Projektdokumentation Modul 326

|  |  |  |
| --- | --- | --- |
| Auftraggeber: |  | R. Probst |
| Autor\*in: |  | Yannis Lee, Jonas Rhbary, Thanabalasingam Maruthan |
| Klasse: |  | IM19A |
| Projekttitel: |  | M326 Memory |

Inhaltsverzeichnis

[Projektbeschreibung 2](#_Toc38719552)

[Anwendungsentwurf 2](#_Toc38719553)

[Testfälle 4](#_Toc38719554)

[Lösungsbeschreibung 4](#_Toc38719555)

[Anhang 4](#_Toc38719556)

Abbildungsverzeichnis

[Abbildung 1: Muster 2](#_Toc1036253)

Tabellenverzeichnis

[Tabelle 1: Muster Anwendungsfallspezifikation 2](#_Toc38719557)

[Tabelle 2: Muster CRC-Karte 3](#_Toc38719558)

# Projektbeschreibung

*Beschreiben Sie in Ihren Worten das Projekt respektive Ihre Vision dazu (Einleitung und Übersicht zum Projekt respektive Projektauftrag).*

## Ziele

*Definieren Sie sich selbst Ziele (mind. 2), die Sie erreichen wollen. Ziele beschreiben, was Sie bis wann (konkreter Zeitpunkt) mit Ihrem Projekt / Ihrer Anwendung erreichen möchten.*

*Hinweis: Stellen Sie sicher, dass die Ziele* [*SMART*](https://de.wikipedia.org/wiki/SMART_(Projektmanagement)) *sind. Unterscheiden Sie insbesondere zwischen zwingend zu erreichenden MUSS-Zielen sowie wünschenswert zu erreichenden KANN-Zielen (diese können Sie bei Bedarf noch priorisieren).*

# Anwendungsentwurf

*Erstellen Sie in den Unterkapiteln die notwendigen Artefakte (z.B. Textbeschreibungen von Anforderungen, UML-Diagramme, Mockups der grafischen Oberfläche), die Sie zur Erstellung der Anwendung benötigen. Darstellungen, welche grösser sind als der zur Verfügung stehende Platz, müssen als Vorschaubild dargestellt, im Anhang hinzugefügt und nachvollziehbar referenziert werden (bspw. «siehe Anhang 1 – Klassendiagramm Anwendung (Bild1.jpg)»).*

## Anwendungsfalldiagramm(e)

*Erstellen und beschreiben Sie hier ein Anwendungsfalldiagramm mit mindestens einem Anwendungsfall und mindestens einem Akteur.*

## Anwendungsfallspezifikationen

*Erstellen Sie hier für jeden Anwendungsfall eine Anwendungsfallspezifikation.*

Tabelle 1: Muster Anwendungsfallspezifikation

|  |  |
| --- | --- |
| ID |  |
| Name |  |
| Kurzbeschreibung |  |
| Akteur(e) |  |
| Auslöser |  |
| Vorbedingung(en) |  |
| Ergebnis |  |
| Nachbedingung(en) |  |
| Ablauf |  |
| Alternativen |  |
| Fehlerfälle |  |
| Testfälle |  |

## Anforderungen

## Funktionale Anforderungen:

* Das Spiel Memory ist für 2 Spieler anwendbar
* Das Spiel ist zwischen 5x5 und 10x10 Felder gross, wobei 5x5, 7x7 und 9x9 keine Optionen sind, da dies eine ungerade Anzahl an Feldern ergibt.
* Mehrere Spiele können hintereinander gespielt werden, dabei wird die Punktzahl der Spiele vorher übernommen.
* Beim Start des Spieles sind die Kartenrückseiten zu sehen
* Bei einem Zug zieht ein Spieler 2 Karten, diese beiden Rückseiten werden dann angezeigt, sind diese identisch, kriegt der Spieler ein Punkt und bekommt einen neuen Zug.
* Sind die Karten nach einem Zug nicht identisch, werden wieder die Rückseiten der Karten angezeigt und der andere Spieler bekommt einen Zug.
* Wenn alle Kartenpaare aufgedeckt sind, wird das Spiel beendet und der Benutzer bekommt die Option ein weiteres Spiel zu starten oder das Programm zu beenden.
* Der Spieler der mehr Punkte erzielt hat, ist der Gewinner.

