Software Requirements Specification

for

CrossFit

Version 1.0

Prepared by Shivalik Pandita, Pratik Pujari

Bharatiya Vidya Bhavan's Sardar Patel Institute of Technology

07 September 2022

Table of Contents

Table of Contents ii
Revision History 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 1
1.5 References 1
2. Overall Description 2
2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Classes and Characteristics 2
2.4 Operating Environment 2
2.5 Design and Implementation Constraints 2
2.6 User Documentation 2
2.7 Assumptions and Dependencies 3
3. External Interface Requirements 3
3.1 User Interfaces 3
3.2 Hardware Interfaces 3
3.3 Software Interfaces 3
3.4 Communications Interfaces 3
4. System Features 4
4.1 System Feature 1 4
4.2 System Feature 2 (and so on) 4
5. Other Nonfunctional Requirements 4
5.1 Performance Requirements 4
5.2 Safety Requirements 5
5.3 Security Requirements 5
5.4 Software Quality Attributes 5
5.5 Business Rules 5
6. Other Requirements 5
Appendix A: Glossary 5
Appendix B: Analysis Models 5
Appendix C: To Be Determined List 6

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This SRS describes the functional and non-functional software requirements for CrossFit version 1.0 and provides the basis for the project. The aim of this product is to make our users independent and aware of their fitness choices. Everyone hustling out there has a different goal towards fitness like gaining muscle weight or losing fat. Our product helps the user get the right meal preps according to their required calorie goal and even the exercises would be recommended if needed so.

1.2 Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents. The document was prepared using Google Docs and uses the "Arial" font type.

1.3 Intended Audience and Reading Suggestions

This SRS document is intended for programming developers, product testers, project managers, and marketing staff; and our stakeholders may review this as a guideline for understanding the requirements and maintaining the main goal of the application. By reading this document, the reader will get a general understanding of the product including the application's functional and non-functional requirements.

1.4 Product Scope

The major scope of this project is to spread awareness and correct information about fitness, in a time where people tend to tilt towards the generic internet to look upon their fitness goals, which often tends to be a big ocean of myths and lies which seem to be true but are really deceiving. In today's busy world, people barely have time for themselves hence they don't really care to verify the information they read on the internet and blindly assume it to be true.
Our project aims to help such people in the most important fitness fields like

- Meal planning
- Exercise recommendations
- Fitness goals of the user(gaining/ losing weight).

1.5 References

User's Perspective about Mobile Fitness Applications

Status of the research in fitness apps: A bibliometric analysis - PMC

2. Overall Description

2.1 Product Perspective

Crossfit is a workout, health, and fitness-related mobile application. It aims to offer proper fitness-related regimens, daily meal planners, and workout sessions. Users will be able to choose from a variety of workouts and can also make a daily workout session routine. The app will generate user-desired custom meals every day according to the calorie limit. Third-party app integrations like Starva, and Fitbit will also enable users to connect with their accounts.

2.2 Product Functions

- Workout Session
  Users can choose between different workout sessions. The app will record all the daily calorie goals and remind the user if he/she has skipped the workout. If the user wishes to make a custom workout plan, a full detailed graph of the workout will be shown.
- Meal Planner
  CrossFit will display a variety of meal plans that the user wishes to choose according to this target calorie per day. This plan may be in the daily category or a continuous week plan.
- Third Party Integrations
  If the user has a Starva account or a Fitbit watch, he/she can connect to the respective account in order to see fitness-related information on CrossFit.
- Share Progress
  CrossFit users can share their in-app progress and can copy custom workout sessions
- Barcode Scanner
  Users can any food product which contains a barcode and will get a detailed view regarding the food product.

2.3 User Classes and Characteristics

Users:

- Registered Users: Authenticated Crossfit users have full access to all features present in the app. They can add custom workout sessions, meal plans, and

daily goals. All the details of the registered users are stored partially in local storage and majorly in the database
- Non-Registered Users: If a user is new to the app, he/she has to register using email in order to enter as the app relies on storing the user's data in the cloud.

