2014

| | |
|---|---|
| **Autor/en:** | Kodras, Özsoy, Scholz, Vogt |
| **Klasse:** | 4AHIT |
| **Letzte Änderung:** | 05.06.2014 |
| **Version:** | 2.0 |

# Protokoll


Protokollierung der einfachen Scheduling-Software mit allen benötigten Funktionen für ein benutzerfreundliches und einfach bedienbares Terminvereinbarungssystem.

# Inhaltsverzeichnis

# Aufgabenstellung

## Terminvereinbarungssystem

update 25.04.14

Es soll ein kollaboratives Terminvereinbarungssystem (ähnlich wie Doodle) erstellt werden, in dem sich Benutzer koordinieren können. Folgende Anforderungen sollen dabei erfüllt werden:

### Benutzer

- Neue Benutzer können sich registrieren
- Existierende Benutzer können sich einloggen
- Man kann nach registrierten Benutzern im System suchen (über ihren Namen).
- Ein Benutzer kann gleichzeitig ein Organisator und Teilnehmer sein.
- Jeder Benutzer kann sich die Events, die er organisiert, oder Events, an denen er teilnimmt, anzeigen lassen.

### Organisator

- ist ein Benutzer, der Events mit Namen und mehreren Termin- und Zeitvorschlägen erstellt und die Einladungen an einige registrierte Benutzer schickt
- darf den Namen, die Termine und Zeiten eines Events ändern, aber nur bevor sich einer der Benutzer zu dem Event angemeldet hat
- darf neue Benutzer zu seinen Events zusätzlich einladen
- darf eingeladene Benutzer wieder löschen, bevor sich diese zu dem Event angemeldet haben
- darf die Events jederzeit löschen
- darf zu seinen Events Kommentare posten
- darf Kommentare zu seinen Events löschen (auch die von anderen Benutzern)
- Nachdem sich alle Benutzer zu einer Einladung angemeldet haben, darf der Organisator einen fixen Termin festlegen.

### Teilnehmer

- wählt aus den vorgeschlagenen Terminen und Zeiten eines Events (eine Checkbox pro Zeitvorschlag reicht)
- darf seine Wahl ändern, bis ein fixer Termin existiert
- darf Kommentare zu Events, zu denen er eingeladen ist, posten

### Notifications

- Ein Teilnehmer wird über jede neue/editierte/gelöschte Eventeinladung notifiziert.
- Weiteres wird ein Teilnehmer notifiziert, sobald ein fixer Termin für ein Event festgelegt wird.
- Ein Organisator wird notifiziert, sobald alle Teilnehmer einen Termin gewählt haben.
- Wenn ein Benutzer zur Zeit einer Notification offline ist, darf diese nicht verloren gehen. Der Benutzer bekommt alle seine versäumten Notifications, sobald er online kommt.

### Events

Es kann zwei Arten von Events geben:

1. Events, bei denen sich die Teilnehmer auf (möglichst) einen Termin einigen sollen (Standardfall). Der Organisator legt letztendlich einen fixen Termin fest.
2. Events, bei denen jeweils nur ein Teilnehmer pro Termin erlaubt ist (z.B. für Elternsprechtag). Der Organisator muss jede Teilnehmer/Termin-Kombination fixieren.

## Aufgabenstellung

Entwickeln Sie ein GUI-Programm, welches das Terminvereinbarungssystem realisiert. Bei der Abgabe müssen Sie die Aufgabe auf mindestens drei Rechnern (mit mehreren gleichzeitig gestarteten Clients) präsentieren.

Beim Starten des Programms müssen der gewünschte Benutzername und die Netzwerkadresse des Servers angegeben werden (kein Passwort erforderlich). Die Registrierung kann automatisch bei der ersten Anmeldung erfolgen.

Achten Sie bei der Implementierung auf die transaktionale Sicherheit. Überlegen Sie sich Situationen, in denen z.B. ein Benutzer versucht, eine Terminwahl zu einem in der Zwischenzeit gelöschten Event zu realisieren. Ihr Programm sollte auf solche und ähnliche Situationen entsprechend reagieren.

Beachten Sie bei der Implementierung, dass die Kommentare in derselben Reihenfolge aufgelistet werden müssen, in der diese von den einzelnen Benutzern abgeschickt wurden. Sie müssen sich auch Gedanken über die Persistenz der Informationen machen. Wenn die Serverinstanz herunterfährt, muss der gesamte Inhalt dauerhaft abgelegt worden sein.

Es reicht ein einfaches, aber funktionales GUI. Sie dürfen (sollten!) dafür Frameworks einsetzen.

### Vorgehensweise

Es sind die Meta-Regeln zu beachten. Dabei ist zu beachten, dass nur durch eine obligatorische Design-Review durch die unterrichtenden Lehrkräfte, das gewählte Design (realisiert und vorgestellt mittels UML Diagrammen) verwendet werden darf. Nachträgliche Änderungen müssen durch einen Change-Request genehmigt werden. Diese müssen in eine Feature/Requirements Liste münden, die z.B. durch User-Stories definiert werden können. Zu bedenken sind auch nicht-funktionale Anforderungen an das System, wie z.B. die Anzeigegeschwindigkeit der ersten Termine und Kommentare.

Des Weiteren sind Programmier-Teams verpflichtend. Diese sind durch eine/n Tester/in und eine/n Programmierer/in definiert. Angenommene Tasks der einzelnen Stories werden gleichzeitig(!) vom Tester und Programmierer behandelt, wobei der Tester die Anforderungen in z.B. Unit-Tests und der Programmierer in den entsprechenden Codeteilen implementiert. Dabei soll sichergestellt sein, dass sofort geeignete Testfälle den gerade eben implementierten Code auf dessen Funktionstüchtigkeit überprüfen.

Integrations- und Systemtests sind verpflichtend. Dabei sind in diesem Fall auch automatisierte GUI-Tests zu verwirklichen. Der Testbericht im Protokoll muss auch eine kontinuierliche Verbesserung der zu erzielenden Story Points ersichtlich machen.

