## Project Specification

‘Limerick Gamers Game Library’ operates in Limerick City. The library hires out games to its members. The company wants to replace their current system in order to allow them operate with greater efficiency. The library has joined an exchange scheme to supplement the stock of purchased games. A major game stockist MGT in Dublin will, for a small fee, supply games on a regular monthly basis. This gives the opportunity for ‘Limerick Gamers Game Library’ to offer a wider choice than would otherwise be possible and to experiment with games that may turn out to be not very popular, but yet of interest.

Therefore the new software system is required to do the following: -

There is a collection of games initially acquired by the library. The library may have multiple copies of a game. New games from MGT must be added to the library catalog. From time to time damaged or out of date games must be deleted from the catalog.

Members pay an initial fee of €20. Games are only are only rented to members. At any one time a member may have only one game rented. Games are rented for 3 nights only. Rental fee is €3 .A late return fee of €1 is charged for each day overdue.

## Functional Requirements

Management are responsible for maintaining the library system. Management can

* Login
* View / add / delete / update staff members
* View / add / delete / update members
* Report on game catalogue
* Report on overdue games
* View / add / delete / update games
* Report on account transactions

Other staff members deal with the customers. They can

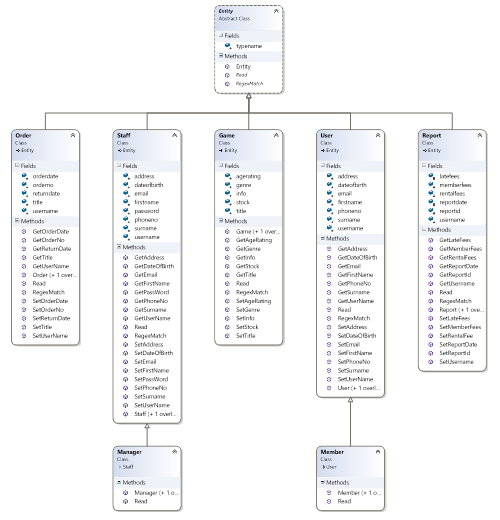
* Login
* View games
* Reserve games for members
* View / add / delete / update members
* Issue games
* Process returned games
* Enquire on availability of game

Additional information

1. Customer Account Transactions must show
   * Rental fees
   * Late return fees
   * Membership fees
2. Validation should be included on all forms

## Class Hierarchy

Class hierarchy of the entities that the system manipulates. All classes derive from Entity, allowing the system to treat all objects the same regardless of the class they belong to. The second tier is made up of the major divisions of entities: Orders, Reports, Games, Staff and Users. Managers derive from staff, and Members derive form Users.

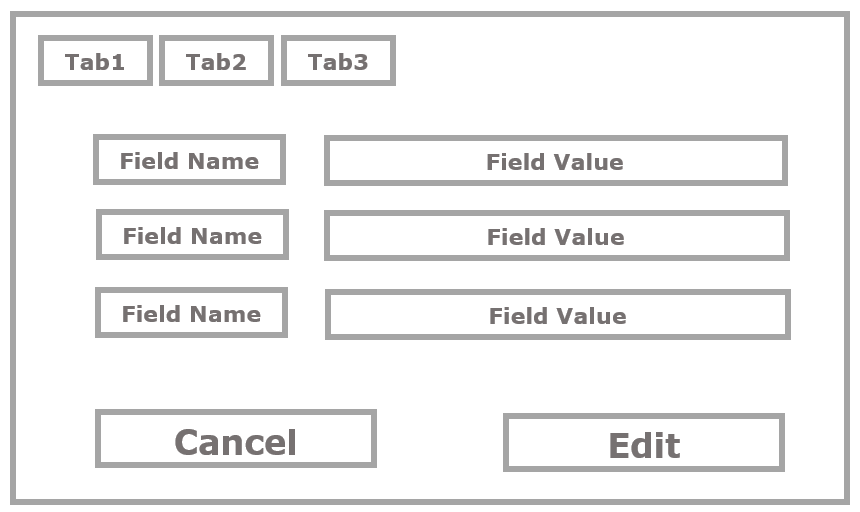


## Wire Frames

The user interface will be driven by a tab bar. The Tabs will be arranged horizontally across the top margin of the window. Each tab is associated with a different class of entity. Each tab will have its own list view which presents the user with a list of all objects of that class. Selecting an item from the list will result in a form being displayed, containing all the details of the selected object, along with buttons form manipulating that object.



*List Page*



*Form Page*

## StoryBoard

|  |  |
| --- | --- |
| 1. Select the Correct Tab | 1. Select an Item from the List |
| 1. Press the appropriate button | 1. Make your changes |
| 1. Commit the changes | 1. Then you’re back at the list view. |

## Prototype of User Interface

## Project Handbook Contents

1. ~~Project Specification~~
2. ~~Functional Requirements~~
3. ~~Class Hierarchy~~
4. ~~Storyboard / Menu Design~~
5. Prototype of User Interface – Main page, Add Member From, Report on Overdue Games
   * 1 per group member
   * This can be done in Word / Power Point / Visual Studio. Put a screen grab into a word document
6. Code Printout
7. Printout of Test Cases
8. Printout of Files before & after each function to show correct working. Note in order to see contents of files, they must be text files not binary files
9. Screen Printouts
10. Report on Project Management
    * Allocation of tasks within the group
    * Individual contribution to the project
    * Did the group adhere to deadlines

## Marking Scheme

1. 5% - Design: Class hierarchy
2. 5% - Usability: HCI / User Interface Design
3. 10% - Testing: Test plan and results
4. 10% - Individual contribution to the group & Project Management
5. 70% - Functionality