Techengers Requirements Specification

Version 1.0

April 19, 2021

Rei Balla

Klejda Kumi

Nikaela Balla

Xhesika Bicaku

Xhois Shaholli

Lidio Hoxha

EPOKA
UNIVERSITY

Table of Contents

1. Executive Summary

1.1 Project Overview

Running a successful fitness business is very demanding and a difficult task given the competitive market. Having a great internal management system for keeping track of business related tasks or services and having attractive features that elevate customer satisfaction can help quite a bit inside this market. This application aims to solve these problems for a specific fitness business in Elbasan.

Almost no other business in this field has a real product that solves the problems mentioned above, that is why we think this can bring big potential value to the business and could help reduce business costs since it aims to save time for staff and customers. This business has an average of 320 memberships and is planning to expand in 2 more locations, which makes our solution a much more attractive option.

1.2 Purpose and Scope of this Specification

In Scope:

The purpose of this project is to allow the business to better organize and manage the staff and other important business logic as well as to allow the business to connect to their customers in a more meaningful and efficient way, enabling them to schedule their exercises and routines, buy diet plans and see the latest or upcoming events in the gym.

Currently, the business doesn’t have any type of database or inventory, only physical records. In this scope our project will provide inventory management, staff management, staff progress and performance, user satisfaction and feedback and other features yet to be added.

All in all, our intended audience is both the staff and customers.

Out of Scope:

In progress.

2. Product/Service Description

Our solution aims to help the internal business relations and management and at the same time enables users to schedule their exercise routines more efficiently and fun while providing them with additional features like progress tracking, calorie planning or getting updates on latest events. The product is meant to be completely managed by the administrator and other employers with frequent technical maintenance depending on future agreements.

2.1 Product Context

In relation to other products in other markets, our solution will provide a similar experience with extra features to a market that is not currently technologically up to date in Albania. Our intention is to make the product self contained and independent, without requiring much maintenance and technical oversight. In concept, other related systems share the same ideology but we believe we will provide a better and more efficient experience.

2.2 User Characteristics

All these users have different dashboards and have different roles.

Users:

1. Customer has :

- can log in to his/her account
- can access the exercise table to get or select them
- can get notifications on Events
- can access the Routine table in order to create, select or delete it
- can access the diet table to select one of them
- can edit their own schedule
- can log out of his/her account

Customers do not need to have any technical abilities, however they need to be familiar with the system and its functionality in order to have a smooth user experience. They can access almost all tables but never control them. They are not involved in the building or organization of the system but they will be the ones who use this product the most.

2. Staff has:

- can log in to his/her account

- can control the Users Table
- can control Exercise Table
- can get Mentoring Users
- can control Routines Table
- can control Events Table
- can control Diet Tables
- can edit his Working Days and edit his AvailableHours
- can get Notes from Admin
- can can log out of his/her account

The staff also has no need to have any technical abilities but will be able to properly use the application as they take on a lot of tasks similar to an administrator. They are the ones who will come up with new ideas for events and exercises, as well as monitor users and their progress. It is assumed that their opinion was taken into consideration when building the software functionalities.

3. Admin has:

- can log in to his/her account
- can control Staff Table
- can control User Table
- can control Salaries Table
- can control Events Table
- can control Diet Plans Table
- can add Notes for the staff
- can can log out of his/her account

The administrator has the greatest responsibility in comparison to the other user. They have no direct relation to the customer, however they communicate with the staff through the notes. They also act as an economist as they handle the salaries. Some technical as well as some financial abilities are needed in order for the application to run properly.

Eventually Staff roles will be separated and new use cases will be made for each role. (ex. Trainers can add exercises/routines but can’t delete users. Etc.)

2.3 Assumptions

It is assumed that:

- Business has an inventory database
- Business has a staff database (salary, staff information etc)
- Users are members of the gym
- All trainers, nutritionists are certified and provide legal advice.
- The Administrator and all staff understand how to use the program.
- Sensitive user information (ex. passwords) are safe and under the protection of the law.
- Administrator has rights to user and staff data.
- Users and staff have a device and internet connection to use the product.

2.4 Constraints

- Server load and capacity constrained by third party hosting service.
- In Django model relationships have ON DELETE CASCADE by default, which is a poor choice because it is not safe — one might lose data because of this.
- The product should be easy to use.
- All client’s requests must be met.
- The product must be finished in June.

