SOFTWARE DEVELOPMENT

PROJECT – ITERATION 1

1. **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 2/02/2019 | V1 | First Sketch | LG |
| 6/02/2019 | V2 | First Iteration | LG |
| 19/03/2019 | V3 | Revision of Document | LG |
| 20/03/2019 | V4 | Final check-up | LG |

1. **General Information**

|  |  |
| --- | --- |
| Project Summary | |
| Project Name | Project ID |
| Hangman Project | 256534 |
| Project Manager | Main Client |
| Loïc Galland (lg222sv) | Hangman enthusiasts can know the paper version of the game and want to try a computer version of the game. |
| Key Stakeholders | |
| * Developer * End-customer | |
| Executive Summary | |
| This project consists of creating the “Hangman” game. The basic idea for this game is that the player is going to guess a word by suggesting letter by letter. It will be implemented in a text-based fashion with Java code language. If a letter given is inside the predefined word, then the letter will be showed at the right position of the word. If the player guesses a wrong letter, the game is building a part of a man getting hanged. The player can have 7 letters wrong before the man is hanged and therefore losing the game. (ground, pole, head, body, left arm, right arm, left leg, right leg).  The game will be created to allow hangman enthusiasts that know the paper version of the game to discover a new version on computer. | |

1. **Vision**

This project consists of creating the “Hangman” game. The basic idea for this game is that the player is going to guess a word by suggesting letter by letter. It will be implemented in a text-based fashion with Java code language. If a letter given is inside the predefined word, then the letter will be showed at the right position of the word. If the player guesses a wrong letter, the game is building a part of a man getting hanged. The player can have 7 letters wrong before the man is hanged and therefore losing the game. (ground, pole, head, body, left arm, right arm, left leg, right leg).

The game will also let the user decide whether to write a letter or try a word. If they get wrong letter or wrong word it will be considered the same and they will therefore go to another part of the hanged man. The players will also be able to create an account by entering a username and a password. They will be able to log in and see the amount of times they won and other statistics that could be important for them.

By creating account, a leader board can be implemented to show the best players with their names, amount of wrong to win the game and in how much time they managed to finish the game.

**Reflection:**

I found that writing the vision was helpful to see what I wanted in the application. It made me think about all the possibilities and all the problem that could come with the program. It also made me think more in details to the different functionalities I wanted to implement for the application. It also helps to see how much time, will be needed to create the all program. And allow us to create a more realist view of what our program could look like.

1. **Project Plan**

The player needs to be able to:

* Draw hangman after every wrong try
  + At the 7th wrong letter = Game Over
* Enter a Letter and see if and where it is on the word
* Choose whether write a letter or whole word.
* When guessing whole word = included in count of wrong words
* Single and Multi-player modes
* The app should have a nice graphical representation of the game that updates when the user is given wrong letter or words

Reflection:

The Project Plan help me to focus more into what could be done and make a more realistic plan. It also helped mapped out my ideas to know exactly what needed to be done and realised what couldn’t be done in the given timeframe.

**Introduction**

For this project, the “Hangman” game will be created. It will be implemented in a text-based fashion with Java code language. The basic idea for this game is that the player is going to guess a word by suggesting letter by letter. If a letter given is inside the predefined word, then the letter will be showed at the right position of the word. If the player guesses a wrong letter, the game is building a part of a man getting hang. The layer can have 8 letters wrong before the man is hanged and therefore losing the game. (ground, pole, head, body, left arm, right arm, left leg, right leg).

* 1. **Justification**

This application should be made because it allows people that are learning programming to show their skills and use them in a bigger project that they have been before. It is also a very famous game where almost everybody knows how to play it. This game will therefore be targeted for people that I have previously played this game on a paper version.

