**[**GYMIFY**] Requirements Specification**

**Version** 0**.**3

April 7, 2023

Use this Requirements Specification template to document the requirements for your product or service, including priority and approval. Tailor the specification to suit your project, organizing the applicable sections in a way that works best, and use the checklist to record the decisions about what is applicable and what isn't.

The format of the requirements depends on what works best for your project.

This document contains instructions and examples which are for the benefit of the person writing the document and should be removed before the document is finalized.

To regenerate the TOC, select all (CTL-A) and press F9.

**Table of Contents**

[**[GYMIFY] REQUIREMENTS SPECIFICATION 1**](#_1mrcu09)

[**VERSION 0.2 1**](#_46r0co2)

[**APRIL 7, 2023 1**](#_2lwamvv)

[**1.**](#_111kx3o) **EXECUTIVE SUMMARY 3**

[1.1](#_3l18frh) Project Overview 3

[1.2](#_206ipza) Purpose and Scope of this Specification 3

[**2.**](#_4k668n3) **PRODUCT/SERVICE DESCRIPTION 3**

[2.1](#_2zbgiuw) Product Context 3

[2.2](#_1egqt2p) User Characteristics 3

[2.3](#_3ygebqi) Assumptions 3

[2.4](#_2dlolyb) Constraints 3

[2.5](#_sqyw64) Dependencies 4

[**3.**](#_3cqmetx) **REQUIREMENTS 4**

[3.1](#_1rvwp1q) Functional Requirements 5

[3.2](#_4bvk7pj) Non-Functional Requirements 5

[*3.2.1*](#_2r0uhxc) *Product Requirements 5*

[**3.2.1.1**](#_4i7ojhp) **User Interface Requirements** 6

[**3.2.1.2**](#_2xcytpi) **Usability** 6

[**3.2.1.3**](#_1ci93xb) **Efficiency** 6

[3.2.1.3.1](#_3whwml4) Performance Requirements 6

[3.2.1.3.2](#_2bn6wsx) Space Requirements 6

[**3.2.1.4**](#_qsh70q) **Dependability** 6

[**3.2.1.5**](#_3as4poj) **Security** 7

[*3.2.2*](#_1664s55) *Organizational Requirements 7*

[**3.2.2.1**](#_49x2ik5) **Environmental Requirements** 7

[**3.2.2.2**](#_2p2csry) **Operational Requirements** 7

[**3.2.2.3**](#_147n2zr) **Development Requirements** 7

[*3.2.3*](#_3q5sasy) *External Requirements 7*

[**3.2.3.1**](#_23ckvvd) **Regulatory Requirements** 7

[**3.2.3.2**](#_ihv636) **Ethical Requirements** 7

[**3.2.3.3**](#_32hioqz) **Legislative Requirements** 7

[3.2.3.3.1](#_1hmsyys) Accounting Requirements 7

[3.2.3.3.2](#_41mghml) Security Requirements 7

[3.3](#_25b2l0r) Domain Requirements 7

[**4.**](#_kgcv8k) **USER SCENARIOS/USE CASES 7**

# Executive Summary

## ***Project Overview***

This project is based on a Web-Based application and acts as a convenient way for an entire gym chain to maintain and manage the gym and it provides a platform for coaches as well as managers to stay organized and work more efficiently. This project is designed for people who want to go to the gym, and have previous experience with gyms. For most people going to the gym happens on a whim and usually those same people lack the motivation to go back, the design and purpose of this application is to make it so these people can keep track of their progress whilst going on the gym and they can stay motivated.

The app will help individuals better communicate with coaches and help them choose their desired courses, the plan they want most and the time and location they want to go to. After the users have selected a course, the coach can choose to accept or decline said course, and they can choose to change the appointed time and location based on their needs and the users needs.

This app was made to make people rely more on courses and coaches rather than just working alone but it also allows them other features to choose from such as an online shop where people can buy gym products and have them delivered at their houses, chatting with their instructors, watching videos from a video catalog, see their schedules and other features. The app provides an effective way to conveniently manage gyms online and on a real time basis, using a system that will include all gyms in a certain city, district or region. It gives all user levels a self-explanatory and easy to use tool which they can fulfill their jobs with.

In order for our product to work in the desired way, the users of the system must be able to correctly and effectively use each feature and by coherently using the in app systems such as reviews and reports we will be able to create a community which will continue improving with the app, that is our goal as every feature of the app is designed with improvement in mind and it will be implemented in such a way that change will be welcomed. The end goal is to have a system that functions without a lot of maintenance on an international level.

## ***Purpose and Scope of this Specification***

The purpose of this document is to create a way to guide us through creating an application without missing any of the requirements while also staying within the scope, and not breaking any guidelines.