2.5 Dependencies

- Python
- Django Framework
- (Future Django packages)
- Bootstrap 4
- Jquery

3. Requirements

- Describe all system requirements in enough detail for designers to design a system satisfying the requirements and testers to verify that the system satisfies requirements.
- Organize these requirements in a way that works best for your project. See Appendix DAppendix D, Organizing the Requirements for different ways to organize these requirements.
- Describe every input into the system, every output from the system, and every function performed by the system in response to an input or in support of an output. (Specify what functions are to be performed on what data to produce what results at what location for whom.)
- Each requirement should be numbered (or uniquely identifiable) and prioritized.

  See the sample requirements in Functional Requirements, and System Interface/Integration, as well as these example priority definitions:

  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

  It may be helpful to phrase the requirement in terms of its priority, e.g., "The value of the employee status sent to DIS **must be** either A or I" or "It **would be nice** if the application warned the user that the expiration date was 3 business days away". Another approach would be to group requirements by priority category.

- A good requirement is:
  - Correct
  - Unambiguous (all statements have exactly one interpretation)
  - Complete (where TBDs are absolutely necessary, document why the information is unknown, who is responsible for resolution, and the deadline)
  - Consistent
  - Ranked for importance and/or stability
  - Verifiable (avoid soft descriptions like "works well", "is user friendly"; use concrete terms and specify measurable quantities)

- Modifiable (evolve the Requirements Specification only via a formal change process, preserving a complete audit trail of changes)
- Does not specify any particular design
- Traceable (cross-reference with source documents and spawned documents).

- ***Functional Requirements***

In the example below, the requirement numbering has a scheme - BR_LR_0## (BR for Business Requirement, LR for Labor Relations). For small projects simply BR-## would suffice. Keep in mind that if no prefix is used, the traceability matrix may be difficult to create (e.g., no differentiation between '02' as a business requirement vs. a test case)

The following table is an example format for requirements. Choose whatever format works best for your project.

For Example:

Req#	**Requirement**	**Comments**	**Priority**	**Date Rvwd**	**SME Reviewed / Approved**
BR_1	The system should register every new user: client, manager, economist, trainer. .	User Roles should be clearly defined in the database	1	20/04/2021	Xhesika Bicaku, Rei Balla
BR_2	Every user has different views and sees different information.	User dashboard with 3 column design, Staff/Admin dashboard with 2 column (as discussed)	1	20/04/2021	Xhesika Bicaku, Rei Balla
BR_3	Each of the trainers can create a routine or exercises and add them to the database.	Exercises and routines should be created for all types of users.	1	20/04/2021	Xhesika Bicaku, Rei Balla
BR_4	Each customer can choose the routine that he wants to train.	Routines should be accessible from every user role.	2	20/04/2021	Xhesika Bicaku, Rei Balla

BR_5	Each customer can create his/her own routine.	Created routines should be updated on screen in realtime.	1	20/04/2021	Xhesika Bicaku, Rei Balla
BR_6	Each customer can delete his/her own routine.	Deleted routines should be updated on screen in realtime.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_7	Each customer can change his/her schedule of training.	The user's dashboard schedule should be editable.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_8	Managers should provide different diet plans that can be selected by the clients.	Diet plans should be confirmed by nutritionists and can be paid by cash or online.	3	20/04/2021	Xhesika Bicaku, Rei Balla
BR_9	The system should provide an event section where the clients get notified about different events in the gym during the week such as : zumba class, cross fit day, abs day ect.	Events should only be viewed on the user dashboard and be controlled by Managers.	3	20/04/2021	Xhesika Bicaku, Rei Balla
BR_10	The trainers can edit the working hours and the available days.	Editable trainers schedules on trainer dashboards.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_11	The managers should control the events happening in the gym.	Access to the event table is only to Managers.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_12	The economist should be responsible for the inventory of the gym.	Inventory includes all gym equipment.	1	20/04/2021	Xhesika Bicaku, Rei Balla
BR_13	The economist should be responsible for the wages of the staff.	Staff will be notified by email for every wage update.	2	20/04/2021	Xhesika Bicaku, Rei Balla

