Abschlussprüfung Sommer 2023

Fachinformatiker für Anwendungsentwicklung

Dokumentation zur betrieblichen Projektarbeit



Entwicklung eines 2D – „Schiffe versenken“ Spiels  
zur Simulation einer Seeschlacht

Abgabedatum:

Neckargemünd,

den 25.08.2023

**Prüfungsbewerber:**

Denis Ojdanic

Martinstraße 4

89518 Heidenheim

**Ausbildungsbetrieb:**

SRH Berufsbildungswerk Neckargemünd GmbH

Im Spitzerfeld 25

69151 Neckargemünd

Inhaltsverzeichnis

## Inhaltsverzeichnis

[Inhaltsverzeichnis 2](#_Toc141104836)

[Abbildungsverzeichnis 3](#_Toc141104837)

[Tabellenverzeichnis 3](#_Toc141104838)

[Verzeichnis der Listings 4](#_Toc141104839)

[Verzeichnis der Listings 4](#_Toc141104840)

[Abkürzungsverzeichnis 4](#_Toc141104841)

[1 Einleitung 4](#_Toc141104842)

[1.1 Projektumfeld 4](#_Toc141104843)

[1.2 Projektziel 5](#_Toc141104844)

[1.3 Projektbegründung 7](#_Toc141104845)

[1.4 Projektschnittstellen 7](#_Toc141104846)

[2 Projektplanung 7](#_Toc141104847)

[2.1 Projektphasen 7](#_Toc141104848)

[2.2 Ressourcenplanung 8](#_Toc141104849)

[2.3 Entwicklungsprozess 9](#_Toc141104850)

[3 Analysephase 9](#_Toc141104851)

[3.1 Ist-Analyse 9](#_Toc141104852)

[3.2 Wirtschaftlichkeitsanalyse 9](#_Toc141104853)

[3.2.1 Make or Buy-Entscheidung 9](#_Toc141104857)

[3.2.2 Projektkosten 10](#_Toc141104858)

[3.2.3 Amortisationsdauer 10](#_Toc141104859)

[3.3 Nutzwertanalyse 10](#_Toc141104860)

[3.4 Anwendungsfälle 10](#_Toc141104861)

[3.5 Qualitätsanforderungen 11](#_Toc141104862)

[3.6 Fachkonzept 11](#_Toc141104863)

[4 Entwurfsphase 11](#_Toc141104864)

[4.1 Zielplattform 11](#_Toc141104865)

[4.2 Architekturdesign 12](#_Toc141104866)

[4.3 Entwurf der Benutzeroberfläche 12](#_Toc141104867)

[4.4 Datenmodell 12](#_Toc141104868)

[4.5 Geschäftslogik 13](#_Toc141104869)

[4.6 Maßnahmen zur Qualitätssicherung 13](#_Toc141104870)

[4.7 Pflichtenheft 13](#_Toc141104871)

[5 Implementierungsphase 14](#_Toc141104872)

[5.1 Implementierung der Datenstrukturen 14](#_Toc141104873)

[5.2 Implementierung der Weboberfläche 14](#_Toc141104874)

[5.3 Implementierung der Spiellogik 15](#_Toc141104875)

[5.3.1 Platzierung der Schiffe 15](#_Toc141104881)

[5.3.2 Anmeldung der Spieler 16](#_Toc141104888)

[5.3.3 Spielzüge 17](#_Toc141104889)

[5.3.4 Statistik 19](#_Toc141104890)

[6 Abnahmephase 20](#_Toc141104891)

[7 Einführungsphase 20](#_Toc141104892)

[8 Dokumentation 20](#_Toc141104893)

[9 Fazit 21](#_Toc141104894)

[9.1 Soll-/Ist-Vergleich 21](#_Toc141104895)

[9.2 Gelerntes 22](#_Toc141104896)

[9.3 Ausblick 22](#_Toc141104897)

[10 Quellenverzeichnis 22](#_Toc141104898)

[11 Anhang 26](#_Toc141104899)

[A1 Detaillierte Zeitplanung 26](#_Toc141104900)

[Abbildung 1: Skizze der Webseite 27](#_Toc141104901)

[A3 Programmablaufplan zur Registrierung zum Spiel 28](#_Toc141104902)

[A4 Programmablaufplan zum Spielvorgang 29](#_Toc141104903)

[A5 Oberflächenentwurf 30](#_Toc141104904)

[A6 Ereignisgesteuerte Prozesskette 31](#_Toc141104905)

[A7 Oberflächenentwürfe 31](#_Toc141104906)

[A8 Screenshots der Anwendung 32](#_Toc141104907)

[A9 Entwicklerdokumentation (Auszug) 33](#_Toc141104908)

[A10 Testfall und sein Aufruf auf der Konsole 34](#_Toc141104909)

[A11 Klasse: ComparedNaturalModuleInformation 36](#_Toc141104910)

[A12 Klassendiagramm 38](#_Toc141104911)

[A13 Benutzerdokumentation (Auszug) 39](#_Toc141104912)

**Abbildungsverzeichnis**

## Abbildungsverzeichnis

[Abbildung 1: Skizze der Webseite 22](#_bookmark56)

[Abbildung 2: Programmablaufplan zum Registrierungsvorgang zum Spiel y](#_bookmark59)

[Abbildung 3: Programmablaufplan zum Spielvorgang vi](#_bookmark60)

[Abbildung 4: Prozess des Einlesens eines Moduls vi](#_bookmark62)

**Tabellenverzeichnis**

## Tabellenverzeichnis

[Tabelle 1: Grobe Zeitplanung 2](#_bookmark13)

[Tabelle 2: Kostenaufstellung 3](#_bookmark23)

[Tabelle 3: Entscheidungsmatrix 5](#_bookmark31)

[Tabelle 4: Soll-/Ist-Vergleich 8](#_bookmark46)

[Tabelle 5: Detaillierte Zeitplanung ii](#_bookmark53)

## Verzeichnis der Listings

## Verzeichnis der Listings

[Listing 1: Testklasse xii](#_bookmark73)

[Listing 2: Klasse ComparedNaturalModuleInformation xiv](#_bookmark75)

**Abkürzungsverzeichnis**

## Abkürzungsverzeichnis

HTML Hypertext Markup Language

CSS Cascading Style Sheets

JS JavaScript

PHP PHP Hypertext Preprocessor

JSON JavaScript Object Notation

PAP Programmablaufplan

**Einleitung**

1. **Einleitung**

## Projektumfeld

Ich bin Auszubildender Fachinformatiker für Anwendungsentwicklung am SRH Berufsbildungswerk Neckargemünd GmbH. Das SRH Berufsbildungswerk Neckargemünd GmbH bildet über 900 Auszubildende in über 40 anerkannten Ausbildungsberufen aus und bietet ihnen somit die Chance, auf die Integration in den ersten Arbeitsmarkt.

Im Rahmen meines betrieblichen Abschlussprojektes bin ich …

Der Auftraggeber ist eine Förderschule mit Förderschwerpunkt körperliche und motorische Entwicklung in Heidenheim. Die Schule bietet Kindern die Möglichkeit an, einen Hauptschulabschluss zu erwerben. Diese Schule hat ca. 200 Schüler insgesamt, unterrichtet grundsätzlich aber in kleinen Klassengrößen, um besser auf die individuellen Bedürfnisse der Schüler eingehen zu können. Das Ziel ist es, die Schüler in ihrer Entwicklung in vollen Umfang zu unterstützen und sie somit auf das spätere Leben trotz motorischen Einschränkungen vorzubereiten.

Der Kontakt ist durch einen Freund meines Vaters entstanden, der Lehrer dieser Schule ist. Er hat mich bei der Schulleitung empfohlen. Diese hatte das Vorhaben bezüglich des Projekts „Schiffe versenken“ danach bei mir in Auftrag gegeben.

Das Projekt soll in Zukunft im EDV-Unterricht zum Einsatz kommen, um mit motorisch beeinträchtigten Personen an die Handkoordination anhand der Mausbewegung zu sensibilisieren. Die Demoversion wird auf einen lokalliegenden Webserver erstellt und getestet, bevor es dort in Zukunft regulär zum Einsatz kommt.

Das Projekt wird im Rahmen der betrieblichen Projektarbeit in der SRH Neckargemünd erstellt.

* + - Kurze Vorstellung des Ausbildungsbetriebs (Geschäftsfeld, Mitarbeiterzahl usw.)
    - Wer ist Auftraggeber/Kunde des Projekts?

## Projektziel

Text

Die Förderschule Ojde-Schule hat mich beauftragt, ein „Schiffe versenken“-Spiel zu entwickeln, um Kindern mit motorischen Einschränkungen den Umgang mit der Computer-Maus zu trainieren.

