Project plan (Cheat engine for games)

Stefan-Nikola

2024

Version history				
Version	Date	Author	Changes	State
1.0	13/03/2024	Stefan-Nikola Stanev	Started Project plan	Finished

Contents

Chapter 1

Project assignment

1.1 Context

Our company "MemoryExploit" offers wide range of software applications, focusing on engines that manipulate the memory of a computer, thus allowing it to make changes in real time.

1.2 Goal of the project

Why?

- We are doing this project because we want to provide the players the options to cheat in their favorite single player titles, so that they can try out what it feels like to have cheats and then realize they should not cheat since it disrupts the balance of the game.

Prefered situation!

- We want to create a environment where players realize the issue of cheating and make them not use it in multiplayer games because it will ruin everyone's enjoyment.

Advantages!

- This software will allow users to experience what it's like to play with cheat. By making the cheat engine work for only single player titles, we ensure that the integrity of multiplayer games is not ruined.

Value to the company!

- The project aligns with our strategic goals of diversifying our offerings, expanding our user base, and boosting profitability.

ICT Product possibilities!

- The cheating engine will offer features such as scanning for running processes, tooling for memory scanning and editing in memory with a possibility of real time notifications that will allow the users to see the made changes immediately.

1.3 Scope and preconditions

Inside scope:

- development of memory manipulation algorithms and features.

- Implementation of scanning processes and memory scanning

Outside scope:

- Features unrelated to scanning processes and memory scans.

- Any specific technology choices already made by the company.

1.4 Strategy

We have chosen an Agile approach, more specifically Scrum, because of it's adaptability to changing requirements and elements on the go and its ability to deliver value to the user.

1.5 Research questions

How do you build cheating engine for games?

* What programming language should be used for the creation of the cheating engine? DOT Research methods to be applied: Library Research, Field Research, Showroom Research.

* What OS should be used as the base of target? Should it be Windows or Linux? DOT Research methods to be applied: Library Research, Field Research, Workshop Research.

* What are the key functionalities and components of a cheating engine? DOT Research methods to be applied: Library Research, Showroom Research, Field Research.

* How is a cheating engine detected so it can be build to avoid detection? Examples of anti cheats: VAC (CS2), VANGUARD (Valorant), Easy Anti-Cheat (Fortnite). DOT Research methods to be applied: Library Research, Field Research, Showroom Research

1.6 End products

Application:

- UI design
- Memory manipulation scanner
- Process scanner

Documentation

- Project plan
- User stories
- Other documents in the future

Chapter 2

Project Organisation

2.1 Stakeholders

- Stanev, Stefan-Nikola, Project lead, Developer, Scrum Master
- Heijman, David, Project mentor, Reviewer, 2 times a week in his lecture

2.2 Communication

* **Weekly lecture meetings with teacher**

- Goal: Discuss and showcase project progress, ask questions, receive guidance and feedback I can apply before next sprint review.
- Location: (R10 Fontys building)
- Timing: Twice a week with teacher
- Attendees: Stanev, Stefan-Nikola; Heijman, David

* **Questions via Email/Teams or Canvas**

- Goal: Address specific questions and seek clarification
- Location: Email, Teams or Canvas
- Timing: Whenever needed

- Attendees: Stanev, Stefan-Nikola, Heijman, David

* Spring Review Meetings (Every 3 weeks)

- Goal: Review completed work from previous sprint, receive feedback and plan for next sprint
- Location: In person (R10 Fontys)
- Timing: Every 3 weeks (After the end of each sprint)
- Attendees: Stanev, Stefan-Nikola; Heijman, David

Chapter 3

Activities and time plan

3.1 Phases of the project

* **Sprint planning and set up (Sprint 1)**

- Project Plan

- User Stories

- Scrum Board

- Interview Report

* **Sprint deliverables of second phase (Sprint 2)**

- Network drawing

- flowchart

- Sprint planning

- Technical design document

- Attack scenarios

* **Sprint deliverables of third phase (Sprint 3)**

- Implementation document

- Code

- test results
- feedback and validation

* **Final deliverables (Final sprint 4)**

- Research document
- Advisory Report
- Powerpoint presentation

3.2 Time plan and milestones

Time plan and milestones			
Phasing	Effort	Start date	Finish date
Sprint 1 Startup	Project Plan User stories Scrum Board Interview Report	08/03/2024	29/03/2024
Sprint 2	Network drawing flowchart sprint planning Technical design document attack scenarios	30/03/2024	19/04/2024
Sprint 3	Implementation document Code test results feedback and validation	20/04/2024	17/05/2024
Sprint 4 (Final sprint)	Research document Advisory Report Powerpoint presentation	18/05/2024	07/06/2024

Chapter 4

Testing strategy and configuration management

4.1 Testing strategy

The testing strategy we'll be using for this project is going to be mainly manual testing because it will do a lot of memory manipulation. There could be other tests being done in the future.

4.2 Test environment and required resources

* **Test environment:**

- Manual testing will be done to ensure real life simulation of the application.

* **Required resources:**

- A Linux or Windows machine that could be used for testing.

4.3 Configuration management

* **Tooling**

- Github will be used as primary management tool for the project.

- Git will be used as the underlying version control system

* **Branching strategy**

- Development will be used as the main branch when developing.

- Main/Master will be used to deploy final version of each sprint.

- Additional branches can be created depending on the requirements and the needs of the developer.

* **Change Requests and Problem Reports:**

- Change requests and problem reports will be managed using Github's issue tracking system.

Chapter 5

Finances and Risk

5.1 Risk and mitigation

- **Risks:**

Risk **Client not available**

- Schedule meetings to avoid missing each other during lecture days.

Risk **Project not matching final requirements**

- Making sure the feedback is received and applied accordingly.

Risk **Software failure**

- Make sure to push at least once a day when changes are made.

Risk **Not enough time**

- Make sure I allocate my time accordingly even if software is not fully finished before deadline, all deliverables should be delivered.