Requirements Engineering

GolfMemSYS

Submitted By: Michael Crowley McDevitt (T00200043)

Computing with Software Development

Date Submitted: 15/May/2020

**Table of Contents**

[1. Introduction/overview 3](#_Toc40353994)

[2. Functional Components 4](#_Toc40353995)

[3. User Requirements 5](#_Toc40353996)

[3.1. GolfMemSYS will manage Members 5](#_Toc40353997)

[3.2. GolfMemSYS will manage Renewals 5](#_Toc40353998)

[3.3. GolfMemSYS will perform Admin 5](#_Toc40353999)

[4. System Requirements 6](#_Toc40354000)

[4.1. System Level Use Case Diagram 7](#_Toc40354001)

[4.2. Manage Members 8](#_Toc40354002)

[**4.2.1.** **Add Membership Type** 8](#_Toc40354003)

[**4.2.2.** **Add Member** 10](#_Toc40354004)

[4.2.3. View Members 13](#_Toc40354005)

[4.2.4. Update Member 16](#_Toc40354006)

[4.3. Manage Renewals 22](#_Toc40354007)

[4.3.1. Issue Notices 22](#_Toc40354008)

[4.3.2. Record Payments 22](#_Toc40354009)

[4.4. Analyse Finance 22](#_Toc40354010)

[4.4.1. Create Yearly Revenue Report 22](#_Toc40354011)

[4.4.2. Carry out Membership Type Analysis 22](#_Toc40354012)

[5. System Model 23](#_Toc40354013)

[5.1. Level-0 DFD 23](#_Toc40354014)

[5.2. Level-1 DFD 24](#_Toc40354015)

[5.3. Level-2 DFD (Process P1: Manage Members) 25](#_Toc40354016)

[5.4. Level-2 DFD (Process P2: Send Renewal Notice) 26](#_Toc40354017)

[6. Data Model (Class Diagram) This section outlines the data stored by the system and gives graphical representations of this data and how it is stored. 27](#_Toc40354018)

[6.1. Class Diagram 27](#_Toc40354019)

[6.2. Relational Schema 28](#_Toc40354020)

[6.3. Database Schema 29](#_Toc40354021)

[7. Conclusion 31](#_Toc40354022)

# Introduction/overview

This document contains details to create a golf membership system that provides functions for member management, member type management, finance management and renewal tracking. The system will have three processes: Manage members, finance management and renewal management. Each will have its own functions.

Manage Members will have four functions: Add Member, Update Member, View Member and Remove Member. Add Member will allow the user to add a new member to the system. Update Members will allow the user to update an existing member. Remove Member will set the status of an existing member to ‘removed’ so that they no longer count towards the renewal process.

Manage Finance will have two functions: Yearly Revenue Analysis and Record Payment. Yearly Revenue Analysis will show the income of the golf course on a yearly basis. Record Payment will allow the system to store the payment paid by each member.

Manage Renewals will have two functions: Issue Renewal Notice and Membership Type Analysis. Issue Renewal Notice allows the system to send out a notice to each member one month prior to the renewal date to give them time and awareness for the renewal to happen. Membership Type Analysis allows the system to show the current number of members and the amount of money the selected Member Type brings in per year based on current member counts signed up to that type. This is useful for finance purposes if demand can change the price of membership according to the number of members of each type.

# Functional Components

This section presets the functional components of the proposed software system.

There are 3 main componets in the proposed system

# User Requirements

This section describes the user requirements (functional components) as high-level abstract statements.

## GolfMemSYS will manage Members

* + 1. GolfMemSYS will add a Membership Type
    2. GolfMemSYS will add a member
    3. GolfMemSYS will update a member
    4. GolfMemSYS will remove a member
    5. GolfMemSYS will view all members

## GolfMemSYS will manage Renewals

* + 1. GolfMemSYS will issue renewal notices one month prior to renewal due date.
    2. GolfMemSYS will record payments

## GolfMemSYS will perform Admin

* + 1. GolfMemSYS will display a yearly revenue report
    2. GolfMemSYS will perform a membership type analysis

# System Requirements

GolfMemSYS consists of three main modules.

The Members module provides functions to add a Member, update Member’s details, view members in the system and add a new membership type for new members to sign up to.

The Renewals module provides functions to issue renewal notices and to record payments.

The Finances module provides functions to display a yearly revenue report and to show a membership type analysis.

## System Level Use Case Diagram

The following system level use case diagram illustrates the high-level system requirements.

Manager

Member

Treasurer

## Manage Members

This module processes all things related to golf club members such as adding new Members or editing information for existing members.

