Dokumentation

Fachinformatiker Anwendungsentwicklung

Thema

**Super Mario Bros. Klon**

Prüfungsteilnehmer: Ricardo Liebig

Betrieb: cbm GmbH Bremen

Wegesende 3-4

28195 Bremen

Projektbetreuer: Sven Lilienthal

Durchführungszeitraum: 16.06.2025 bis 27.06.2025

Inhaltsverzeichnis

[1 Einleitung 2](#_Toc198802761)

[2 Projektbeschreibung 2](#_Toc198802762)

[2.1 Projektumfeld 2](#_Toc198802763)

[2.2 Organisatorische Vorgaben 2](#_Toc198802764)

[2.3 Ziel des Projektes 3](#_Toc198802765)

[2.4 Projektabgrenzung 3](#_Toc198802766)

[2.5 Projektschnittstellen 4](#_Toc198802767)

[3 Projektplanung 4](#_Toc198802768)

[3.1 Ist-Analyse 4](#_Toc198802769)

[3.1.1 Netzwerkplan Ist-Zustand 4](#_Toc198802770)

[3.2 Soll-Planung 5](#_Toc198802771)

[3.3 Projektablaufplan 5](#_Toc198802772)

[3.4 Personalplanung 6](#_Toc198802773)

[3.5 Personalkosten 6](#_Toc198802774)

[3.6 Nutzwertanalyse 6](#_Toc198802775)

[3.6.1 Bewertungskriterien 6](#_Toc198802776)

[3.6.2 Bewertung der Technologien 7](#_Toc198802777)

[3.6.3 Ergebnis und Entscheidung 7](#_Toc198802778)

[4 Realisierung 8](#_Toc198802779)

[4.1 Technische Umsetzung 8](#_Toc198802780)

[4.2 Spielmechanik 8](#_Toc198802781)

[4.3 Manuelles Testing 8](#_Toc198802782)

[4.3.1 Testziele 9](#_Toc198802783)

[4.3.2 Durchführung der Tests 9](#_Toc198802784)

[4.3.3 Fazit des manuellen Testens 9](#_Toc198802785)

[5 Grafische Benutzeroberfläche 9](#_Toc198802786)

[5.1 Aufbau und Elemente 10](#_Toc198802787)

[5.2 Technische Umsetzung 10](#_Toc198802788)

[6 Qualitätssicherung 10](#_Toc198802789)

[6.1 Projektstruktur und Codequalität 11](#_Toc198802790)

[6.2 Benutzerfreundlichkeit 11](#_Toc198802791)

[6.3 Performance und Stabilität 11](#_Toc198802792)

[7 Projektabschluss 12](#_Toc198802793)

[7.1 Übergabe 12](#_Toc198802794)

[7.2 Fazit und Ausblick 12](#_Toc198802795)

[A Anhang 12](#_Toc198802796)

[A.1 UML-Klassendiagramm 12](#_Toc198802797)

[A.2 Gantt Diagramm 13](#_Toc198802798)

[A.3 Mockup 13](#_Toc198802799)

# Einleitung Alle Projektphasen, von der Konzeption über die Entwicklung bis hin zu den Tests, wurden von einer einzelnen Person durchgeführt. Die Entscheidung für Java und Swing als technologische Grundlage ermöglichte eine effiziente Umsetzung sowie die Nutzung bereits vorhandener Kenntnisse.

# Projektbeschreibung

## Projektumfeld

Das Projekt wurde auf einem Windows-PC in einer lokalen Entwicklungsumgebung umgesetzt. Für die Entwicklung kam Java 11 in Verbindung mit Swing zum Einsatz. Als IDE wurde IntelliJ IDEA verwendet, um eine effiziente und produktive Entwicklungsumgebung zu gewährleisten. Die grafische Benutzeroberfläche wurde vollständig mit Swing gestaltet, was eine einfache Handhabung und schnelle Anpassung der Benutzeroberfläche ermöglichte.