## Termine

8.5.2014 Ads: Design Review - Deadline für Design der Applikation!
4.6.2014 23:55: Deadline Abgabe (fix - keine Verlängerung möglich!)
5.6.2014: Abnahme Interview (15min / Gruppe)

Dazwischen müssen die Teamleader wöchentlich einen Review Termin selbstständig wahrnehmen.

## Bewertungskriterien

- Dokumentation
- Design
- Implementierung
- Funktionalität
- Tests/System

## Arbeitsaufteilung mit Aufwandschätzung

| Zuständige Person(en) | Task | Beschreibung | Geschätzte Zeit in h |
|---|---|---|---|
| Kodras | GUI-Mockups | GUI-Mockup für das Client Programm (am besten mit https://moqups.com/), soll alle wichtigen Funktionen enthalten. 3 Pros und Cons | 4 |
| Vogt | Use-Case-Diagram | Requirementsanalyse der Angabe anhand von USE-Case-Diagrammen und der dazugehörigen Beschreibung (erweiterte Punkteliste der Angabe) | 3 |
| Özsoy | Statemachine-Diagram | Erstellung eines Zustandsdiagrammes für SigmaScheduler in ASTAH und zusätzliches Exportieren des ASTAH-Files als Bilddatei | 2 |
| Özsoy | Dokumentation | Erstellung, Design, Zusammenführung aller Grafiken und Erweiterung/Ergänzung der Dokumentation unter Beachtung der Meta-Regeln | 2 |
| Scholz | Applikationsaufteilung | Aufteilung der Anwendung in Applikationen (Client/Server) und den verwendeten Technologien | 3 |
| Scholz | Teammanagement | Aufteilung und Beschreibung der Arbeitspakete für die Teammitglieder | 1 |
| Kodras, Vogt | UML-Export | Exportieren des ASTAH-UML-Diagramms als JAVA-Dateien | ½ |
| Özsoy | Technologiebeschreibung | Technologiebeschreibung von Hibernate, Vaadin, Glassfish-Server und MySQL | 2 |
| Kodras, Vogt | Hibernate-Mapping | Mappen der zu persistierenden Klassen | 4 |
| Özsoy | Verbesserung: Statemachine-Diagram | Verbessern der Aufteilung der Elemente des Zustanddiagramms | 1 |
| Özsoy | Ersetzen der Diagramme und Aktualisierung der Dokumentation (ink. Änderung der Technologiebeschreibung) | Die in der derzeitigen Dokumentation vorhandenen Diagramme mit den Diagrammen im EMF-Format austauschen und die restliche Dokumentation aktualisieren (anpassen). | ½ |
| Kodras, Özsoy, Vogt | Hibernate-Mapping-Debugging | Debuggen des Mappings aufgrund der aufgetretenen Fehler | 6, 5, 4 |
| Scholz | GUI-Prototyping | Prototyp für die GUI designen | 2 |
| Özsoy, Vogt | Hibernate JUnit-Test-Cases | JUnit-Test-Cases für die gemappten Tabellen schreiben (Schreiben in Datenbank und Auslesen aus der Datenbank mittels HQL) | 2 |
| Vogt | Testing der Programmfunktionen | Alle Funktionen des Programms werden getestet. | 4 |
| Kodras | GUI-Testing | Testen der GUI-Funktionalitäten mittels Selenium | 3 |
| Özsoy | Ant-Deployment | Erstellen der Datenbank (bestenfalls auch Installation, User und Passwort beachten); Installation des Application-Servers; Deploy des WAR-Files | 3 |
| Scholz | GUI für Eventerstellung | Fertigstellung der GUI für die Event-Erstellung | 4 |

**Geschätzter Gesamtzeitaufwand**

| Person | Zeitaufwand in h |
|---|---|
| Kodras | 17 ½ |
| Özsoy | 17 |
| Scholz | 10 |
| Vogt | 20 ½ |
| **Summe:** | **65** |

## Endzeitaufteilung

| Zuständige Person(en) | Task | Geschätzte Zeit in h | Tatsächliche Zeit in h | Kommentar |
|---|---|---|---|---|
| Kodras | GUI-Mockups | 4 | 3 | Erwartete Zeit zum Einarbeiten in das Mock-Up-Tool war zu hoch (Das Toll ist sehr leicht zu bedienen und zu verstehen) |
| Vogt | Use-Case-Diagram | 3 | 4 | Einlesen in die UML-Use-Case-Spezifikation |
| Özsoy | Statemachine-Diagram | 2 | 5 | Einlesen in die UML-Statemachine-Spezifikation; Änderungen/Verbesserungen nach der Erstellung des Diagramms |
| Özsoy | Dokumentation | 2 | 4 | Verbesserung, dynamische Gestaltung und Updaten der Dokumentation |
| Scholz | Applikations-aufteilung | 3 | 5 | Verzögerung, da 1. Designentwurf mit Client-/Server-Applikation verworfen wurde |
| Scholz | Teammanagement | 1 | 1 | / |
| Kodras, Vogt | UML-Export | ½ | ¼ | Keine Probleme beim ASTAH-Export |
| Özsoy | Technologiebeschreibung | 2 | 1 ½ | Problemlose Beschreibung der Technologien |
| Kodras, Vogt | Hibernate-Mapping | 4 | 3 ½ | / |
| Özsoy | Verbesserung: Statemachine-Diagram | 1 | 2 | Die Anpassung der Diagramme hat mehr Zeit benötigt als wie geschätzt |
| Özsoy | Ersetzen der Diagramme und Aktualisierung der Dokumentation (ink. Änderung der Technologiebeschreibung) | ½ | 1 | Das Anpassen hat etwas mehr Zeit in Anspruch genommen als wie geschätzt |
| Kodras, Özsoy, Vogt | Hibernate-Mapping-Debugging | 6, 5, 4 | 3, 10, 2 | Das Mapping hat nicht funktioniert wie gewollt und aus diesem Grund hat es mehr Zeit in Anspruch genommen |
| Scholz | GUI-Prototyping | 2 | 2 | / |
| Özsoy, Vogt | Hibernate JUnit-Test-Cases | 2 | 4 | Die Test-Cases waren schwieriger zu implementieren als wie gedacht |
| Vogt | Testing der Programmfunktionen | 4 | 11 | Unerwartete Fehler sinnd aufgetreten |
| Kodras | GUI-Testing | 3 | | |
| Özsoy | Ant-Deployment | 3 | | |
| Scholz | GUI für Eventerstellung | 4 | | |

