HANGMAN PROJECT

Osama Zarraa
Linneuniversitet

Date
2019-04-18

Contents

1 | Revision History

Date	Version	Description	Author
2018-02-08	1.0.0	Planning, writing the skeleton of the program	Osama Zarraa
2018-02-21	1.0.1	Design and modelling	Osama Zarraa
2018-03-24	1.0.2	Testing	Osama Zarraa
2018-03-24	1.0.3	Final	Osama Zarraa

2 | General Information

Project Summary	
Project Name	Project ID
Hangman game	1DV_10001_HM
Project Manager	Main Client
Osama Zarraa	+13 age user
Key Stakeholders	
End users, developers, testers	
Executive Summary	
Hangman game is a word-guessing game with limited number of wrong guesses with extra features like user registration and statistics with points system to players.	

3 | Vision

Hangman project is a text-based and a word-guessing game which consists of a word, players and an image that is drawn based on player's answer that is given character by character. Game starts with loading old files that represents players, scores and old games. After loading, a menu will appear telling the user to choose an option from:

- Registering a player
- Selecting an old player
- Start a game
- Showing players scores
- Exit

After starting a game, a "store" will choose a random word from a list which is loaded previously. Program will print an underscore characters represents the missed word, without showing the real characters, and players can guess the word by giving a character. Each time a player gives a character, program will check if this character is part of the missed word. If character is included in the word the program will replace underscores that matches the given character with the right characters. Otherwise a figure of a hangman will be completed by every wrong answer given by the players. The game has maximum eight or ten wrong answers to complete the hangman figure. If the figure of the hangman is completed the game will be ended and saved in a store for later statistics, else the player who gave the last successful character will be considered as a winner and a log will be saved as part of a point system. The program will also have a time limit which represents that the deadline of the game.

Reflection:

To find out how to write a vision for the product was not that hard because after I read the teacher feedback for about how to write. Online material also helped me on how the vision should be written, and what is the purpose of a vision document in every project plan. In general, I think the vision should be in a separated document not inside the project plan document it does make more sense. Have the vision inside the project plan made some confusion to me since I think it does not have much to do with the project plan.

4 | Project Plan

4.1 Introduction

Hangman project is a text-based and a word-guessing game which consists of a word, players and an image that is drawn based on player's answer that is given character by character. This document presents the four iterations which the development of this project goes through.

4.2 Justification

This reason why this game is made is that there is no hangman games in the market that have the same functionality that this game provides. Such as: sitting a timer for each player when a they start the game. Also, each player has a history in the system record their scores and name.

4.3 Stakeholders

Stakeholder	Description
Software developer	Responsible to develop the requirement and all the features of the game
Software tester	Responsible to test the features of the game
End-user	Gamers, at age 13+ who have interest puzzle games.

4.4 Resources

Resource	Description
Man power	Only one, who is responsible for development and testing process.
Tools	IntelliJ 2019.1: as IDE Java 11
Available time	4 hours per day during this period: From 2019.01.21 to 2019.03.24 Holidays are excluded.

4.5 Hard- and Software Requirements

Hardware: Computer with operating system such as Windows, macOS or Linux.
Software: IDE(Intellij), JDK (Java development kit), JRE (Java runtime environment)

4.6 Overall Project Schedule

8th February 2019: Project plan.

21th February 2019: Working version of the game with UML.

8th March 2019: Testing.

4.7 Scope, Constraints and Assumptions

Hangman game is a console application. It is a puzzle game, based on word-guessing game where a player starts the game and start guessing a word by providing characters. A new part of the hangman image will be drawn every time the player fails to guess the right character.

Game starts with loading the words and players files. The words file has the words that a player guesses from. Also, the players files contains the players information of (id, name, scores). The features that the game provides are:

- Start game
- Play game, with time limit game
- List words
- Remove a word
- Add word
- List players (with their scores)
- Register player
- Remove a plyer
- Log in player
- Quit

When the player starts the game, the system saves the current time, the players start entering characters, for each wrong character a part of the hangman image shows up and the system will check the time if it exceeded 3 minutes, if the time is exceeded or the player enters 9 wrong characters then the player loses. If the player guesses the word correctly in less 3 minutes and less than 9 wrong guesses then the player wins the game.

