IEEE 830-1998 Software Requirements Specification

# TITLE: "Fit-Connect: Always a click away from your next training session!"

# Table of Contents



## 1. Introduction

### 1.1 Purpose

The purpose of this document is to provide a Software Requirements Specification (SRS) for the development of “Fit-Connect”, a web application that enables users to hire trainers for training sesions or consultations quickly and conveniently. This document describes the requirements of the web application from the perspectives of the user, system, and trainer.

### 1.2 Document Conventions

This document is written using the IEEE 830-1998 standard for software requirements specifications. It will include user, system, and trainer requirements for the completion of the project.

### 1.3 Intended Audience and Reading Suggestions

This document is intended for the developers and designers working on the "Fit-Connect" project. It's recommended that readers start from the beginning and read the document from start to finish to get a complete picture of the project requirements.

### 1.4 Product Scope

The scope of "Fit-Connect" is to create a web application that allows users to schedule training sessions or consultations with ease, help users find the right trainer for their liking, and enables the trainer to view and enroll for available jobs. It will also provide an easy and secure payment experience and an orderly review system.

### 1.5 References

1. IEEE 830-1998: IEEE Standard for Software Requirements Specifications (IEEE, 1998).
2. Django 4.1. documentation

### 1.6 Overview

The "Fit-Connect" web application will be a platform for users to request training sessions or consultations. The application will also provide a secure payment system and a review system for users to rate their experience with trainers.

The sections of this document are as follows:

- Section 2: Overall Description

## 2. Overall Description

### 2.1 Product Perspective

Fit-Connect is a web application designed to help users find the perfect gym trainer for their project, quickly and easily.

The user interface is designed to be intuitive and easy to use, with an Admin dashboard, customer dashboard, trainer dashboard, and a public part with a list of submited jobs and trainers (if you are a customer), and a list of requested jobs (if you are a trainer) filterable by date. Users can click on a particular Trainer's profile and view their previous jobs and reviews. There is also an in-app messaging system for users to communicate with their trainers.

Users can post job requests

Users can post jobs and receive bids from trainers. Trainers are delivered through the app. Payment is handled using a cryptocurrency token. If there is a dispute or disagreement between users and trainers, an admin user will review the case and make a decision.

Fit-Connect also offers a rating system for trainers, as well as a searchable database of trainers so users can easily find the perfect trainer for their liking. Unfortunately, there is no automated quality assurance process.

Fit-Connect is a self-contained web application that will be hosted on a web server. It will be accessible through a web browser.

### 2.2 Product Functions

"Fit-Connect" will provide the following functions:

- User Interface: Users will be able to navigate between pages on the web application, communicate with trainers, view the names of the trainers assigned to the jobs, contact a trainer after assigning a job, and view the estimated cost of their trainers jobs.
- Job Posting and Assignment: Users will be able to post jobs and receive bids from trainers and request a specific trainer for their trainer job.
- Job Delivery and Payment: Web application will handle the delivery of disputes or disagreements between users and trainers, payments, and invoicing.
- Review System: The web application will include a rating system for trainers, as well as allow users to review the quality of their jobs.
- Job and Trainers Search: The web application will provide a searchable database of trainers, and will verify the credentials and qualifications of trainers.

## 2.3 User Classes and Characteristics

The product will have three logical user classes: Admin, User, and Trainer. Admin is a user class with a superuser status, and can access the Admin dashboard. Admin users can create, edit, and delete users and trainers.

Users can post jobs and trainers can submit bids for jobs.

### 2.3.1 Admin

Admin users will be able to manage the web application, including managing users, managing trainers, and managing jobs. Admin users will also be able to review disputes and disagreements between users and trainers. Since the app wil use a cryptocurrency token, admin users will be able to manage the token allocation to users and trainers.

### 2.3.2 User

Users will be able to post jobs, assign jobs to trainers, and view the status of their jobs. Users will also be able to review the quality of their trainers.

### 2.3.3 Trainer

Trainers will be able to view available jobs, submit bids, and view the status of their jobs.

## 2.4 Operating Environment

The web application will be hosted on a web server. It will be accessible through a web browser.