BR_14	The economist should keep track of the broken equipment in the gym in order for them to be replaced.	Administrators should be notified for every broken equipment.	3	20/04/2021	Xhesika Bicaku, Rei Balla
BR_15	Trainers should be able to edit the exercises and routines if necessary.	All updates to exercise and routines will be updated in real time.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_16	Clients should see their progress.	Progress must be viewable at each user's profile.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_17	The system should keep track of the clients that come in the gym during the day.	Number and information of active clients should be accessible to admins, managers etc.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_18	The manager should control the staff and their salaries.	Economists and staff should be notified for every update.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_19	Financial activity should be controlled by the manager.	The economist can not control the balance but only get information and calculate results.	2	20/04/2021	Xhesika Bicaku, Rei Balla
BR_20	The manager should control the events and can add events.	Events should be updated for every user role.	3	20/04/2021	Xhesika Bicaku, Rei Balla
BR_21	The administrator can add notes or notifications in the system.	Admin's notes can be viewed by the staff.	3	20/04/2021	Xhesika Bicaku, Rei Balla

3.1 Non-Functional Requirements

3.1.1 User Interface Requirements

- User Experience must follow Google's Material Design and Material Studies.
- It should be implemented in Bootstrap 4, jquery and custom css only.
- Application must run in every browser.
- Every dashboard must follow the 2-3 column grid layout.

- User dashboard will have a 3 column grid layout.
- Staff and Admin dashboard should have 2 column grid layout.
- Every page must be mobile compatible.
- Every user role must be prevented from accessing pages of other user roles. Such as user accessing manager view etc.
- Every staff dashboard should contain interactable editable tables not only views.
- User exercises and routines should be public to other user profiles.
- Other requirements will be added in the future.

3.1.2 Usability

- The system should be easy to use by the members of the gym, the trainers ,manager and the economist.
- The clients should be able to create an account to log in into their personal pages.
- The system should be organized in a way that user error possibility is minimized.
- Trainers should be able to access the clients routines during the working hours.
- It should be easy for the users to organize their schedules of working hours.
- Clients should be able to use all the gym’s facilities.

3.1.3 Performance

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 a 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”.

3.1.3.1 Capacity

Include measurable capacity requirements (e.g., the number of simultaneous users to be supported, the maximum simultaneous user load, per-user memory requirements, expected application throughput)

3.1.3.2 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).

3.1.3.3 Latency

Include explicit latency requirements, e.g., the maximum acceptable time (or average time) for a service request.

3.1.4 Manageability/Maintainability

3.1.4.1 Monitoring

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

3.1.4.2 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.

3.1.4.3 Operations

Specify any normal and special operations required by the user, including:

- periods of interactive operations and periods of unattended operations
- data processing support functions
- backup and recovery operations
- safety considerations and requirements
- disaster recovery and business resumption

3.1.5 System Interface/Integration

Specify the use of other required products (e.g., a database or operating system), and interfaces with other systems (e.g., UWHires package interfaces with PubCookie and ODS, HEPPS system interfaces with Budget system). For each interface, define the interface in terms of message format and content. For well-documented interfaces, simply provide a reference to the documentation.

Outline each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

3.1.5.1 Network and Hardware Interfaces

Specify the logical characteristics of each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

3.1.5.2 Systems Interfaces

Example systems interface requirements:

A. System1-to-System2 Interface

The <external party> will create and send a fixed length text file as an email attachment to System2mail@u.washington.edu to be imported into the System2 system for payroll calculation. This file must be received on EDIT day by 4:00 PM in order to be processed in the EDIT night run. The requirements below document the file specifications, data transfer process, and specific schedule. This file is referred to as "FileName" in this document.

File Structure and Format

A1. The FileName file is a fixed length text file.

A2. The FileName file is an unformatted ASCII file (text-only).

A3. The FileName file contains a batch totals record and several detail records.

File Description: Batch Totals Record

A4. The batch totals record can be placed at the beginning, in the middle, or at the end of the file.

A5. The batch totals record contains the following:

- Record Type (value: XA)
- Process Type (value: A)
- Batch Number (3 digit number assigned by Payroll Dept)
- Origin Code (AIG)
- Total number of detail records
- Total deduction amount

File Description: Detail Records

A6. The FileName file contains a row for each record meeting xxx criteria.

