Software Requirements Specification

Version 1.0

May 20, 2021

Programming Languages Dictionary System

Mahmoud Mahmoud

Gaafar Soliman

Nour Hammouda

Maya Al-Okda

Submitted as Homework

Of Software Engineering

In Al-Baath University

Supervised By

Eng. Nahla Saad Al-deen

Table of Contents

1.0. Introduction .......................................................................................................................... 4

1.1. Purpose ............................................................................................................................... 4

1.2. Scope of Project .................................................................................................................. 4

1.3. Glossary .............................................................................................................................. 4

1.4. References .......................................................................................................................... 5

1.5. Overview of Document ........................................................................................................ 5

2.0. Overall Description ............................................................................................................... 6

2.1 System Environment .......................................................................................................... 6

2.2 Functional Requirements Specification .......................................................................... 7

2.2.1 User Use Cases ........................................................................................... 7

Use case: Search Use Case..................................................... 8

Use case: Submit Use Case..................................................... 9

2.2.2 Admin Use Cases......................................................................................... 10

Use case: Receive Function .................................................... 11

Use case: Add Function …….................................................... 12

Use case: Edit Function …….................................................... 13

Use case: Remove Function .................................................... 14

Use case: Add Admin ……….................................................... 15

Use case: Remove Admin ………............................................. 16

2.3 User Characteristics ........................................................................................................17

2.4 Non-Functional Requirements .........................................................................................17

3.0. Requirements Specification ................................................................................................18

3.1 External Interface Requirements ......................................................................................18

3.2 Functional Requirements .................................................................................................19

3.2.1 Search for function ........................................................................................19

3.2.2 Submit a function ..........................................................................................20

3.2.3 Receiving a function .....................................................................................20

3.2.4 Adding a new function...................................................................................21

3.2.5 Edit an existing Function ..............................................................................21

3.2.6 Remove an existing function.........................................................................22

3.2.7 Adding a new admin ......................................................................................23

3.2.8 Remove an existing admin ............................................................................23

3.3 Detailed Non-Functional Requirements ...............................................................................24

3.3.1 Logical Structure of the Data ............................................................................................24

3.3.2 Security ............................................................................................................................25

**1.0 Introduction**

*1.1. Purpose*

The purpose of this document is to present a detailed description of the dictionary system. It will help programmers to search for functions and how to use it and what arguments should they pass, it will ease the work and make the maximum effectiveness of time and resources.

*1.2. Scope of Project*

This dictionary system will be a global Web Service to help programmers around the world finding functions to do what they need, it could be built-in or user defined, the system is managed by multiple administrators, the users could submit new functions to be published but this these functions will need to be reviewed and approved by the administrators, the communication between administrators and users will be by via E-Mail.

The System will contain a relational database containing a list of functions and similar functions in other programming languages.

*1.3. Glossary*

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Function Page | A page where the function from dictionary is displayed. |
| Admins (Administrators) | Users with high Privileges like add, remove, modify a function page. |
| Database | Collection of all the information monitored by this  system. |
| Field | A cell within a form. |

*1.4. References*

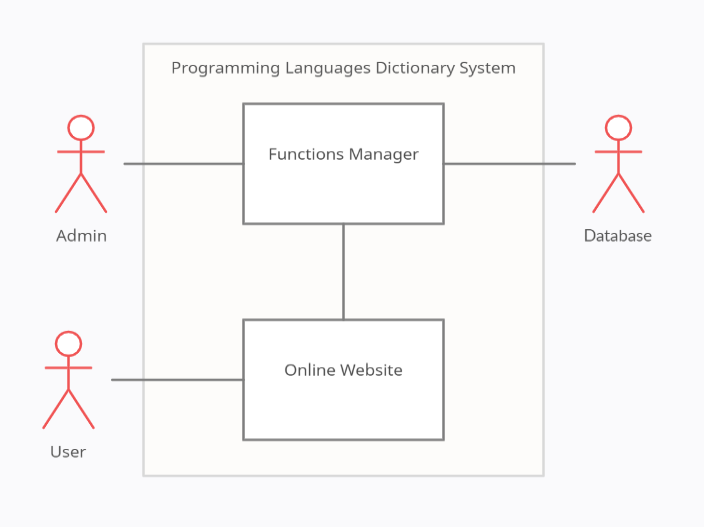
* IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.
* [www.sourcecodesolutions.in](http://www.sourcecodesolutions.in)

*1.5. Overview of Document*

* This system is a web-based application that helps programmers to find their needs faster and more efficient, in the next chapter you will find an informal overview of the functionality of the product.
* And in the third chapter you will find a detailed description for the developers in technical terms about the functionality of the product.
* Both of these chapters describe the same software but are intended for different audiences.