Das Projekt soll webbasiert als Demoversion zunächst auf einen lokalliegenden Server installiert werden. Der Grund für die webbasierte Entwicklung ist der, dass auf den Schüler-PCs keine zusätzliche Software dafür installiert werden soll. Die Schüler-PCs sind alle standardmäßig mit einem Browser ausgestattet. Somit ist die webbasierte Umsetzung am besten geeignet. Diese Demoversion ist zu Testzwecken auf 2 Spieler im Netzwerk beschränkt und kann in Zukunft erweitert werden, um mehrere Matches gleichzeitig im Netzwerk zu ermöglichen.

Es wird eine HTML-Datei erstellt, die mit JavaScript interaktiv bedienbar wird. Auf dem Server wird eine Textdatei mit Daten der Positionen der Schiffe und Schüsse abgelegt. Diese Informationen werden mit PHP über Ajax ausgelesen und im HTML dynamisch mit JavaScript und CSS grafisch dargestellt.

Es wurden Features und Regeln festgelegt, die im Programm umgesetzt werden sollen.

Folgende Features sollen implementiert werden:

* Spieler gegen Spieler
* Online über einen Webserver nutzbar
* Statistik für Züge, Trefferquoten und verlorene Schiffe

Es wurden folgende Regeln definiert:

1. Die Spielfeldgröße beträgt 10 x 10 Felder.
2. Der anzufangende Spieler soll zufällig ausgewählt werden.
3. Die Schiffe dürfen nicht aneinanderstoßen.
4. Die Schiffe dürfen nicht über Eck gebaut sein oder Ausbuchtungen besitzen.
5. Die Schiffe dürfen auch am Rand des Spielfelds liegen.
6. Die Schiffe dürfen nicht diagonal aufgestellt werden.
7. Jeder Spieler verfügt über insgesamt zehn Schiffe (in Klammern die Größe):

* ein Schlachtschiff (5 Felder)
* zwei Kreuzer (je 4 Felder)
* drei Zerstörer (je 3 Felder)
* vier U-Boote (je 2 Felder)

Mit dem Spiel „Schiffe versenken“ soll eine Verbesserung der motorischen Fähigkeiten der Schüler auf lange Sicht erzielt werden. Hierbei wird der Fokus auf die Bedienung mit der Computer Maus gelegt.

* + - Worum geht es eigentlich?
    - Was soll erreicht werden?

## Projektbegründung

Wie in 3.2.1 Make-or-Buy-Entscheidung erklärt, handelt es sich hierbei um Individualsoftware und konnte nicht für den Kunden passend gekauft werden. Trotz langen Recherchen wurde kein geeignetes Produkt für den Kunden gefunden. Somit musste das Produkt kundengerecht entwickelt werden.

## Projektschnittstellen

Damit die Daten der Spieler, wie Spielernamen, Aufstellungen der Schiffe und Spielzüge, in die Textdatei geschrieben werden können, ist es nötig, mit JS die Informationen aus dem HTML‑Dokument zu lesen. Diese werden dann über Ajax an einen PHP‑Skript gesendet, die die erhaltenen Daten dann in eine Textdatei als JSON‑String schreibt. Dies ist wichtig, da die Spieler auf die Daten entsprechend zugreifen müssen und diese Daten, wie gemachte Spielzüge des Gegners, ebenfalls wieder in das HTML‑Dokument eingetragen werden müssen.

1. **Projektplanung**

## Projektphasen

Tabelle für grobe Zeitplanung…

Tabelle 1: Grobe Zeitplanung

|  |  |
| --- | --- |
| Projektphasen | geplante Zeit |
| Kundengespräch | 1 h |
| Projektplanung | 4 h |
| GUI‑Entwurf | 4 h |
| Erstellung der GUI | 6 h |
| Implementierung der Features | 12 h |
| Implementierung der Regeln | 16 h |
| Funktionsprüfung & Qualitätskontrolle | 4 h |
| Soll‑Ist‑Vergleich | 1 h |
| Abnahme | 2 h |
| Dokumentation | 30 h |
| Summe | 80 h |

* + - In welchem Zeitraum und unter welchen Rahmenbedingungen (z.B. Tagesarbeitszeit) findet das Projekt statt?
    - Verfeinerung der Zeitplanung, die bereits im Projektantrag vorgestellt wurde.

**Beispiel**

[Tabelle 1](#_bookmark13) zeigt ein Beispiel für eine grobe Zeitplanung.

Eine detailliertere Zeitplanung ist in [Tabelle 5](#_bookmark53) in Anhang [A1](#_bookmark52) zu sehen.

|  |  |
| --- | --- |
| **Projektphase** | **Geplante Zeit** |
| Analyse | 9 h |
| Entwurf | 20 h |
| Implementierung | 30 h |
| Abnahme | 1 h |
| Einführung | 1 h |
| Dokumentation | 9 h |
| **Gesamt** | **80 h** |

**Tabelle 1: Grobe Zeitplanung**

## Ressourcenplanung

Text

* + - Detaillierte Planung der benötigten Ressourcen (Hard-/Software, Räumlichkeiten usw.).
    - Ggfs. sind auch personelle Ressourcen einzuplanen (z.B. unterstützende Mitarbeiter).
    - Hinweis: Häufig werden hier Ressourcen vergessen, die als selbstverständlich angesehen werden (z.B. PC, Büro).

## Entwicklungsprozess

Das Projekt wurde nach der Wasserfallmethode entwickelt. Die Vorteile darin liegen in der einfachen Umsetzung für die Demoversion mit einem klaren Ergebnisvorschlag, das dem Kunden präsentiert werden soll. Dadurch soll der Kunde einen konkreten Eindruck des Ergebnisses für die Vollversion bekommen und gewünschte Änderungen beziehungsweise Erweiterungen mitteilen. Somit erfolgt nach der Einführung der Demoversion eine stärkere Einbeziehung der Ojde-Schule.

1. **Analysephase**

## Ist-Analyse

Wie schon in 1.1 Projektumfeld erwähnt, ist die Ojde‑Schule eine Förderschule für Schüler mit motorischen Einschränkungen. Um die Schüler bei der Entwicklung zu unterstützen, malten die Schüler in Kunst Mandalas mit Farben nach Wunsch aus. Dies sollte ihre Koordination mit ihren Händen verbessern, um sie auf das spätere Leben trotz motorischen Einschränkungen vorzubereiten.

Durch die Entwicklung des Spiels „Schiffe versenken“ soll für das Unterrichtsfach „EDV“ eine weitere Möglichkeit der Förderung entstehen, die gezielt auf dem Umgang mit Desktop‑PCs angepasst ist. Hierbei soll für die Schüler gezielt die Koordination der Mausbewegungen verbessert werden, die später im Leben beziehungsweise im Beruf benötigt werden könnten.

Die Ojde-Schule betreibt einen lokalen Webserver, mit dem alle in der Schule befindlichen Windows 10 PCs beziehungsweise Laptops mit Mozilla Firefox und Google Chrome zugreifen können. Auf dem Schulserver ist PHP bereits installiert und dieser wird vom IT‑Lehrer verwaltet.

## Wirtschaftlichkeitsanalyse

Im Folgendem wird der wirtschaftliche Aspekt des Projektes betrachtet.



### Make or Buy-Entscheidung

Das Projekt soll kundengerecht für die Ojde-Schule entwickelt werden. Dabei wird Zugriff auf den Quellcode benötigt, um bei Bedarf gewünschte Anpassungen vornehmen zu können. Somit ist ein Kauf eines Standardproduktes nicht geeignet und muss individuell angepasst entwickelt werden.

### Projektkosten

Die Kosten der Durchführung des Projektes setzen sich aus Ressourcenkosten und Personalkosten zusammen. Der Stundensatz für Personal wird auf 70 € angesetzt. Die Sachmittelkosten betragen 40 € für Hardware und Software für den Zeitraum der Durchführung des Projektes. Sowohl die Kosten für Hard- und Software, als auch der Stundensatz wurden von der Abteilung Finanzcontrolling angegeben.

Berechnung der Gesamtpersonalkosten:

70 € / Stunde \* 80 Stunden = 5.600 €  
Somit belaufen sich die Gesamtkosten für das Projekt auf 5.640 €.

### Amortisationsdauer

Eine Amortisierung ist nicht berechenbar. Es wird lediglich durch die Entwicklung der Demoversion ein Teil der Vorarbeit geleistet. Dies hat zur Folge, dass die spätere Arbeitszeit für das finale Produkt eingespart wird.

## Nutzwertanalyse

Text (Fließtext)

* Darstellung des nicht-monetären Nutzens (z.B. Vorher-/Nachher-Vergleich anhand eines Wirtschaftlichkeitskoeffizienten).

**Beispiel**

Ein Beispiel für eine Entscheidungsmatrix findet sich in Kapitel [4.2](#_bookmark30) ([Architekturdesign](#_bookmark30)).

## Anwendungsfälle

Das Projekt soll im Schulunterricht der Ojde-Schule im Fach „EDV“ begleitend zum Einsatz kommen. Die Schüler

Bezug PAP nehmen.