**Tatsächlicher Gesamtzeitaufwand**

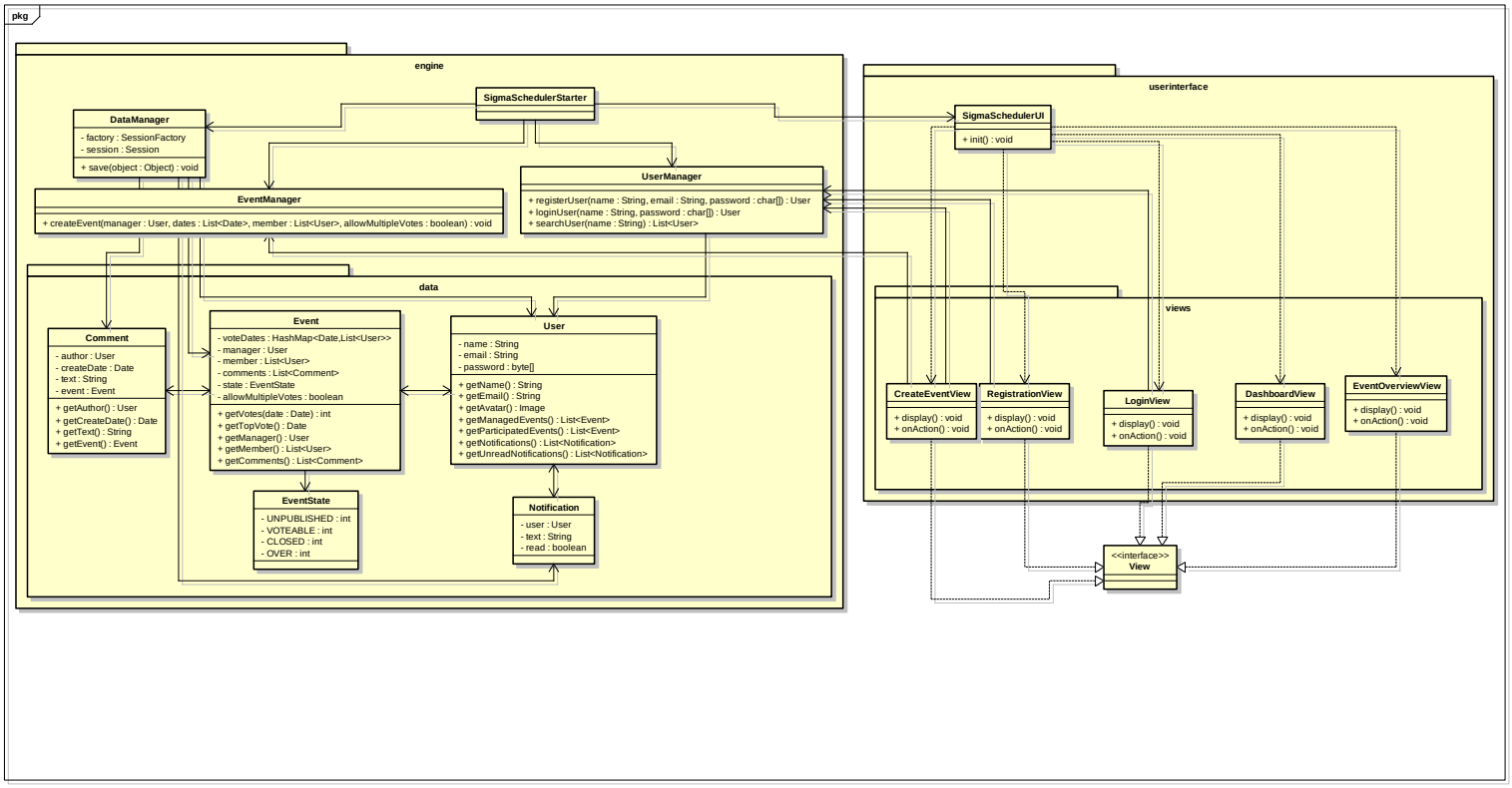
| Person | Zeitaufwand in h |
|---|---|
| Kodras | 9 ¾ |
| Özsoy | 27 ½ |
| Scholz | 8 |
| Vogt | 13 ¾ |
| **Summe:** | **59** |