**2.0 Overall Description**

*2.1. System Environment*

Figure 1 – System Environment

The System has only 2 Actors Admin and User, Both can communicate via E-mail, the User can access the online website and submit new functions, the Admins can use Functions Manager to add, modify, remove, or add new Functions.

*2.2. Functional Requirements Specification*

This section will show the use cases of each Actor in this System.

*2.2.1. User Use Cases*

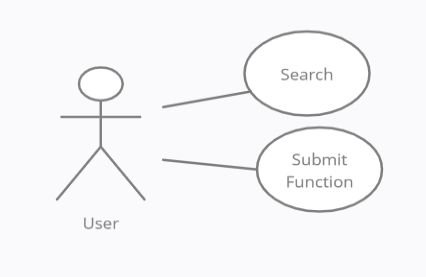
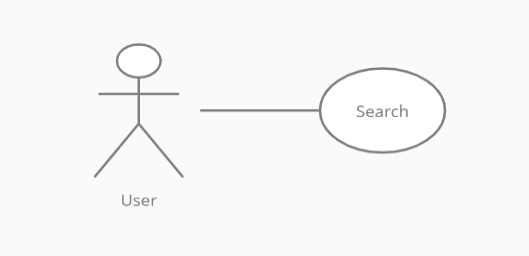


Figure 2 – User Use Cases

**Search Use Case**

**Use Case:** Search for Function

**Diagram:**



**Brief Description**

The User accesses the Online Website, searches for a Function.

**Initial Step-By-Step Description**

Before this use case can be initiated, the User has already accessed the Online Website.

1. The User chooses to search by author name, category, or keyword.

2. The system displays the choices to the User.

3. The User selects the Page desired.

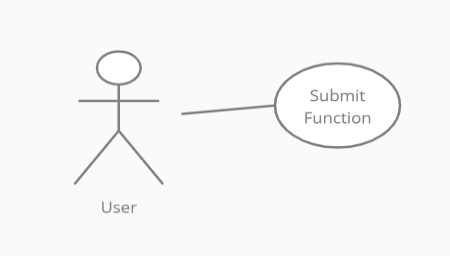
4. The system presents page of the function to the user.

**Xref**: Section 3.2.1

**Submit Use Case**

**Use Case:** Submit a new Function

**Diagram:**



**Brief Description**

The User either submits a new function or resubmits an edited function.

**Initial Step-By-Step Description**

Before this use case can be initiated, the User has already accessed the Online Website.

1. The User chooses the *Submit new function* button.

2. The System Shows a fillable Interface to submit the function.

3. The User fills in the Subject, Description, and write the function info (User can attach a file instead) and then submit it.

4. The System generates a request and store it internally.

5. The System generates and sends an email acknowledgement.

**Xref**: Section 3.2.2

*2.2.2. Admin Use Cases*

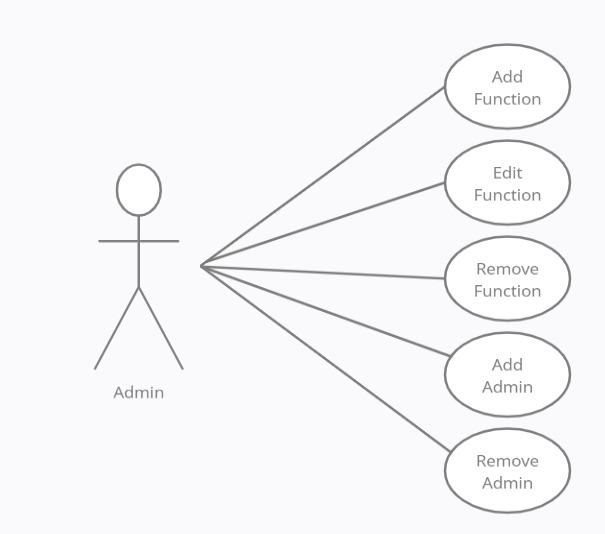
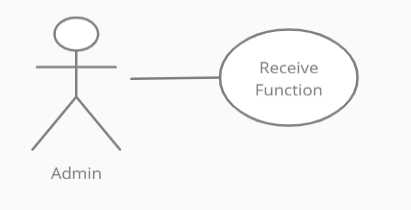


Figure 3 – Admin Use Cases

**Receive Function Use Case**

**Use Case:** Receive Function

**Diagram:**



**Brief Description**

The Admin accesses the Function Manager Page in the Online Website, and see the added function requests.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Admin has already accessed the Function Manager Page in the Online Website.