2.4 Operating Environment

CrossFit will support Andriod(primary)/IOS phones. This app is based on the Flutter, and Dart SDK and uses Firebase as the backend to store and retrieve the data of the users. Also, local data storage will be used to store the temporary data related to the workout sessions of the users.

2.5 Design and Implementation Constraints

- In order for the app to work correctly, a working Andriod/IOS phone should be a necessity.
- The app must have access to an active network connection to the Internet and allow all the permissions that the app demands.
- In order to access their account, users should enter the correct username and password in order to gain access or sign-up if new.

2.6 User Documentation

NA

2.7 Assumptions and Dependencies

- User must have a working phone with an Andriod API level 20+, IOS 10 version, and internet connection
- Users should accept all the requirements and permissions of the app.
- User must have a basic idea of navigating an app and should understand the English language.
- Registered User should remember his/her password if auto-login isn't enabled
- Firebase is used as the backend for storing and retrieving data of the users.(https://console.firebase.google.com/)
- Firebase database should be online, accepting and sending requests as intended, and mustn’t time out the connection.
- Packages from https://pub.dev/ will be used as the development of the project progresses.

3. External Interface Requirements

3.1 User Interfaces

i) Login/ Sign up page:
This would be the first page that the user would see. The user has to log into his account or make one if he has not made it prior.

ii) Home Screen:
There will be multiple small sections that would be leading to further screens. There is a calorie tracker which shows updated calorie intake of the day along with the remaining calorie intake of the day. The next section is the recommended exercises which on a smaller scale shows how many of the user of the prescribed exercise has done it according to the target set. The next section is a meal planner which is a customized meal plan for the user in accordance with the fitness/calorie goal of the user.

iii) Calorie management screen:
In this screen, the user can select his calorie intake for a day. The user's data will be fetched from his sign-in data and using an API, the application will recommend the daily calorie intake the user should have for each scenario i.e., cutting, maintenance, and bulking. The user can also manually select his daily calorie goal according to his interests.

iv) Exercise recommendation:
This screen helps the user find the right exercise and the right amount of the same to reach their fitness goals.

v) Meal Planner:
This screen helps the user to decide what he should be eating throughout the day to reach his calorie goal. The user will be asked about the number of meals he would like to consume and according to the fitness goal, customized meal plans will be shown to the user.

3.2 Hardware Interfaces

The user must have either Android API level 20+ or IOS 10 version. the user must also have a stable internet connection, preferably with a download speed of over 512 kbps.

3.3 Software Interfaces

Dart and flutter would be used to construct our application which majorly includes the front end and the functioning of the application like touch responses etc. The app

backend will be able to connect to a database to store user information. Data will be stored locally as well as in firebase.

3.4 Communications Interfaces

For communication, the application would require the user's email id for creating an account in order to proceed.

4. System Features

Our app majorly focuses on the user's daily meal plan and workout routine. Registered users can prepare a special meal and workout plans based on their weight class. If the user has access to a smartwatch/other third-party fitness apps, it can be integrated with the app using authentication. It also contains a calorie tracker which reminds the user's daily calorie intake.

4.1 Workout Session

4.1.1 Description and Priority

Users can see different workout sessions and choose from the various workouts to see their details. If the app doesn't have a specific workout, the user has the option to add a new custom workout to exercise. This custom workout is can be seen by the other users if the creator of the workout shares it.

Priority: High

4.1.2 Stimulus/Response Sequences

Viewing different workout session
Stimulus: The user scrolls from the list of a workout session.
Response: New data from the API is loaded as the scroll continues

Stimulus: The user logins into the homepage and clicks on the particular workout session
Response: The user is redirected to a new page that shows the workout/ workout session in detail.

Creating a custom workout routine:
Stimulus: The user clicks on the "Add Workout" button
Response: The user is then redirected to a page that shows a form-type layout and has to fill in the details in order to save the workout.

Stimulus: The user clicks on the Save button in order to save the workout session.

Response: After all the details is been verified, user data is saved to the database.

Editing a custom workout routine:
Stimulus: After the user navigates to the Custom Workout Page and clicks the required workout routine to edit.
Response: After clicking the routine, the user is redirected to an edit page that contains prefilled previous information which the user can edit

Stimulus: The user changes all the data from the workout session and clicks on the save button
Response: After a successful edit, the user is taken back to the home screen.

