Quality plan for To-do list

**File Name:** C:\Users\Kolkwitzia\Documents\SE\BIT707\_A1\_QualityPlan\_5086526.docx

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Description** | **Date** | **Author** |
| 1-0 | Initial Version | 5th April | Jody Gibson |

# Overview

This quality plan covers the above project and will be used throughout said project to ensure quality is kept to a high standard so that it will be a high-quality application once complete.

# 1. Quality Management issues for this project

**Issue:** Parts of the written code do not work as expected

**Explanation:** Written parts of the project code work but do not work in the intended way

**Management plan:**

* Check each part of the code is working before moving onto the next

**Issue:** Created designs are not implemented correctly/not adhered to

**Explanation:** Designs that are created as part of the planning process are not used to help complete the project

**Management plan:**

* Having the designs set up in a way that is obvious and easy to follow

**Issue:** Stated requirements are missed or partially implemented

**Explanation:** Functional requirements are missed and/or non-functional requirements are missed or not fully implemented

**Management plan:**

* Keep the list of requirements handy and check the requirements with each stage of the project

# 2. Planned Quality Management Activities

1. **Planning Stage (Assignment 1)**
   1. Documents are named with repeatable naming conventions to be able to connect them to the stated project.
   2. Documents are placed in a suitably named folder
   3. Folder is backed up to a separate place from where it was initially saved and placed.
   4. Planning documents are checked against each other to ensure that each document is correct and related to each document.
2. **Requirements Engineering Stage (Assignment 1)**
   1. Requirements written down are checked to make sure none are missed.
   2. Requirements are written into separate lines and are checked against the original written requirements to make sure that all requirements are included
   3. Created user-stories are double checked alongside the requirements to make sure they are all answered for.
   4. Programs created for the basic architecture are using easy to read naming conventions
   5. The created program is checked alongside the requirements to ensure that they are followed and implemented to a degree.
3. **Design Stage (Assignment 2)**
   1. Documents are named aptly and easy to recognise that they are part of this project.
   2. Folder is also named correctly and can be easily traced to this project
   3. Created diagrams are checked against the allocated principles to ensure that they follow them.
   4. Created ui interface designs are check against diagrams to ensure they match up with them.
4. **Construction Stage (Assignment 3)**
   1. Test plans are checked against previous documents to ensure everything is included and in place.
   2. Second iteration of application is checked to make sure that the first iteration is added and working.
   3. As sections of code work correctly, they are committed to git to be able to be rolled back to in case of the code breaking when something is added and/or changed.
5. **Transition Stage (Assignment 3)**
   1. All documents are checked to ensure that everything is included (which is why naming conventions help)
   2. Zip folder is created with a recognisable name
   3. Program files within zip folder are extracted and tested to make sure they work

# 3. Discussion: Quality Management in different projects

The program that is created hasn’t been tested within the user group and as such isn’t what they are looking for or they find it difficult to use. In this case, testing the application as it is being developed to check the user group is happy with it would be useful in managing this.

The requirements weren’t accurately sourced from the client and are therefore inaccurate and incomplete when written down meaning that the entire program would most likely be wrong or lacking specific things that the client thought they requested. To manage this the requirements should be double checked with the client to ensure they are accurate and what the client wants.

The developer wasn’t given the correct requirements or they weren’t communicated to them clearly from the analyst so the program is coded incorrectly and/or missing requirements due to miscommunication between the two. To manage this, the developer should double check with the analyst that requirements are as written and that they understand what the analyst means.