## 2.5 Design and Implementation Constraints

The web application will be written in Python using the Django framework. It will be hosted on a web server. The data will be stored in a SQLite database. No further design and implementation constraints are known at this time.

## 2.6 User Documentation

No documentation will be provided to users.

## 2.7 Assumptions and Dependencies

The application will use a cryptocurrency token to handle payments. For the first version of the application, the amounts of tokens each user have will be recorded as a field in the database and will be managed by admin users. The token deposit and withdrawal funtionality will not be implemented in this version of the application.

The application will not have an automated quality assurance process.

# 3. Specific Requirements

## 3.1 Functional Requirements

### 3.1.1 User Requirements

#### 3.1.1.1 User Registration

**Req U1** - The web application will allow users to register for an account.

The web application will allow users to register for an account. Users will provide their name, email address, and password. The web application will verify that the email address is unique and not already used by another user.

#### 3.1.1.2 User Login

**Req U2** - The web application will allow users to log in to their account.

The web application will allow users to log in to their account. Users will provide their email address and password. The web application will verify that the email address and password match the information in the database.

#### 3.1.1.3 User Logout

**Req U3** - The web application will allow users to log out of their account.

The web application will allow users to log out of their account. Users will click on a "Log Out" button in the Navigation. The web application will terminate the user's session.

#### 3.1.1.4 User Dashboard

**Req U4** - The web application will provide a dashboard for users to view their account information and manage their jobs.

The web application will provide a dashboard for users to view their account information and manage their jobs. The dashboard will include the following information:

- User's name
- User's email address

- User's token balance
- List of jobs posted by the user
- A list of bids for jobs posted by the user
- A link to the trainer's profile for each bid
- A link to the trainer's profile for each accepted job
- Messages from trainers ordered by date from newest to oldest

##### 3.1.1.5 User Profile

**Req U5** - The web application will provide a profile for users to view and edit their account information.

The web application will provide a profile for users to view and edit their account information. The profile will include the following information:

- User's name
- User's email address
- User's posted jobs
- User's completed jobs

The user will be able to change their name or their password on this page.

##### 3.1.1.6 Job Posting

**Req U6** - The web application will allow users to post jobs.

The web application will allow users to post jobs. Users will provide the following information:

- Job title
- Job description
- Job date
- Job time

Job title will be a droppdown menu giving you the option to choose from "Training session" or "Consultations". Job description would have pre-written examples.

##### 3.1.1.7 Trainer Listing

**Req U7** - The web application will allow users to view a list of trainers on the platform.

The web application will allow users to view a list of trainers. Users will click on a "Trainers" button in the Navigation. The web application will display a list of trainers. The list will include the following information:

- Trainer's name
- A link to the trainers's profile
- Trainer's rating
- Trainer's number of completed jobs
- A "Post a Job" button

There will be a button to post a job dirrectly to a trainer.

#### 3.1.1.8 Job Assignment

**Req U8** - The web application will allow users to accept a bid from a trainer and assign a job to the trainer.

The web application will allow users to accept a bid from a trainer and assign a job to the trainer. Users will click on a "Accept" button next to the bid. The web application will assign the job to the trainer and notify the trainer that the job has been assigned to them.

#### 3.1.1.9 Job Status

**Req U9** - The web application will allow users to view the status of their jobs.

The web application will allow users to view the status of their jobs. Users will click on a "View Status" button next to the job. The web application will display the status of the job.

#### 3.1.1.10 Job Review

**Req U10** - The web application will allow users to review the quality of done job.

The web application will allow users to review the quality of the job. Users will click on a "Review" button next to the job. The web application will display a form for the user to provide a review. The web application will verify that the review is between 1 and 5 stars.

After the user submits the review, the web application will update the trainer's rating. The web application will calculate the new rating by calculating the average of all the reviews for the trainer. The web application will update the trainer's rating in the database. The web application will transfer the tokens for the job from the users balance to the trainers balance.

#### 3.1.1.11 User Messaging

**Req U11** - The web application will allow users to send messages to trainers.

The web application will allow users to send messages to trainers. Users will click on a "Message" button next to the accepted job. The web application will display a form for the user to provide a message. The web application will verify that the message is not empty.