### **Add Membership Type**

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Membership Type** | |
| **Use Case Id** | 1 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function adds a Membership type to the GolfMemSYS system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** The Manager invokes the Add Membership Type function  **Step 3:** Manager enters required data:   * TypeID (char 2) * TypeName (char 50) * TypePrice   **Step 4:** Manager confirms that the Membership Type is to be added. | **Step 2:** Display UI  **Step 5:** The system validates the data entered:   * All fields must be entered * TypeID cannot already exist * TypeName must not be numeric * The TypePrice must be numeric and > zero   **Step 8:** The System saves the data in the Members files:   * TypeID * TypeName * TypePrice   **Step 8:** Show confirmation Message.  **Step 9:** Reset UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 5:** Invalid Information is found  **Step 6:** Show suitable error message  **Step 7:** Return to Step 3: |
| **Conclusions** | A new Member Type will be added to the system | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### **Add Member**

This function adds a new member to the system. Each member is identified by a unique MemberID. When a member is added, the member status is set to “Active” .

Manager

Member

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Member** | |
| **Use Case Id** | 2 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | Member | |
| **Description** | This function adds a Member to the GolfMemSYS system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** The Manager invokes the Add Member function  **Step 6:** Manager enters required data:   * MemberID * FirstName * LastName * Street * Town * County * Mobile * Email * MemberStatus * SignUpDate * TypeID * Paid     **Step 7:** Manager confirms that the Member is to be added. | **Step 2:** System assigns the next MemberID  **Step 3:** The system retrieves details of Membership Types from the Member Types file and loads on UI  **Step 4:** Retrieve counties from Counties file and load on UI  **Step 5:** Display UI  **Step 8:** The system validates the data entered:   * All fields must be entered * Member Name must not contain digits. * Email must be valid format   **Step 9:** System assigns SignUpDate to current system date.  **Step 10:** system assigns a default Member Status “Active” and default “Paid” status to n (no).    **Step 11:** The System saves the data in the Members files:   * MemberID * FirstName * LastName * Street * Town * County * Mobile * Email * MemberStatus * SignUpDate * TypeID * Paid   **Step 12:** Show confirmation Message.  **Step 13:** Reset UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | Step 8: Invalid Data entered  Step 9: Display an appropriate error message  Step 10: Return to step 6 |
| **Conclusions** | The member is added to the system | |
| **Post conditions** | This member can now pay/renew fees | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### View Members

**Description:**

This function lets the user view all members in the system, and all their information.

**Use Case Diagram:**

Manager

**Use Case Narrative:**

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **View Members** | |
| **Use Case Id** | 3 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function shows all members in the GolfMemSYS system. | |
| **Preconditions** | The system needs to load all current members | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** The Manager triggers the View Members function  **Step 4:** Manager can view all members in the system.  **Step 5:** Manager exits the UI | **Step 2:** Retrieves the summary details of current members from the members file and the Member Types file  **Step 3:** Display UI  **Step 6:** The system closes the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The manager can view members in the system and select one to see all details for. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Member

Description:

This function will allow the manager to update information for an existing user.  
You can select a member from the system and then edit any information that is related to them.

It is validated, then the new information is updated in the system.

**Use Case Diagram**

Member

Manager

**Use Case Narrative:**

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Member** | |
| **Use Case Id** | 4 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | Member | |
| **Description** | This function updates an existing Member’s details in the GolfMemSYS system. | |
| **Preconditions** | The system needs to load all current members. | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** The Manager triggers the Update Member function  **Step 4:** Manager can view all members in the system and can select one to update.  **Step 6:** The manager updates the details.  **Step 7:** The manager confirms the update. | **Step 2:** Retrieves the list of current members from the system.  **Step 3:** Display UI  **Step 5:** The system shows the current member information on the UI form.    **Step 8:** The system validates the information.  **Step 9:** The system saves the information to the Member file.   * MemberID * MemberType * FirstName * LastName * Street * Town * County * Mobile * Email * MemberStatus * SignUpDate * TypeID * Paid   **Step 10:** The system shows a confirmation message  **Step 11:** The system resets the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 8:** The System fails to validate the data  **Step 9:** Display appropriate error message  **Step 10:** Return to step 6 |
| **Conclusions** | The manager can select a member from the list and update their details. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

**Activity Diagram**

**Manager**

Retrieve active member list

Invokes Update

Member

Retrieve active member list

Manager picks member from list to view all the details for.

Display list of members from the Members File

System puts the details for selected member into boxes for editing

Manager updates the details and confirms the changes.

System validates the data entered

Valid data?