Deleting a custom workout routine:
Stimulus: The user navigates to the “Custom Workout Page” and clicks on it
Response: The custom workout page is opened and the user can see different options to select from.

Stimulus: The user clicks the delete button to remove the workout session.
Response: The respective workout session has been removed from the app after confirmation from the user.

Start a workout session
Stimulus: The user clicks on a specific/custom workout to start working on the exercise
Response: The app will notify the user that the selected workout is starting and record a timer until the exercise finishes.

Exit a workout session
Stimulus: The user wants to exit the workout and clicks on the exit workout button to stop training.
Response: The workout is stopped and shows the details of the time user was exercising.

Show Recorded Workouts
Stimulus: The user clicks on the ‘Workout history tabs.
Response: The user is navigated to the workout history page and is shown a list of workouts that he/she has done in the past.

4.1.3 Functional Requirements

Case 1: User doesn’t fill in required details on the page
Exception Response: The user is alerted regarding which of the details are necessary to proceed

Case 2: User enters the same custom workout name as that of the already existing workout name
Exception Response: The user is alerted that workout already exists and should choose a different name

4.2 Customized Mean Planner

4.2.1 Description and Priority

The user can create meal plans according to their daily calorie, the app will auto-generate the meal plan using the Spoonacular API. This meal planner is stored and used for further repeating use. Users can add allergies to the plan and the system will avoid the listed allergies.

Priority: Medium

4.2.2 Stimulus/Response Sequences

Viewing different meal plans
Stimulus: The user chooses the target calorie count and clicks on Search.
Response: User-submitted details are packaged into the API's body and sent. After the API is fetched, different meal plans template is presented before the user according to their needs.

Stimulus: The user chooses a daily/weekly meal plan and along with target calories clicks Search.
Response: User-submitted details are packaged into the API's body and sent. After the API response is returned, the user has a list of the different meal plans according to the requirement

Adding a custom meal plan:
Stimulus: After the user has a list of meal templates, clicks on any meal plans and proceeds to save.
Response: The respective meal plan is saved as default and stored in the database.

Deleting a custom meal plan:
Stimulus: The user navigates to the My Meals and clicks on it
Response: The My Meal page loads and the user can see his previous as well as current meal plans.

Stimulus: The user clicks the delete button to remove the specific meal plan.
Response: The user is again prompted with the option that if he/she is sure that the meal plan is sure to be deleted.

Stimulus: The user wants to delete the meal plan after seeing the prompt confirmation
Response: The meal plan that the user selected has been removed from the My Meals page.

4.2.3 Functional Requirements

Case 1: User gives incorrect/invalid input in target calories
Exception Response: The user is alerted if the target calories are below a certain value, too high, or invalid.

4.3 Third Party Integrations

4.3.1 Description and Priority

If the user has third-party app accounts like Strava or Fitbit, he/she can connect the account with the app and can view all the activity. The user has to have an account in order to complete the whole process. Information from the account is fetched using the API

Priority: Mediocre

4.3.2 Stimulus/Response Sequences

Adding Starva Account to the Crossfit App
Stimulus: The user clicks on the “Connect Starva”
Response: The user is navigated to the Auth Page where the required credentials of a Starva account is required.

Stimulus: The user fills in all the details and clicks on the Login button
Response: The user is navigated to the permissions page where a list of permission regarding Crossfit taking control of the account is displayed.

Stimulus: The user chooses all permission to give access and clicks submit.
Response: The Starva Auth Page authenticates the information and then redirects to the main home page.

Adding Fitbit Account to the Crossfit App
Stimulus: The user clicks on the “Connect Fitbit”
Response: The user is navigated to the Auth Page where required credentials of a Fitbit account is required.

Stimulus: The user fills in all the details and clicks on the Login button
Response: The user is navigated to the permissions page where a list of permission regarding Crossfit taking control of the account is displayed.

Stimulus: The user chooses all permission to give access and clicks submit.
Response: The Fitbit Auth Page authenticates the information and then redirects to the main home page.