## 2.2 Organisatorische Vorgaben

Das Projekt wurde im Zeitraum vom 16.06.2025 bis 27.06.2025 durchgeführt und durfte 80 Arbeitsstunden nicht überschreiten.

## Ziel des Projektes

## Das Ziel dieses Projekts war die Entwicklung eines einfachen 2D-Plattformspiels, das sich am klassischen Super Mario Bros orientiert. Dabei wurde die Programmiersprache Java in Kombination mit der Swing-Bibliothek für die grafische Benutzeroberfläche verwendet. Es sollte eine funktionale Basisversion eines Spiels entstehen, bei der der Spieler einen Charakter durch ein einziges Level steuert, Hindernissen ausweicht, Gegner überwindet und das Levelziel erreicht.

## Projektabgrenzung

Einzelne Handlungsschritte im Detail sind aus zeitlichen Gründen nicht Bestandteil dieses Projekts. Der Fokus liegt auf der Konzeption und Umsetzung der Kernfunktionen eines Super Mario Bros.-Klons sowie der technischen Einbindung einer Highscore-Speicherung über eine SQL-Datenbank. Die folgende Liste gibt einen Überblick über den groben Projektablauf und zeigt, welche Punkte Bestandteil des Projekts waren und welche nicht:

* Entwurf und technische Richtlinien für das Spielsystem definieren
* Aufbau der Swing-Oberfläche zur Interaktion mit dem Spiel
* Implementierung der Spiellogik mit Bewegung, Angriff und Ressourcenverwaltung
* Anbindung einer SQL-Datenbank zur Highscore-Speicherung
* Implementierung der Benutzerinteraktion
* Durchführung eines Testdurchlaufs inklusive Datenbankeintrag
* Weitere Features wie Level-Auswahl, dynamische Levelpfade oder Online-Ranking sind nicht Bestandteil des Projekts

Damit bleibt das Projekt in einem klar umrissenen Rahmen. Es liefert eine funktionierende Basisversion, die bei Bedarf in zukünftigen Arbeitsschritten erweitert werden kann.

## Projektschnittstellen

Eine zentrale Schnittstelle im Projekt ist die SQL-Datenbank, die mit der Java-Anwendung über JDBC kommuniziert. Nach jedem abgeschlossenen Spiel wird entweder der Spielname und der Highscore beim Erreichen des Levelendes in der Datenbank gespeichert. Die Datenbankinteraktionen umfassen das Hinzufügen von Datensätzen sowie das Abrufen und Anzeigen des Highscores und die Zeit die der Spieler für das jeweilige Level beanspruchte. Dies geschieht über vorbereitete SQL-Abfragen, die durch die Anwendung ausgelöst werden.

3 Projektplanung

## 3.1 Ist-Analyse

Die Ist-Analyse beschreibt die Ausgangssituation vor Beginn des Projekts.

Es lagen keine bestehenden Projektstrukturen, Quellcodes oder vorbereiteten Module vor. Die Entwicklungsumgebung musste vollständig eingerichtet werden, einschließlich:

* Installation von Java 11 SDK
* Einbindung von Swing
* Installation und Konfiguration der IDE
* Einrichtung eines lokalen MySQL-Servers zur Speicherung von Spielergebnissen
* Konfiguration eines einfachen Datenbank-Schemas mit einer Tabelle zur Speicherung von Spielername und Punktestand

Die Netzwerkanbindung diente ausschließlich für Recherche- und Updatezwecke. Das Projekt selbst lief vollständig lokal ohne externe Abhängigkeiten.

### 3.1.1 Netzwerkplan Ist-Zustand

Da es sich um ein rein lokal durchgeführtes Einzelprojekt handelte, existierte keine strukturierte Netzwerkinfrastruktur. Die gesamte Entwicklungs- und Laufzeitumgebung lief auf einem einzigen Rechner. Eine Datenbankverbindung bestand lediglich lokal zur eigenen MySQL-Instanz. Es gab keine Verbindungen zu externen Servern oder Netzwerken.