* Welche Anwendungsfälle soll das Projekt abdecken?
* Einer oder mehrere interessante (!) Anwendungsfälle könnten exemplarisch durch ein Aktivitätsdiagramm oder eine EPK detailliert beschrieben werden.

**Beispiel**

Ein Beispiel für ein Use-Case-Diagramm findet sich im Anhang [A3](#_bookmark55).

## Qualitätsanforderungen

Der Fokus für Qualität wurde auf die Funktionsfähigkeit der Demoversion gelegt. Diese soll als Vorlage zur späteren Weiterentwicklung für die Vollversion dazu beitragen, die grundlegende Funktionsweise mit dem Spielablauf der Ojde-Schule ein Prototyp zu demonstrieren, der durch Kundenwünsche noch angepasst werden kann.

## Fachkonzept

Wie ich Features umsetzen will. Feature 1 setze ich so um…

* Auszüge aus dem Lastenheft/Fachkonzept, wenn es im Rahmen des Projekts erstellt wurde.
* Mögliche Inhalte: Funktionen des Programms (Muss/Soll/Wunsch), User Stories, Be- nutzerrollen

**Beispiel**

Ein Beispiel für ein Lastenheft findet sich im Anhang [A2](#_bookmark54).

1. **Entwurfsphase**

## Zielplattform

Wie in 3.1 IST‑Analyse beschrieben, betreibt die Schule einen lokalen Webserver, auf dem alle im Schulnetz befindenden Geräte mit einem Browser zugreifen können. Des Weiteren ist es erwünscht zusätzliche Installationen auf den PCs beziehungsweise Laptops zu vermeiden, wenn möglich. Somit bietet sich eine webbasierte Entwicklung an, damit die Schüler der Ojde-Schule ohne zusätzliche Software einfach über den Browser, wie Mozilla Firefox oder Google Crome, auf das Spiel zugreifen können.

Um zusätzliche Installationen zu vermeiden wird somit HTML, CSS zur Gestaltung der Spielseite eingesetzt, die durch JavaScript interaktiv bedienbar wird und über Ajax ein PHP Skript aufruft, das die Verwaltung der benötigten Daten während eines Spiels in eine Textdatei übernimmt.

## Architekturdesign

Das Spiel „Schiffe versenken“ läuft linear ab. Zunächst müssen die Spieler ihre Schiffe platzieren, sich dann in eine Textdatei registrieren lassen und können anschließend abwechselnd ihre Spielzüge tätigen. Aus diesem Grund wurde sich für eine funktionell orientierte Programmierung entschieden, jedoch mit Beihilfe der JSON Klasse Spiel, die nur als Vorlage dient, die dann durch eine Funktion als JSON-String in die Textdatei geschrieben wird. Sie speichert die nötigten Daten, die während eines Spielmatches regelmäßig benötigt werden.

## Entwurf der Weboberfläche

Zuerst wurde eine Skizze für die webbasierte Benutzeroberfläche erstellt. (siehe Anhang Abbildung 1) Darauf wurde geachtet, dass sie vollständig mit der Maus bedienbar gestaltet wird. Diese beinhaltet sowohl Textfelder für Spielernamen, als auch Spielfeldern für die jeweiligen Spieler. Ebenfalls wurde eine Schiffanzeige, sowie ein Bereich für Statusmeldungen vorgesehen.

## Datenmodell

Zur Datenspeicherung der Demoversion wurde sich aufgrund der leichten Handhabung für eine Textdatei entschieden. Diese soll für die einfachere Verarbeitung Informationen als JSON-String abspeichern. Die Textdatei soll als JSON-String Daten bezüglich des laufenden Spiels speichern.

Dabei werden sowohl die Spielernamen und die Aufstellungen beider Spieler des aktuellen Spiels benötigt. Zusätzlich muss die Information, welcher Spieler den Spielzug hat, mit abgelegt werden. Der Grund dafür ist, dass die Spieler den gegnerischen Spielzug bei sich eintragen muss. Dieser wird als „Token“ bezeichnet.

## Geschäftslogik

PAP…

* + - Modellierung und Beschreibung der wichtigsten (!) Bereiche der Geschäftslogik (z.B. mit Komponenten-, Klassen-, Sequenz-, Datenflussdiagramm, Programmablaufplan, Struktogramm, EPK).
    - Wie wird die erstellte Anwendung in den Arbeitsfluss des Unternehmens integriert?

**Implementierungsphase**

**Beispiel**

Ein Klassendiagramm, welches die Klassen der Anwendung und deren Beziehungen untereinander darstellt, kann im Anhang [A12](#_bookmark76) eingesehen werden.

Die EPK in Anhang [A6](#_bookmark61) zeigt den grundsätzlichen Ablauf beim Einlesen eines Moduls.

## Maßnahmen zur Qualitätssicherung

Während der Entwicklung wurde nach jeder erstellten Funktion ein Unittest durchgeführt, um die ordnungsmäßige Lauffähigkeit sicherzustellen. Des Weiteren wurde nach Fertigstellung des Spiels ein White-Box Test zur ganzheitlichen Funktionsüberprüfung durchgeführt. Abschließend wurde noch ein praktischer Blackbox-Test durchgeführt, um unerwartete Fehlerquellen auszuschließen. Die Demoversion soll jedoch nach der Abnahme vor Ort ebenfalls durch die Ojde-Schule getestet werden, um Feedback zur Vollversion zu erhalten.

## Pflichtenheft

Erstellen…

* + - Auszüge aus dem Pflichtenheft/Datenverarbeitungskonzept, wenn es im Rahmen des Projekts erstellt wurde.

**Beispiel**

Ein Beispiel für das auf dem Lastenheft (siehe Kapitel [3.6](#_bookmark27)) aufbauende Pflichtenheft ist im Anhang [A4](#_bookmark57) zu finden.

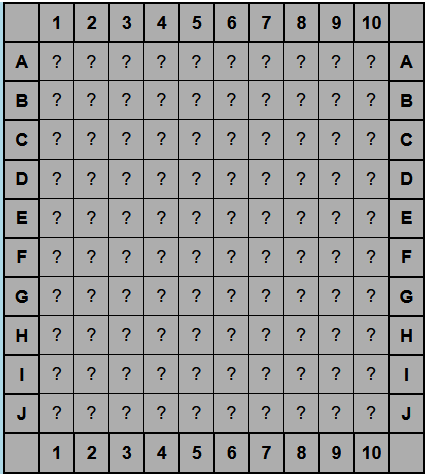
1. **Implementierungsphase**

## Implementierung der Datenstrukturen

Das in 4.4 geplante Datenmodell wurde durch eine benötigte Information erweitert. Bei der Entwicklung ist aufgefallen, dass die Spieler ebenso die Information bei einem gemachten Spielzug benötigen, ob das gegnerische Schiff getroffen wurde oder sogar versenkt wurde. Somit wurde die Information des letzten gemachten Spielzuges im Token miteingetragen. Dazu wurde zur Abgrenzung noch ein Trennzeichen eingefügt, damit die Daten des Tokens durch eine Funktion getrennt werden.

* + - Beschreibung der angelegten Datenbank (z.B. Generierung von SQL aus Modellierungswerkzeug oder händisches Anlegen), XML-Schemas usw.

## Implementierung der Weboberfläche

Es wurde für die Spielfelder des Spielers und des Gegners in HTML jeweils eine Tabelle erstellt, die mit CSS grafisch dargestellt wird. (siehe Abbildungen unten)  
 

Links ist das eigene Spielfeld zu sehen, deren belegbaren Felder mit Wasser „W“ vorbelegt sind. Rechts befindet sich das noch unbekannte gegnerische Spielfeld, deren noch unbekannten Felder mit „?“ vorbelegt sind.

Jede ansprechbare Zelle wurde mit einer ID zugewiesen, die dem Aufbau eines Koordinatensystems folgt. Sie setzt sich auf Zeile als erste Ziffer und Spalte als folgende Zahl zusammen. Der Ursprung ist das Feld „A1“, wobei das „a“ die Zeile beschriftet und die „1“ die Spalte. Die Tabellen der Spielfelder sind quadratisch bestehend aus 10x10 Feldern. Somit reicht die Beschriftung bis „J10“.

Wenn der Spieler mit dem Mauszeiger über eine Wasserfläche fährt, wird der Hintergrund dieser Fläche schwarz dargestellt. Dies dient zur Hilfe für die Orientierung, um eine genaue Auswahl der Spielfläche zu ermöglichen.

## Implementierung der Spiellogik

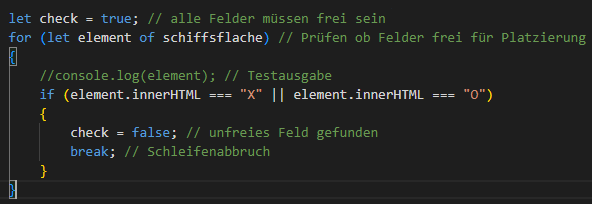


Folgend wird die Umsetzung der Spiellogik beschrieben.

