongoDB database is opened via port 27017 for data transfer to occur
* Bcrypt encryption is used when the password is stored at rest in MongoDB to improve security

# System Features

## Register User

4.1.1 Description and Priority

The app will prompt for user information during registration. This is a high priority feature.

* + 1. Stimulus/Response Sequences

|  |
| --- |
| Sequence |
| 1. The user launches the app for the first time. 2. The user is prompted with the option to register. 3. The user clicks on the register option. 4. The user must input their desired profile information.  * Alt: If the user enters an already previously registered username, the app will prompt the user to change the username * Alt: If the desired password does not meet the requirements, the user will be prompted to change to a new one.  1. The user must input their personal details. 2. The user must input their current abilities. 3. The user must input their target abilities. 4. The app will create a new user account for the user. |

4.1.3 Functional Requirements

* 1. The app must prompt the user for their registration information for first time users.
     1. The app must prompt for the user’s username, which must have at least one character and less than 10 characters.
     2. The app must prompt for the user’s account password which length must be at least 8, have at least 1 letter, have at least 1 number and must not contain the user’s username.
  2. The app must prompt for the user’s personal details.
     1. The app must prompt for the user’s name.
     2. The app must prompt for the user’s date of birth.
        1. The app will calculate the user’s age, which must be between 18 and 45 years old.
        2. The app must prompt for the user’s residential postal code, which must be a valid address.
  3. The app must prompt for the user’s current fitness abilities.
     1. The app must prompt for the user’s current number of push-ups he can perform in a minute, which must be between 0 and 100.
     2. The app must prompt for the user’s current number of sit-ups he can perform in a minute which must be between 0 and 100.
     3. The app must prompt for the user’s current timing for his 2.4km run in seconds, which must be between 0 and 1100.
     4. The app must calculate the user’s current IPPT score grading using his current fitness abilities and the IPPT scoring metrics.
     5. The app must calculate the user’s current IPPT grade using his current IPPT score.
  4. The app must prompt for the user’s target IPPT score.
     1. The app must prompt for the user’s target number of push-ups performed in a minute, which must be between 0 and 100.
     2. The app must prompt for the user’s target number of sit-ups performed in a minute, which must be between 0 and 100.
     3. The app must prompt for the user’s target 2.4km run timing in seconds, which must be between 0 and 1100.
     4. The app must calculate the user’s target IPPT score grading using his target fitness abilities and the IPPT scoring metrics.
     5. The app must calculate the user’s target IPPT grade using his target IPPT score.

## Authenticate User

4.2.1 Description and Priority

The app will authenticate the user’s username and password before the user can use the app’s functionalities. This is a high priority feature.

4.2.2 Stimulus/Response Sequences

|  |
| --- |
| Sequence |
| 1. The user opens the app and is directed to log in page. 2. The app will prompt the user for his username and password. 3. The app will authenticate the user input and returns the result. 4. If successful, the app will redirect the user to the Plans Screen and fetch the user’s details  * Alt: If user input fails the authentication, the app will display a general error message stating the error and prompt the user to try again. |

4.2.3 Functional Requirements

* 1. The app must prompt for the user’s login details.
     1. The app must prompt for the user’s username.
     2. The app must prompt for the user’s password.
  2. The app must react to the status of the login attempt.
     1. Upon successful login, the app will redirect the user to the Plan Screen and fetch user details
     2. Upon unsuccessful login, the app will display an error message.

## Create Personalised Training Plan

4.3.1 Description and Priority

The app presents user with a personalized workout plan consisting of daily training exercises, extra related exercises and nearby training venues. This is a high priority feature.