The specification addresses the following aspects of the design process:

* Documentation of all product features and functionality
* Technical overview of the application
* Requirements Explanation
* Product Interfaces
* Thorough explanation of all ways the application will be used
* Dependencies and Constraints

Out of the scope of the project are:

* Legislative Requirements
* Financial Aspects of the project
* Unnecessary Requirements with QoL features (unneeded)

# Product/Service Description

Our application will allow members of the gym to flexibly access any particular gym. The user can buy membership that will be accessible at any gym that is part of the same chain. The user can choose a course, which is split into 2 categories, group courses and individual courses. For the individual ones we have provided each member with the ability to flexibly arrange the desired time and location by having the option of communicating directly with their coach and creating the plan that will suit them the most. Also they will be able to watch videos, and purchase a variety of products from our online shop.

The coach will be able to have its own courses and will benefit from the larger user base of this system by having more people reach them. They can submit an application form to the managers with the courses they want to teach and after acceptance, they will be able to accept and decline members, report them and chat with them and manage their schedules according to their desire.

## ***Product Context***

This software is a tool which serves the current owners of the gyms and the people working in them. It establishes an easy way to interact with customers and coaches, by providing them with a number of different ways to achieve their goals, such as personal courses and group sessions with the idea of training in mind. It also allows for a better management of user data and overall gym functionality by keeping track of every process performed within the environment. Since to this day, this type of system is yet to introduce itself within our country we choose the initiative to design such a software.

Its main focus is the user, and it is designed with accessibility and pragmaticality in mind. It is there to provide a simple solution to current problems and an effective way of organizing and managing information. It is a real time tracking system that helps all people going in and out of gyms.

## ***User Characteristics***

* Member
* choose membership plan
* choose course (and gym coach for specific type of course)
* buy from an online shop
* set-up appointment with their coach
* review a course after finishing it
* chat with coach/customer service within the app
* watch videos from a video catalog
* Coach/Trainer/Gym Expert
* Create and publish their own courses
* decline/accept customers course
* set-up appointment with customer
* chat with their customers
* create custom plan based on customer availability
* review gym equipment / recommend new equipment
* report customer / cancel ongoing course
* Manager
* approve reviews / reports ( from both users and coaches)
* analyze equipment complaints (whether the complaint was needed or just skip it)
* order new equipment
* manage users (includes banning/unbanning)
* manage coaches and their courses
* manage gym schedule (will include the ability to close the gym for certain reasons)
* manage income/expenses
* see statistics about customer flow

## ***Assumptions***

* Browser
* Storage

We will assume that the user will have storage and a browser installed on their computer.

## ***Constraints***

* usability between old and new versions of software
* Server Resources ( depending on how much processing power we have at hand)
* User Base ( slow connection due to too many users)
* Hardware restrictions based on old computers
* Storage ( very little but it may be a constraint)
* Php ( will prove to be difficult to implement certain features)

## ***Dependencies***

List dependencies that affect the requirements. Examples:

* Our software will depend on current web programming languages, and its features
* It will depend on the browser and device / os that will be using it

# Requirements

**Priority Definitions**

The following definitions are intended as a guideline to prioritize requirements.

* Priority 1 – The requirement is a “must have” as outlined by policy/law
* Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
* Priority 3 – The requirement is a “nice to have” which may include new functionality

## ***Functional Requirements***

| **Req#** | **Requirement** | **Comments** | **Priority** | **Date Rvwd** | **SME Reviewed / Approved** |
| --- | --- | --- | --- | --- | --- |
| BR\_01 | The app will open with different views for different user levels (Members, Coaches, Managers) | The app will be able to differentiate users based on their credentials when they first registered | 1 | 4/14/2023 | Kevin Cela |
| BR\_02 | The app must be able to correctly handle in-app transactions. | The user will be able to purchase products just by entering their credit card information. | 1 | 4/14/2023 | Kevin Cela |
| BR\_03 | The user will log in using email and password or phone number and password | The app will have 2 ways of logging in. | 2 | 4/14/2023 | Kevin Cela |
| BR\_04 | The user can choose from different membership plans | The app will have a couple of different options to choose a plan from. | 3 | 4/14/2023 | Kevin Cela |
| BR\_05 | The user will receive immediate feedback on purchases and when choosing an option. | The server will need to update on real time | 1 | 4/14/2023 | Kevin Cela |
| BR\_06 | The user can contact customer service within the app | The user will be able to click on an option to contact or inform the customer service. | 2 | 4/14/2023 | Kevin Cela |
| BR\_07 | The user can choose from different available courses. | The courses will all be listed in a separate page. | 1 | 4/14/2023 | Kevin Cela |
| BR\_08 | The coach can create or change existing courses on their account. | So that they don't have to fill a form or contact the manager. | 1 | 4/15/2023 | Kevin Cela |
| BR\_09 | The manager will be able to generate records of all individuals currently assigned to that gym. | It will be needed by every manager. | 1 | 4/15/2023 | Kevin Cela |
| BR\_010 | The user can enroll and quit a course whenever they want. | It is important to allow user freedom. | 2 | 4/15/2023 | Kevin Cela |
| BR\_11 | The manager will be provided with all product information. | They will see which needs restocking and how many were bought within a certain time period. | 2 | 4/15/2023 | Kevin Cela |
| BR\_12 | The coach will be able to change and set-up appointments based on their schedule. | To make an effective scheduling method for coaches. | 1 | 4/15/2023 | Kevin Cela |
| BR\_13 | The coach can decline/accept members. | They will also provide a reason as to why a user was declined. | 1 | 4/15/2023 | Kevin Cela |
| BR\_14 | The manager can generate records for all income and expenses.. | Such as employer salaries and membership incomes.. | 1 | 4/15/2023 | Kevin Cela |
| BR\_15 | The user can select a video from the catalog to watch from. | A separate section within the app will have a number of different videos to select from. | 3 | 4/15/2023 | Kevin Cela |