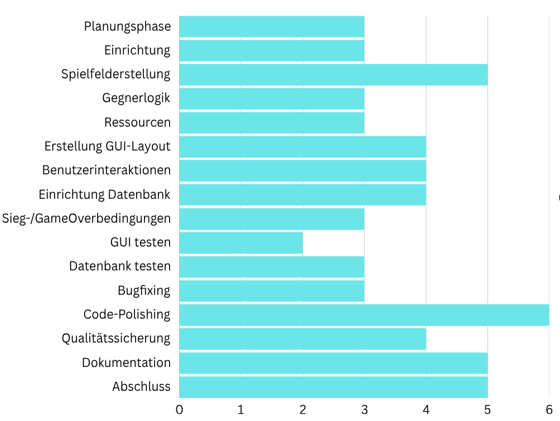
Ein Netzwerkplan im klassischen Sinne war daher nicht erforderlich.

Für zukünftige Erweiterungen, beispielsweise zur Bereitstellung eines Online-Highscores oder eines Multiplayer-Modus, müsste das Projekt um entsprechende Netzwerkschnittstellen ergänzt werden.

## 3.2 Soll-Planung

Im Rahmen der Soll-Planung wurde definiert, wie der Super Mario Bros.-Klon technisch aufgebaut sein soll, welche Spielfunktionen umzusetzen sind und welche Struktur die Anwendung erhalten soll. Der Schwerpunkt lag auf der Realisierung eines funktionalen Spielflusses, bei dem Gegner erscheinen und die Kamera dem Spieler durch das Level folgt. Das Spiel sollte modular aufgebaut sein, um spätere Erweiterungen zu erleichtern.

## 3.3 Projektablaufplan

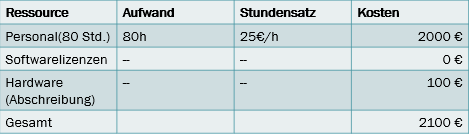


## 3.4 Personalplanung

Das Projekt wurde vollständig allein durchgeführt. Es gab keine Aufgabenverteilung im klassischen Sinne. Alle Rollen wie Planung, Entwicklung, Testing und Dokumentation wurden von einer Person übernommen. Die Personalplanung war somit sehr übersichtlich:

| **Rolle** | | **Zuständigkeit** |
| --- | --- | --- |
| Projektleitung | | Zeitplanung, Projektstrukturierung, Zieldefinition |
| Softwareentwicklung | | Implementierung der Anwendung mit Java und Swing |
| Datenbankverwaltung | | Einrichtung der SQL-Datenbank und Anbindung via JDBC |
| Test-/ Qualitätssicherung | | Funktionstests, Fehleranalyse, Korrekturmaßnahmen |
| Dokumentation | Erstellung dieser Projektdokumentation | |

## 3.5 Personalkosten



## 3.6 Nutzwertanalyse

Im Rahmen des Projekts wurde eine Nutzwertanalyse durchgeführt, um die geeignetste Programmiersprache für die Entwicklung des Super Mario Bros.-Klons zu identifizieren. Zur Auswahl standen Java, Python und JavaScript, wobei besonderes Augenmerk auf Eigenschaften gelegt wurde, die für Spieleentwicklung, insbesondere im 2D-Bereich sowie auf eine saubere Architektur und Erweiterbarkeit entscheidend sind.

### 3.6.1 Bewertungskriterien

Folgende Bewertungskriterien wurden festgelegt und gewichtet:

