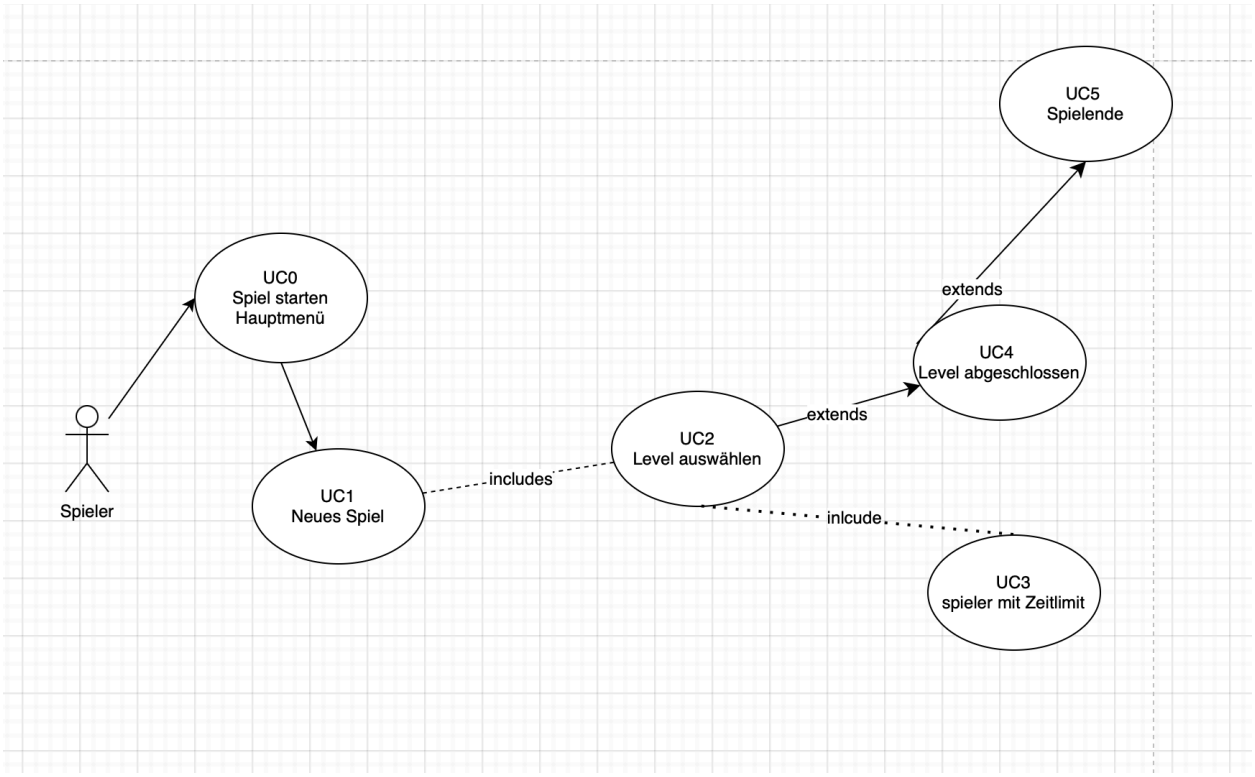
Use Case Modellierung – Data-Security-Memory

“data-security-Memory” ist ein Lernspiel, bei dem der Spieler durch klassisches Memory etwas über Datensicherheit lernt.



Use Case 0: Spiel starten

Use Case: Spiel starten/ Hauptmenü öffnen
Akteur: Spieler
Ziel:
Vorbedingung:

- Das Spiel ist installiert und bereit zu starten

Ereignisfluss:
Standard:

1. Der Spieler öffnet die Anwendung.
2. Das System zeigt das Startmenü mit dem Titel „Data-Securit-Memory“.
3. Das System lädt alle benötigten Ressourcen (Kartendaten und Inhalt zu Datensicherheit, Grafiken, Sounds).