Starva Activities View
Stimulus: After the user has authenticated with the Starva Account and clicks on the Starva page
Response: The user is redirected to a page where all activities of the user, and account details are shown.

Fitbit View
Stimulus: After the user has connected his/her Fitbit account with CrossFit and clicks on the Fitbit page.
Response: The user can see all details related to the account and its user activity.

4.3.3 Functional Requirements

Case 1: User gives doesn't give permission to access its account details

Exception Response: The app won't have details of the respective account and will prompt the user to again sign in to the account and give permissions

4.4 Calorie Management System

4.4.1 Description and Priority

After the user registers through the app and collects meal plan according to the user's need, the app calculates the calories consumed after all the meal times has been set,

Priority: Low

4.4..2 Stimulus/Response Sequences

Calorie Management Setup
Stimulus: The user clicks on the tab "My Calorie Intake"
Response: If the user has never entered this page, it will ask all the details regarding the meal timing of the users.

Stimulus: The user completes filling in all the details and clicks the Save button
Response: The user is shown the current calories along with other details like BMI, height, weight, etc.

Stimulus: The user clicks on the tab “My Calorie Intake” after filling in the details.
Response: The user is redirected to the My calorie page and is shown today’s calorie intake and can see last week's calorie intake.

4.4.3 Functional Requirements

Case 1: The user doesn’t give incomplete information regarding the meal time cycle of the plan.
Exception Response: The user will be alerted according to the context of which of the form fields are incomplete

4.5 Barcode Scanner

4.3.1 Description and Priority

If the user wishes to find the calorie content, protein, carbohydrates, and other nutrients, he/she can open the bar code scanner will find the details regarding the product.

Priority: Lowest

4.3.2 Stimulus/Response Sequences

Barcode Scanning

Stimulus: The user clicks on the “Scan any item” tab

Response: The user is navigated to the Scanner page which waits for a barcode to be scanned.

Stimulus: The user points to a product with a barcode scanner open.

Response: After the barcode is in the visible area, it scans the barcode and shows the information on the screen.

4.3.3 Functional Requirements

Case 1: User doesn't give permission to access its camera

Exception Response: The app won't have any activity as the camera isn’t accessible and will prompt the user to give permission again.

Case 2: The user waits for the barcode output, but no output is generated due to some issue.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

i)The data in the database should be updated in less than 1 second.
ii) The load time of UI should be within 30 milliseconds.
iii) The API should respond in less than 2.5 seconds.

5.2 Safety Requirements

In the meal planner section of the application, the meals shown are after being taken care of by the user's allergies and food choices.
In the exercise recommendations, the exercises are recommended after taking care of the user's movement conditions like joint pain, fractures, muscle misalignment etc.

5.3 Security Requirements

All data of the users will be stored safely on the database using their privacy features. Thus, there will be no threat of data loss.

5.4 Software Quality Attributes

The app has a simple UI with easily distinguishable windows which lead to specific screens. It also has easy navigation between the various pages on the website. All the features provided by the app are easy to understand and use.

5.5 Business Rules

The application is basically non for profit and free to use for the public. Only the current devs are allowed to upgrade the application features and there are no other special admin-like roles whatsoever. The application is solely meant for the users out there to benefit from it. Hence there is no business alignment on the application like payment gateways, specialized plans, etc.

6. Other Requirements

Once the system is completely developed, a few sessions will be required in order to find and eliminate bugs and glitches in the application. After those sessions, it's required that a member of the development team should spend some time upgrading the app if needed.

Appendix A: Glossary

i) Flutter: It is an open-source framework to create high-quality, high-performance mobile applications across mobile operating systems - Android and iOS. It provides a simple, powerful, efficient, and easy-to-understand SDK to write mobile applications

ii) Dart: It is a client-optimized language for developing fast apps on any platform. Its goal is to offer the most productive programming language for multi-platform development, paired with a flexible execution runtime platform for app frameworks. its google's language and supports flutter quite well

iii) Google Firebase: It is a Google-backed application development software that enables developers to develop iOS, Android, and Web apps. Firebase provides tools for tracking analytics, reporting and fixing app crashes, and creating marketing and product experiment.

Appendix B: Analysis Models

Flowchart for firebase authentication: