

2022

Pflichtenheft „SOKI“

IM RAHMEN DER PRÜFUNGSLEISTUNG SOFTWARETECHNIK
SOMMERSEMESTER 2022

SELINA ZESEWITZ-THIEL, CORA V. STOKOWSKY, LUKAS KEINERT, JULIE L.
HAUPT

Inhaltsverzeichnis

Einleitung	2
1. Zielbestimmung.....	2
1.1 Musskriterien	2
a. Hauptmenü	2
b. Speicherfunktion	2
c. Befehleingabe.....	2
d. Anforderungen an das Spiel	4
1.2 Wunschkriterien.....	4
a. Hauptmenü	4
b. Befehleingabe.....	4
c. Sprache	4
d. Anforderungen an das Spiel	4
1.3 Abgrenzungskriterien.....	4
a. Spielprinzip	4
2. Produkteinsatz	5
2.1 Anwendungsbereiche	5
2.2 Zielgruppe	5
2.3 Betriebsbedingungen	5
3. Produktübersicht.....	5
4. Funktionale Anforderungen	6
4.1 Hauptmenü	6
4.2 Steuerung des Spiels	6
4.3 Befehle des Spiels.....	6
4.4 Speicherfunktion	7
5. Nicht-Funktionale Anforderungen.....	7
5.1 Zeitverhalten	7
5.2 Robustheit	7
6. Produktdaten.....	8
7. Laufzeitumgebung	8
7.1 Software.....	8
8. Entwicklungsumgebung.....	8
8.1 Software.....	8
8.2 Orgware	8
9. Glossar	9

Einleitung

Das Programm mit dem Arbeitstitel "SOKI", ist ein horrorbasiertes Textadventure das von Team SOKI im Rahmen einer Prüfungsleistung entwickelt wird. In diesem Dokument werden die Details zur Entwicklung der Anwendung festgehalten und Pflichtvorgaben festgelegt, welche bis zur Fertigstellung zu erfüllen sind.

1. Zielbestimmung

1.1 Musskriterien

a. Hauptmenü

- *"Neues Spiel starten"*: Der Nutzer hat die Möglichkeit ein neues Spiel zu starten. Hierbei öffnet sich das Spielfenster am Anfang der Story und das Spiel hat noch keine Eingaben wie Name oder Befehle empfangen.
- *"Spiel laden"*: Der Nutzer lädt das vorherige Spiel. Den Startpunkt bildet hierbei der letzte Standpunkt des Spielers, bevor das Spiel beendet wurde. Liegen noch keine Speicherdaten vor, ist diese Option deaktiviert.
- *"Programm beenden"*: Der Nutzer kann das Programm beenden. Die Fenster des Spiels schließen sich und alle Funktionen des Spiels werden beendet.

b. Speicherfunktion

- Im Verlauf des Spieles werden automatisch die Informationen zum aktuellen Fortschritt abgespeichert.
- Die abgespeicherten Daten lassen sich bei einem erneuten Programmaufruf über „Spiel laden“ abrufen.

c. Befehleingabe

- Während des Spiels hat der Nutzer die Möglichkeit den Fortgang der Story sowie seine Handlungsmöglichkeiten und Hilfsfunktionen mittels der Eingabe von Textbefehlen zu steuern.
- Das Programm soll Fehleingaben erkennen und entsprechend mit ihnen umgehen.