### Platzierung der Schiffe

Nach dem das Design der Oberfläche der Webseite für das Spiel „Schiffe versenken“ fertiggestellt wurde, wurde zunächst die Platzierung der Schiffe ermöglicht. Dazu wurden die Aktionen des Spielers gespeichert, durch die ein Spieler Start- und Endkoordinaten des eigenen Spielfeldes des Schiffes durch einen Klick auf das gewünschte Spielfeld auswählen kann. Dabei wird einerseits überprüft, ob das Schiff gerade platziert wurde und andererseits wird die Schifflänge auf Gültigkeit überprüft. Diese muss einschließlich von 2 bis 5 reichen, da andere Längen für Schiffe laut Spielregeln nicht vorgesehen sind. (siehe Quellcodedokumentation von Ajax.js Zeile 104 – 124)

Hierbei werden die für die Platzierung benötigten Spielfeldflächen zunächst in eine Liste geschrieben. Es werden zunächst alle in der Liste enthaltenden Flächen überprüft, ob diese frei zur Platzierung des Schiffes sind. Dies ist nur der Fall, wenn keines der Flächen bereits durch ein Schiff belegt wurde oder es sich auch nicht um ein Sperrfeld handelt. (siehe Ausschnitt aus Quellcodedokumentation unten)

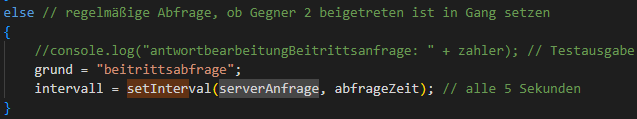


Sollte ein Feld des Schiffes doch nicht frei zur Platzierung eines Schiffes sein, wird eine Meldung an den Spieler ausgegeben, dass ihn diesbezüglich darauf hinweist. Eine Sperrfläche wird direkt um das Schiff nach der Platzierung erstellt, um zu verhindert, dass Schiffe aneinander platziert werden können. Dies dient zur Einhaltung der gewünschten Regeln für das Spiel.



### Anmeldung der Spieler

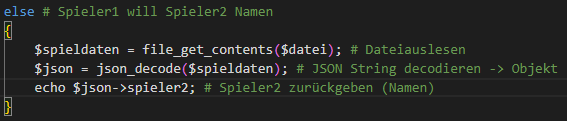
Nachdem alle Schiffe platziert wurden, können Spieler nach Eingabe des Spielernamens sich in ein Spiel eintragen lassen. Hierbei ist das Vorhandensein eines „;“ im Namen unzulässig, da die Informationen des Tokens durch dieses Zeichen getrennt werden. Dieser Eintrag erfolgt in eine Textdatei als JSON‑String. Beim Klick auf die Schaltfläche „Spiel beginnen“ wird der Spielername und die Aufstellung der Schiffe gespeichert in einem Array als JSON-String an PHP gesendet. Der erste Spieler, der sich einträgt, prüft nach einem erfolgreichen Eintrag in eine Textdatei regelmäßig, ob ein zweiter Spieler beigetreten ist. Hierbei wird der Grund der Serveranfrage auf „Beitrittsabfrage“ gesetzt und im Intervall von 5 Sekunden abgefragt. (siehe Ausschnitt aus Quellcodedokumentation unten)



Zusätzlich bekommt der erste Spieler, der sich einträgt, die ID „Spieler1“ übergeben, die nach Spielbeginn mit jedem getätigten Spielzug mitgesendet wird. Somit wird auch das Problem umgangen, dass beide Spieler mit denselben Spielernamen nicht gegeneinander spielen könnten.

Der zweite Spieler erhält nach erfolgreichem Eintrag den Namen des ersten Spielers und die Information ob er den ersten Spielzug hat. Dies wurde durch eine einfache Funktion zur Generierung einer Zufallszahl umgesetzt und wird dann als Token in die Textdatei miteingetragen. Der Token speichert sowohl Spieler-ID von dem Spieler, der gerade den Spielzug hat, als auch den zuletzt durchgeführten Spielzug. Dieser ist bei Spielbeginn nicht vorhanden und dient dazu, bei Tokenwechsel für den anderen Spieler die Information des gegnerischen Spielzuges beim eigenen Spielfeld einzutragen.

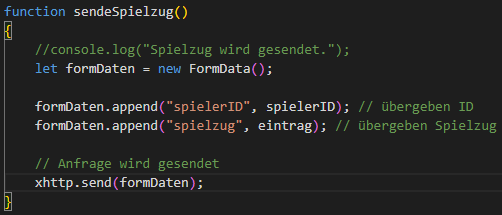
Nach dem Beitritt des zweiten Spielers wird dieser auch vom ersten Spieler durch die erwähnte PHP‑Abfrage nach Intervall der Textdatei entdeckt, wodurch dieser auch den Spielernamen seines Gegners erhält. (siehe Ausschnitt aus Quellcodedokumentation unten)



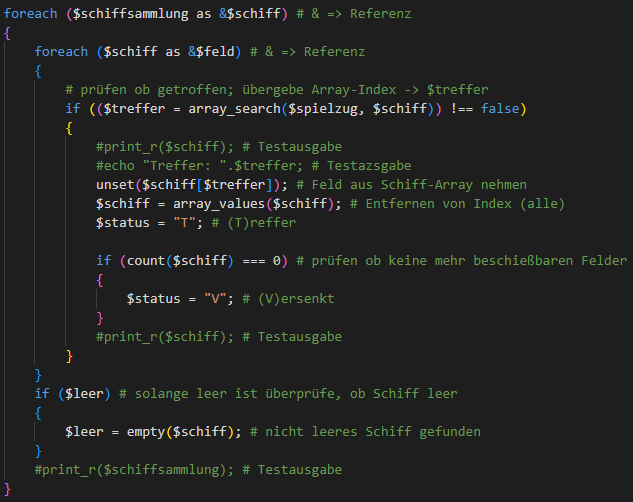
Danach überprüft er regelmäßig ob er mit dem Spielzug dran ist. Dies geschieht durch die Abfrage des Tokens in der Textdatei.

### Spielzüge

Je nachdem, welcher Spieler beim ersten Spielzug im Token eingetragen wurde, erhält dieser die Möglichkeit für den Spielzug. Beim ersten Spielzug enthält der Token noch keine Information zum letzten Spielzug, da noch kein Spielzug getätigt wurde. Der Spieler mit dem Spielzug kann dann auf ein Feld des gegnerischen Spielfelds klicken, das noch nicht beschossen wurde. Andererseits kommt eine Meldung an den Spieler, dass dieses Feld bereits beschossen wurde. Nach der Auswahl des Feldes im gegnerischen Spielfeld wird die ID des Feldes mit der Spieler ID durch eine Serverabfrage mit Grund „Spielzug“ an PHP gesendet, um diese einzutragen. (siehe Ausschnitt aus Quellcodedokumentation unten)



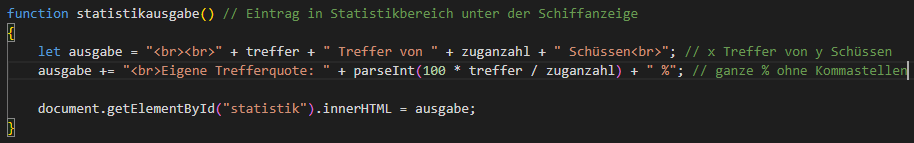
Nach dem Eintrag kommt die Rückmeldung vom PHP‑Skript mit der Information, ob es sich um einen Treffer handelt oder sogar Versenkt. Falls es sich beim Versenken um das letzte gegnerische Schiff handelt, kommt die Information, dass der Spieler gewonnen hat. Dies wird durch das PHP‑Skript festgestellt, in dem der Status in einer erstellten Funktion durch die ganze Schiffaufstellung jedes Schiff überprüft, ob ein Treffer vorliegt. (siehe Ausschnitt aus Quellcodedokumentation unten)



Nach dem Spielzug wechselt der Token und beinhaltet die neue Information des letzten Spielzuges. Dieser wird durch die regelmäßige Serveranfrage gelesen. Wird eine Änderung des Spielers festgestellt, wird der Spielzug des Gegners eingetragen, und es wird eine Statusmeldung für den Spieler ausgegeben, nach dem die Information des letzten Spielzuges verarbeitet wurde.

### Statistik

Für die Umsetzung der Statistik wurden zusätzliche Variablen benötigt, die die Spielzüge und Treffer abspeichern, woraus dann die Berechnung der Trefferquote in Prozent erfolgt. Die Ausgabe der Statistik erfolgt unter der Schiffanzeige am Ende eines Spielmatches für beide Spieler. Dies wurde in eine Funktion ausgelagert. (siehe Ausschnitt aus Quellcodedokumentation unten)



1. **Abnahmephase**

Text

* + - Welche Tests (z.B. Unit-, Integrations-, Systemtests) wurden durchgeführt und welche Ergebnisse haben sie geliefert (z.B. Logs von Unit Tests, Testprotokolle der Anwender)?
    - Wurde die Anwendung offiziell abgenommen?

