Project Texting Spree

The following document is meant to help the reader understand the premiss of this project better as well as define the main focus of the project as well as its core systems. There are also systems going to be defined that are nice to have but not crucial for the success of the project as a whole as well as the good handling and feeling of the software.

## **Starting Point**

In this day and age, it is only too common for bad stories to be written. Not only do mainstream movies portrait such garbage but also the game industry is hit by this crisis. The stories written are often boring and there is no engagement for the reader to actually want to know the characters portrayed in these stories. Speaking of characters, the stories I am talking about often have no compelling plot and the development of character stagnates to the point where we get only one-dimensional characters, that do not resemble a normal human in the slightest.

Another point is, that many people today do simply not enjoy reading through a story and think to themselves “If I was in his/her place, I would have done it like this!”. And this is where this project steps in. The project codenamed “Texting Spree” is a playable story in the format of a novel. It aims to tell an interesting and compelling story and motivate players to read the story as well as invest themselves into the world, characters and lore and make decisions that shape the rest of the story plot in a way the player actually wants.

So, depending on the players choices, other options are opened to the player and the story progresses differently for every decision made. Other than that, there are other systems going to be implemented, such as and inventory where the player can pick up items that will hinder or help him progress the story.

## **People involved**

In the following chapter we are going to have a look at all the people who are going to be involved in making this project a reality. First of all, the following people are going to be named “Three Trolls Games” as a pseudo-company that will form somewhen in the future. The table below lists all the people involved in the project as well as their titles within the future company.

|  |  |  |
| --- | --- | --- |
| **Person** | **Title** | **Description/Work area** |
| Bärfuss Frederic | Lead Story Designer | The lead story designer is responsible for the story plot that will be used in this project. His primary focus is to make sure that the plot has no inconsistencies and plot holes. |
| Dulla Kastriot | Game Director | The game director is the project leader. He has the responsibility for the entire project, its success or downfall. His work area is to make sure that every other participant of the project has what he needs to fulfil their role as good as possible and also, he will help wherever he can.  The game director is also the primary decision maker in the project and other lead roles have to converse with him first before changing any specifications of the project. So, it’s the game director that approves or declines features and he oversees changes to the story. |
| Dulla Kastriot | Programer | Programmers are responsible for the technical implementation of the features approved by the game director and the lead programmer. They will primarily work with the source code and its components. |
| Dulla Kastriot | Assistant Story Designer | The assistant story designer assists the lead story designer to write the story of the project. He makes sure that there are no inconsistencies or plot holes in the story. |
| von Ballmoos Kevin | Lead Programmer | The lead programmer is the chief of programming and responsible for the technical implementation of the project. He is primarily in touch with the game director and works closely with his team to make sure the code is reusable and stable. |
| von Ballmoos Kevin | Lead Game Designer | The lead game designers job is to make sure that the game elements are coherent and make sense. He is responsible for the usability and user friendliness as well as the overall design of the project. |

Table 1: Job descriptions

During the project, there can be other roles that need to be filled. These roles will therefore be added to the table above and the document will be updated.

## **Development and Engine**

The development of this project is primarily made with the Unity Engine (version 2021.3.5f1 or higher). The components that ware to be developed or used in this project are going to be downloaded from the Unity Asset Store and are to be used as placeholders for the rest of the development process. Once the core systems and functionality has been established, own assets are going to be created and used instead of the aforementioned Unity Assets.

The code as well as the developed C# scripts are going to be created with the IDE known as Rider (version 2022.2.2 or higher). If the individual programmer desires, he can change the IDE to one of his likings. If the Lead Programmer decides not to use any other IDEs for compatibility reasons or otherwise, Rider is to be used in all cases.

## **Core Systems**

In the following table the core systems are defined as must-have functions and are therefore to be implemented into the final game under all circumstances. In the following table we can see a name as well as a description for all the functions made. The more important part here, however, is the status of development that can be seen.