1. The Admin chooses the *Received Functions* Button in the Function Manager Page.

2. The system displays the received functions to the Admin.

3. The Admin selects a Function Page desired.

4. The system presents page of the function to the Admin.

5.The Admin can Fill the Author field or Edit the page then Submit the Function (or the Admin can remove the received Function).

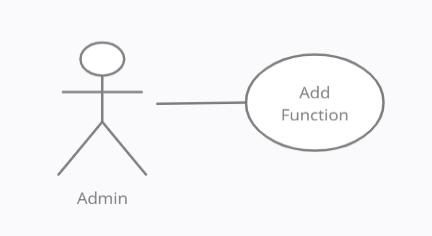
6.The System verifies the information and returns the Admin to the Function Manager main page.

**Xref**: Section 3.2.3

**Add Function Use Case**

**Use Case:** Add Function

**Diagram:**



**Brief Description**

The Admin accesses the Function Manager Page in the Online Website, and add a new function.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Admin has already accessed the Function Manager Page in the Online Website.

1. The Admin chooses the *Add new Function* Button in the Function Manager Page.

2. The system displays the Interface to the Admin.

3. The Admin Fills the information of the function and submit it.

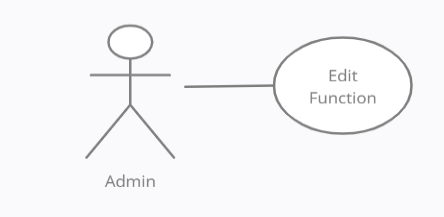
4. The System verifies the information and returns the Admin to the Function Manager main page.

**Xref**: Section 3.2.4

**Edit Function Use Case**

**Use Case:** Edit Function

**Diagram:**



**Brief Description**

The Admin accesses the Function Manager Page in the Online Website, and edit an existing function.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Admin has already accessed the Function Manager Page in the Online Website.

1. The Admin chooses the *Edit an existing Function* Button in the Function Manager Page.

2. The system displays the Interface filled of old function information to the Admin.

3. The Admin edits the information of the function and resubmit it.

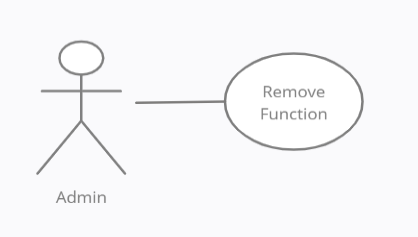
4. The System verifies the information and returns the Admin to the Function Manager main page.

**Xref**: Section 3.2.5

**Remove Function Use Case**

**Use Case:** Remove Function

**Diagram:**



**Brief Description**

The Admin accesses the Function Manager Page in the Online Website, and remove an existing function.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Admin has already accessed the Function Manager Page in the Online Website.

1. The Admin chooses the *Remove an existing Function* Button in the Function Manager Page.

2. The system displays the Interface filled of old function information to the Admin and ask him to make sure.

3. The Admin press *Yes Button* from the interface.

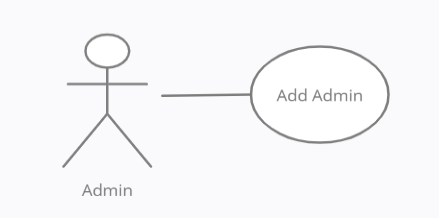
4. The System remove the Function and returns the Admin to the Function Manager main page.

**Xref**: Section 3.2.6

**Add Admin Use Case**

**Use Case:** Add Admin

**Diagram:**



**Brief Description**

The Admin accesses the Function Manager Page in the Online Website, and add a new Admin.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Admin has already accessed the Function Manager Page in the Online Website.