# Designüberlegung

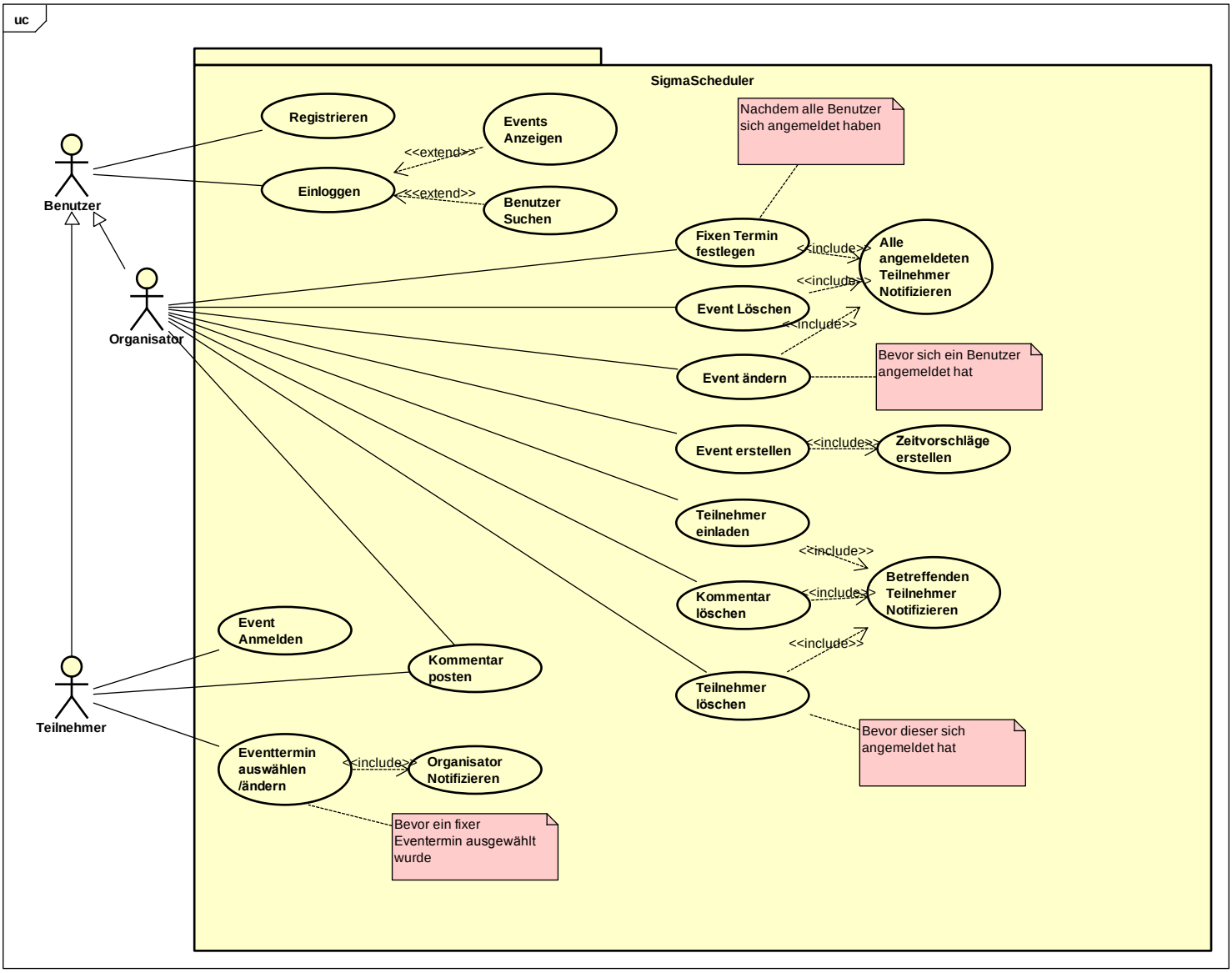
## Applikationsaufteilung

Unser erster Designentwurf, war eine Aufteilung in eine Client und eine Server Applikation. Die Persistierung der Daten sollte der Server mit Hibernate lösen und die Kommunikation zwischen Client und Server mittels RMI realisiert werden. Das Client Programm wäre in dieser Variante eine einfache Java Applikation mit Swing GUI gewesen. Die Serverapplikation hätte den Datenbankzugriff mittels Hibernate implementiert gehabt. Nach einigen Überlegungen, sind wir im Team zu dem Entschluss gekommen, dass die Anwendung besser als Web Anwendung umgesetzt werden sollte. Gründe dafür sind vor allem die leichtere sowie Betriebssystem unabhängige Bedienung jedes Benutzers und der Entfall der Client-Server Kommunikation. Der neue Designvorschlag, sieht eine einzige Server Application vor (Java EE), welche mittels Vaadin Framework das grafische Interface im Browser realisiert, und die Daten wie schon davor mit Hibernate persistiert.