A7. Each row in the FileName file contains the following fields, comma-delimited and encased in double-quotes where the data includes commas or spaces:

- Employee Id
- Record Type
- Process Date (MMDDYY)
- XYG Number
- Element Code
- Amount
- Amount Sign
- Year Flag
- Total Amount
- Total Amt Sign

3.1.6 Security

3.1.6.1 Protection

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

3.1.6.2 Authorization and Authentication

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

3.1.7 Data Management

Specify the requirements for any information that is to be placed into a database, including

- types of information used by various functions
- frequency of use
- data access rules
- data entities and relationships
- integrity constraints

- data retention
- valid range, accuracy, and/or tolerance
- units of measure
- data formats
- default or initial values

3.1.8 Standards Compliance

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.

3.1.9 Portability

If portability is a requirement, specify attributes of the system that relate to the ease of porting the system to other host machines and/or operating systems. For example,

- Percentage of components with host-dependent code;
- Percentage of code that is host dependent;
- Use of a proven portable language;
- Use of a particular compiler or language subset;
- Use of a particular operating system;
- The need for environment-independence - the product must operate the same regardless of operating systems, networks, development or production environments.

3.1.10 Other Non-Functional Requirements

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

3.2 Domain Requirements

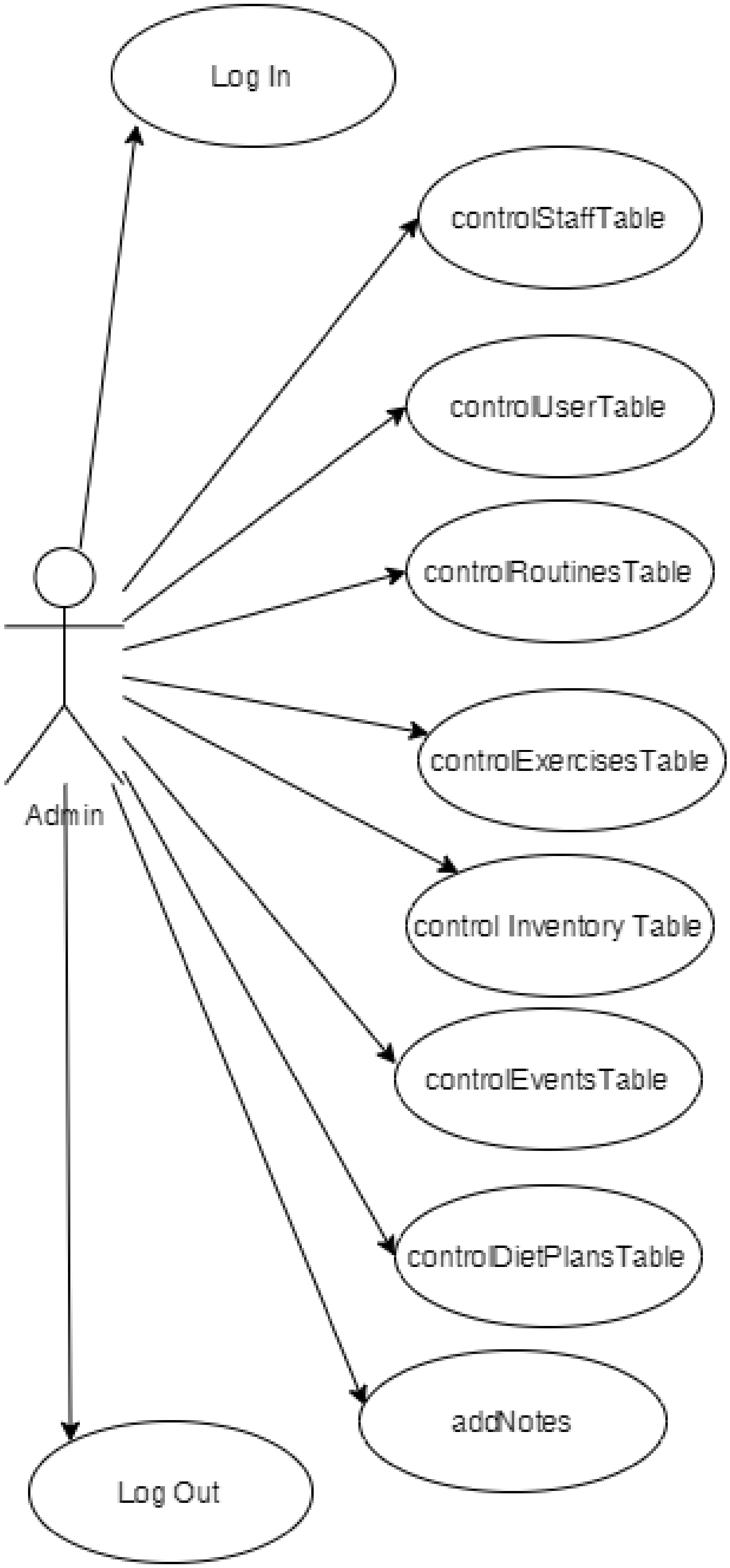
Everything related to the domain that might be needed in the project shall be mentioned here. Sometimes the domain Requirements might be thought as part of either functional or non-functional requirements.

