|  |  |
| --- | --- |
| A picture of a winding road and trees  Library Management system  Assessment 1 | Abstract  Planning phase of the CS106 Library management system.  Ian Teves, Jack Giddens, Jordan Jenkins  CS106 |

Contents

[Project Description (Pre-planning) 2](#_Toc116509957)

[Timeline & Constraints 2](#_Toc116509958)

[Team Members & Responsibilities 2](#_Toc116509959)

[Software Process Model Used 2](#_Toc116509960)

[Tools Used 2](#_Toc116509961)

[Software Requirements (Planning) 3](#_Toc116509962)

[Identifying Functional & Non-functional Requirements 3](#_Toc116509963)

[Functional Requirements 3](#_Toc116509964)

[Non-functional Requirements 3](#_Toc116509965)

[Getting user requirements (User Testing) 3](#_Toc116509966)

[Interviewing (Open + Closed) 3](#_Toc116509967)

[Ethnographic research 3](#_Toc116509968)

[User Stories + Scenarios 3](#_Toc116509969)

[User Requirements 3](#_Toc116509970)

[Classifying/Organising gathered requirements 3](#_Toc116509971)

[Prioritising/negotiating requirements 3](#_Toc116509972)

[Requirement Specification document 4](#_Toc116509973)

[Requirement Validation by tutor 4](#_Toc116509974)

[UI Design/Prototype 4](#_Toc116509975)

[Sketches 4](#_Toc116509976)

[Lo-fi Frames 4](#_Toc116509977)

[Hi-fi Frames 4](#_Toc116509978)

[Screen Layout 4](#_Toc116509979)

[Main and Secondary Windows (Dialogue Boxes and pop-ups) 4](#_Toc116509980)

[Functions of each window/screen 4](#_Toc116509981)

[Form elements in each screen/window 4](#_Toc116509982)

[User-testing 4](#_Toc116509983)

[Presentation 4](#_Toc116509984)

# Project Description (Pre-planning)

## Timeline & Constraints

Timeline info goes here…

This project has two major constraints:

1. Time – We have a large project to complete and submit within a short period.
2. Resources – We are to put it in few words, quite unfamiliar with the tools used for making the graphical interface.

## Team Members & Responsibilities

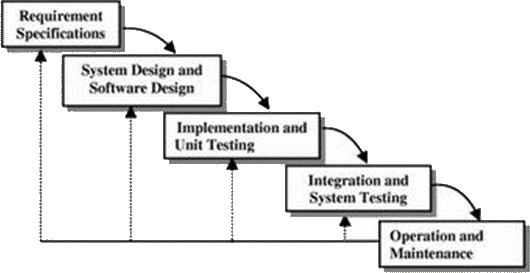
Team members:

* Ian Teves
* Jack Giddens
* Jordan Jenkins

|  |  |  |
| --- | --- | --- |
| Responsibilities | | |
| Jack | Ian | Jordan |
|  |  |  |

## Software Process Model Used

The team came together in a meeting and decided that use the iterative waterfall approach to design and develop the library management system.

* Iterative Waterfall - The iterative waterfall model is a workflow whereby we work on chunks of the project at a time, and we can’t move on to the next part without finishing the last. Though when we reach the end of the cycle, we can go back to a previous block if there is something to improve.

## Tools Used

To develop the library information system, we will need a few tools to get the job done. We will be dealing with a GUI, file managing, classes and objects and countless loops and algorithms for searching through data associated with each user.

* Visual Studio - Visual Studio is what will be used for writing backend code, such as dealing with files for storing user’s data working and working with objects.
* Qt Creator - Qt is the standard framework used in class for creating a GUI. It makes creating a GUI on a desktop or for mobile extremely straightforward and easy to edit if the design isn’t working out.
* Excel - As we’ll need to store user data, one of the best ways to do it is using a CSV (comma-separated values) file. Excel can read and display in its cell format, which will make for much easier debugging.
* Figma – To create lo-fi and hi-fi prototypes, Figma will be used because of its simplicity to design, also its ability to recreate our ideas to the finest details.

# Software Requirements (Planning)

## Identifying Functional & Non-functional Requirements

A functional requirement defines the function(s) that the app should perform.

In the library app, for example, a user should be able to search for a title, and then the system takes the data input, searches the database, and should output any result that may be relevant.

Non-functional requirements describe how the app will deal with the work.

If multiple users were to use the library app at the same time, it should be able to maintain fast loading times.

### Functional Requirements

* Requirements go here…

### Non-functional Requirements

* Requirements go here…

## Getting user requirements (User Testing)

Include questions asked…

### Interviewing (Open + Closed)

Results go here…

### Ethnographic research

Observations during testing go here…

### User Stories + Scenarios

Basically, the users give feedback if something didn’t work how they’d expect, they come up with a few sentences illustrate how they want it to work.

### User Requirements

* Bullet point them here…

## Classifying/Organising gathered requirements

## Prioritising/negotiating requirements

## Requirement Specification document

* Use case Diagrams with explanation in case of functional requirements
* Detailing functional and non-functional requirements and assumptions

[How to Write a Software Requirements Specification (SRS Document) | Perforce](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document)

## Requirement Validation by tutor

# UI Design/Prototype

## Sketches

## Lo-fi Frames

## Hi-fi Frames

## Screen Layout

## Main and Secondary Windows (Dialogue Boxes and pop-ups)

## Functions of each window/screen

## Form elements in each screen/window

## User-testing

# Presentation