**Beispiel**

Ein Auszug eines Unit Tests befindet sich im Anhang [A10](#_bookmark71). Dort ist auch der Aufruf des Tests auf der Konsole des Webservers zu sehen.

1. **Einführungsphase**

Text

* + - Welche Schritte waren zum Deployment der Anwendung nötig und wie wurden sie durchgeführt (automatisiert/manuell)?
    - Wurden Ggfs. Altdaten migriert und wenn ja, wie?
    - Wurden Benutzerschulungen durchgeführt und wenn ja, Wie wurden sie vorbereitet?

1. **Dokumentation**

dsfngsdtretre

* + - Wie wurde die Anwendung für die Benutzer/Administratoren/Entwickler dokumentiert (z.B. Benutzerhandbuch, API-Dokumentation)?
    - Hinweis: Je nach Zielgruppe gelten bestimmte Anforderungen für die Dokumentation (z.B. keine IT-Fachbegriffe in einer Anwenderdokumentation verwenden, aber auf je- den Fall in einer Dokumentation für den IT-Bereich).

**Beispiel**

Ein Ausschnitt aus der erstellten Benutzerdokumentation befindet sich im Anhang [A13](#_bookmark78).

Die Entwicklerdokumentation wurde mittels *PHPDoc* automatisch generiert. Ein beispielhafter Auszug aus der Dokumentation einer Klasse findet sich im Anhang [A9](#_bookmark69).

1. **Fazit**

Fehler:

* bei x10-Feldern korrigiert (1h)
* Sperrflächen falsch erstellt
* Ungewollte Rekursion erzeugt

## Soll-/Ist-Vergleich

Wie sich herausgestellt hat, benötigte die Implementierung der Features, dass Spieler gegeneinander spielen können mehr Zeit als die Implementierung der Regeln. Der Aufwand für die Anmeldung der Spieler, die Verarbeitung der Spielzüge für den Spielvorgang und die Auswertung der Spielzüge war im Vergleich zu den Regeln für die Platzierung der Schiffe deutlich höher. (siehe Tabelle unten)

Tabelle 2: Zeitvergleich

|  |  |  |  |
| --- | --- | --- | --- |
| Zeitrechnung | | | |
| Projektphasen | geplante Zeit | benötigte Zeit | Differenz |
| Kundengespräch | 1 h | 1 h | 0 h |
| Projektplanung | 4 h | 4 h | 0 h |
| GUI‑Entwurf | 4 h | 4 h | 0 h |
| Erstellung der GUI | 6 h | 5 h | -1 h |
| Implementierung der Features | 12 h | 20 h | +8 h |
| Implementierung der Regeln | 16 h | 9 h | -7 h |
| Funktionsprüfung & Qualitätskontrolle | 4 h | 4 h | 0 h |
| Soll‑Ist‑Vergleich | 1 h | 1 h | 0 h |
| Abnahme | 2 h | 2 h | 0 h |
| Dokumentation | 30 h | 30 h | 0 h |
| Summe | 80 h | 80 h | 0 h |

Trotz dieser Fehleinschätzung wurden alle Features und alle Regeln umgesetzt und die Demoversion fristgerecht fertiggestellt und ist voll funktionsfähig.

## Gelerntes

Text

* + - Was hat der Prüfling bei der Durchführung des Projekts gelernt (z.B. Zeitplanung, Vorteile der eingesetzten Frameworks, Änderungen der Anforderungen)?

## Ausblick

Text (max halbe Seite)

* + - Wie wird sich das Projekt in Zukunft weiterentwickeln (z.B. geplante Erweiterungen)?

**Quellenverzeichnis**

1. Quellenverzeichnis

**Internetrecherchen**

https://www.w3schools.com/jsref/jsref\_includes.asp

https://www.w3schools.com/jsref/jsref\_includes\_array.asp

https://www.w3schools.com/html/html\_table\_borders.asp

https://www.w3schools.com/js/js\_timing.asp

https://www.w3schools.com/html/html\_tables.asp

https://www.w3schools.com/html/html\_css.asp

https://www.w3schools.com/xml/ajax\_xmlhttprequest\_response.asp

https://www.w3schools.com/php/php\_oop\_constructor.asp

https://www.w3schools.com/jsref/jsref\_split.asp

https://www.w3schools.com/jsref/jsref\_slice\_array.asp

https://www.w3schools.com/jsref/jsref\_charat.asp

https://www.w3schools.com/jsref/jsref\_parseint.asp

https://www.w3schools.com/js/js\_switch.asp

https://www.w3schools.com/jsref/jsref\_push.asp

https://www.w3schools.com/howto/howto\_css\_three\_columns.asp

https://www.w3schools.com/cssref/pr\_dim\_width.php

https://www.w3schools.com/howto/howto\_css\_aspect\_ratio.asp

https://www.w3schools.com/css/css\_dimension.asp

https://www.w3schools.com/jsref/jsref\_keys.asp

https://www.w3schools.com/js/js\_json\_arrays.asp

https://www.w3schools.com/js/js\_json.asp

https://www.w3schools.com/php/php\_oop\_classes\_objects.asp

https://www.w3schools.com/php/php\_json.asp

https://www.w3schools.com/php/func\_math\_rand.asp

https://www.w3schools.com/jsref/met\_win\_clearinterval.asp

https://www.w3schools.com/php/func\_var\_empty.asp

https://www.w3schools.com/php/func\_string\_explode.asp

https://www.w3schools.com/php/php\_looping\_foreach.asp

https://www.w3schools.com/php/func\_array\_replace.asp

https://www.w3schools.com/php/func\_array\_filter.asp

https://www.w3schools.com/jsref/jsref\_foreach.asp

https://www.w3schools.com/jsref/jsref\_filter.asp

https://www.w3schools.com/jsref/jsref\_obj\_array.asp

https://www.w3schools.com/js/js\_arrays.asp

https://www.php.net/manual/en/function.empty.php

https://www.php.net/manual/en/function.array-filter.php

https://www.php.net/manual/en/function.getrandmax.php

https://www.php.net/manual/en/function.rand.php

https://www.php.net/manual/en/functions.returning-values.php

https://www.php.net/manual/en/function.file-put-contents.php

https://www.php.net/manual/en/function.json-encode.php

https://www.php.net/manual/en/function.json-decode.php

https://www.php.net/manual/de/function.isset.php

https://www.php.net/manual/en/language.types.object.php

https://www.php.net/manual/en/function.json-encode.php

https://www.php.net/manual/en/function.json-decode.php

https://www.php.net/manual/en/function.file-exists.php

https://www.php.net/manual/en/language.oop5.magic.php

https://www.php.net/manual/en/language.oop5.visibility.php

https://www.php.net/manual/de/function.explode.php

https://www.php.net/manual/de/control-structures.foreach.php

https://www.php.net/manual/en/control-structures.for.php

https://www.php.net/manual/en/function.array-replace.php

https://www.geeksforgeeks.org/how-to-create-equal-width-table-cell-using-css/

https://www.geeksforgeeks.org/javascript-array-values-method/

https://linuxhint.com/javascript-remove-index-from-array/

https://linuxhint.com/enable-disable-input-fields-using-javascript/

https://stackoverflow.com/questions/72321480/how-to-loop-through-and-reference-with-an-array-in-an-array

https://stackoverflow.com/questions/52163894/javascript-reference-in-for-of-loop

https://stackoverflow.com/questions/17511273/how-to-replace-elements-in-array-with-elements-of-another-array

https://stackoverflow.com/questions/3954438/how-to-remove-item-from-array-by-value

https://stackoverflow.com/questions/41877868/php-a-way-to-get-property-without-getter

https://stackoverflow.com/questions/2156712/how-to-float-3-divs-side-by-side-using-css

https://stackoverflow.com/questions/58202591/three-divs-with-different-width-side-by-side- those-left-and-right-fixed

https://stackoverflow.com/questions/5445491/height-equal-to-dynamic-width-css-fluid-layout

https://stackoverflow.com/questions/21799852/css-width-same-as-height

https://stackoverflow.com/questions/11243075/css-scale-height-to-match-width-possibly-with-a-formfactor

https://stackoverflow.com/questions/22893866/css-dynamically-calculate-width

https://stackoverflow.com/questions/59696160/calc-pixel-height-dynamically-based-on-https://stackoverflow.com/questions/25990938/how-to-remove-keys-from-php-array

https://stackoverflow.com/questions/2295496/convert-array-to-json

https://stackoverflow.com/questions/10525744/css-table-cell-equal-width

https://stackoverflow.com/questions/10525744/css-table-cell-equal-width

https://stackoverflow.com/questions/7693224/how-do-i-right-align-div-elements

https://stackoverflow.com/questions/50936764/storing-key-value-pairs-in-an-array-in-javascript

https://stackoverflow.com/questions/1144705/best-way-to-store-a-key-value-array-in-javascript

https://stackoverflow.com/questions/42526032/how-to-find-if-element-with-specific-id-exists-or-not

https://stackoverflow.com/questions/9329446/loop-for-each-over-an-array-in-javascript

https://stackoverflow.com/questions/12504042/what-is-a-method-that-can-be-used-to-increment-letters

https://dev.to/dillionmegida/arraysplice-for-removing-replacing-or-adding-values-to-an-array-1k6c

https://www.tutorialrepublic.com/faq/how-to-delete-php-array-element-by-value-not-key.php

https://www.reddit.com/r/PHP/comments/kkrzks/getterssetters\_vs\_public\_properties/

https://www.beberlei.de/post/building\_an\_object\_model\_\_no\_setters\_allowed

https://php.budgegeria.de/frgcebc

https://coding-champ.com/tutorials/php/getters-and-setters

https://www.daniweb.com/programming/web-development/threads/100792/pass-a-2d-array-with-ajax