- Der Nutzer muss bei der Befehlseingabe nicht auf Groß- und Kleinschreibung achten, da diese vom System verarbeitet und abgefangen wird. Die Funktionsweise des Spiels wird so nicht beeinträchtigt. Umlaute müssen jedoch umschrieben werden.
- Dem Nutzer stehen verschiedene Befehle mit eigenen Funktionsweisen zur Verfügung:
 - ❖ *<Hilfe>*: Es wird eine Liste der verfügbaren Befehle mit einer Kurzerklärung ausgegeben.
 - ❖ *<Menu>*: Das Spiel erfasst den letzten Standpunkt des Spielers und speichert diesen. Danach kehrt der Nutzer ins Hauptmenü zurück.
 - ❖ *<Beenden>*: Der letzte Standpunkt in der Story wird erfasst und gespeichert. Im Anschluss wird das Programm mit allen Funktionen beendet.
 - ❖ *<Untersuche {Ziel}>*: Der Spieler untersucht ein Objekt in seiner Umgebung und erhält Informationen dazu. Mögliche Ziele dieser Aktion sind Items, Charaktere und Orte sowie Umgebungsgegenstände.
 - ❖ *<Nimm {Item}>*: Das Item wird aufgesammelt und im Inventar verstaut.
 - ❖ *<Inventar>*: Eine Liste der sich aktuell im Inventar befindenden Items wird ausgegeben.
 - ❖ *<Benutze {Item} (mit {Ziel})>*: Das gewählte Item wird eingesetzt. Abhängig vom Gegenstand muss/ kann ein Ziel gewählt werden. Mögliche Ziele sind andere Items, Objekte in der Umgebung sowie Figuren.
 - ❖ *<Interagiere mit {Ziel}>*: Der Spieler führt eine Aktion passend des gewählten Ziels aus. Mögliche Ziele sind Charaktere und Umgebungsobjekte. Ist das Ziel eine Figur wird eine Konversation mit dieser gestartet.
 - ❖ *<Gehe zu {Ort}>*: Der Spieler bewegt sich zum gewählten Ort.
 - ❖ *<Wähle {Option}>*: In speziellen Situationen beispielsweise Konversationen können weitere Eingaben abgefragt werden. Diese werden in der Regel vorgeschlagen und als Liste zur Verfügung gestellt.

d. Anforderungen an das Spiel

- Die Handlungsstränge des Adventures wurden so ausgeplant, dass der Nutzer sich nicht selbst in Sackgassen befördern kann.

1.2 Wunschkriterien

a. Hauptmenü

- "Achievements": Im Verlauf der Handlung können für bestimmte Aktionen Errungenschaften gesammelt werden. Eine Übersicht der erreichten und Hinweise auf mögliche weitere kann über das Hauptmenü abgerufen werden.
- „Spracheinstellung:“ Wenn eine anderssprachige Version des Spiels vorhanden ist, kann im Hauptmenü diese ausgewählt werden.

b. Befehleingabe

- "Easter-Eggs": Optionale Befehle die nicht zum Satz der Standarteingaben gehören und kleine Anspielungen oder unerwartete Interaktionen beinhalten.

c. Sprache

- Eine Englische Version des Spiels ist verfügbar. Diese Übersetzung beinhaltet alle Elemente des Spiels.

d. Anforderungen an das Spiel

- Der durch das Spiel ausgegebene Text wird in einer Art Schreibanimation dargestellt.
- Objekte, Orte, Figuren und Items mit denen der Spieler aktuell über die verschiedenen Befehle interagieren kann, werden im Text gesondert hervorgehoben.

1.3 Abgrenzungskriterien

a. Spielprinzip

- Das Spiel ist ein reines Singleplayer-Spiel und enthält keine Mehrspielerelemente oder -funktionen.
- Abgesehen von Text und dessen eventuelle Animation gibt es keine grafischen Darstellungen im Spiel. Das Logo und eventuelle Achievements können Abbildungen im Hauptmenü sein.

2. Produkteinsatz

2.1 Anwendungsbereiche

Das Programm wird von einem Einzelnutzer zum Spielen eines Textadventures genutzt.