1. The Admin chooses the *Add new Admin* Button in the Function Manager Page.

2. The system displays the Interface to the Admin.

3. The Admin Fills the information of the Admin and submit it.

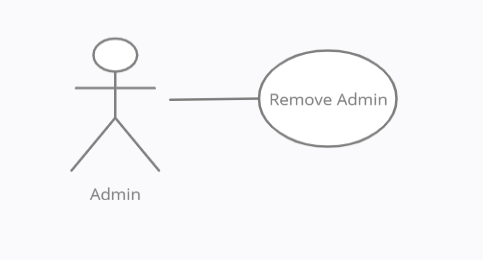
4. The System verifies the information and returns the Admin to the Function Manager main page.

**Xref**: Section 3.2.7

**Remove Admin Use Case**

**Use Case:** Remove Function

**Diagram:**



**Brief Description**

The Admin accesses the Function Manager Page in the Online Website, and remove an existing Admin.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Admin has already accessed the Function Manager Page in the Online Website.

1. The Admin chooses the *Remove an existing Admin* Button in the Function Manager Page.

2. The system displays the Interface filled of the Admin which need to be removed information to the Admin and ask him to make sure.

3. The Admin press *Yes Button* from the interface.

4. The System remove the Admin and returns the Admin to the Function Manager main page.

**Xref**: Section 3.2.8

*2.3. User Characteristics*

* The User is expected to be Internet literate and be able to use a search engine.
* The Admin is expected to be Windows literate and to be able to use button, pull-down menus, and similar tools.

*2.4. Functional Requirements*

* The Online Website will be on a server with high-speed Internet capability.
* The physical machine to be used will be determined by the Owner.
* The software developed here assumes the use of a tool such as Tomcat for connection between the Web pages and the database.
* The speed of the User’s connection will depend on the hardware used rather than characteristics of this system.
* The Function Manager will run on the User’s PC and will contain a MySQL database.
* MySQL is already installed on this computer and is a Windows operating.

**3.0 Requirements Specification**

*3.1. External Interface Requirements*

* The only link to an external system is the link to the Dictionary

Database to verify the authority of an Admin or to modify a Function.

* The Database fields of Admin are admin’s name, User ID number, email address, and Password.
* The Add Admin use case requests a list of User names, User ID numbers and email addresses when adding a new Admin. It returns a Boolean for Admin status when updating an Admin.
* The Database fields of Function are function’s name, Function ID number, Description, and Code.
* The Edit/Remove Function use case requests a list of Function names, Function ID numbers when editing/removing a existing Function.

*3.2. Functional Requirements*

*3.2.1. Search for Function*

|  |  |
| --- | --- |
| **Use Case Name** | Search for Function |
| **Xref** | Section 2.2.1 Search for Function |
| **Trigger** | The User accesses the Online Website |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The User chooses how to search the Web site. The choices: are by Author, by Category, and by Keyword.  2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of a Function with multiple authors, each is contained in the list.  3. The User selects an author.  4. The system creates and presents a list of all Functions by that author in the database.  5. The User selects a function.  6. The system displays the Function page for the article.  7. The User selects to return to the main page list or to the previous list. |
| **Alternative Paths** | **In step 2**, if the User selects to search by category, the system creates and presents a list of all categories in the database.  3. The User selects a category.  4. The system creates and presents a list of all functions in that category in the database.  **Return to step 5.**  **In step 2**, if the User selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.  3. The User enters a keyword or phrase.  4. The system searches all functions with  that keyword or phrase (in description) and creates and presents a list of all such functions in the database.  **Return to step 5.** |
| **Postcondition** | None |
| **Exception Paths** | The User may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when the function was added. |

*3.2.2. Submit a Function*

|  |  |
| --- | --- |
| **Use Case Name** | Submit a Function |
| **Xref** | Section 2.2.1 Submit a Function |
| **Trigger** | The User selects add a function button |
| **Precondition** | The Web is displayed with add a function button |
| **Basic Path** | 1. The system displays a fillable form.  2. The User fills the fields (function name, function description, function code, function language, function categories “Optional”, user’s email, author name) and select Submit.  3. The system verify the info and display a response message.  4. The User is returned to the website main page. |
| **Alternative Paths** | **In step 2,** if any field except categories is not filled probably an error message will appear explaining the problem and user won’t be redirected to home page. |
| **Postcondition** | The System will save the info internally |
| **Exception Paths** | The User may abandon the submitting at any time |
| **Other** | None |