**Kriterium Gewichtung**

Einfachheit der Datenbankanbindung 10%

Performance für 2D-Spiele 20%

Strukturierbarkeit / OOP 30%

Cross-Plattform-Fähigkeit 10%

Einarbeitungszeit 5%

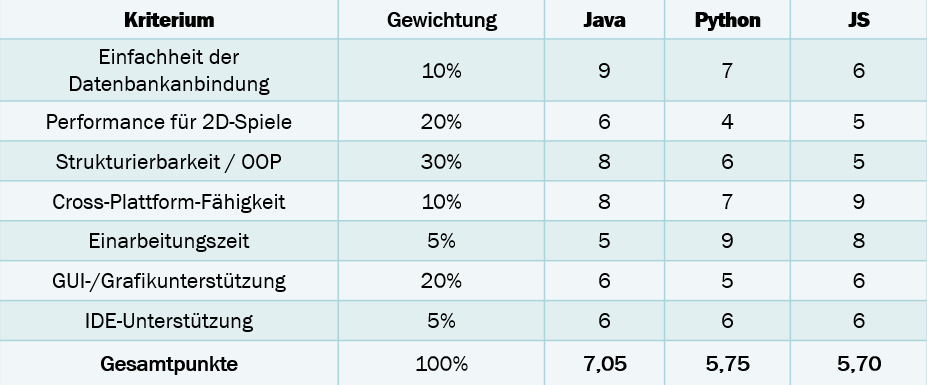
GUI-/Grafikunterstützung 20%

IDE-Unterstützung 10%

**Gesamt 100%**

### 3.6.2 Bewertung der Technologien

Die drei Technologien wurden anhand der obigen Kriterien mit Punktzahlen von 1 (sehr schlecht) bis 10 (sehr gut) bewertet:



### 3.6.3 Ergebnis und Entscheidung

Die Nutzwertanalyse zeigt, dass Java mit einem Gesamtnutzwert von 7,05 deutlich vor Python (5,75) und JavaScript (5,70) liegt. Java überzeugt vor allem in den Bereichen Performance, Objektorientierung Cross-Plattform-Fähigkeit und GUI-/Grafikunterstützung. Aufgrund dieser Bewertung fiel die Entscheidung, den Super Mario Bros.-Klon mit Java zu realisieren.

Die Auswahl bietet nicht nur eine saubere Architektur mit objektorientierten Prinzipien, sondern auch eine gute Performance und Erweiterbarkeit für künftige Features wie Highscore-Tabellen oder Mehrspieler-Modi.

# 4 Realisierung

## 4.1 Technische Umsetzung

## Die Entwicklung des Spiels erfolgte modular und Schritt für Schritt, wobei Spiellogik, Benutzeroberfläche und Datenbank klar getrennt wurden, um den Code übersichtlich und erweiterbar zu halten. Das Spiel wurde komplett in Java mit Swing für die Grafik umgesetzt, die Datenbankanbindung erfolgte über JDBC. Die Spielwelt besteht aus einem 2D-Raster mit Plattformen, Gegnern und Hindernissen. Die zentrale Steuerung regelt den Spielablauf, Kollisionen sowie Zustände wie Start und Game Over. Der Spieler steuert die Figur mit Tastatureingaben, während Gegner festgelegte Bewegungsmuster folgen. Die Oberfläche zeigt wichtige Infos wie Punkte und Leben. Highscores und benötigte Zeit werden über eine Datenbank-Anbindung gespeichert.

## 4.2 Spielmechanik

## Der Spieler steuert eine Figur durch ein seitlich scrollendes 2D-Level, weicht Hindernissen aus, besiegt Gegner durch gezieltes Springen und sammelt Punkte in Form von Münzen. Ziel ist es, das Ende des Levels zu erreichen, ohne alle Leben zu verlieren. Die Spielfigur reagiert auf Tasteneingaben zur Bewegung und zum Springen. Kollisionen mit Gegnern führen zum Verlust eines Lebens. Das Level ist so gestaltet, dass Timing und Reaktion gefordert sind, wodurch ein klassisches 2D-Plattformspiel entsteht.

## 4.3 Manuelles Testing

Da im Rahmen dieses Projekts keine automatisierten Unit-Tests durchgeführt wurden, wurde das Testing hauptsächlich durch manuelles Testen realisiert. Dabei wurden alle wesentlichen Spielmechaniken, die Benutzeroberfläche sowie die Interaktion mit der Anwendung manuell überprüft, um sicherzustellen, dass das Spiel wie erwartet funktioniert und benutzerfreundlich ist.