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Object

https://softwareengineering.stackexchange.com/questions/321534/what-json-structure-to-use-for-key-value-pairs

https://www.blogseite.com/html-div-nebeneinander-anordnen/

https://www.computerhilfen.de/info/css-tricks-3-divs-zentriert-nebeneinander-anzeigen-ohne-float.html

https://www.thesitewizard.com/css/make-table-cells-same-size.shtml

https://blog.hubspot.com/website/center-div-css

https://codedamn.com/news/frontend/use-css-to-put-div-side-by-side

dynamic-width-and-ratio-with-css

https://support.pega.com/question/how-calculate-dynamic-width-css-code-based-screen-resolution

https://mademyday.de/height-equals-width-with-pure-css/

https://www.cssportal.com/css-properties/size.php

https://www.helpster.de/tabulator-in-html-erzeugen-so-gelingt-s\_123496

https://www.lima-city.de/thread/html-text-mit-tabulator-abstand

https://www.home.unix-ag.org/juergen/selfhtml/absatz.html

https://sentry.io/answers/how-do-i-add-a-tab-space-instead-of-multiple-non-breaking-spaces/

https://forum.freecodecamp.org/t/hover-effect-not-working/592288

**Anhang**

1. Anhang

## A1 Detaillierte Zeitplanung

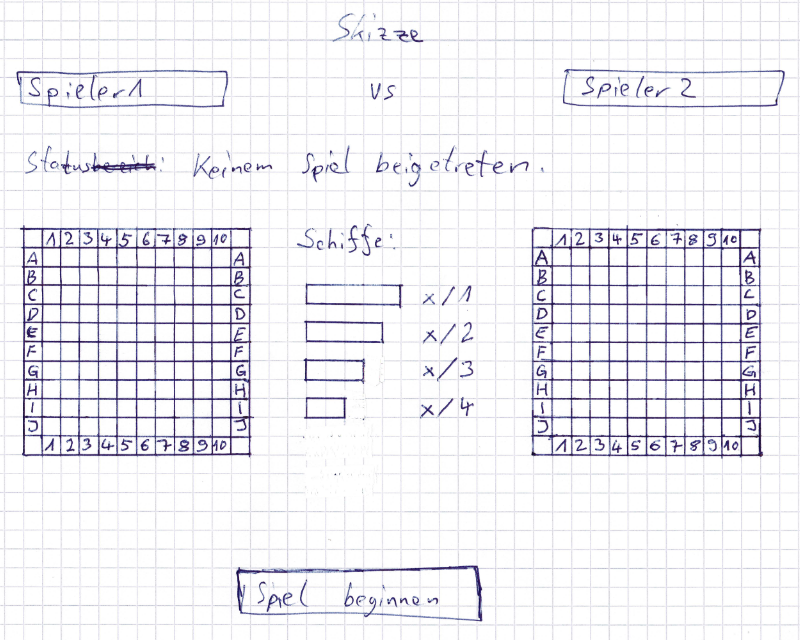
|  |  |  |  |
| --- | --- | --- | --- |
| **Analysephase** |  |  | **9 h** |
| 1. Analyse des Ist-Zustands |  | 3 h |  |
| 1.1. Fachgespräch mit der EDV-Abteilung | 1 h |  |  |
| 1.2. Prozessanalyse | 2 h |  |  |
| 2. „Make or buy“-Entscheidung und Wirtschaftlichkeitsanalyse |  | 1 h |  |
| 3. Erstellen eines Use-Case-Diagramms |  | 2 h |  |
| 4. Erstellen des Lastenhefts mit der EDV-Abteilung |  | 3 h |  |
| **Entwurfsphase** |  |  | **20 h** |
| 1. Prozessentwurf |  | 3 h |  |
| 2. Datenbankentwurf |  | 3 h |  |
| 2.1. ER-Modell erstellen | 2 h |  |  |
| 2.2. Konkretes Tabellenmodell erstellen | 1 h |  |  |
| 3. Erstellen von Datenverarbeitungskonzepten |  | 4 h |  |
| 3.1. Verarbeitung der CSV-Daten | 1 h |  |  |
| 3.2. Verarbeitung der SVN-Daten | 1 h |  |  |
| 3.3. Verarbeitung der Sourcen der Programme | 2 h |  |  |
| 4. Benutzeroberflächen entwerfen und abstimmen |  | 2 h |  |
| 5. Erstellen eines UML-Komponentendiagramms der Anwendung |  | 4 h |  |
| 6. Erstellen des Pflichtenhefts |  | 4 h |  |
| **Implementierungsphase** |  |  | **30 h** |
| 1. Anlegen der Datenbank |  | 1 h |  |
| 2. Umsetzung der HTML-Oberflächen und Stylesheets |  | 5 h |  |
| 3. Programmierung der PHP-Module für die Funktionen |  | 23 h |  |
| 3.1. Import der Modulinformationen aus CSV-Dateien | 2 h |  |  |
| 3.2. Parsen der Modulquelltexte | 3 h |  |  |
| 3.3. Import der SVN-Daten | 2 h |  |  |
| 3.4. Vergleichen zweier Umgebungen | 4 h |  |  |
| 3.5. Abrufen der von einem zu wählenden Benutzer geänderten Module | 3 h |  |  |
| 3.6. Erstellen einer Liste der Module unter unterschiedlichen Aspekten | 5 h |  |  |
| 3.7. Anzeigen einer Liste mit den Modulen und geparsten Metadaten | 3 h |  |  |
| 3.8. Erstellen einer Übersichtsseite für ein einzelnes Modul | 1 h |  |  |
| 4. Nächtlichen Batchjob einrichten |  | 1 h |  |

|  |  |  |
| --- | --- | --- |
| **Abnahmetest der Fachabteilung** |  | **1 h** |
| 1. Abnahmetest der Fachabteilung | 1 h |  |
| **Einführungsphase** |  | **1 h** |
| 1. Einführung/Benutzerschulung | 1 h |  |
| **Erstellen der Dokumentation** |  | **9 h** |
| 1. Erstellen der Benutzerdokumentation | 2 h |  |

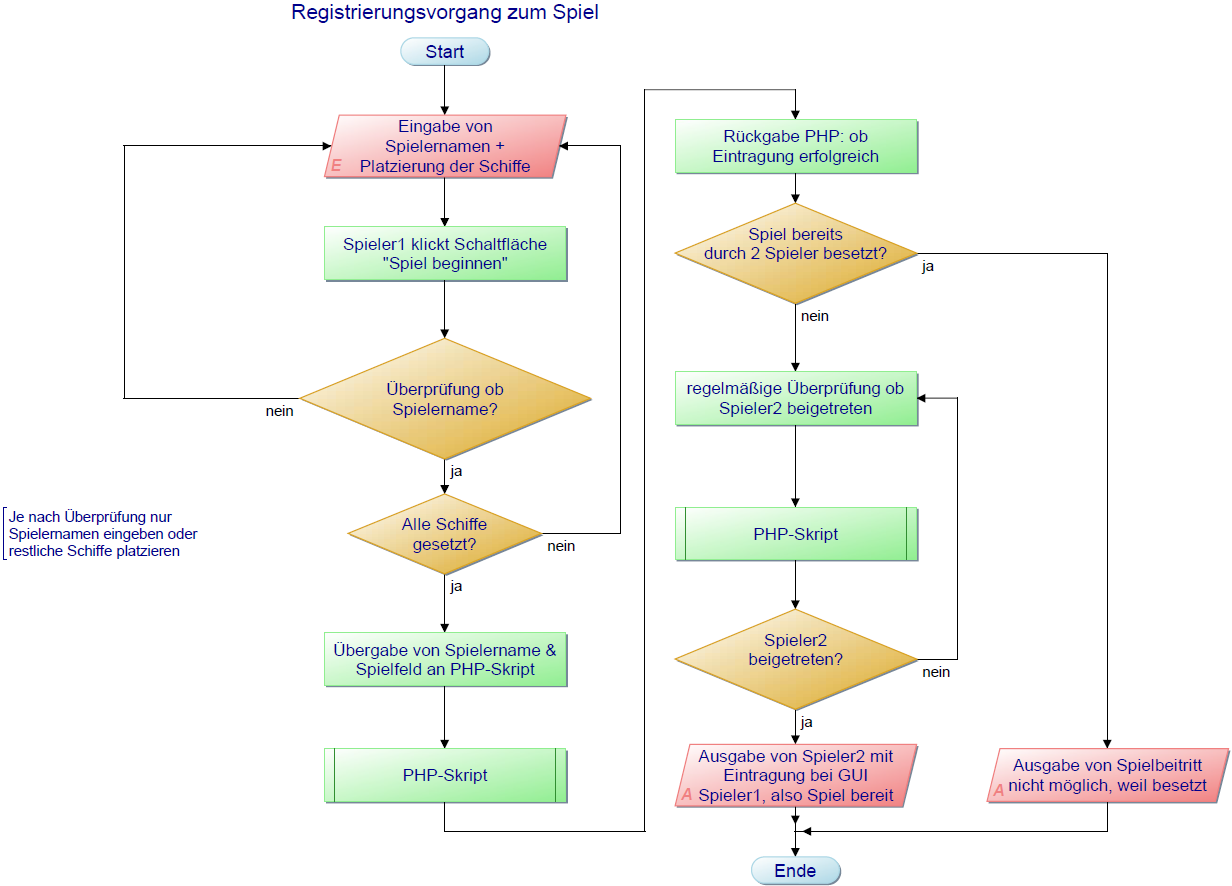
|  |  |  |
| --- | --- | --- |
| 2. Erstellen der Projektdokumentation |  | 6 h |
| 3. Programmdokumentation |  | 1 h |
| 3.1. Generierung durch *PHPdoc* | 1 h |  |
| **Gesamt** |  | **80 h** |