* 1. **Stakeholders**

The different stakeholders are:

* The Developer will want to create a well-structured code to make it more maintainable in the future.
* The potential player will want to able to play the game without any issue. The player will also want the game to be easy to understand and to have a minimal graphical Interface with images and text.
  1. **Resources**

The resources available for this project are:

|  |  |
| --- | --- |
| **Time Used** | **Resource** |
| 5h | Book: Software Engineering by Ian Sommerville |
| 5h | The Internet |

* 1. **Hard- and Software Requirements**

The developer will use the JDK IntelliJ IDEA ULTIMATE 2018.3.5 to code the program in the programming language Java 11. To code, the developer will use a laptop to create this application.

The player will use the JRE of IntelliJ IDEA ULTIMATE 2018.3.5 to run the application. To play, the player will use a portable or desktop computer.

* 1. **Overall Project Schedule**

This is the Overall Project Schedule for Assignment 1.

|  |  |
| --- | --- |
| **Tasks** | **Deadline** |
| Write Vision | Week 6 |
| Write Project Plan | Week 6 |
| Create Skeleton code | Week 6 |
| Write a risk analysis | Week 6 |
| Create time log | Week 6 |

* 1. **Scope, Constraints and Assumptions**

Scope:

The developer is supposed to implement:

* A Graphical Interface
* A place to write the guesses
* The wrong guesses should be showed on the screen
* A hint on how many letters there is on the word should be showed on the screen.
* A graphical representation of the Hangman. The graphical representation should change every time that the player guesses a wrong letter.
* Keep track of the wrong guesses. 7 wrong guesses allowed.
* Text that appears when the player has won or lost.
* A menu where the player can choose to go play or to exit the game.

Out of Scope:

All these things below will not be implemented due to lack of time:

* Multiplayer Mode.
* “.exe” version of the game.

Constraints:

* The Game runs into an IDE.
* Only 1 player can play.

Assumptions:

The player needs to do all the things below to be able to play the game:

* Download all the files for the game from GitHub.
* Put all these files into an IDE.
* Run the class called HangManMain.
* The player can now play.

1. **Iterations** 
   1. **Iteration 1**

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimation Time** | **Due Date** |
| Read about Project Planning | 2h | Week 6 |
| Write Vision | 20min | Week 6 |
| Write Introduction for Project Plan (PP) | 10 min | Week 6 |
| Write Justification for PP | 15 min | Week 6 |
| Write Stakeholders for PP | 20 min | Week 6 |
| Write Resources for PP | 5 min | Week 6 |
| Write Hardware and Software for PP | 20 min | Week 6 |
| Write Overall Project Schedule for PP | 30 min | Week 6 |
| Write Scope, constraints and assumptions for PP | 1h | Week 6 |
| Write Iteration 1 Table | 20 min | Week 6 |
| Read about Risk Analysis | 3h | Week 6 |
| Write Lists of risks | 20 min | Week 6 |
| Write Strategies for risks | 20 min | Week 6 |
| Create time log | 5 min | Week 6 |
| Create skeleton code | 5 h | Week 6 |

* 1. **Iteration 2**

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimation Time** | **Due Date** |
| Create Class Diagram | 15 min | Week 8 |
| Write Fully Dressed Use Case | 30 min | Week 8 |
| Create Use Case Diagram | 15 min | Week 8 |
| Create State Machine | 30 min | Week 8 |
| Implement Code | 1 h | Week 8 |

* 1. **Iteration 3**

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimation Time** | **Due Date** |
| Create Manual Test Case | 1h | Week 10 |
| Write JUnit test | 1h30min | Week 10 |
| Running test | 40 min | Week 10 |
| Checking the code | 30 min | Week 10 |
| Write Report of Testing | 30 min | Week 10 |

* 1. **Final Iteration**

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimation Time** | **Due Date** |
| Update Planning Document | 120 | Week 12 |
| Write Reflections | 30 | Week 12 |
| Implement Last Piece of Code | 60 | Week 12 |
| Create Tests for the Project | 30 | Week 12 |
| Run Tests | 20 | Week 12 |
| Write Final Report | 60 | Week 12 |

For the final Iteration, I have decided to implement an additional functionality to the game. Now the user can write the full word directly. To test this implementation, I will use Manual Testing because it is easier to test.

