SOFTWARE DEVELOPMENT PROJECT - HANGMAN

Mattias Ruljeff
Rulles Hack

Project name	Hangman
Revision	4
Project ID	1
Project manager	Mattias Ruljeff
Main Client	High School‘s with programming orientation
Key stakeholders	Developer
Executive Summary	For teachers teaching programming and students who wants to learn programming in an easy and fun way

Content

1. Revision History

Date	Revision	Description	Author
2020-02-03	1	Added this document.	MR
2020-02-13	2	Added User Cases, iteration 2 and Check-state diagram.	MR
2020-02-17	3	Changed the vision. Added reflections on the vision, risks and the project plan.	MR
2020-02-21	4	Changed the Check-state diagram, planned time for hangman-game changed to 10 hours. Rewritten UC case 1-2-3. Added class diagram.	MR

Full name of Author	
Initials	**Author name**
MR	Mattias Ruljeff

2. Vision

The game will be targeted to High Scholl’s and their students, for understanding programming structures and how they themselves can develop games in an IDE.

This game shall get students interested in programming, and games is often a good way to get students engaged and exited.

This will be a good template for teachers for teaching. The teachers can inspire the students to make their own custom features in the game and how to implement them.

In the Hangman-game, a player should guess a given word, letter by letter. It shall be a game that the player think is worth spending time with, being easy to get started playing with via the menus, that shall me minimal in choices that the player can make. The scoreboard in the end of the game will give the player motivation to play the game repeatedly.

Reflections on the Vision

The vision is hard to come up with when having a project that contains such a small game.

To motivate the team that develops the game, and to have a good vision is necessary, also as a selling point to the customer!

The vision should reflect the goal of the development, the things shat will make the game special, to make the development team work in the same direction for the same goal! That is what I wanted to accomplish with my vision.

The task also included a special feature. I chose to give the game a scoreboard, so that the player wants to come back and beat the best player. That is my way of giving the player a “goal” to the game, be the best player.

3. Project plan

Introduction

- A game of hangman, played by a player in the computers console window.
- The player shall be able to insert a player name.
- The animation-steps of the hangman-game are:
    1. The ground.
    2. The vertical pole.
    3. The horizontal pole.
    4. The supporting pole for the vertical pole.
    5. The rope.
    6. The head and torso.
    7. The arms.
    8. The legs.
- The player will be given eight chances to guess the letters in the word.
- When the player loose, a scoreboard will be presented and then the game restarts.

Justification

The Hangman game will be a good teachers-tool to learn students how to program, use the console window, use Gitlab to save project-files and download project-files.

Stakeholders

The development team, high Scholl's and students.

Resources

Book: Sommerville software engineering.
Lectures: Videos in 1DV600 course-page.
Software: An IDE of choice (preferable Visual Studio Code), Gitlab and Gitbash.

Hard- and Software Requirements

Hardware: A computer.
Software: An IDE, GitLab and Gitbash.

Overall Project Schedule

Week 6 (Complete assignment 1)
Week 9 (Complete assignment 2)
Week 11 (Complete assignment 3)
Week 12 (Submit complete project)

Scope, Constraints and Assumptions

Scope:

- The game will initially not be connected to any database.
- The game will be playable from the start. The teacher or the students will be able to modify and add functionality to the game.

Constraints:

- The game will only be playable in the terminal window, and will not be a stand-alone app. The player must have all the files containing the game and an IDE of choise.

Assumptions:

- The project will assume that the player knows how to use Visual Studio Code (or any similar IDE) to start the game via the console window. The player knows how to install Node.js latest version and the packages that is included in the game.

Reflections on the project plan

The project plan was easier to write than the Vision because the project plan is more how the game is meant to be structured, so that the developers of the game and the stakeholders in the project plan knows how the final product shall work. It was harder to find good ways of writing the Scope, Constraints and the Assumptions.