4. User Scenarios/Use Cases

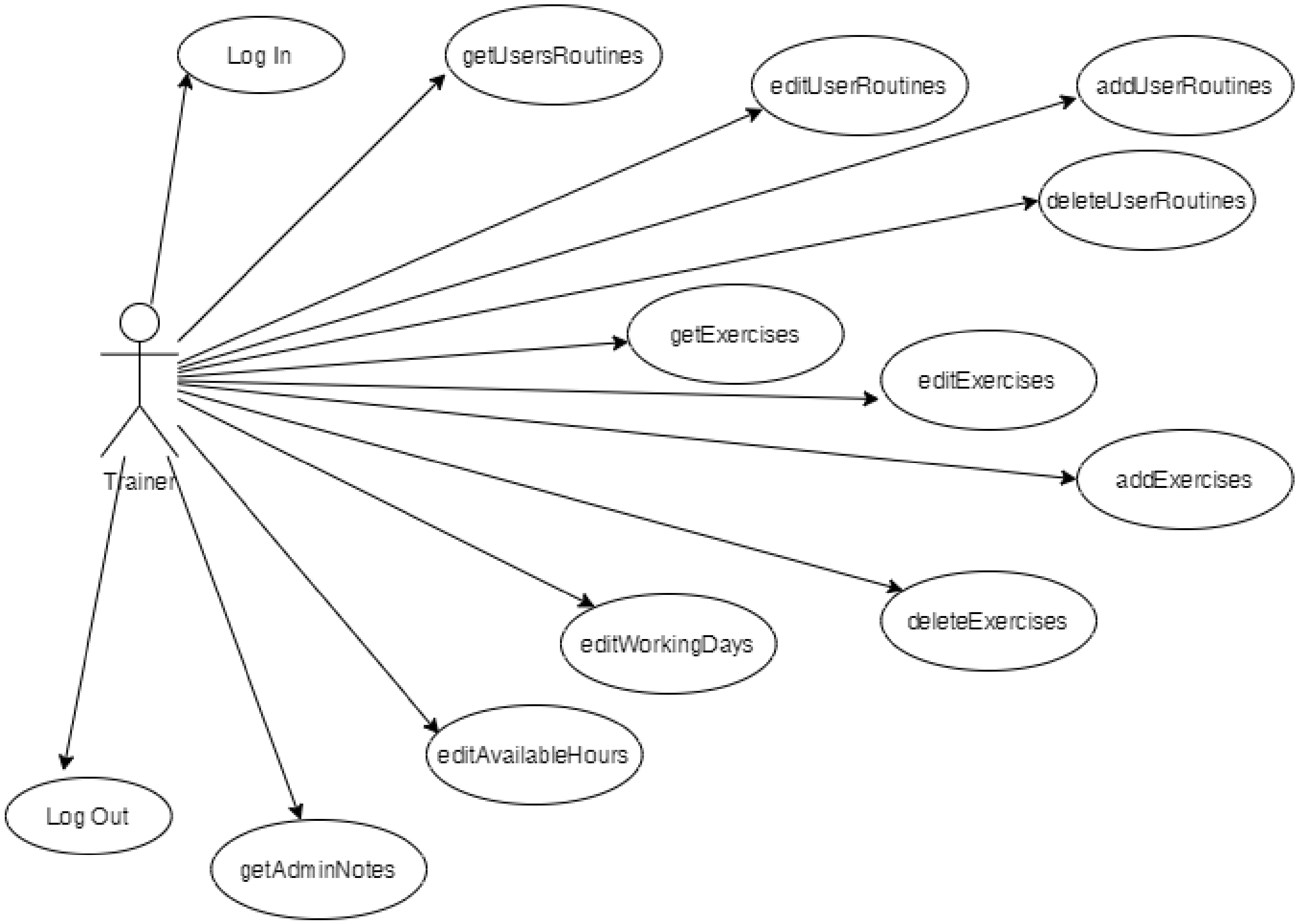
Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