4.8 Reflection

Writing a project plan was a bit hard because it is my first time in this course to write such a document like this. I Had to go through many online templates of project plans so I know what is purpose of it. Was confused between the different templates online and the template that was given to us in this course. However, it was a good experience for me when I rewrite it more than once to fulfill all the requirements.

5 | Iterations

5.1 Iteration 1

Task	Description	Estimated Time(hours)	Deadline
Read documentation	Reading the documentation about how to write the project plan	16	8th February 2019
Prepare working tools	Prepare the tools required for the project	8	8th February 2019
Create the project basic files and classes	Create the first version of the project	2	8th February 2019
Write the project plan	Write the project plan file including the iterations and time log and risks	8	8th February 2019

5.2 Iteration 2

Task	Description	Estimated Time(hours)	Due Time
Read documentation	Reading the documentation about the class diagram and state machine diagram besides reading about use cases and fully dressed use cases	16	21th February 2019
Create Use Case Diagram	Sketching the use case diagram	8	21th February 2019
Write Fully dressed use case	Write a fully dressed use case for the second use case	8	21th February 2019
Create State Machine diagrams	Sketch the state machine diagram to the second use case	16	21th February 2019

	with and without time limit feature		

5.3 Iteration 3

Task	Description	Estimated Time(hours)	Deadline
Read documentation	Reading the documentation about how to make manual and automated testing	16	8th March 2019
Write manual tests	Create manual tests for use cases 1,2,3	8	8th March 2019
Write Automated tests	Create automated tests for the Word class and for the Game class	8	
Write report	Write the final test report with images about the automated tests	8	8th March 2019

5.4 Iteration 4

Task	Description	Estimated Time(hours)	Deadline
Write the plan	Write the plan for the whole project	8	18th April 2019
Model the non-completed features	Create the state machine diagram and use case diagram for the new features	8	18th April 2019
Implement features	Complete the features that are not implemented in the previous iterations	8	18th April 2019
Test the new added features	Testing the new features including the players class	8	18th April 2019

6 | Risk Analysis

In this project the main three risks are: sickness, hardware failure, software crashing and learning difficulties.

6.1 List of risks

Risk	Impact	Probability	Strategies
Sickness	Not submitting the project at before the deadline	10%	Have a strict plan to finish the deliverables before few days before the deadlines
Learning difficulties	Hinder the development process to progress.	50%	Try to use slack to ask. Study in studies groups where I can share experience and ask and learn more.
Software crash		10%	Save files multiple times
Hardware damage	Losing the project files.	10%	Backup all the files that are related to

6.2 Reflection:

Being prepared for the risks is something mandatorily in SDLC (Software development life cycle). It is a way to keep the developer within the timeframe of the project and give him a chance to overcome them without affecting the whole project. Moreover, it gives the developer a future perspective of what would happen and how to make a backup plan for issues that will come.

6.2 Strategies

For sickness issue, I will try to get help from my colleges and I will consider adding more time from the beginning to solve this issue.

For hardware failure issue, I will use version control system such as GitHub for preventing losing the project files.

For the Software crash, I will make copies of the files that I work on so I can go back to them if needed.

7 | Time log

Iteration	Task	Actual time(hours)
Iteration 1		
	Reading documentation	15
	Prepare the tools	8
	Create first version of the project	6
	Write a project plan	17
Iteration 2		
	Read documentation	16
	Create use case diagram	8
	Write fully dressed use case	8
	Create State Machine diagrams	16
Iteration 3		
	Read documentation	12
	Write manual tests	8
	Write automated tests	7
Iteration 4		
	Write the project plan	8
	Model the non-completed features	7
	Implement features	9
	Test the new added features	8