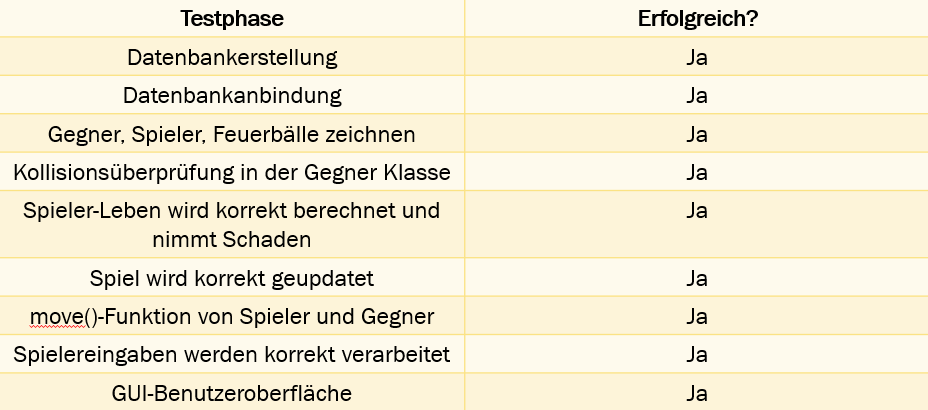
### 4.3.1 Testziele

Die Hauptziele des manuellen Testens waren:

* **Funktionalität prüfen:** Sicherstellen, dass alle Spielmechaniken (z. B. Spielerplatzierung, Gegnerplatzierung, Ressourcenverwaltung) ordnungsgemäß funktionieren.
* **Benutzeroberfläche testen:** Überprüfung der Benutzerfreundlichkeit und Lesbarkeit der GUI, Testen der Interaktivität.
* **Fehlerfreiheit:** Identifikation und Behebung von visuellen oder funktionalen Fehlern, wie z. B. falsche Berechnungen.

### 4.3.2 Durchführung der Tests

Die Tests wurden in mehreren Iterationen durchgeführt:



### 4.3.3 Fazit des manuellen Testens

### Durch manuelles Testen konnten viele Bereiche des Spiels gezielt überprüft und verbessert werden. Auch wenn diese Methode keine vollständige Abdeckung wie bei automatisierten Tests bietet, konnten wichtige Fehler frühzeitig erkannt und behoben werden. Vor allem Spielfluss, Steuerung und Stabilität ließen sich so realistisch einschätzen. Insgesamt hat sich das manuelle Testen als sinnvoll und zuverlässig erwiesen, um die Funktionalität und Benutzerfreundlichkeit des Spiels sicherzustellen.

# 5 Grafische Benutzeroberfläche

Die grafische Benutzeroberfläche (GUI) stellt den zentralen Zugangspunkt für die Interaktion der Spieler mit dem Super Mario Bros.-Klon dar. Sie wurde benutzerfreundlich, übersichtlich und modular aufgebaut, um sowohl eine einfache Bedienung als auch eine klare Trennung der Zuständigkeiten im Code zu gewährleisten.

## 5.1 Aufbau und Elemente

Die GUI wurde mit Java Swing umgesetzt. Sie gliedert sich in mehrere Hauptkomponenten:

* **Spielanzeige:**  
  Zeigt das Spielfeld, Ressourcen des Spielers (z. B. Leben, Punkte), sowie Platzierung Spieler und Gegner  
  → Klassen: MarioGame, GamePanel, Player, Enemy
* **Spielende:**  
  Nach Spielende (Sieg) erscheint ein Dialogfenster mit der erreichten Punktzahl und der Option, seinen Namen einzutragen und das Spiel zu beenden.  
  → Klassen: MarioGame, GamePanel, Database

## 5.2 Technische Umsetzung

Die GUI-Komponenten wurden nach dem Model-View-Controller (MVC)-Prinzip entworfen:

* Model: Verwaltung des Spielzustands (z. B. Spiellogik, Gegner, Spieler)
* View: Darstellung der Oberfläche (Swing-Komponenten)
* Controller: Verarbeitung der Benutzereingaben (Tastatureingaben)

Dies ermöglicht eine saubere Trennung von Darstellung und Logik sowie eine leichtere Erweiterbarkeit (z. B. neue Gegner).