|  |  |  |
| --- | --- | --- |
| **Naming** | **Description** | **Status** |
| Dialogue System | In this system we can create multiple dialogues that are used in the game to progress the story. This is going to be the one core system the project depends most on.  The player is going to be able to make choices that will progress the story in a different direction, based on the decisions made by the player, the dialogue takes another turn and therefore can be either a game over for the player or it can lead to a new chapter, treasure or item.  The player should also be able to go back in the dialogue to be able to read the story that up until the last checkpoint. | In development. |
| Save System | The player should be able to save the game at any point in the story. All his decisions as well as his items and the page of the story where he is at are to be saved securely.  Eventually we can build in an auto save function that saves the game at certain checkpoints. | In development. |
| Loading System | The player is able to load one of the made saves including all of his choices as well as his inventory and current status. | In development. |
| Title Screen / Menu Screen | There must be a title screen developed for the game. In this screen all the normal menu points should be available, for example “New Game”, “Continue”, “Load Game” as well as “Options” and “Quit Game”. | In development. |
| Inventory System | An inventory should be implemented that allows the player to pick up items during the playthrough. These items can manipulate the choices made and unlock or lock certain choices for the player. These items should be visible in a separate menu and a tooltip with a description is going to be needed for the individual item as well as a number indicating how many of said items are in the inventory. | In development. |
| Checkpoint System | During the game there should be multiple checkpoints where the game is automatically save. These checkpoints are going to be spread throughout a chapter so that the player can easily reload the game in case of death or when he is stuck in a situation that he does not want.  A checkpoint could probably be set when a very important decision is going to be made by the player. | In development. |
| Pause menu | By pressing the “Esc”-Key on the keyboard the player should be able to access the pause menu. This menu is basically a copy of the already implemented menu with the addition that the player can chose to go back to the title screen instead of quitting the game completely. | In development. |
| Maps and Drawings | Throughout the game there should be multiple maps for the player to be able to see where he is at the moment. Especially at the beginning there should be a world map to show the players in what kind of environment he is.  In later sections of the game there should be maps of cities, villages or even drawings of different rooms or characters. A good example here is found in the game “Pillars of Eternity: Deadfire” where the book shows a clear picture of the environment on one side, while telling the story in the other side. | In development. |
| Main character | At the beginning of the game the player must choose between three main characters who all have a different prologue but follow the same story in a different way. Depending on the choice different dialogues are triggered. | In development. |
| Music | In the prototype build of the game, there are going to be at least 3 soundtracks that depict an idle state as well as a more sinister and more lighthearted state of the story. Once the project has advanced to a more advanced state there can be an idea for more soundtracks as well as sound effects that are played during the game for example depicting a large scale battle or a lively tavern. | In development. |

Table 2: Core systems

## **Advanced systems**

Im folgenden Kapitel definieren wir die Systeme und Funktionen, deren Entwicklung massgeblich zur Erhöhung der Qualität des Projektes beiträgt, allerdings nicht funktional zwingend sind. Diese Funktionen können, müssen aber nicht eingebaut werden. Sie werden daher als Nice-To-Have-Funktionen definiert.

|  |  |
| --- | --- |
| **Namensgebung** | **Funktionsbeschreibung** |
| Kampfsystem | Es soll ein Kampfsystem eingebaut werden, bei dem der Spieler in der Lage ist, einen Kampf gegen verschiedene Gegner zu führen.  Wie das Kampfsystem auszusehen hat, ist noch nicht entschieden. |
| Interaktive Karte | Eine Karte, die es dem Spieler erlaubt sich nach eigenem Gutdünken auf ihr zu bewegen. Die ihm aber auch wichtige Informationen wie z.B. die nächstgelegene Stadt liefert. |
| Handelssystem | Der Spieler kann sich an diverse Händler wenden, um gefundene Gegenstände zu verkaufen und andere Gegenstände zu kaufen. Es wird daher auch eine Währung in das Spiel integriert. |
| Fähigkeiten | Der Spieler hat je nach Protagonisten Wahl eine Anzahl von Fähigkeiten, die es ihm erlaubt Situationen bzw. Entscheidungen einfacher zu treffen oder es ihm gar erlauben Entscheidungen auf die «Bestmögliche» Weise zu lösen. |
| Charaktererstellung | Der Spieler ist in der Lage sich einen eigenen Charakter zu erstellen inkl. Eigenem Namen und Geschlecht. |
| Charakter Attribute | Der Spieler verfügt über diverse Attribute wie z.B. Stärke, Geschick, Ausdauer und Charisma, die es ihm erlauben diverse Entscheidungen einfacher zu treffen.  Dies erlaubt es dem Spieler, basierend auf einer gewissen Chance, die z.B. mit der Zahl des Attributes multipliziert wird (z.B. Change sich an den Wachen vorbeizuschleichen = 5%, Geschicklichkeit = 10, Resultat = 0.5 X 10 = 5. Somit 5% + 5% = Endresultat) Entscheidungen besser zu bewältigen. |
| Uhrzeitsystem | Ein System das die Tages- und Nachtzeit der Welt festhält. |
| Event Director | Eine unsichtbare Entität, die basierend auf diversen Entscheidungen bzw. Systemen wie dem Uhrzeitsystem oder die Interaktive Karte, Events triggered, die den Spielverlauf beeinflussen. |
| Audio | Audio für das Spiel. Darunter ist Musik sowie diverse Audioeffekte zu verstehen. |
| RNG-System | Diverse Entscheidungen können sich negativ auf die Chance auswirken. Zum Beispiel gibt es eine geringe Chance während einer Scene von einem Dach zu fallen und sich zu verletzen. |
| Hunger/Durst System | Der Spieler erhält während des Spiels Hunger bzw. Durst, je nachdem wie lange er ohne diese Dinge verbracht hat. Essenziel hierfür ist das Uhrzeitsystem. |
| Sprachsupport | Das Spiel soll in den Sprachen Deutsch/Französisch/Albanisch und Chinesisch. |
| NPC Party Members | Der Spieler kann während dees Spiels mehrere Partymembers finden und in seine Gruppe laden. Diese können ihm im Kampf beistehen bzw. Quests geben oder aber nützliche Informationen zu bestehenden Quests. |
| NPC Party Members Portrais | NPC Party Members bzw. wichtige Charactere haben ein eigenes Portrait. |
| Lager System/Unterschlupf? | Für zum interagieren mit den NPC Charakteren der Party. |