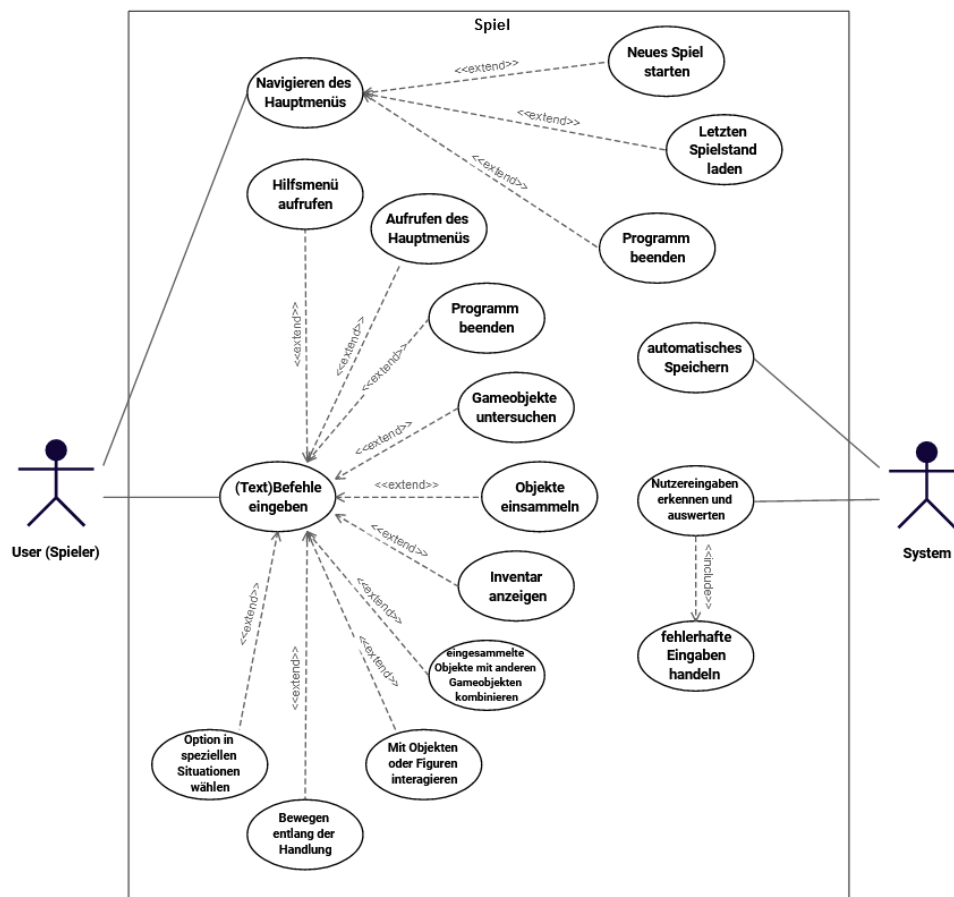
2.2 Zielgruppe

Das Programm soll von Personen über 14 Jahren zum Entertainment genutzt werden. Der Stil des Spiels und die Story sollen eine Personengruppe ansprechen, die sich für das Horror/ Mystery Genre interessieren.

2.3 Betriebsbedingungen

Um das Programm nutzen zu können sind Java und Java FX auf dem Anwendercomputer benötigt. Es wird des Weiteren genügend Speicherplatz und ein Windows-Betriebssystem vorausgesetzt.

3. Produktübersicht



4. Funktionale Anforderungen

4.1 Hauptmenü

- /F101/ Das Programm lädt einen neuen Spielstand. Es verfügt an diesem Punkt über keine Spielerdaten.
- /F102/ Der letzte Standpunkt des letzten Spielstandes wird aufgerufen. Der Benutzer setzt sein Spiel an dieser Stelle fort.
- /F103/ Das Programm mit all seinen Funktionen wird beendet.

4.2 Steuerung des Spiels

- /F201/ Benutzereingaben werden vom Programm untersucht. Wird ein übereinstimmender Befehl dazu gefunden, führt es diesen aus.
- /F202/ Invalide Benutzereingaben werden abgefangen. Der Benutzer wird zur erneuten Eingabe aufgefordert.