System saves information to the member file

NO

YES

System shows confirmation message and reset UI

## Manage Renewals

### Issue Notices

### Record Payments

## Analyse Finance

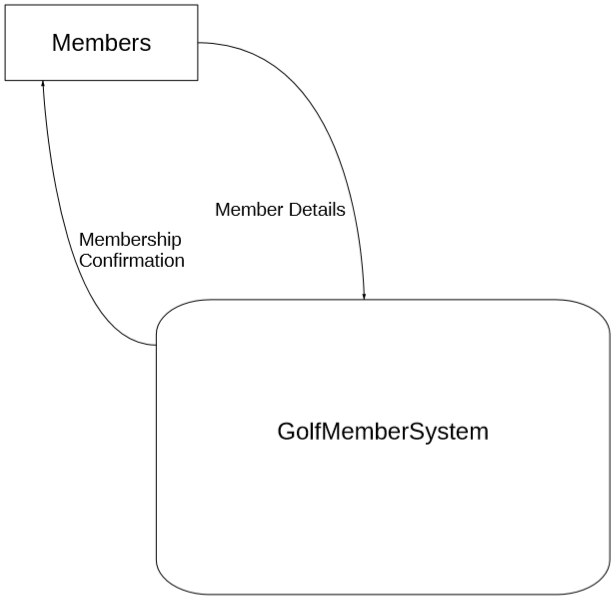
### Create Yearly Revenue Report

### Carry out Membership Type Analysis

# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD



## Level-1 DFD

A picture containing text, map

Description automatically generated

## Level-2 DFD (Process P1: Manage Members)

A picture containing screenshot

Description automatically generated

## Level-2 DFD (Process P2: Send Renewal Notice)

A screenshot of a cell phone

Description automatically generated

# Data Model (Class Diagram) This section outlines the data stored by the system and gives graphical representations of this data and how it is stored.

## Class Diagram

GOLFMEMBERSLOG

* MEMBERID
* FIRSTNAME
* LASTNAME
* STREET
* TOWN
* COUNTY
* MOBILE
* EMAIL
* MEMBERSTATUS
* SIGNUPDATE
* PAID
* TYPEID

COUNTY

* COUNTY

1

Must have

1

Must have

Renewals

* MEMBERID
* FIRSTNAME
* LASTNAME

Must have

1

MEMBERTYPE

* TYPEID
* TYPENAME
* TYPEPRICE

## Relational Schema

MEMBERTYPES (TYPEID, TYPENAME, TYPEPRICE)

COUNTYLIST (COUNTY)

GOLFMEMBERSLOG (MEMBERID, FIRSTNAME, LASTNAME, STREET, TOWN, COUNTY, MOBILE, EMAIL, MEMBERSTATUS, SIGNUPDATE, TYPEID, PAID)

RENEWALS (RENEWALID, MEMBERID, AMOUNTPAID)

## Database Schema

A definition of the database to be implemented.   
This includes primary key, foreign key and other constraints to be implemented.

**Schema:** GOLFMEMSYS schema

**Relation:** MEMBERTYPES  
Attributes:  
TYPEID NVARCHAR2(2) NOT NULL  
TYPENAME NVARCHAR2(50)  
TYPEPRICE NUMBER(5,2)  
**PRIMARY KEY:** TYPEID

**Relation:** COUNTYLIST  
Attributes:  
COUNTY NVARCHAR2(50) NOT NUL  
**Primary Key:** COUNTY  
  
**Relation:** GOLFMEMBERSLOG  
Attributes:  
MEMBERID NVARCHAR(3) NOT NULL  
FIRSTNAME NVARCHAR2(50)  
LASTNAME NVARCHAR2(150)  
STREET NVARCHAR2(30)  
TOWN NVARCHAR2(30)  
COUNTY NVARCHAR2(30)  
MOBILE NVARCHAR2(10)  
EMAIL NVARCHAR2(150)  
MEMBERSTATUS NVARCHAR2(10)  
SIGNUPDATE DATE  
TYPEID NVARCHAR2(2) NOT NULL  
PAID NVARCHAR2(1) NOT NULL  
**Primary Key:** MEMBERID  
**Foreign Key:** TYPEID  
  
**Relations:** RENEWALS  
Attribute:  
RENEWALID NUMBER(2) NOT NULL  
MEMBERID NUMBER(3) NOT NULL  
AMOUNTPAID NUMBER(5,2) NOT NULL  
**Primary Key:** RENEWALID  
**Foreign Key:** MEMBERID

# Conclusion

The system can carry out the functions described above.   
It is also able to automatically assign the sign up date based on the current time and date of the system.