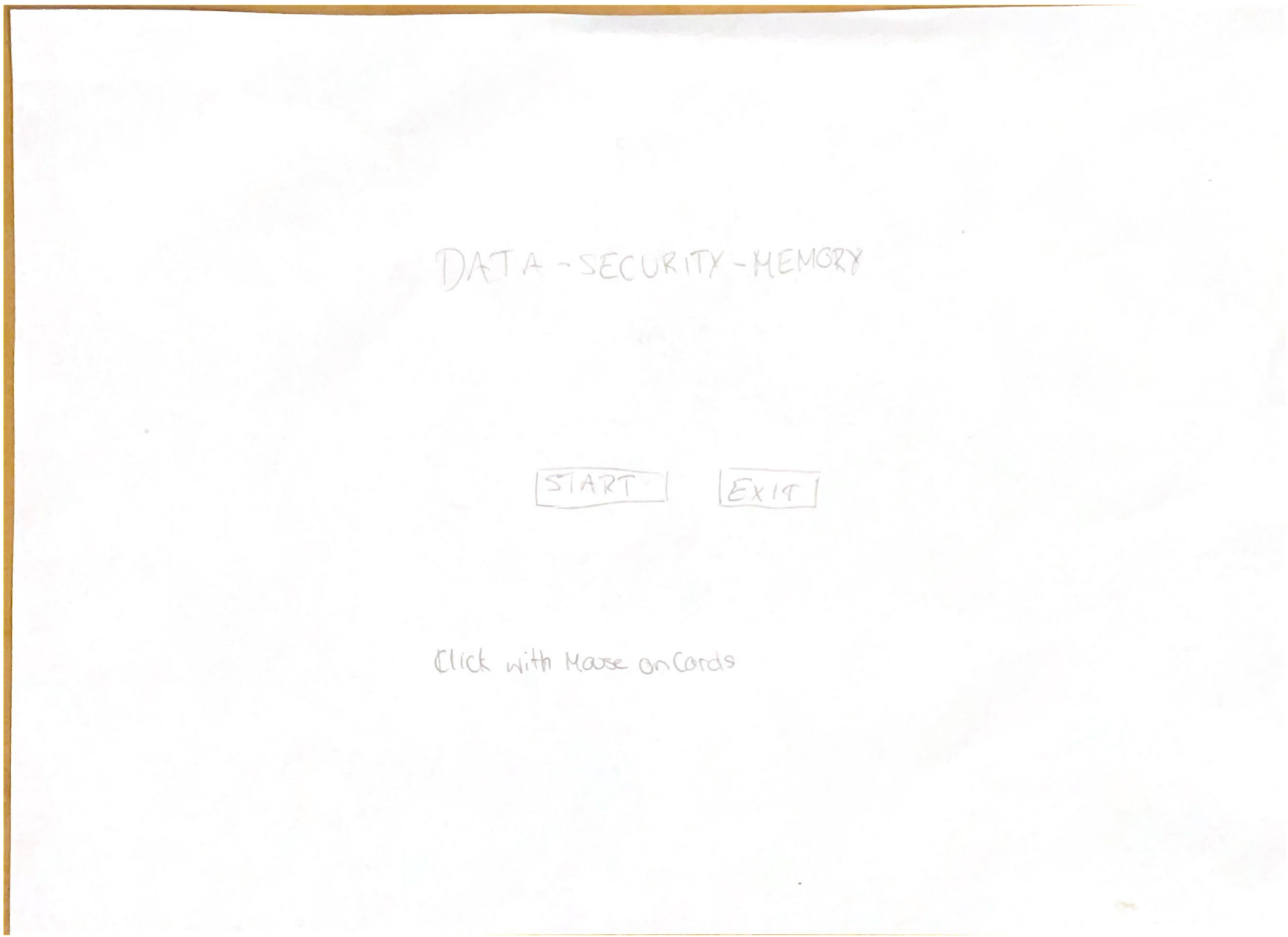
4. Das System prüft, ob alle Dateien und Daten vorhanden sind und vollständig sind.
5. Das System zeigt das Startmenü (mit den Optionen: „Start“, „Exit“).

Alternativen:

- Bei Fehlen von wichtigen Dateien → Das System gibt eine Fehlermeldung wieder.