After the user submits the message, the web application will send the message to the trainer assigned to the job. The web application will display the message on the trainer's dashboard.

#### 3.1.1.12 Job listing

**Req U12** - The web application will allow users to view a list of jobs they posted.

The web application will allow users to view a list of jobs they posted. Users will click on a "My Jobs" button in the Navigation. The web application will display a list of jobs they posted. The list will include the following information:

- Job title
- Job description
- Job date

- Job time
- Job bids
- A link to the trainer's profile (if assigned)
- A "Message Trainer" button (if assigned)

##### 3.1.1.13 Job Dispute

**Req U13** - The web application will allow users to dispute a job.

The web application will allow users to dispute a job. Users will click on a "Dispute" button next to a completed job. The web application will display a form for the user to provide a dispute. The web application will verify that the dispute is not empty.

The dispute will be handled by the administrator manually. The dispute has to contain the following information:

- Details about the job
- The user that posted the job
- The trainer that completed the job
- The reason for the dispute (provided by the user that posted the job)

### 3.1.2 Trainer Requirements

Trainer requirements focus on the specific funtionalities that trainers will be able to use.

Trainer use the same registration and login process as users. The only difference is that during the registration process, users will be able to click a checkbox stating that they wish to be a trainer.

User interface options intented for the trainers will be visible only to users that selected the option to be a trainers during the registration process.

User interface options and functions for trainers and users will differ accordingly.

#### 3.1.2.1 Choose to be a Trainer

**Req T1** - The web application will allow trainer to register as a trainer.

The web application will allow trainers to register as a trainer. During the registration process, trainers will be able to click a checkbox stating that they wish to be a trainer.

#### 3.1.2.2 Trainer Dashboard

**Req T2** - The web application will provide a dashboard for trainers to view their account information.

The web application will provide a dashboard for trainers to view their account information. The dashboard for trainers will include the following information:

- Trainer's name
- Trainer's email address
- Trainer's token balance
- Messages from users ordered by date from newest to oldest

- A list of submitted bids
- A link to the users's profile for each submitted bid
- A list of assigned jobs
- A link to the users's profile for each assigned job
- A list of completed jobs
- A link to the users's profile for each completed job
- Trainer's rating

##### 3.1.2.3 Trainer Profile

**Req T3** - The web application will provide a profile for trainers to view their account information.

The web application will provide a profile for trainers to view their account information. The profile for trainers will include the following information:

- Trainer's name
- Trainer's email address
- A list of submitted bids
- A list of assigned jobs
- A list of completed jobs
- Trainer's rating

##### 3.1.2.4 Job Listing

**Req T4** - The web application will allow trainers to view a list of jobs posted by users.

The web application will allow trainers to view a list of jobs posted by users. Trainers will click on a "Jobs" button in the Navigation. The web application will display a list of jobs posted by users. The list will only show unassigned jobs and/or jobs assigned specifically for that trainer. The list will include the following information:

- Job title
- Job description
- Job date
- Job time
- A link to the user's profile
- A "Bid on Job" button
- A "Message User" button

##### 3.1.2.5 Job Bidding

**Req T5** - The web application will allow trainers to bid on jobs posted by users.

The web application will allow trainers to bid on jobs posted by users. The jobs will be listed in the Job Listing page as discribed in **Req T4**. Trainers will click on a "Bid on Job" button next to the job. The web application will display a form for the user to provide a bid. The web application will verify that the bid is a positive number.

After the trainer submits the bid, the web application will send the bid to the user who posted the job.

#### 3.1.2.6 Job Status

**Req T6** - The web application will allow trainers to view the status of their jobs.

The web application will allow trainers to view the status of their jobs. The possible statuses are: "Assigned", "Bid In Progress", "Completed". The web application will display the status of the job.

### 3.1.3 Admin Requirements

Admin requirements focus on the specific funtionalities that Admins will be able to use. Admins are created by the system administrator in the Admin Dashboard. The Admins use the Django Admin interface to manage the system.