4.3.2 Stimulus/Response Sequences

|  |
| --- |
| Sequence |
| 1. The user navigates to the Plan Page of the app. 2. The app displays the number of days till the user’s IPPT Test. 3. The app recommends the nearest IPPT testing venue from the user’s residential location. 4. When the user navigates to the daily training section of the page, the app recommends the number of repetitions of Push-ups and Sit-ups to perform within a minute and the duration for a 2.4km run. 5. When the user navigates to the additional related exercise section of the page, the app recommends a workout of exercises as well as a suitable nearby training venue |

4.3.3 Functional Requirements

* 1. The app must display the number of days from the current date until the user’s IPPT test date.
  2. The app must recommend the nearest IPPT testing venue from the user’s residential location.
     1. The app must query the database for the IPPT testing venue nearest to the user’s residential location.
     2. The app must display the nearest IPPT testing venue for the user’s reference.
  3. The app must recommend the appropriate daily training, consisting of sit-ups, push-ups and 2.4km run timings to be performed based on the exercise algorithm.
     1. The app must recommend the appropriate repetition of push-ups for the user to be complete within a minute.
     2. The app must recommend the appropriate repetition of sit-ups for the user to be complete within a minute.
     3. The app must recommend the appropriate timing for the user to complete a 2.4km run.
     4. For each activity, the app must display the status of the activity.
        1. The app must allow users to mark the completion of a particular activity.
  4. The app must recommend additional related exercises that can improve the user’s fitness abilities after the user has completed his daily training exercises.
     1. The app must query from its database for a recommended workout that is suitable for the fitness activities in which the user needs to improve upon.
     2. The database must return a recommended workout that is catered to those areas for improvement.
     3. The app must display the recommended workout to users in an appropriate format.
        1. For each exercise on the recommended workout, the exercise information must include the name of the exercise.
        2. For each exercise on the recommended workout, the exercise information must include how many repetitions to perform.
        3. For each exercise, on the recommended workout, the exercise information must contain the status of each exercise.
           1. The app must allow users to mark the completion of a particular exercise.
     4. The app must recommend a suitable nearby venue for the user to engage in exercise.
        1. The app must query from its database for an exercise venue that has the required equipment to do the workout and is the closest to the user’s residential address.

## Track Fitness Progress

4.4.1 Description and Priority

The app must have the option for the user to log any of the 3 exercises of Push-ups, Sit-ups and 2.4km run for progress tracking. This is a high priority feature.

4.4.2 Stimulus/Response Sequences

|  |
| --- |
| Sequence |
| 1. The user navigates to the training page of the app 2. The app provides options for Push-ups, Sit-ups and 2.4km run 3. When push-ups is selected, the app will track the number of proper push-ups the user completes within 1 minute, while providing live-feedback during that 1 minute. 4. When sit-ups is selected, the app will track the number of proper sit-ups the user completes within 1 minute, while providing live-feedback during that 1 minute. 5. When 2.4km run is selected, the app will obtain the user’s latest run data from Strava by use case 4.4 and calculate the estimated duration to complete a 2.4km run. |

4.4.3 Functional Requirements

* 1. The app must be able to accurately track and correct the user’s push-ups during his attempts.
     1. The app must allow the user to begin an attempt which will start a 60 second timer.
     2. The app must classify the correct push-up form from an incorrect one with an accuracy of 90%.
     3. The app must count the number of correct push-ups performed by the user within the attempt.
     4. The app must display live feedback on the screen to correct the user’s form during the attempt.
     5. The app must have the option for users to submit the results of their attempt to be stored.
     6. The app must have the option for users to redo their attempt as many times as desired.
  2. The app must be able to accurately track and correct the user’s sit-ups during his attempts.
     1. The app must allow the user to begin an attempt which will start a 60 second timer.
     2. The app must classify the correct sit-up form from an incorrect one with an accuracy of 90%.
     3. The app must count the number of correct sit-ups performed by the user within the attempt.
     4. The app must display live feedback on the screen to correct the user’s form during the attempt.
     5. The app must have the option for users to submit the results of their attempt to be stored.
     6. The app must have the option for users to redo their attempt as many times as desired.
  3. The app must be able to track the user’s 2.4km run timing.
     1. The app must be able to retrieve the user’s latest running information from Strava through API calls.
        1. The app must retrieve the run’s details of distance and duration.
     2. The app must calculate the duration the user takes to complete a 2.4km
     3. The app must store the estimated duration.

## Discover Nearby Users

4.5.1 Description and Priority

The app will allow the user to discover other nearby and message one another to arrange meet-ups to exercise together. This is a high priority feature.

4.5.2 Stimulus/Response Sequences

|  |
| --- |
| Sequence |
| 1. The user navigates to the social page of the app 2. The app will display a list of users generated for the user 3. The user has the option to message with any user on the list to arrange for an exercise meet-up. |

4.5.3 Functional Requirements

* 1. The app shall recommend to the user a maximum of 10 users who are within a radius of 300m from the user’s residential addresses.
     1. For each of the recommended users, the app must display the user’s details which includes name, profile picture, IPPT score and IPPT grade
  2. The app must allow the user to be able to start a private chat with recommended users.

# Non-functional Requirements

## Performance Requirements

* The app must be able to begin operation within 30 seconds of opening the app
  + Reduces the waiting time of the user to improve the user experience.
* The app must be able to upload and fetch data from the system database within 1 minute for every call.
  + Increases the efficiency of the user’s experience to allow them to reduce time wasted.
* The app will not crash during use.

## Safety Requirements

* The app will not count the incorrect form of the user while doing the exercise
  + Prevents the user from continuing with their incorrect form, which can cause harm to their bodies through injuries

## Security Requirements

* User Login Details
  + The user will not be allowed to register with a password that includes their username, does not include both letters and numbers, and is shorter than 8 characters.
* User Data
  + The user will not be able to access the personal data of other users through any means.
  + There will be no localized storage of user data within the app. All data will be stored on the database.
  + Only the user will be able to modify their data.

## Software Quality Attributes

* Availability
  + The app must always be available for a new user to register a new account.
* Correctness
  + The app will always display the correct profile data of the user.
* Flexibility
  + The app will be usable with older versions of the mobile phone’s operating system.
* Maintainability
  + The app must be able to receive updates without having to change the entire source code.
  + The app must include SOLID design principles.
* Reliability
* The app will function when the user boots it within 30 seconds.
* The app will be able to judge the user’s form within 10 seconds of selecting the train exercise use case.
* Reusability
  + Users must be able to use the app daily.
  + The app must be able to track a user’s exercises 3 times a day corresponding to the number of sets of exercises the user is required to complete.
* Robustness
  + The app must not crash when the user tries to register an invalid user.
  + The app must not crash when the user fails to authenticate and login.
* Testability
  + The developer must be able to easily test the output when tracking the user’s exercise completion and give verbose, accurate feedback on the user’s exercise form.
* Usability
  + The app must function on different devices, and the user should have their information stored in the database so that they can maintain and track their information on different devices.

## Business Rules

* Users
  + Register account
  + View user data
  + Modify user data
* Developers
  + Manage data fields
  + Develop source code
  + Troubleshoot bug fixes

# Appendix A: Glossary

|  |  |
| --- | --- |
| Term | Definition |
| IPPT | Individual Physical Proficiency Test. An annual fitness test administered to Singaporeans males who have undergone National Service and are between the ages 18 to 45 years old. |
| Fitness ability | Refers to a collection of repetition of push-ups within a minute, repetition of sit-ups within a minute, and duration for a 2.4km run timing |
| IPPT score | Measured using a standardized IPPT scoring metrics which depends on the user’s fitness ability and adjusts for the age of individuals. |
| IPPT grade | Pass, Pass with Incentives, Silver or Gold  Grade awarded according the standardized IPPT grading system which solely depends on the IPPT score |
| IPPT test date | The date of the user’s IPPT test date is determined by the day before his date of birth. |
| API | Refers to application programming interface, which is a set of definitions and protocols for building and integrating application software. |
| Exercise venue | Refers to parks, training facilities and gyms. |
| Correct form | Refers to IPPT’s standard for a push-up or a sit-up respectively. |
| Distance/Nearest | Euclidean distance will be used for distance measurement requirements of the app. |
| Database | Refers to an organised collection of structured information, or data, typically stored electronically in a server. |
| Record | Writing to a cloud database |
| Strava | A well-known external mobile application that users use to track user’s running information such as distance and duration. |
| Appropriate repetition | Recommended fitness ability must be reasonable for the user to accomplish without being too ambitious while ensuring users are progressively challenged to improve through the usage of the exercise algorithm |
| Exercise algorithm | A function that considers the user’s past fitness abilities while taking into consideration his target fitness level and duration to his IPPT test, to recommend the appropriate target fitness ability. |
| Workout | A collection of different exercises a user can perform to improve his fitness ability. |

# Appendix B: Analysis Models

Chart, box and whisker chart

Description automatically generated