*3.2.3. Receiving a Function*

|  |  |
| --- | --- |
| **Use Case Name** | Receive a Function |
| **Xref** | Section 2.2.2 Receive Function |
| **Trigger** | The Admin selects *Received Functions* button from Function Manager |
| **Precondition** | The Function Manager web page is displayed with *Received Functions* button |
| **Basic Path** | 1. The system displays a fillable form with the function info that submitted by user.  2. The Admin review the fields and select Submit.  3. The system verify the info and display a response message.  4. The Admin is returned to the Function Manager page. |
| **Alternative Paths** | **In step 2,** the Admin can edit any field and submit it.  **Return to Step 3.**  **In step 2,** the Admin can discard instead of submit.  **Return to Step 4.**  **In step 3,** if any field except categories is not filled probably an error message will appear explaining the problem and Admin won’t be redirected to Function Manager page. |
| **Postcondition** | The System will save the function to the database |
| **Exception Paths** | The Admin may abandon the submitting at any time |
| **Other** | The fillable form will contain (function name, function description, function code, function language, function categories “Optional”, user’s email, author name) fields. |

*3.2.4 Adding a new function*

|  |  |
| --- | --- |
| **Use Case Name** | Add a new Function |
| **Xref** | Section 2.2.2 Add a new Function |
| **Trigger** | The Admin selects *Add a new Function* button from Function Manager |
| **Precondition** | The Function Manager web page is displayed with *Add a new Function* button |
| **Basic Path** | 1. The system displays a fillable form.  2. The Admin fill the fields and select Submit.  3. The system verify the info and display a response message.  4. The Admin is returned to the Function Manager page. |
| **Alternative Paths** | **In step 3,** if any field except categories is not filled probably an error message will appear explaining the problem and Admin won’t be redirected to Function Manager page. |
| **Postcondition** | The System will save the function to the database |
| **Exception Paths** | The Admin may abandon the submitting at any time |
| **Other** | The fillable form will contain (function name, function description, function code, function language, function categories “Optional”, author name) fields. |

*3.2.5 Edit an existing Function*

|  |  |
| --- | --- |
| **Use Case Name** | Edit an existingFunction |
| **Xref** | Section 2.2.2 Edit an existingFunction |
| **Trigger** | The Admin selects *Edit an existing Function* button from Function Manager |
| **Precondition** | The Function Manager web page is displayed with *Edit an existing Function* button |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all Functions in the database.  2. The Admin select a Function from the list.  3. The system displays a fillable form with the function info.  4. The Admin edit the wanted fields and select Submit.  5. The system verify the info and display a response message.  6. The Admin is returned to the Function Manager page. |
| **Alternative Paths** | **In step 4,** if any field except categories is not filled probably an error message will appear explaining the problem and Admin won’t be redirected to Function Manager page. |
| **Postcondition** | The System will update the function in the database |
| **Exception Paths** | The Admin may abandon the editing at any time |
| **Other** | The fillable form will contain (function name, function description, function code, function language, function categories “Optional”, author name) fields. |

*3.2.6 Remove an existing Function*

|  |  |
| --- | --- |
| **Use Case Name** | Remove an existingFunction |
| **Xref** | Section 2.2.2 Remove an existingFunction |
| **Trigger** | The Admin selects *Remove an existing Function* button from Function Manager |
| **Precondition** | The Function Manager web page is displayed with *Remove an existing Function* button |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all Functions in the database.  2. The Admin select a Function from the list.  3. The system displays a grid of the function info with Yes/No buttons.  4. The Admin selects Yes.  5. The system verify the request and display a response message.  6. The Admin is returned to the Function Manager page. |
| **Alternative Paths** | **In step 3,** if the Admin selects No he/she is redirected to the Function Manager page.  **In step 5,** if any error accords a message will appear explaining the problem and Admin won’t be redirected to Function Manager page. |
| **Postcondition** | The System will remove the function in the database |
| **Exception Paths** | The Admin may abandon the removing at any time |
| **Other** | The grid info contains (function name, function description, function code, function language, function categories, author name) fields. |