I have also updated the Planning Document to what It was supposed to look like after I have got the feedback. I have also written reflection to explain how I was feeling about the different parts of the “Hangman” project.

1. **Risk Analysis** 
   1. **List of risks**

This is the list of all the risks for the “Hangman” project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability** | **Impact** | **Description** |
| 1) Hard disk crash | Low | Catastrophic | The hard disk could crash, the developer will lose all the software created |
| 2) Size underestimation | Moderate | Serious | The project is bigger than expected and will therefore take more time and money to complete |
| 3) time underestimation | Moderate | Serious | The time needed for this project is more than the estimated one. |
| 4) Staff Sickness | Low | Tolerable | The developer could be sick and therefore the project will take more time to complete. |

* 1. **Strategies**

Prepare for the risks by having strategies for avoiding the risks as well as minimising the impact of them if they do occur.

|  |  |
| --- | --- |
| **Risk** | **Strategy** |
| 1) Hard disk crash | *Minimization strategy* – The developers should save the project online every time an important functionality is implemented. |
| 2) Size underestimation | *Avoidance strategy* – Investigate all the requirements of the project to check how the project can be done. |
| 3) Time underestimation | *Avoidance strategy* – Investigate all the requirements needed for this project in detail to have the most accurate time estimation. |
| 4) Staff sickness | *Contingency plan* – Add some extra days where nothing is planned in case of a staff sickness to still implement the project in time. |

**Reflection:**

This part of the assignment was first hard to understand and to know exactly the type of risks part of this project. I can now see why this part is very important for the developer. It will help me a lot when I will be implementing the different codes. It was also hard for me to know what impact the different risks can have on the project as I am lacking experience in this field.

1. **Time log**

**Assignment 1 – Time Log**

|  |  |  |
| --- | --- | --- |
| **Task to Do** | **Time Estimated (min)** | **Time Taken (min)** |
| Read about Project Planning | 120 | 130 |
| Write Vision | 20 | 20 |
| Write Introduction for Project Plan (PP) | 15 | 20 |
| Write Justification for PP | 15 | 10 |
| Write Stakeholders for PP | 20 | 15 |
| Write Resources for PP | 15 | 5 |
| Write Hardware and Software for PP | 20 | 5 |
| Write Overall Project Schedule for PP | 30 | 50 |
| Write Scope, constraints and assumptions for PP | 60 | 30 |
| Write Iteration 1 Table | 20 | 30 |
| Read about Risk Analysis | 180 | 160 |
| Write Lists of risks | 20 | 30 |
| Write Strategies for risks | 20 | 20 |
| Create time log | 15 | 20 |
| Create skeleton code | 300 | 600 |

The skeleton code took way more time than I expected because I bumped into some difficulties and bugs when I started implementing the code. I also had to learn few things to implement some part of the code.

**Assignment 2 – Time Log**

|  |  |  |
| --- | --- | --- |
| **Task to Do** | **Time Estimated (min)** | **Time Taken (min)** |
| Create Class Diagram | 15 | 20 |
| Write Fully Dressed Use Case | 30 | 40 |
| Create Use Case Diagram | 15 | 15 |
| Create State Machine | 30 | 60 |
| Implement Code | 60 | 90 |

**Assignment 3 – Time Log**

|  |  |  |
| --- | --- | --- |
| **Task to Do** | **Time Estimated (min)** | **Time Taken (min)** |
| Write Manual Test Case | 60 | 90 |
| Create JUnit test | 90 | 120 |
| Running test | 40 | 20 |
| Checking the code | 30 | 25 |
| Write report of testing | 30 | 60 |

**Assignment 4 – Time Log**

|  |  |  |
| --- | --- | --- |
| **Task to Do** | **Time Estimated (min)** | **Time Taken (min)** |
| Update Planning Document | 120 | 240 |
| Write Reflections | 30 | 40 |
| Implement Last Piece of Code | 60 | 30 |
| Create Tests for the Project | 30 | 30 |
| Run Tests | 20 | 15 |
| Write Final Report | 60 | 60 |