## ***Non-Functional Requirements***

### **Product Requirements**

#### **User Interface Requirements**

The user interface will be split into 4 different interfaces based on the user level at the time of using the app.

* Login Interface

It will be the basic interface where each separate user can login to. It will have a button for logging in and another to redirect to the sign up page which will be the page where each user will register. Also the user will be able to view some general information about gym chain.

* Member Interface

The member interface will include a home page where they can view their profile, their membership type and expiration date, their schedule for that day, and their current course progress.

They will have the option to travel to other pages using the menu at the top of the page.

* The video catalog

In the video catalog users will be able to watch videos on how to perform different exercises and using the menu they can also go back to other pages.

* The online shop

Users here will be able to choose from different products to purchase from and they will have the option to select the payment method, the option to choose where to deliver the product and an option to print the receipt after the payment has been made.

* The courses interface

It will display all currently available courses with their price, the coach that is providing them and their rating, the members will have the option to check the courses they are enrolled in, and will have the option to choose to chat with their coach. Also they will be able to check their schedule, review the course and report the coach assigned to them by icons at the top of the screen.

* The membership interface

Here members will have the option to select from many different membership plans which will be displayed as cards with their features below their icons. Each option will provide a pop up with more information and users can select to buy the membership or not.

#### **Usability**

Include any specific usability requirements, for example,

Learnability

* The user documentation and help should be complete
* The help should be context sensitive and explain how to achieve common tasks
* The system should be easy to learn

(See <http://www.usabilitynet.org/>)

#### **Efficiency**

##### ***Performance Requirements***

Specify static and dynamic numerical requirements placed on the system or on human interaction with the system:

* Static numerical requirements may include the number of terminals to be supported, the number of simultaneous users to be supported, and the amount and type of information to be handled.
* Dynamic numerical requirements may include the number of transactions and tasks and the amount of data to be processed within certain time period for both normal and peak workload conditions.

All of these requirements should be stated in measurable form. For example, "95% of the transactions shall be processed in less than 1 second" rather than “an operator shall not have to wait for the transaction to complete”.

##### ***Space Requirements***

#### **Dependability**

**Availability**

Include specific and measurable requirements for:

* Hours of operation
* Level of availability required
* Coverage for geographic areas
* Impact of downtime on users and business operations
* Impact of scheduled and unscheduled maintenance on uptime and maintenance communications procedures
* reliability (e.g., acceptable mean time between failures (MTBF), or the maximum permitted number of failures per hour).

**Reliability**

**Monitoring**

Include any requirements for product or service health monitoring, failure conditions, error detection, logging, and correction.

**Maintenance**

Specify attributes of the system that relate to ease of maintenance. These requirements may relate to modularity, complexity, or interface design. Requirements should not be placed here simply because they are thought to be good design practices.

**Integrity**

#### **Security**

Specify the factors that will protect the system from malicious or accidental access, modification, disclosure, destruction, or misuse. For example:

* encryption
* activity logging, historical data sets
* restrictions on intermodule communications
* data integrity checks

Specify the Authorization and Authentication factors. Consider using standard tools such as PubCookie.

### **Organizational Requirements**

Requirements which are a consequence of organizational policies and procedures e.g. process standards used, implementation requirements, etc

#### **Environmental Requirements**

#### **Operational Requirements**

#### **Development Requirements**

### **External Requirements**

* + Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

#### **Regulatory Requirements**

#### **Ethical Requirements**

#### **Legislative Requirements**

Specify the requirements derived from existing standards, policies, regulations, or laws (e.g., report format, data naming, accounting procedures, audit tracing). For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values

##### **Accounting Requirements**

##### **Security Requirements**

## ***Domain Requirements***

Everything related to the domain that might be needed in the project shall be mentioned here. Sometimes the domain Requirements might be thought of as part of either functional or non-functional requirements.