**Tabelle 5: Detaillierte Zeitplanung**

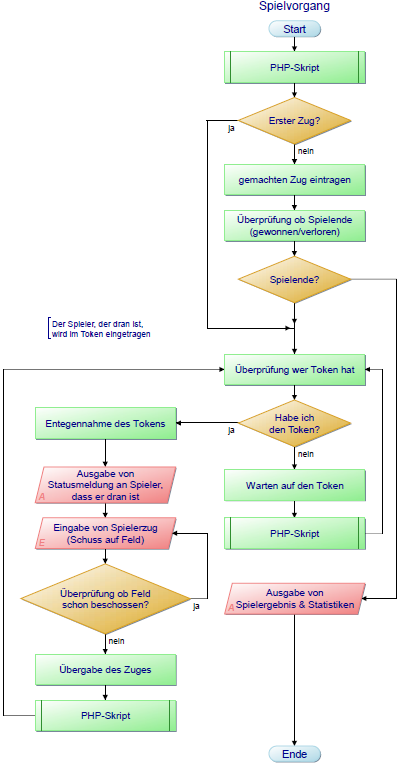
## Abbildung 1: Skizze der Webseite



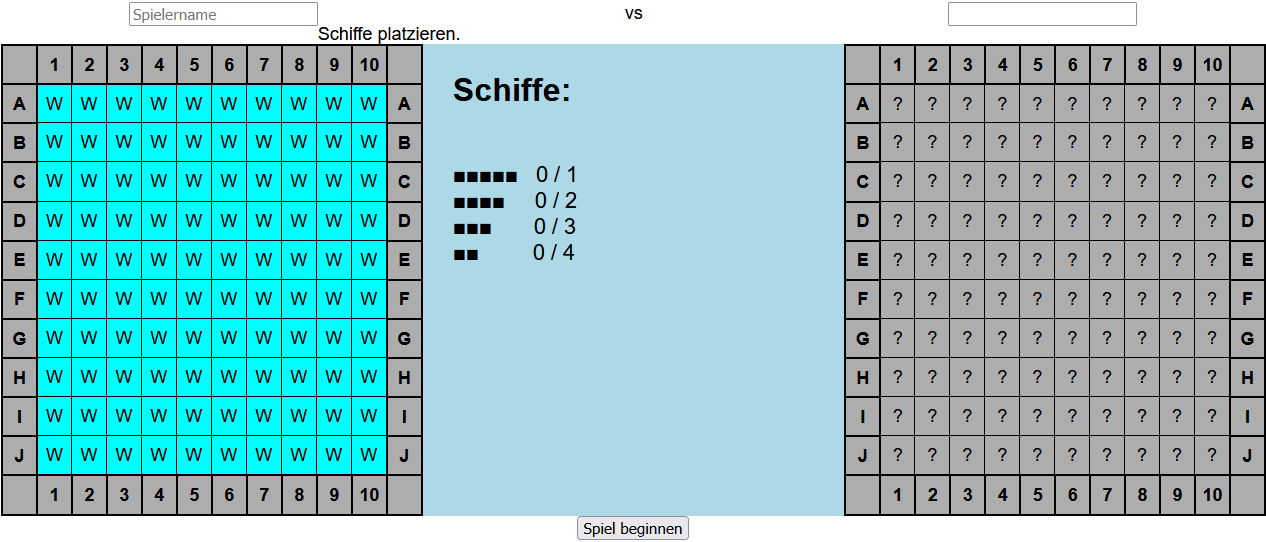
## A3 Programmablaufplan zur Registrierung zum Spiel