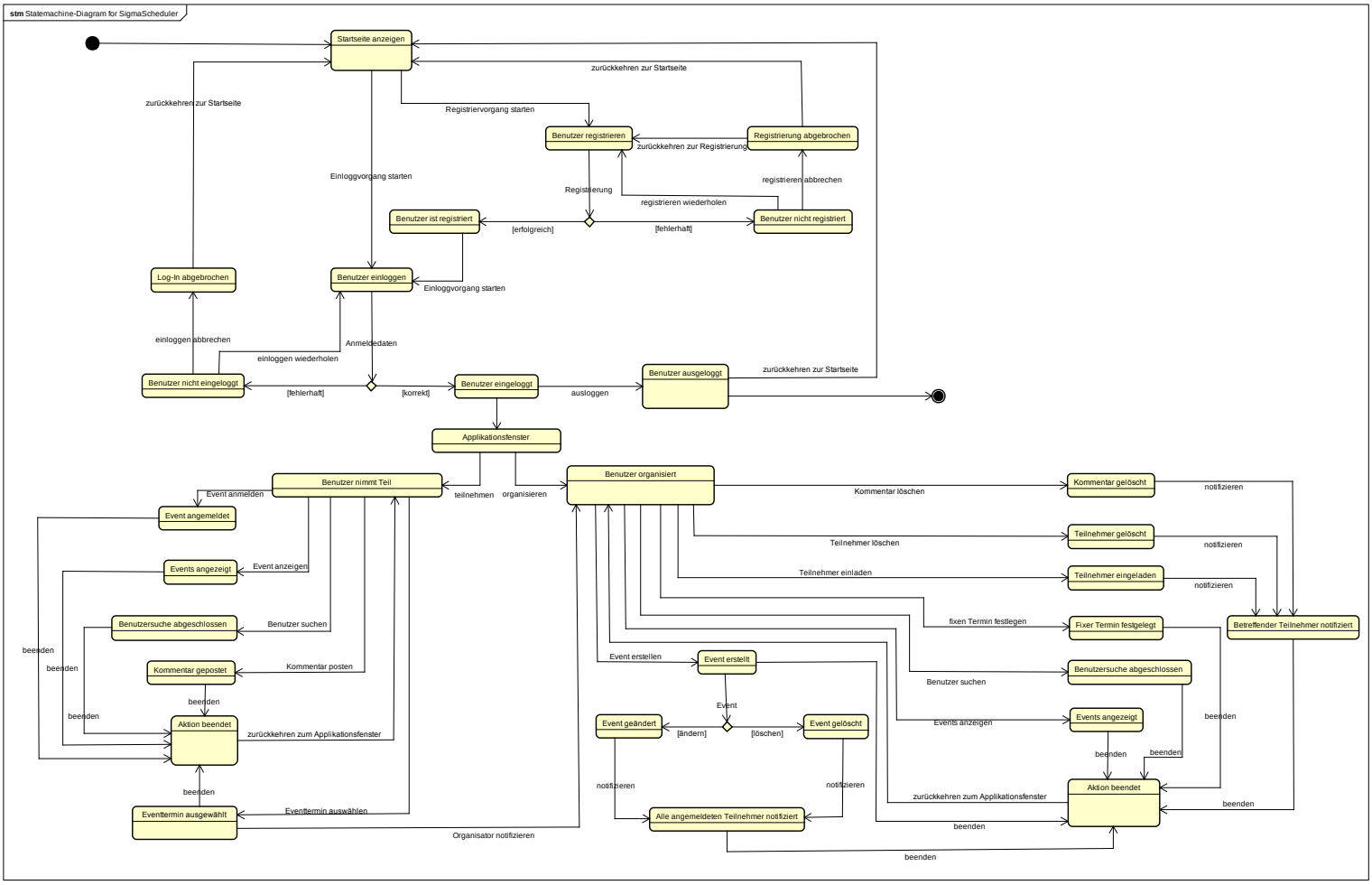
## Klassendiagramm



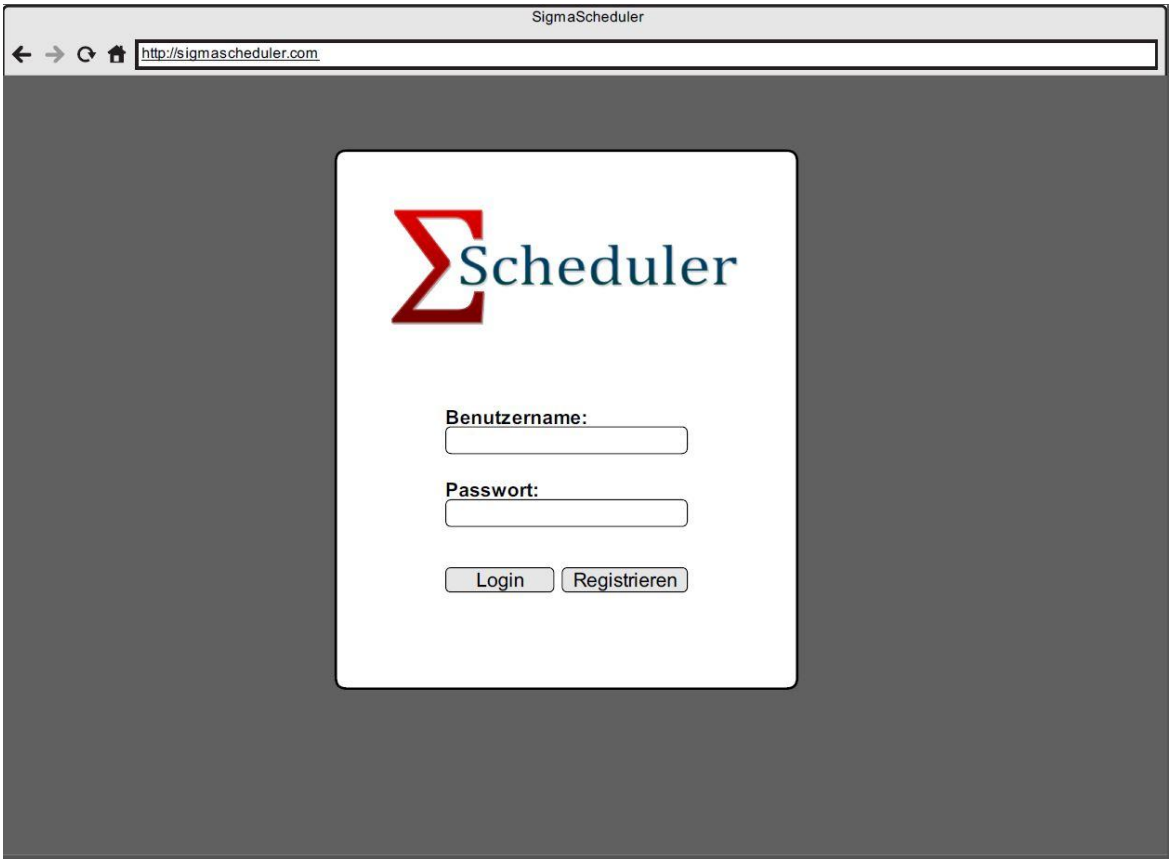
## Requirementsanalyse (Use-Case-Diagram)