- 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

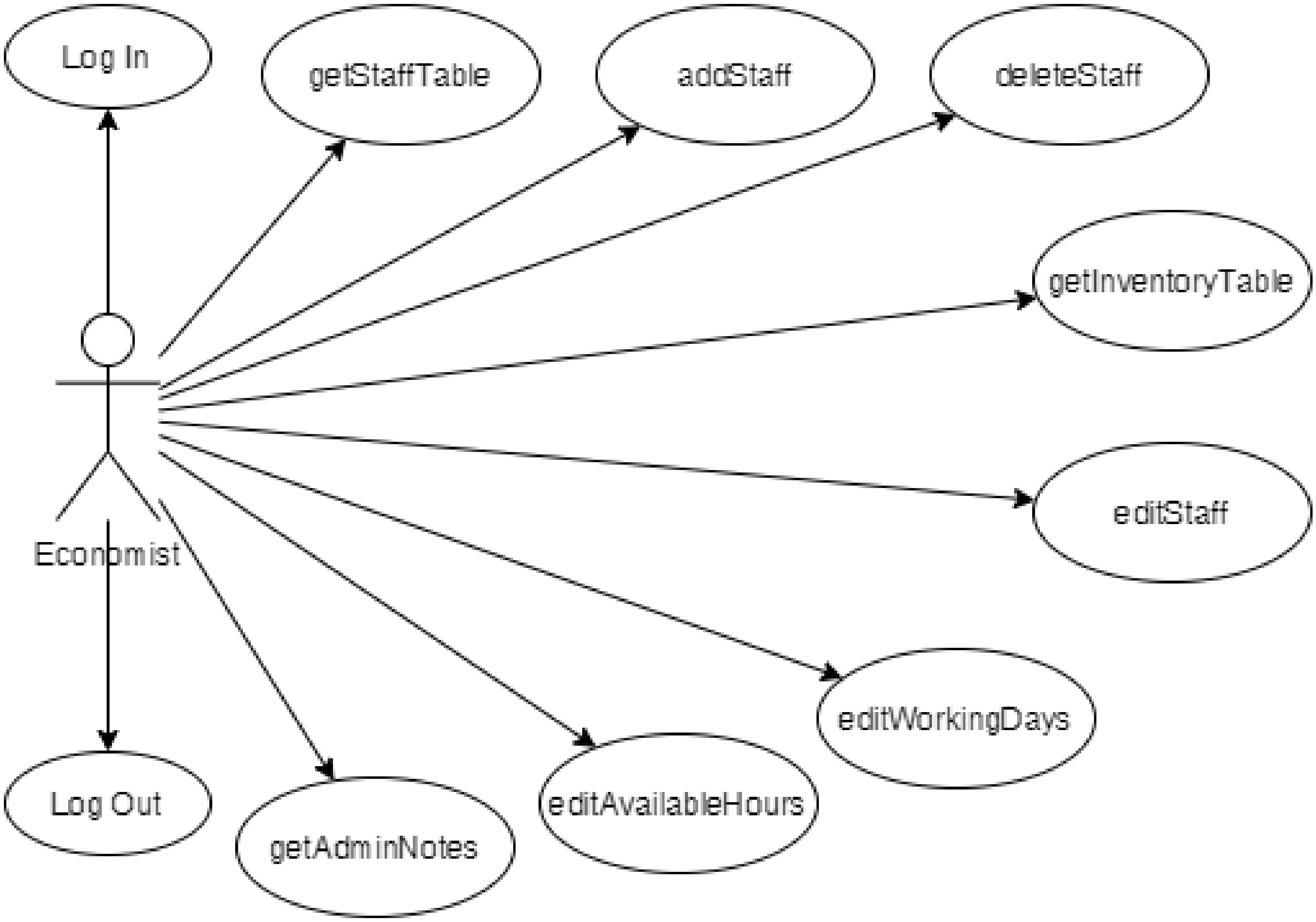
Use Case 1 - Administrator



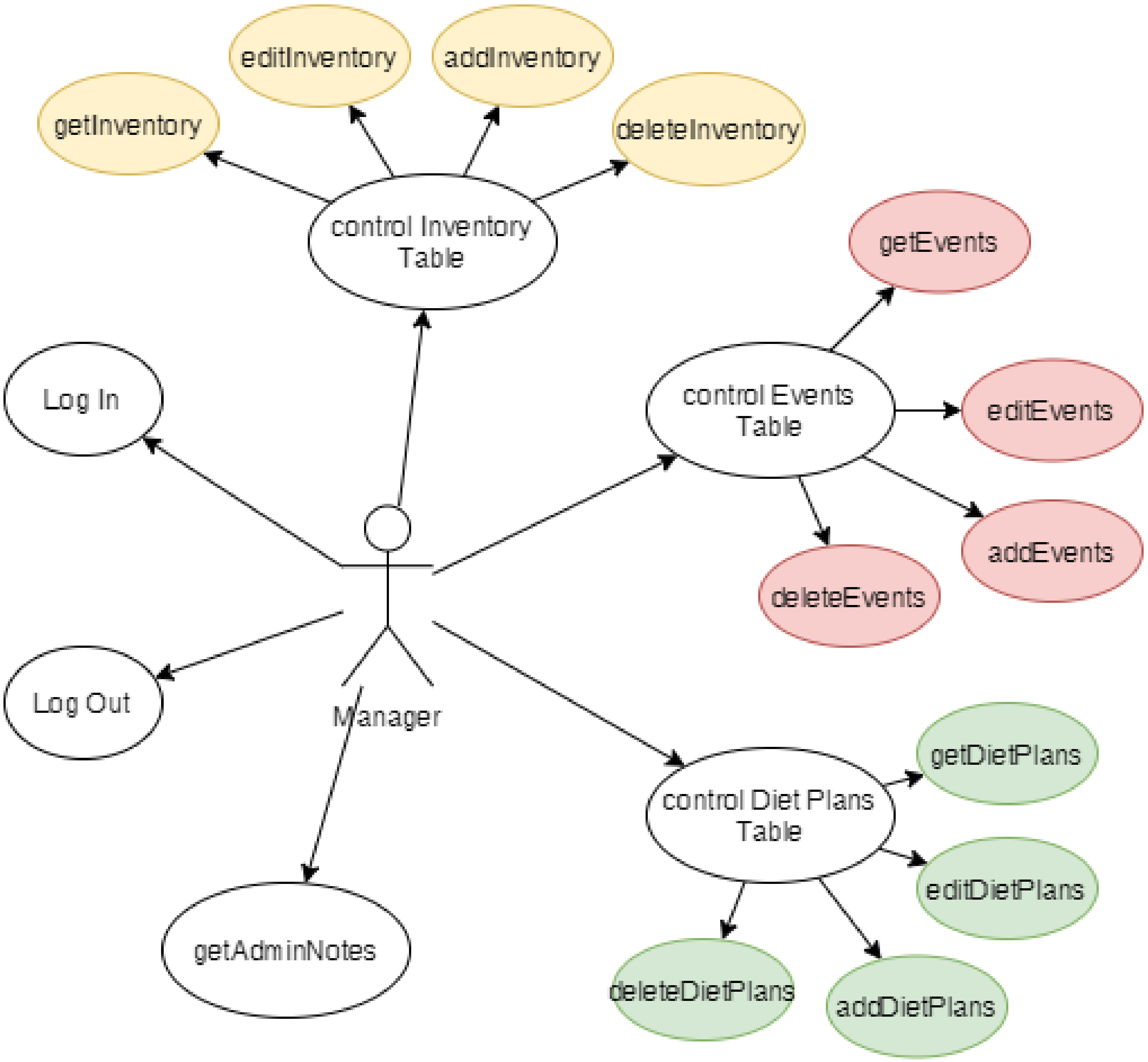
Use Case 2 - Staff - Trainer



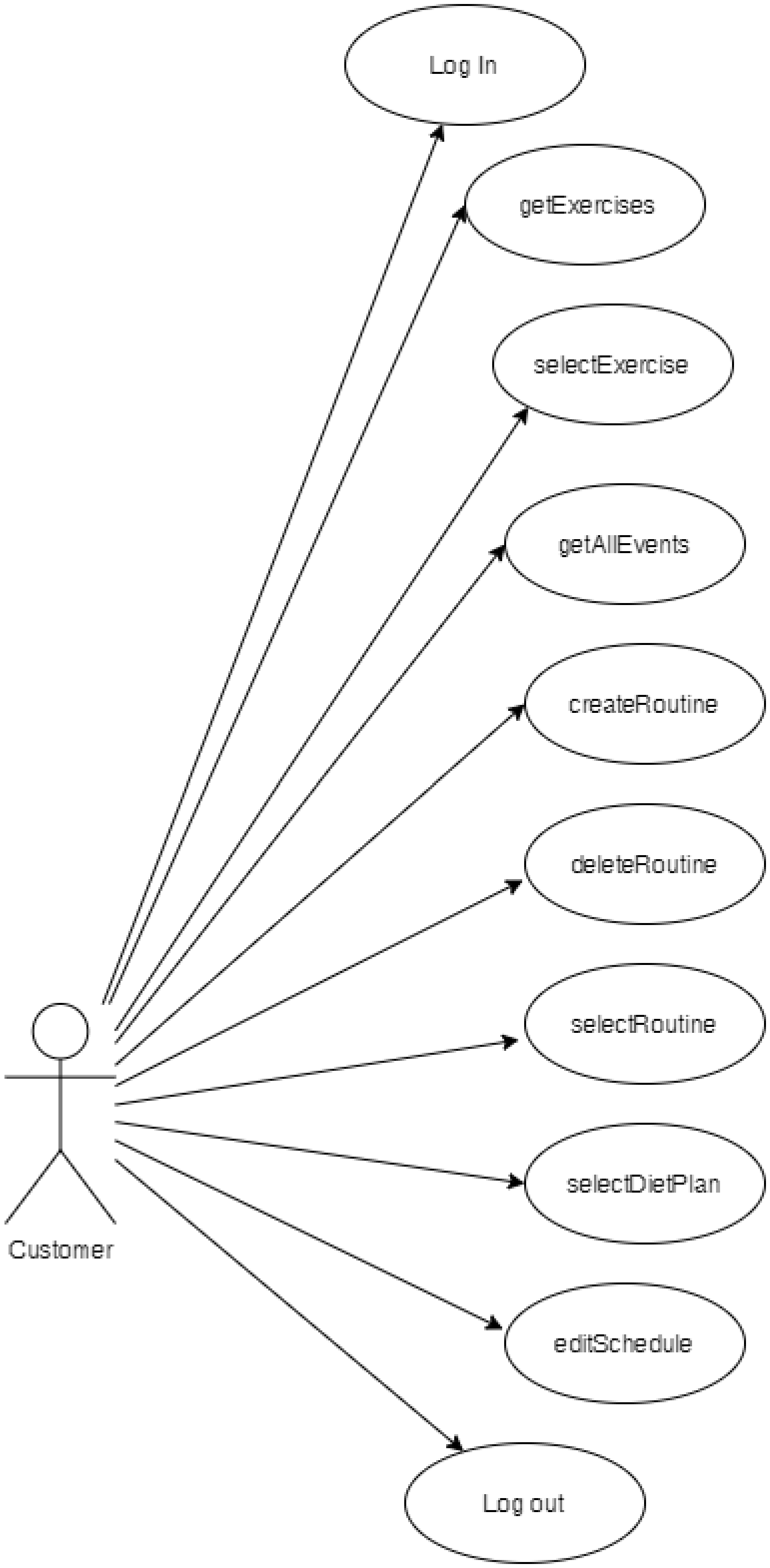
Use Case 3 - Staff - Economist



Use Case 4 - Staff - Manager



Use Case 5 - User - Customer



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.

Appendix A. Definitions, Acronyms, and Abbreviations

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

Appendix B. References

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

Appendix C. 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	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted

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-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											

Appendix D. 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.