# 6 Qualitätssicherung

Zur Sicherstellung der Qualität des entwickelten Super Mario Bros.-Klons wurden während des gesamten Projektverlaufs verschiedene Maßnahmen getroffen. Obwohl keine automatisierten Tests zum Einsatz kamen, wurde besonderer Wert auf Stabilität, Benutzerfreundlichkeit und Wartbarkeit gelegt. Die Qualitätssicherung lässt sich in mehrere Teilbereiche gliedern.

## 6.1 Projektstruktur und Codequalität

Bereits bei der Planung wurde darauf geachtet, das Projekt in klar getrennte Komponenten zu unterteilen. Die Spiellogik, Benutzeroberfläche und Datenverarbeitung wurden in logisch abgeschlossenen Klassen und Paketen organisiert. Diese modulare Struktur erleichterte nicht nur die Entwicklung, sondern auch die spätere Fehlersuche und Erweiterung.

Im Verlauf der Umsetzung wurde der Quellcode regelmäßig überprüft und überarbeitet. Dabei lag der Fokus auf Lesbarkeit, Wiederverwendbarkeit und sauberer Methodengestaltung. Die Einhaltung objektorientierter Prinzipien und die Verwendung aussagekräftiger Bezeichner sorgten für eine gute Wartbarkeit und Verständlichkeit des Programmcodes.

## 6.2 Benutzerfreundlichkeit

## Ein wichtiger Teil der Qualitätssicherung war die Gestaltung der Benutzeroberfläche. Sie wurde bewusst einfach und übersichtlich gehalten, damit sich auch unerfahrene Spieler schnell zurechtfinden. Elemente wie Buttons, Punktestand-Anzeige oder Steueroptionen wurden mehrfach überarbeitet, um die Bedienung so klar und intuitiv wie möglich zu machen. Dabei half auch das Feedback von Testpersonen, die Hinweise zur Verständlichkeit, Anordnung und allgemeinen Übersicht gaben. Diese Rückmeldungen wurden gezielt genutzt, um die Benutzerführung weiter zu verbessern.

## 6.3 Performance und Stabilität

Die technische Stabilität des Spiels war ein zentrales Qualitätsziel. Besonders bei hoher Spiellast, etwa bei vielen gleichzeitig aktiven Gegnern und Türmen, wurde die Performance beobachtet. Durch Optimierungen in den Zeichenroutinen und eine effiziente Ereignisverarbeitung konnte das Spiel auch in solchen Situationen flüssig laufen.

# 7 Projektabschluss

## 7.1 Übergabe

Am 27.06.2025 fand die Übergabe an Herrn Lilienthal über GitHub statt. So ist es dem Dozenten möglich gewesen das Projekt ausgiebig zu prüfen.