4.3 Befehle des Spiels

- /F301/ Ein Hilfsmenü wird aufgerufen, welches alle für den Spieler nutzbaren Befehle auflistet und deren Funktionsweise kurz erklärt.
- /F302/ Der Standpunkt des Spielers wird gespeichert und das Hauptmenü wird aufgerufen.
- /F303/ Der Standpunkt des Spielers wird gespeichert und das Programm mit all seinen Funktionen beendet.
- /F304/ Orte, Gegenstände und Charaktere werden untersucht. Das System gibt Informationen bezüglich dieser aus.
- /F305/ Gegenstände einsammeln und dem Inventar hinzufügen.
- /F306/ Auflisten des aktuellen Inventars des Spielers. Es werden dabei nur Gegenstände angezeigt, die auch für den Benutzer sichtbar sein sollen.
- /F307/ Gegenstände aus dem Inventar werden benutzt. Dazu werden sie auf Orte, Umgebungsobjekte oder Charaktere

angewendet. Zusätzlich können manche Gegenstände mit anderen kombiniert werden. Nach dem Verwenden können Gegenstände im Inventar bleiben oder vom System aus diesem gelöscht werden.

/F308/ Interagieren mit Gegenständen, um diese zu benutzen, oder mit Charakteren, um diese anzusprechen.

/F309/ Der Nutzer navigiert sich entlang der Handlungsstränge.

/F310/ Der Nutzer wählt eine vom System vorgegebene Dialogoption aus.

4.4 Speicherfunktion

/F401/ Das Programm speichert automatisch den Standpunkt des Spielers. So geht bei eventuellem Absturz des Spiels kein Fortschritt verloren.

/F402/ Daten des Benutzers werden vom System erfasst und gespeichert. [Spielername, Inventar (Gegenstände und deren Sichtbarkeit), Durchgang]

5. Nicht-Funktionale Anforderungen

5.1 Zeitverhalten

Das System muss innerhalb von 30 Sekunden auf die Nutzereingaben reagiert haben.

5.2 Robustheit

Das Spiel soll mindestens so lange im Dauerbetrieb benutzt werden können, bis ein Ende der Storyline erreicht wurde.

6. Produktdaten

/D01/ Die Textblöcke, die das System braucht, um die Handlung zu gewährleisten.

/D02/ Spielerdaten (Name, Standpunkt, Item, Itemsichtbarkeit, Durchgang)

7. Laufzeitumgebung

7.1 Software

Um das Spiel für den Anwender verfügbar zu machen, ist Java sowie JavaFX auf dem Anwendercomputer erforderlich. Diese werden beim Download unserer Datei mitgegeben.

8. Entwicklungsumgebung

8.1 Software

Zur Entwicklung der Anwendung wurden drei Programme verwendet:

- IntelliJ, um den Code zu verfassen,
- GitHub, um auf separaten Geräten am gleichen Projekt arbeiten zu können und
- SceneBuilder zum Entwurf des User-Interfaces.

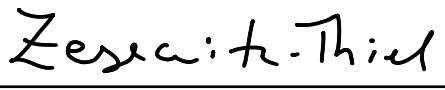
8.2 Orgware

Um das Projekt zu organisieren und koordinieren wurden folgende Anwendungen benutzt:

- GitHub Issues zur Aufteilung der Aufgaben und Planen der Projektphasen.
- Google Tabellen zur Erfassung der aufgewendeten Zeit.
- Google Documents, um wichtige Informationen zusammenzutragen und gemeinsam die Dokumentation zu bearbeiten.
- Microsoft Word und Adobe Acrobat zur Formatierung und Signierung jeglicher Dokumente, beispielsweise dem Pflichtenheft.
- Discord für eine schnelle und unkomplizierte Kommunikation.

9. Glossar

SOKI	Arbeitstitel des Textadventures
Team SOKI	Das Entwicklerteam hinter SOKI bestehend aus Selina Zesewitz-Thiel, Cora V. Stokowsky, Lukas Keinert und Julie L. Haupt
Textadventure	Computerspiel, bei welchem der Benutzer nur durch Textkommandos agieren kann


Selina Zesewitz-Thiel


Cora V. Stokowsky


Lukas Keinert


Julie L. Haupt