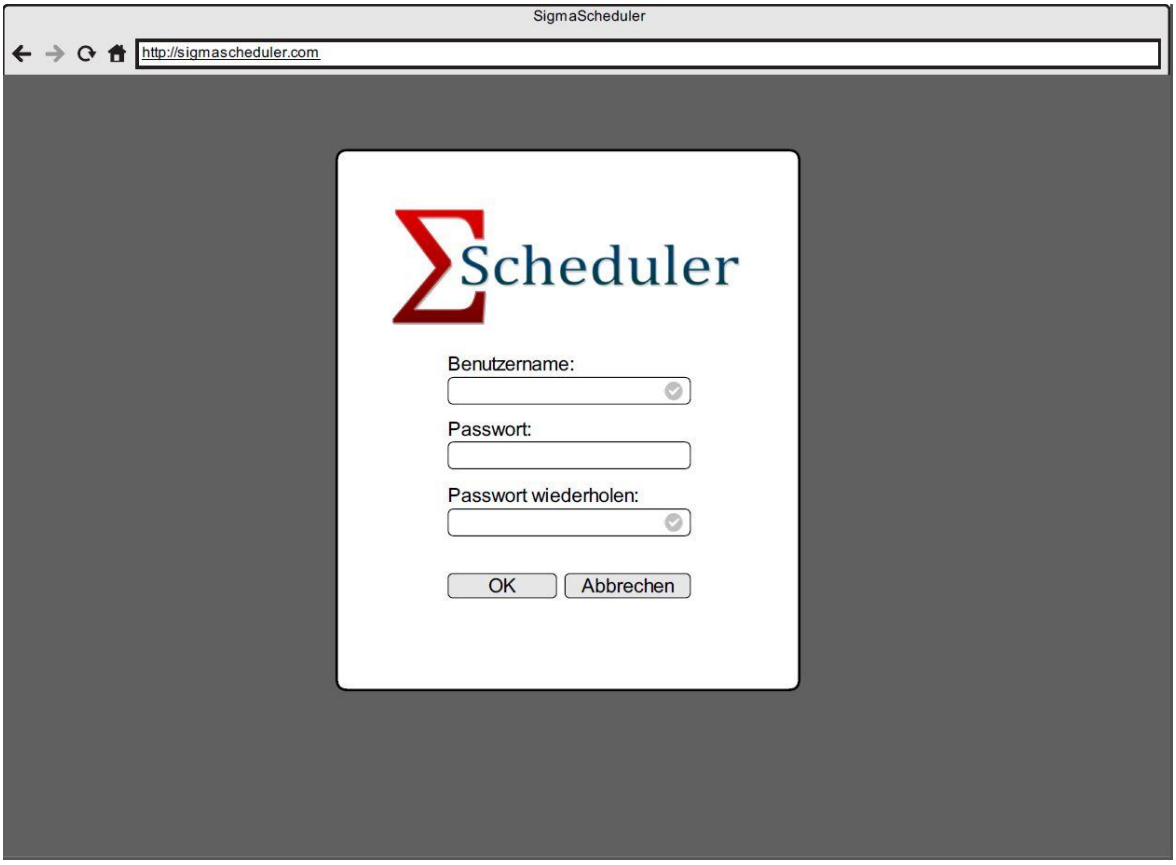


## Statemachine-Diagram (Zustandsdiagramm)

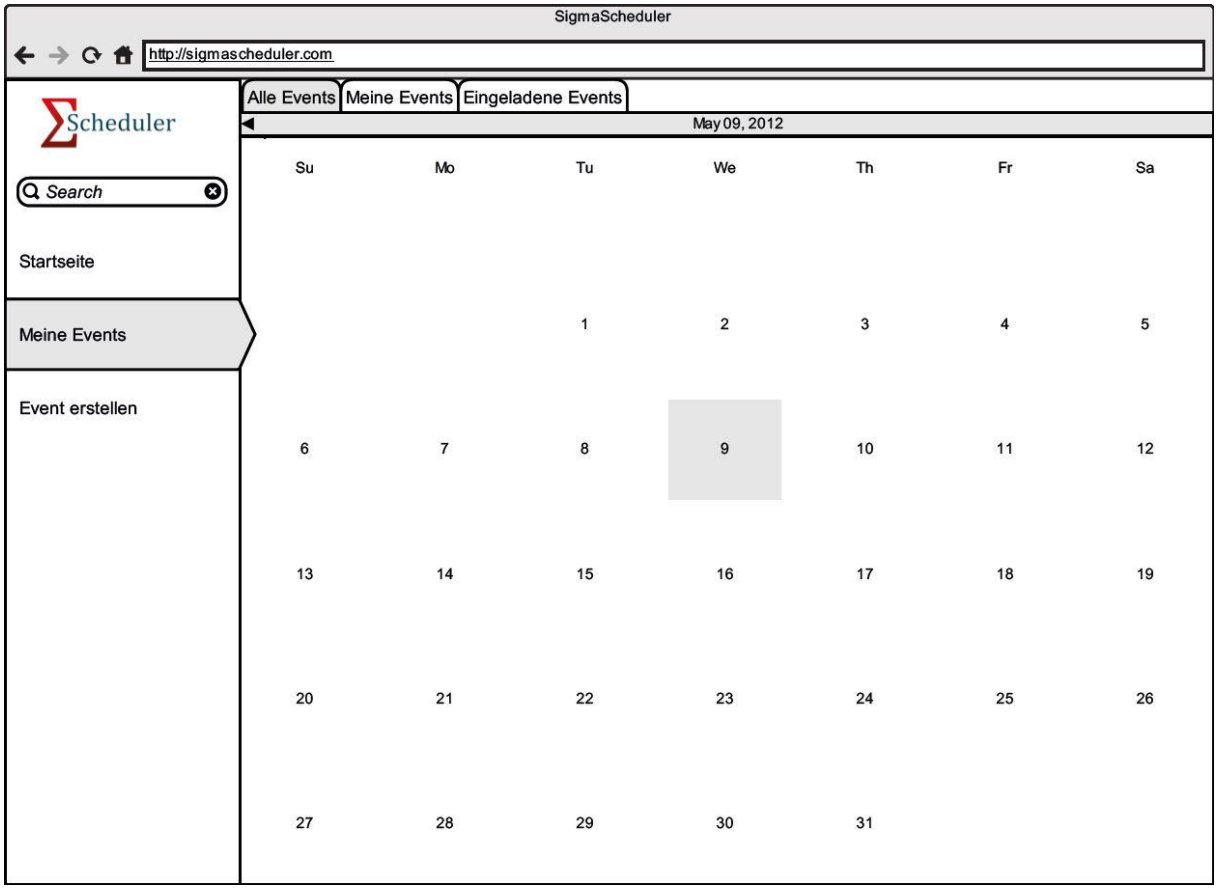


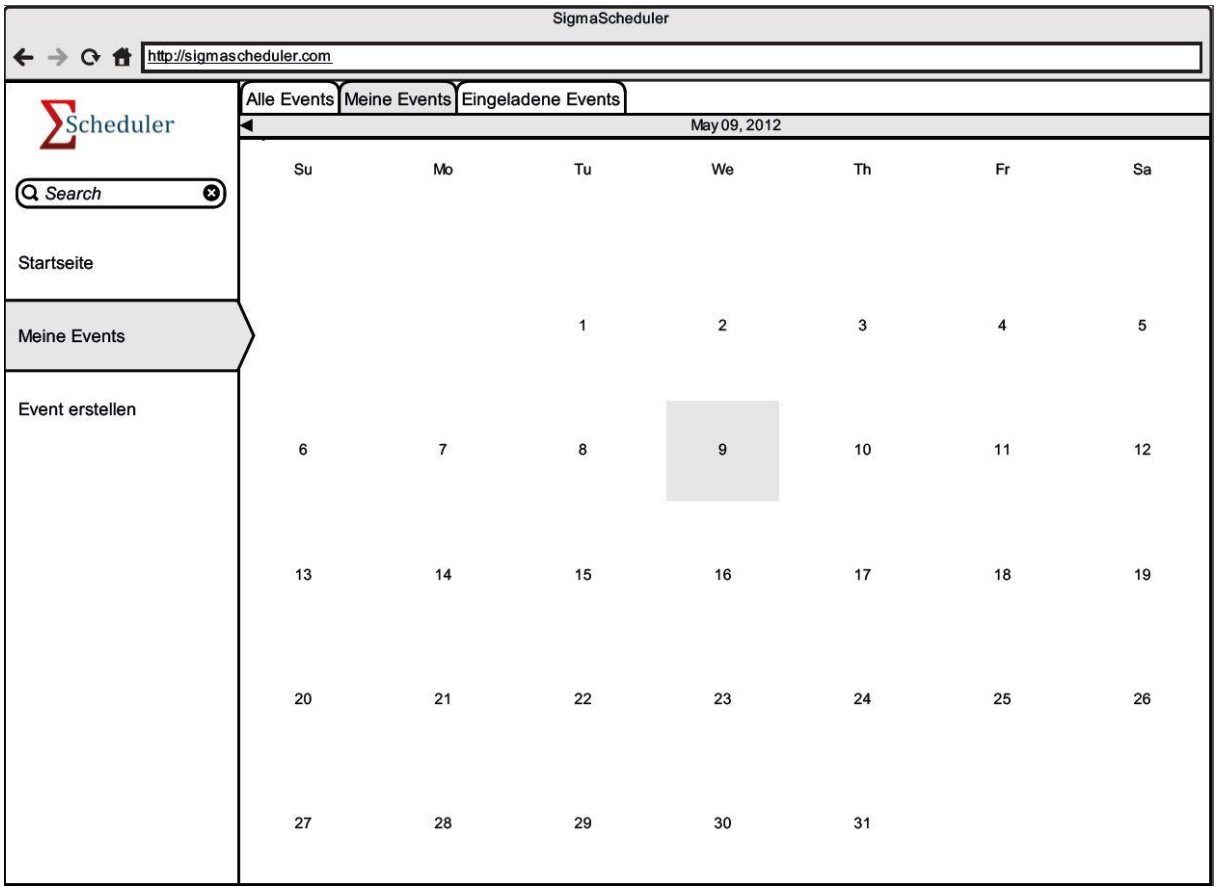