Fakultative Anforderungen:

* Es kann Joker-Karten geben, diese geben eine höhere Punktzahl als normale Karten.
* Alle Spiele können einzeln gespeichert werden und als Ausgabe können die Spieler mit Anzahl der Spiele, der erreichten Punkte und dem Wert Punkte pro Spiel ausgegeben werden.
* Es kann einen Timer geben, wenn dieser abgelaufen ist, ist der andere Spieler am Zug.

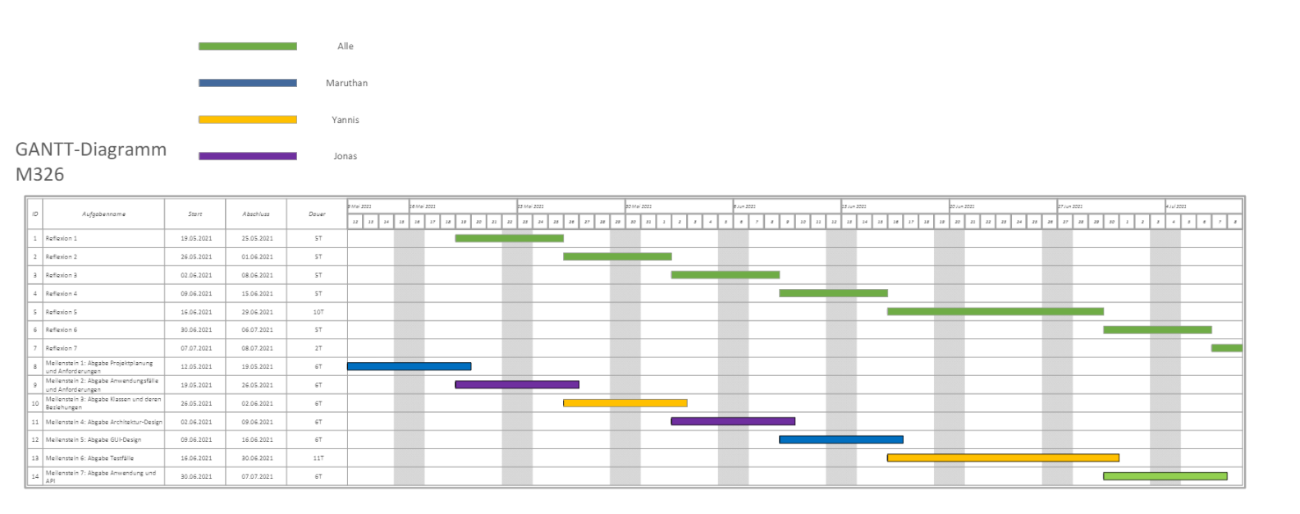
Nichtfunktionale Anforderungen:

* Das Programm ist in Java mit der OO-Technik geschrieben
* Das GUI wird mit Java-Swing realisiert und es ist das MVC-Pattern zu benutzen.
* Die Umsetzung basiert auf selbst erstellten Klassen-, Sequenz-, Zustands- und Aktivitätsdiagrammen, welche die Anwendung in groben Zügen beschreiben.

## Projektplanung

Aufteilung unter den Gruppenmitgliedern:

* Wir haben jeweils für jede Woche einen Gruppenchef bestimmt, der die Verantwortung für diese Woche und auch für die jeweilige Abgabe in dieser Woche.
* Wir haben 7 Abgaben, die letzte Abgabe ist diese, in der wir das ganze Programm abgeben müssen, deshalb haben wir diese Woche alle zusammen übernommen.
* Die restlichen 6 Abgaben haben wir durch 3 geteilt, was jeweils 2 Wochen als Gruppenchef für jedes Gruppenmitglied macht.
* Der Gruppenchef übernimmt die Verantwortung für die jeweilige Gruppe und deren Abgabe, das heisst aber nicht dass er der Einzige ist, der diese Woche den Auftrag macht, sondern wir bearbeiten die Aufträge jeweils gemeinsam.

Zeitliche Aufteilung:

* Wie Sie hier sehen können, haben wir zeitlich eingeplant, dass jede Woche jeder von uns 3 Gruppenmitgliedern eine Reflexion schreiben muss.
* Der Gruppenchef für die erste Woche war Maruthan, danach wird Jonas den 2. Meilenstein übernehmen und Yannis wird sich um die Abgabe der Klassen und deren Beziehungen als Gruppenchef kümmern.
* In der 2. Hälfte des Quartals wird Jonas den 4. Meilenstein «Architektur-Design» leiten und danach wird Maruthan die Abgabe des Gui-Designs übernehmen.
* In der letzten Woche, in der wir einen Gruppenchef zugewiesen haben, wird Yannis die Testfälle machen, und diese natürlich auch testen, bevor er die Verantwortung für deren Abgabe übernimmt.
* In der letzten Woche haben wir uns dazu entschieden dass wir alle zusammen die Verantwortung übernehmen und dann spätestens am 07.07.2021 das ganze Programm «Memory» abgeben.
* Wie Sie ebenfalls sehen können haben wir geplant dass wir jede Woche einen Meilenstein bearbeiten und diesen auch abgeben werden, da ohne den heutigen Tag vom 12.05.2021 uns 7 Schultage am BZZ übrig bleiben, für 7 Abgaben.