#### 3.1.3.1 Admin Dashboard

**Req A1** - The web application will provide a dashboard for admins to manage the system.

The web application will provide a dashboard for admins to manage the system. The dashboard will use the Django Admin interface. The dashboard will allow admins to manage the following:

- Users (create, edit, delete)
- Jobs (create, edit, delete)
- Apply a token balance to a user

#### 3.1.3.2. Apply a Token Balance to a User

**Req A2** - The web application will allow admins to apply a token balance to a user.

The web application will allow admins to apply a token balance to a user. The web application will allow the admins to find a user in the Admin dashboard and to change the user's token balance. The web application will update the user's token balance in the database. The web application will display the new token balance on the user's dashboard.

**Constrainst and considerations:** The token balance is used for payments in the system. The token balance is applied to the user's account when the user completes a job. The token balance is deducted from the user's account when the user assigns a job. The deposit and withdrawal of the tokens is outside the scope of this application.

The admins will verify the user's deposit outside of the application and will apply the token balance to the user's account in the application. Likewise, the admins will process the user's withdrawal requests outside of the application and will deduct the token balance from the user's account in the application.

In this way, the application will not be responsible for the communication between the users and the admins. All the communication between the users and the admins regarding the token balances will be done outside of the application. The admins will then apply the token balance to the user's account through the Admin dashboard.

**Note:** The token balance is not a currency (yet). The token balance is a number that represents the number of tokens that the user has in their account. The token balance is used for payments in the system.

#### 3.1.3.3 Dispute Resolution

**Req A3** - The web application will allow admins to resolve disputes.

The web application will allow admins to resolve disputes. The web application will allow the admins to find a dispute in the Admin dashboard and to resolve the dispute. The dispute will be resolved by the admin manually. The admin will contact the job poster outside the application and will resolve the dispute manually. All the changes to the users' ratings or token balances will be done manually by the admin from the Admin dashboard.

Once the dispute is resolved, the admin will close the dispute. The admin will close the dispute by changing the status of the dispute to "Closed" in the Admin dashboard. The web application will display the status of the dispute as "Closed" on the user's dashboard.

### 3.2 Usability Requirements

The web application will be designed to be easy to use. The web application will be designed to be intuitive and easy to learn.

### 3.3 Reliability Requirements

None a this time

### 3.4 Performance Requirements

None a this time

### 3.5 Supportability Requirements

None a this time

### 3.6 Design Constraints

None a this time

## 4. System Interfaces

### 4.1 User Interfaces

The application will have a user interface that is accessible via a web browser. The interface will be designed to be intuitive and user-friendly, optimized for mobile and desktop devices.

The web application will use a standard web interface with a top navigation bar and a main content area. The web application will be primarily used on a desktop computer and a responsive design is not required.

The top navigation bar will have the following links:

- On the left side:
    - Logo (link to the home page)
    - Jobs (a link to the Job listing page) if the trainer is logged in
    - Post a Job (a link to the Post a Job page) if the user is logged in

    - Trainers (a link to the Trainers page) if the user is logged in

- On the right side:

    - Login (a link to the Login page) if the user/trainer is not logged in
    - Profile (a link to the Profile page) if the user/trainer is logged in
    - Logout (a link to the Logout page) if the user/trainer is logged in
    - My Jobs (a link to the My Jobs page) if the user/trainer is logged in

### 4.2 Hardware Interfaces

The application will be hosted on a secure web server running on Linux or Windows.

### 4.3 Software Interfaces

The application will use Django 4.0, the database will be SQLite, frontend framework will be Bootstrap.

### 4.4 Communications Interfaces

The application will have a secure communication interface for secure data transfer.

## 5. Other Nonfunctional Requirements

### 5.1 Security Requirements

The application will use the Django's default User management system. The application will use Django's default authentication system. The application will use Django's default authorization system. The application will use Django's default password reset system.

### 5.2 Safety Requirements

The application must be designed to prevent the user from performing any unsafe actions or operations.

### 5.3 Business Rules

The application will have business rules in place to ensure the safety and security of user data.

### 5.4 Quality Attributes

The application must have a high quality user interface with good usability and responsiveness. The application must also have an uptime of at least 99.

## 6. Other Requirements

### 6.1 Business Requirements

None a this time

### 6.2 Regulatory Requirements

None a this time