## GUI Mockup

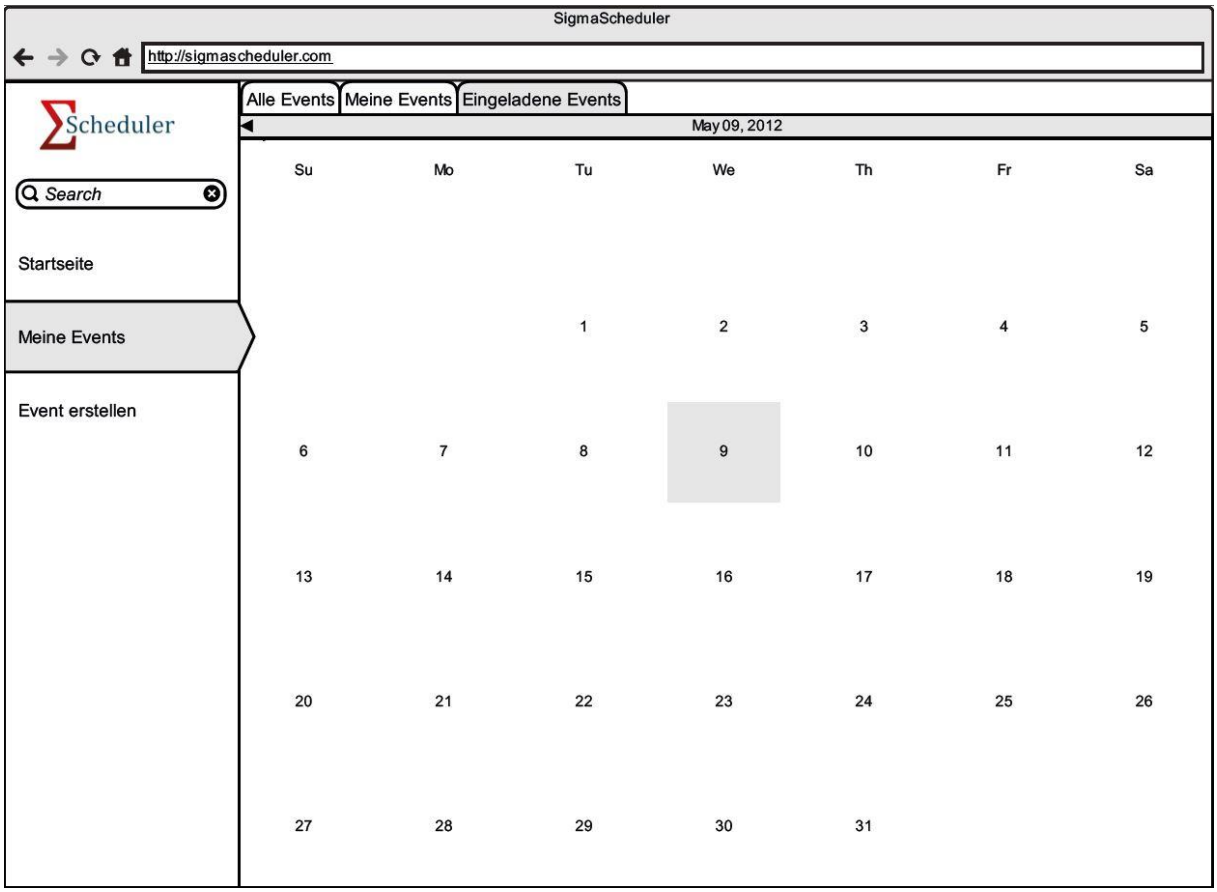

SigmaScheduler
http://sigmascheduler.com
Scheduler
Benutzername:
Passwort:
Login
Registrieren