## Aktivitätsdiagramm(e)

*Erstellen Sie hier mindestens je ein Aktivitätsdiagramm für den gesamten Spielablauf sowie für den Ablauf einer Spielrunde und beschreiben Sie diese nachvollziehbar.*

## CRC-Karten

*Leiten Sie aus dem Projektauftrag mindestens 4 Fachklassen (d.h. keine Hilfsklassen wie «ImageLoader») ab und beschreiben Sie diese einzeln als CRC-Karte.*

Tabelle 2: Muster CRC-Karte

|  |  |
| --- | --- |
| Class: | |
| Responsibilities: | Collaborators: |

## Klassendiagramm(e)

*Erstellen Sie ein Klassendiagramm, dass möglichst vollständig ist und mindestens die CRC-Karten konkretisiert und beschreiben Sie das Diagramm.*

## Zustandsdiagramm(e)

*Erstellen Sie mindestens ein Zustandsdiagramm, welches den nächsten möglichen Zug (spielbare Karte) darstellt und beschreiben Sie dieses.*

## Sequenzdiagramm(e)

*Erstellen Sie mindestens ein Sequenzdiagramm, welches den Programmstart bis hin zum ersten Zug darstellt und beschreiben Sie dieses.*

## Mockup(s)

*Erstellen Sie Mockups (Entwürfe der grafischen Oberfläche) zu Ihrem Spiel und beschreiben Sie diese nachvollziehbar. Gehen sie dabei auf die verwendeten Elemente und Struktur (LayoutManager) ein.*

# Testfälle

*Beschreiben Sie in den Unterkapiteln «JUnit-Testfälle» sowie «Manuelle Testfälle» die Testfälle, welche Sie verwenden, um die fehlerfreie Arbeitsweise Ihrer Anwendung nachzuweisen.*

*Wichtige Elemente sind:*

* *Eindeutige Testfallnummer und Kurzbezeichnung*
* *Beschreibung des Testfalls*
* *Testdaten mit erwartetem Testresultat*

*Am Ende müssen Sie die Durchführung der Testfälle angemessen und nachvollziehbar protokollieren. Die Protokolle der Testfälle sind dem Anhang hinzufügen.*

## JUnit-Testfälle

*Beschreiben Sie hier JUnit-Testfälle (mind. 10) gemäss obigem Schema, die Sie umsetzen werden.*

## Manuelle Testfälle

*Beschreiben Sie hier manuelle Testfälle (mind. 5) gemäss obigem Schema, die nicht automatisiert werden können und die Sie manuell ausführen werden.*

# Lösungsbeschreibung

*Beschreiben Sie hier, wie Ihre Lösung aussieht und was sie kann respektive was noch nicht umgesetzt wurde (z.B. wichtige Hinweise zum Programmcode). Bewerten Sie dazu die von Ihnen gesetzten Ziele und Anforderungen und kommentieren Sie den Unterschied zwischen Ihrem IST-Ergebnis und dem ursprünglich definierten SOLL-Zustand. Machen Sie schliesslich einen kurzen Ausblick, wie Ihre Anwendung weiterentwickelt werden könnte.*

*Erstellen Sie zusätzlich eine API-Beschreibung für Entwickler sowie ein Benutzerhandbuch (inkl. Installationsanleitung) für Anwender Ihres Spiels.*

# Anhang

*Hinweis: Anhänge müssen mindestens in einem Format vorhanden sein, welches ohne Zusatzprogramme dargestellt werden kann. Zu den akzeptierten Formaten gehören Bilddateien (JPEG, PNG, GIF), gängige Microsoft Office Dateien und PDFs.*

*Folgende Anhänge werden erwartet:*

* *Testprotokolle*
* *API-Dokumentation*
* *Benutzerhandbuch*
* *Programm inkl. Source Code*
* *Deklaration Eigenleistung*
* *Optional: Weitere Dateien*