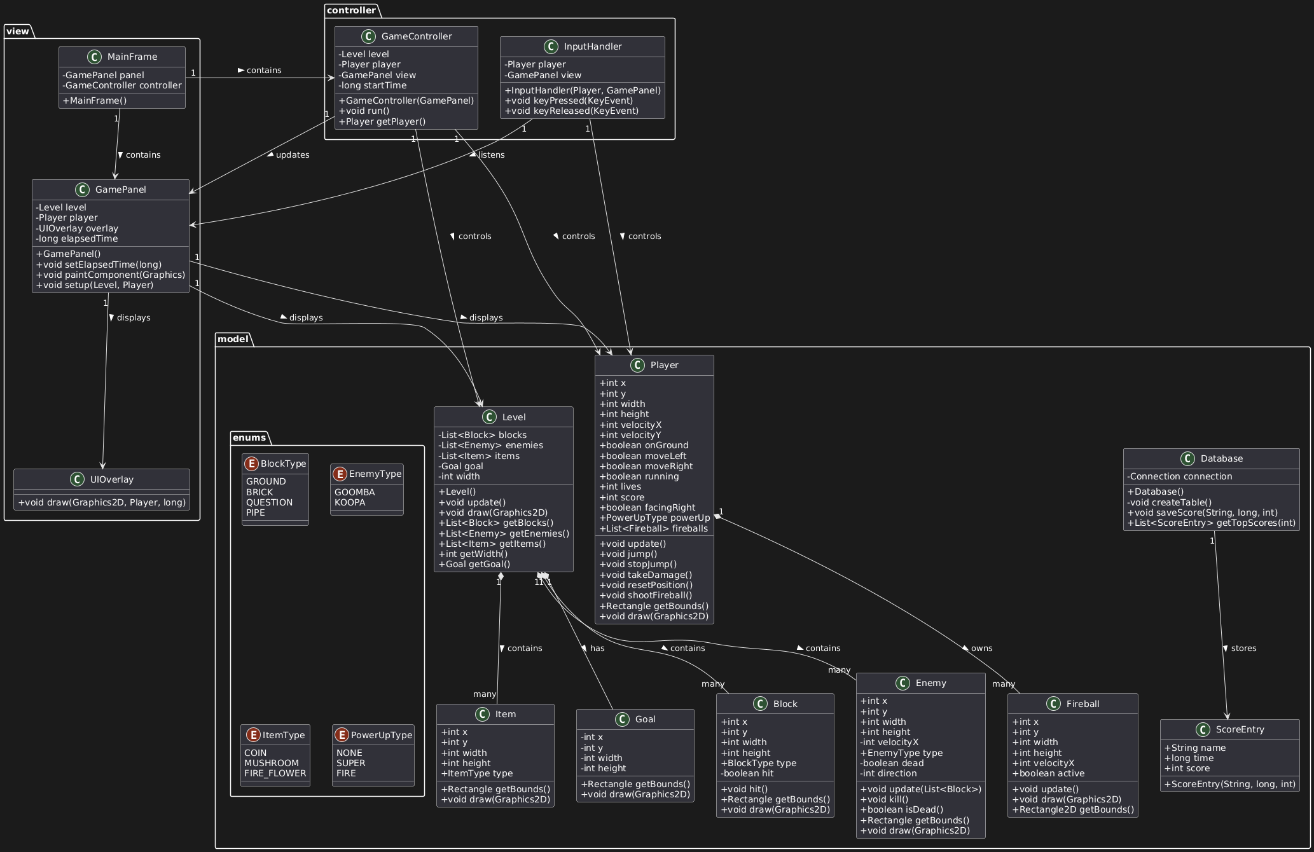
Auf diesem Wege konnte man die wichtigsten funktionalen und nicht-funktionalen Anforderungen überblicken und testen.

## 7.2 Fazit und Ausblick

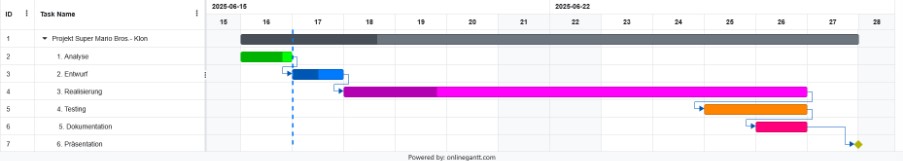
Das Projekt wurde erfolgreich abgeschlossen. Trotz Abweichungen bei den Ist-Zeiten war es möglich die angegebene Bearbeitungszeit von 80 Stunden einzuhalten. Dabei konnte ich viele Erfahrungen im Umgang mit Java und SQL-Datenbanken sammeln.

# A Anhang

## A.1 UML-Klassendiagramm



## A.2 Gantt Diagramm



## A.3 Mockup