Nachbedingung:

- Der Spieler sieht das Startmenü
- Auswählbar von Optionen


DATA - SECURITY - MEMORY
START
EXIT
Click with Mouse on Cards

Use Case 1: Neues Spiel

Use Case: Neues Spiel anfangen
Akteur: Spieler
Ziel: Spieler soll alle Paare aufdecken und dabei über Datensicherheit lernen.
Vorbedingung

- Das Spielfeld ist vollständig geladen
- Alle Karten sind zu Beginn verdeckt

Ereignisfluss:

1. Der Spieler klickt auf eine Karte → Das System deckt die Karte auf.
2. Der Spieler klickt auf zweite Karte → Das System deckt zweite Karte auf.
3. Das System prüft, ob die Karten zusammenpassen. („Virus“ ←→ „Virus“)
4. Wenn das Paar richtig ist → Karten bleiben offen.
5. Wenn falsch → Karten werden nach kurzer Zeit wieder verdeckt.

Nachbedingung:
Alle Kartenpaare sind aufgedeckt.

Use Case 2: Level auswählen

Akteur: Spieler
Ziel: Spieler möchte ein bestimmtes Level mit passender Schwierigkeit spielen.

Vorbedingung:

- Das Spiel ist gestartet und das Hauptmenü sichtbar

Ereignisfluss:

1. Spieler klickt auf „neues Spiel“.
2. Das System zeigt eine Levelauswahl (z. B. 1-3)
3. Der Spieler wählt ein Level.
4. Das System lädt das entsprechende Spielfeld (z. B. mehr Karten, weniger Zeit)

Nachbedingung:

- Das gewählte Level wird gestartet.

Use Case 3: Spielen mit Zeitlimit

Akteur: Spieler
Ziel: Spieler möchte das Level innerhalb der vorgegebenen Zeit erfolgreich abschließen.

Vorbedingung:
- Das Spielfeld ist geladen.
- Ein Timer ist aktiviert.

Ereignisfluss:
1. Der Timer startet beim ersten Klick auf eine Karte.
2. Der Spieler deckt Kartenpaare auf.
3. Wenn der Timer abläuft, bevor alle Paare aufgedeckt wurde → das System zeigt eine „Verloren“- Nachricht an.
4. Wenn alle Paare vor Ablauf des Timers gefunden werden, → das System zeigt eine „Gewonnen“ Nachricht.

Nachbedingung:
- Das Level ist entweder gewonnen oder verloren.

Alternativen:
- Spieler verliert →Möglichkeit, das Level ernetu zu spielen oder ins Hauptmenü.
- Spieler gewinnt → System bietet an, das nächste Level zu spielen.

Use Case 4: Level abgeschlossen

Akteur: Spieler
Ziel: Fortschritt speichern oder nächstes Level beginnen.

Vorbedingung:
- Der Spieler hat ein Level erfolgreich beendet

Ereignisfluss:
1. Das System zeigt das Ergebnis (Zeit, Punkte, …)
2. Der Spieler wählt
    a. „Nächstes Level“→ System lädt das nächste Level.
    b. „zurück zum Hauptmenü“→ System zeigt Hauptmenü

Nachbedingung:
- Der Fortschritt ist gespeichert, und der Spieler befindet sich im gewählten Zustand.

Use Case 5: Spiel-Ende

Use Case: Das Spiel end und die Zeit wird angezeigt.
Akteur: Spieler
Ziel: Das Spielergebnis sehen und ein neues Spiel starten.

Ereignisfluss:

1. Das System erkennt, dass keine verdeckten Karten mehr vorhanden sind.
2. Das System zeigt das Endergebnis.
3. Der Spieler klickt auf „Neues Spiel“ oder „Beenden“
4. Das System führt die gewählte Aktion aus

Alternativen:

a) Spieler wählt „Neues Spiel“ → Das System startet UC1 erneut.
b) Spieler wählt „Beenden“ → Das System kehrt zum Hauptmenü zurück.

Nachbedingung:
Das Spiel ist abgeschlossen oder neu gestartet.