## A4 Programmablaufplan zum Spielvorgang

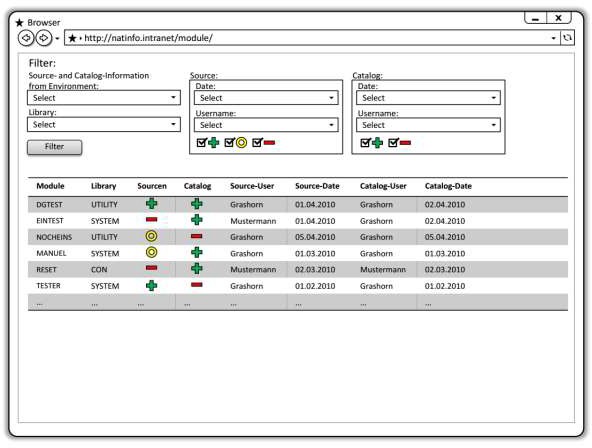


## A5 Oberflächenentwurf

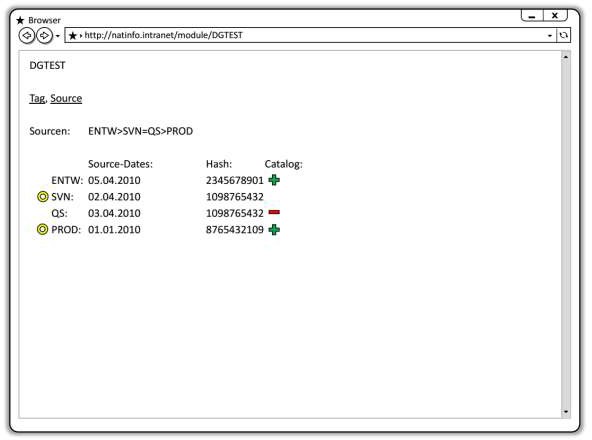
****

## A6 Ereignisgesteuerte Prozesskette

## A7 Oberflächenentwürfe



**Abbildung 5: Liste der Module mit Filtermöglichkeiten**



**Abbildung 6: Anzeige der Übersichtsseite einzelner Module**

## A8 Screenshots der Anwendung

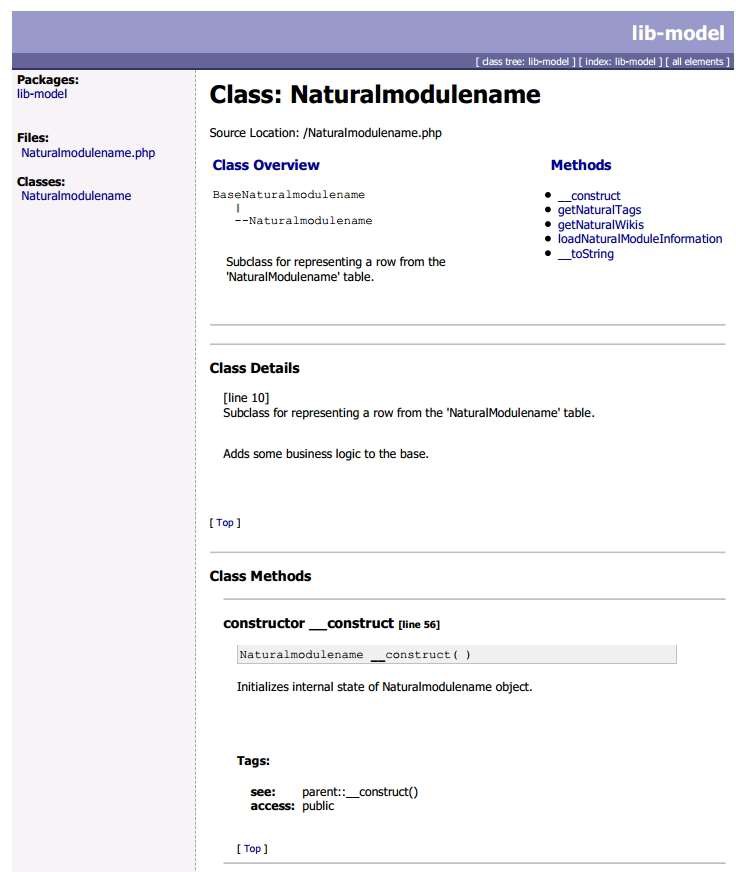


**Abbildung 7: Anzeige und Filterung der Module nach Tags**



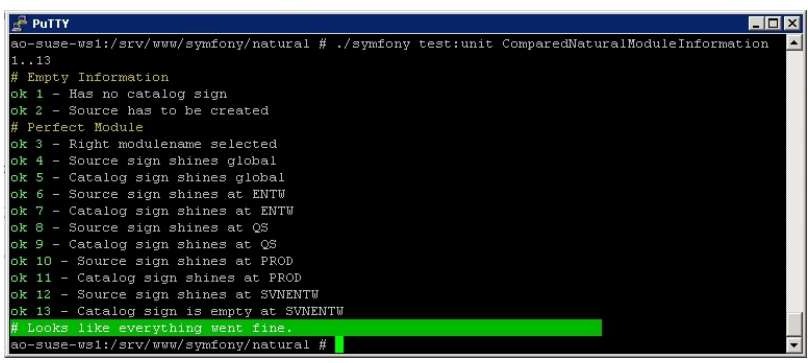
**Abbildung 8: Liste der Module mit Filtermöglichkeiten**

## A9 Entwicklerdokumentation (Auszug)



**Abbildung 9: Auszug aus der Entwicklerdokumentation mit *PHPDoc***

## A10 Testfall und sein Aufruf auf der Konsole



**Abbildung 10: Aufruf des Testfalls auf der Konsole**

$t->comment('Empty Information');

$emptyComparedInformation = new ComparedNaturalModuleInformation(array());

$t->is($emptyComparedInformation->getCatalogSign(), ComparedNaturalModuleInformation::EMPTY\_SIGN, 'Has no catalog sign');

$t->is($emptyComparedInformation->getSourceSign(), ComparedNaturalModuleInformation::SIGN\_CREATE, 'Source has to be created');

$t->comment('Perfect Module');

$criteria = new Criteria();

$criteria->add(NaturalmodulenamePeer::NAME, 'SMTAB');

$moduleName = NaturalmodulenamePeer::doSelectOne($criteria);

$t->is($moduleName->getName(), 'SMTAB', 'Right modulename selected');

$comparedInformation = $moduleName->loadNaturalModuleInformation();

$t->is($comparedInformation->getSourceSign(), ComparedNaturalModuleInformation::SIGN\_OK, 'Source sign shines global');

$t->is($comparedInformation->getCatalogSign(), ComparedNaturalModuleInformation::SIGN\_OK, 'Catalog sign shines global');

$infos = $comparedInformation->getNaturalModuleInformations(); foreach($infos as $info) {

$env = $info->getEnvironmentName();

$t->is($info->getSourceSign(), ComparedNaturalModuleInformation::SIGN\_OK, 'Source sign shines at ' . $env);

if($env != 'SVNENTW') {

$t->is($info->getCatalogSign(), ComparedNaturalModuleInformation::SIGN\_OK, 'Catalog sign shines at ' . $info->getEnvironmentName());

} else {

$t->is($info->getCatalogSign(), ComparedNaturalModuleInformation::EMPTY\_SIGN, 'Catalog sign is empty at ' . $info->getEnvironmentName());

}

}

**Listing 1: Testklasse**

## A11 Klasse: ComparedNaturalModuleInformation

Kommentare und simple Getter/Setter werden nicht gezeigt.

class ComparedNaturalModuleInformation { const EMPTY\_SIGN = 0;

...

const SIGN\_ERROR = 5;

private $naturalModuleInformations = array(); public static function environments() {

return array("ENTW", "SVNENTW", "QS", "PROD");

}

public static function signOrder() {

return array(self::SIGN\_ERROR, self::SIGN\_NEXT\_STEP, self::SIGN\_CREATE\_AND\_NEXT\_STEP, self::SIGN\_CREATE,

self::SIGN\_OK);

}

public function construct(array $naturalInformations) {

$this->allocateModulesToEnvironments($naturalInformations);

$this->allocateEmptyModulesToMissingEnvironments();

$this->determineSourceSignsForAllEnvironments();

private function allocateModulesToEnvironments(array

$naturalInformations) {

foreach ($naturalInformations as $naturalInformation) {

$env = $naturalInformation->getEnvironmentName(); if(in\_array($env, self::environments())) {

$this->naturalModuleInformations[array\_search($env, self::environments())] = $naturalInformation;

}

}

}

private function allocateEmptyModulesToMissingEnvironments() { if(array\_key\_exists(0, $this->naturalModuleInformations)) {

$this->naturalModuleInformations[0]-

>setSourceSign(self::SIGN\_OK);

}

for($i = 0;$i < count(self::environments());$i++) { if(!array\_key\_exists($i, $this-

>naturalModuleInformations)) {

$environments = self::environments();

$this->naturalModuleInformations[$i] = new EmptyNaturalModuleInformation($environments[$i]);

$this->naturalModuleInformations[$i]-

>setSourceSign(self::SIGN\_CREATE);

}

}

}

private function containsSourceSign($sign) {

foreach($this->naturalModuleInformations as $information) { if($information->getSourceSign() == $sign) {

return true;

}

}

return false;

private function containsCatalogSign($sign) {

foreach($this->naturalModuleInformations as $information) { if($information->getCatalogSign() == $sign) {

return true;

}

}

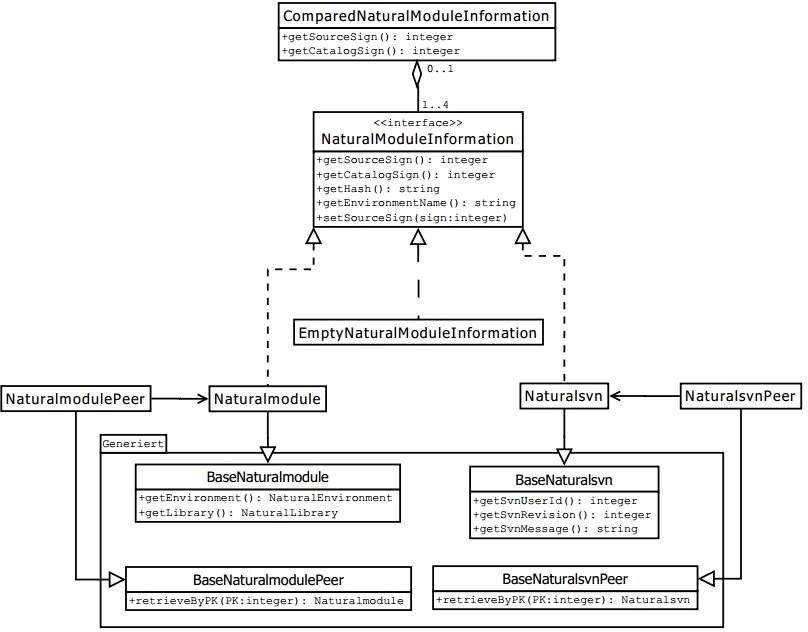
return false;

}

}

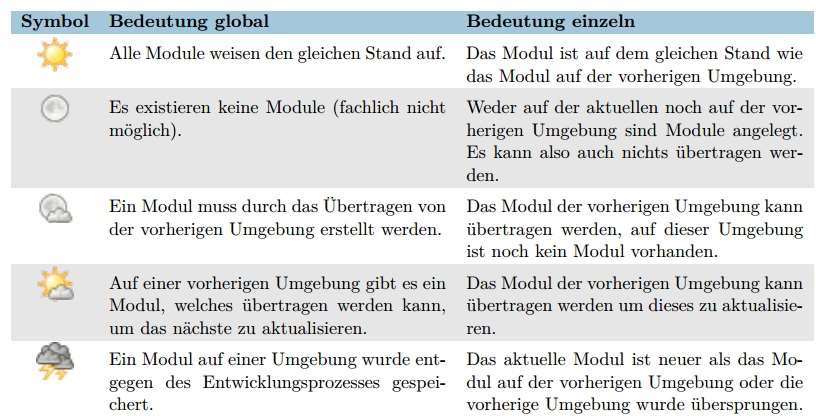
**Listing 2: Klasse ComparedNaturalModuleInformation**

## A12 Klassendiagramm



**Abbildung 11: Klassendiagramm**

## A13 Benutzerdokumentation (Auszug)



**Abbildung 12: Auszug aus der Benutzerdokumentation**