4. Iterations

Included parts:

- Read Software engineering.
- Create Hangman-game.
- Create documentation.
- Plan iterations.

Iteration 1

- Create documentation.
- Start to build the Hangman-game. Add all necessary files for the project.
- Read chapter 2, 3, 22 and 23.
- Watch video-lecture 2 and 3.

Iteration 2

- User case diagrams.
- User cases.
- Class diagram.
- Check-state diagram.
- Introduce more functionality to the game.
- Read chapter 4, 5, 6, 7, 15 and 20.
- Watch lecture 4, 5, 6, 7 and 8.

Iteration 3

- Game-testing.
- Read chapter 8.
- Watch lecture 9, 10 and 10.5.

Iteration 4

5. Risk Analysis

List of risks

Risk	Affect	Description	Probability	Effects
Hardware failure	Project	If the computer/hard-drive crashes	Low	Catastrophic
Sickness	Project and product	If I get sick and can't go to school	Moderate	Serious
Size underestimate	Project and product	If the added features are too complicated/hard to integrate	Moderate	Serious
Requirement change	Project and product	If the product specifications change	Moderate	Tolerable

Strategies

Strategies to avoid and minimize the impact of the risks:

a. Hardware/software – Save all the files to Gitlab. Backup-computer available.
b. Sickness – Work from home in case of sickness
c. Requirement change – Build modules that can easily be modified
d. Size underestimate – Skip extra functions

Reflections on the risk analysis

The risks in this project are not many. This project is a small project and the hardware and software is not extensive. The changing of requirements is a risk that is more likely to happen, but that may be because the development team comes up with additional functions to make the game more playable and interesting.
Risk in a project is the hardest thing to plan. It is always har to know how to avoid risks, but if a project has risk management plans it is less likely that the risk will make much damage to the project and take too long time. It is also easier to re-plan if something happens and the project have a plan-B.

6. Time Log

Task	Planned time (hours)	Actual time (hours)	Starting date	Latest update	End date
Project plan	10	9 so far	2020-02-03 11:00	2020-02-17 14:30	-
Hangman-game	10	4 so far	2020-01-31 15.30	2020-02-21 14:00	-

7. User Cases

User-cases diagrams in separate files:	/Project-plan/Diagrams/UC/Use case revs
State-diagrams in separate files:	/Project-plan/Diagrams/State/Check-state revs
Class-diagrams in separate files:	/Project-plan/Diagrams/State/Class-diagram revs

UC 1 Start Game	
Preconditions:	None
Post conditions	The game menu is shown
Main Scenario	
1. Starts when the Player starts a session of Hangman-game. 2. The Systems presents the main menu, presenting a welcome message, the options to play or to quit the game. 3. The Player starts the game.	
Alternative scenarios	
2.1 The Player quits the game. 2.2 The System prompts if the Player wants to quit the game. (See UC 3)	

UC 2 Play the game	
Preconditions:	Player has chosen to play the game
Post conditions	The UC 1 or UC 3 is triggered
Main Scenario	
1. The Player enters a name. 2. The System prompt the player to enter a letter. 3. The Player enters a letter. 4. Repeat step 2-3 until the Player has guessed the right word. 5. The System prompts if the player wants to play again. (return to UC 1)	
Alternative scenarios	
1.1 The player doesn't enter a name, the system asks the player to enter a name. 2.1 The system detects that the letter is wrong, creates parts of the hanged man and presents it to the player. 2.2 The player guessed wrong 8 times, loses the game. (See UC 3)	

Continues on the next page.

UC 3 Quit the game	
Preconditions:	The game session is running
Post conditions	The game session is terminated
Main Scenario	
1. Starts when the Player wants to quit the game, win or lose. 2. The System prompts for confirmation. 3. The Player confirms. 4. The System terminates.	
Alternative scenarios	
3.1 The Player does not confirm. 3.2 The System returns to UC 1 start game	

8. Handing in