Table 3: Nice to have functions

## **Goals**

Das Ziel dieses Dokumentes ist eine solide Grundlage für die Dokumentation des Projektes zu bieten sowie den Mitgliedern einen Einblick in die Zukunft bzw. den momentanen Stand der Entwicklung zu liefern.

Des Weiteren ist es das Ziel der Entwickler bzw. «Three Troll Games» einen lauffähigen Prototypen des Projektes inkl. Aller Must-Have-Funktionen bis am 31.12.2023 fertigzustellen. Darin nicht notwendig sind Animationen sowie Assets, Design und Audio.

## **Code and versioning**

Der Code ist neben den Assets das Herzstück des Projekts. Der Code wird grundsätzlich nur von den Programmierern oder dem Chief Programer bearbeitet. Dabei sind diverse Standards in Bezug auf die Syntax, die Kommentare sowie die Erstellung von Methoden und Klassen anzuwenden.

Der Code wird grundsätzlich mittels Github versioniert. Es gibt neben dem Master-Branch noch den Develop-Branch, sowie den Administration-Branch innerhalb der Versionskontrolle. Für alle weiteren Entwicklungen an dem Projekt, müssen sogenannte Feature-Branches erstellt werden. Folgende Regeln gelten bei der Erstellung der Feature-Branches:

1. Der Feature-Branch wird im lowercase Camel Case geschrieben. **Beispiel**: feature\MainMenu
2. Die Feature-Branches werden mittels Backslasch von der eigentlichen Codestruktur getrennt. Das heisst es wird **feature\<Arbeit>** benannt. **Beispiel**: feature\MainMenu.
3. Alle Branches werden entweder vom Chief Programer oder vom Game Director in den Develop-Branch und später in den Master gemerged. Es darf kein eigener Merge stattfinden.
4. Nach erledigter Arbeit muss ein Pull-Request zum Chief Programmer oder zum Game Director gemacht werden. Erst nach Bestätigung einer der Beiden, darf ein neuer Branch mit einem neuen Feature erstellt werden.

Für eine genauere Dokumentation der Code-Standards sowie der Namenskonventionen während des Projektes kann das Dokument Coding Conventions.docx konsultiert werden.

* 1. **XML story files**

Die Geschichte soll über mehrere Kapitel entstehen. Die Kapitel selbst in die Dialogue-Boxen zu schreiben ist unglaublich mühselig und kann schnell zu Unübersichtlichkeit führen. Dies führt unweigerlich dazu, dass man Fehler innerhalb der Geschichte macht und somit nicht in der Lage ist, die gewünschte Qualität der Geschichte sicherzustellen.

Um diesen Problem entgegenzuwirken, haben wir uns ein XML File erstellt, dass in der Lage ist vom Dialogue-Editor eingelesen zu werden und die einzelnen Story-Brocken und Kapitel in die jeweiligen Nodes zu speichern. Um das im Code zu ermöglichen, muss das File allerdings ein spezielles Formt haben, damit der Code in der Lage ist die Geschichte richtig abzufüllen und schlussendlich in die jeweiligen Nodes zu fügen. Es folgt ein Beispiel der XML-Struktur, wie sie erstellt werden muss, damit das File richtig eingelesen werden kann.

## **Mock-up**

In the following section we see the mock-up idea of how the game will look in it’s prototype state. This mock-up is by no means final and is subject to change. Should there be new mock-ups drawn, the old ones have to be marked by a clear sign that they are old versions and therefor no longer in use. No mock-up is to be deleted from this document.