*3.2.7 Adding a new Admin*

|  |  |
| --- | --- |
| **Use Case Name** | Add a new Admin |
| **Xref** | Section 2.2.2 Add a new Admin |
| **Trigger** | The Admin selects *Add a new Admin* button from Function Manager |
| **Precondition** | The Function Manager web page is displayed with *Add a new Admin* button |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all Users in the database.  2. The Admin select a User from the list.  3. The system displays an info grid with *Make Admin* button.  4. The Admin selects *Make Admin* button.  3. The system verify the info and display a response message.  5. The Admin is returned to the Function Manager page. |
| **Alternative Paths** | None |
| **Postcondition** | The System will update the user info in the database |
| **Exception Paths** | The Admin may abandon the adding at any time |
| **Other** | The info grid will contain (User name, User ID, User E-mail) fields. |

*3.2.8 Remove an existing Admin*

|  |  |
| --- | --- |
| **Use Case Name** | Remove an existingAdmin |
| **Xref** | Section 2.2.2 Remove an existingAdmin |
| **Trigger** | The Admin selects *Remove an existing Function* button from Function Manager |
| **Precondition** | The Function Manager web page is displayed with *Remove an existing Admin* button |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all admins in the database.  2. The Admin select an admin from the list.  3. The system displays a grid of the admin info with Yes/No buttons.  4. The Admin selects Yes.  5. The system verify the request and display a response message.  6. The Admin is returned to the Function Manager page. |
| **Alternative Paths** | **In step 3,** if the Admin selects No he/she is redirected to the Function Manager page.  **In step 5,** if any error accords a message will appear explaining the problem and Admin won’t be redirected to Function Manager page. |
| **Postcondition** | The System will update the user info in the database |
| **Exception Paths** | The Admin may abandon the removing at any time |
| **Other** | The info grid will contain (User name, User ID, User E-mail) fields. |

*3.3 Detailed Non-Functional Requirements*

*3.3.1 Logical Structure of the Data*

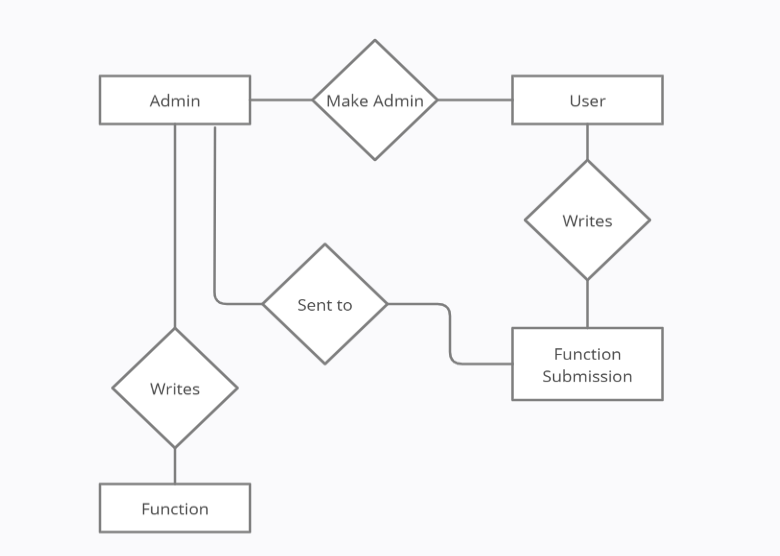


Figure 4 – Logical Structure of the System Function Manager

**Admin/User Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data item** | **Type** | **Description** | **Comment** |
| ID | Number | Identification number | Unique |
| Name | Text | Name of the User |  |
| E-mail Address | Text | Email Address | Unique |
| Admin | Boolean | Is the user Admin or not |  |

**Function Submission Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data item** | **Type** | **Description** | **Comment** |
| Submission ID | Number | Identification number | Unique |
| Description | Text | Description of the function |  |
| Code | Text | Function code |  |
| Language | Text | Programming language |  |
| Author | Text | Name of the Author | Could be Multiple Valued |
| Categories | Text | To help find the Function | Could be Multiple Valued |
| User Email Address | Text | Email Address | To send replay if required |

**Function Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data item** | **Type** | **Description** | **Comment** |
| Submission ID | Number | Identification number | Unique |
| Description | Text | Description of the function |  |
| Code | Text | Function code |  |
| Language | Text | Programming language |  |
| Author | Text | Name of the Author | Could be Multiple Valued |
| Categories | Text | To help find the Function | Could be Multiple Valued |

*3.3.2 Security*

* The server on which the Online Website resides will have its own security to prevent unauthorized add/edit/remove access, there is no restriction on read access.
* Only the Admin will have physical access to the machine and the program on it. There is no special protection built into this system other than to provide the Admin with add/edit/remove access to the Online website.