Please provide all necessary non-functional requirements, similar to the requirements explained in the lesson slides or in the textbook.

# User Scenarios/Use Cases

Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

* Describes a significant business need
* Identifies, documents, and ranks the problem that is driving the scenario
* Describes the business and technical environment that will resolve the problem
* States the desired objectives
* Shows the “Actors” and where they fit in the business model
* Is specific, and measurable, and uses clear metrics for success

APPENDIX

The appendixes are not always considered part of the actual Requirements Specification and are not always necessary. They may include

* Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
* Supporting or background information that can help the readers of the Requirements Specification;
* A description of the problems to be solved by the system;
* Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the Requirements Specification should explicitly state whether or not the appendixes are to be considered part of the requirements.

1. **Definitions, Acronyms, and Abbreviations**

Define all terms, acronyms, and abbreviations used in this document.

1. **References**

List all the documents and other materials referenced in this document.

1. **Requirements Traceability Matrix**

The following trace matrix examples show one possible use of naming standards for deliverables (FunctionalArea-DocType-NN). The number has no other meaning than to keep the documents unique. For example, the Bargaining Unit Assignment Process Flow would be BUA-PF-01.

For example (1):

| **Business Requirement** | **Area** | **Deliverables** | **Status** |
| --- | --- | --- | --- |
| BR\_LR\_01  The system should validate the relationship between Bargaining Unit/Location and Job Class.---Comments: Business Process = "Assigning a Bargaining Unit to an Appointment" (Priority 1) | BUA | BUA-CD-01  Assign BU Conceptual Design | Accepted |
| BUA-PF-01  Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BUA-PF-01  Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BR\_LR\_09  The system should provide the capability for the Labor Relations Office to maintain the job class/union relationship.---Comments: Business Process = "Maintenance" (Priority 1) | BUA | BUA-CD-01  Assign BU Conceptual Design | Accepted |
| BUA-PF-02  BU Assignment Rules Maint Process Flow Diagram | ReadyForReview |

For example (2):

| **BizReqID** | **Pri** | **Major Area** | **DevTstItems DelivID** | **Deliv Name** | **Status** |
| --- | --- | --- | --- | --- | --- |
| BR\_LR\_01 | 1 | BUA | BUA-CD-01 | Assign BU Conceptual Design | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-DS-02 | Bargaining Unit Assignment DB Modification Description | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-PF-01 | Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-UCD-01 | BU Assign LR UseCase Diagram | ReadyForReview |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-001 | BU Assignment by PC UseCase - Add Appointment and Derive UBU | Reviewed |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-002 | BU Assignment by PC UseCase - Add Appointment (UBU Not Found) | Reviewed |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-006 | BU Assignment by PC UseCase - Modify Appointment (Removed UBU) | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-CD-01 | Assign BU Conceptual Design | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-DS-02 | Bargaining Unit Assignment DB Modification Description | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-PF-02 | BU Assignment Rules Maint Process Flow Diagram | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-UCD-03 | BU Assign Rules Maint UseCase Diagram | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-045 | BU Assignment Rules Maint: Successfully Add New Assignment Rule | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-051 | BU Assignment Rules MaintUseCase: Modify Rule | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-053 | BU Assignment Rules MaintUseCase - Review Assignment Rules | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-057 | BU Assignment Rules MaintUseCase: Inactivate Last Rule for a BU | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UI-02 | BU AssignRules Maint UI Mockups | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-021 | BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Success | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-027 | BU Assignment Rules Maint TestCase: Modify Rule - Success | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-035 | BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Error Condition | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-049 | BU Assignment Rules Maint TestCase: Modify Rule - Error Condition | ReadyForReview |

For example (3):

| **BizReqID** | **CD01** | **CD02** | **CD03** | **CD04** | **UI01** | **UI02** | **UCT01** | **UCT02** | **UCT03** | **TC01** | **TC02** | **TC03** | **TC04** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BR\_LR\_01 |  |  | X |  | X |  | X |  |  | X |  | X |  |
| BR\_LR\_09 | X |  |  | X |  | X |  |  | X |  | X |  | X |
| BR\_LR\_10 | X |  |  | X |  |  |  |  | X |  | X |  |  |
| BR\_LR\_11 |  | X |  |  |  |  |  |  |  |  |  |  |  |

1. **Organizing the Requirements**

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

**By System Mode**

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

**By User Class**

Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

**By Objects**

Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

**By Feature**

A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

**By Stimulus**

Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

**By Response**

Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

**By Functional Hierarchy**

When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

**Additional Comments**

Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.