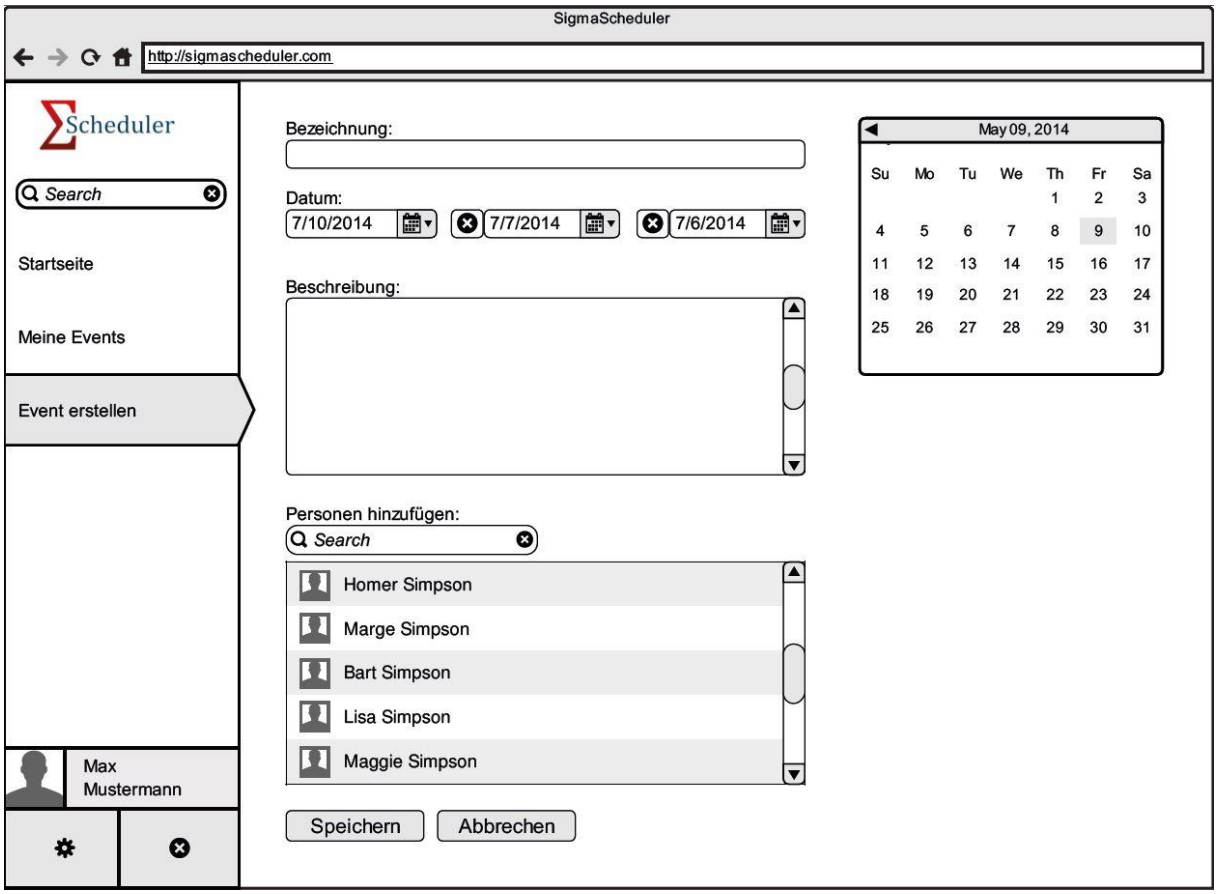

SigmaScheduler
http://sigmascheduler.com
Scheduler
Benutzername:
Passwort:
Passwort wiederholen:
OK
Abbrechen


SigmaScheduler
http://sigmascheduler.com
Scheduler
Search
Startseite
Meine Events
Event erstellen
Max
Mustermann
May 09, 2014
Su Mo Tu We Th Fr Sa
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
Max
Mustermann
Anstehende Events:
Heute (09.05.2014):
8:00 SYT-Wiederholung
9:50 Design-Abnahme
Sonntag (11.05.2014):
3:00 Wiener Schulball - Afterparty
20:00 Grillfest
Samstag (17.05.2014):
20:00 Grillfest


SigmaScheduler
http://sigmascheduler.com
Scheduler
Search
Startseite
Meine Events
Event erstellen
Alle Events
Meine Events
Eingeladene Events
May 09, 2012
Su Mo Tu We Th Fr Sa
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31


SigmaScheduler
http://sigmascheduler.com
Scheduler
Search
Startseite
Meine Events
Event erstellen
Alle Events
Meine Events
Eingeladene Events
May 09, 2012
Su Mo Tu We Th Fr Sa
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31


SigmaScheduler
http://sigmascheduler.com
Scheduler
Search
Startseite
Meine Events
Event erstellen
Alle Events
Meine Events
Eingeladene Events
May 09, 2012
Su Mo Tu We Th Fr Sa
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31


SigmaScheduler
http://sigmascheduler.com
Scheduler
Search
Startseite
Meine Events
Event erstellen
Max Mustermann
Bezeichnung:
Datum:
7/10/2014
7/7/2014
7/6/2014
Beschreibung:
Personen hinzufügen:
Search
Homer Simpson
Marge Simpson
Bart Simpson
Lisa Simpson
Maggie Simpson
Speichern
Abbrechen
May 09, 2014
Su Mo Tu We Th Fr Sa
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

# Technologiebeschreibung

## Hibernate

### Allgemein

Hibernate ist ein Open-Source-Persistenz- und ORM-Framework für Java. Für .NET ist eine portierte Version namens NHibernate verfügbar. [4]

Die Hauptaufgabe von Hibernate ist ORM(=Object-Relational Mapping). Dies ermöglicht es, gewöhnliche Objekte mit Attributen und Methoden (im Java-Bereich POJOs genannt) in relationalen Datenbanken zu speichern und aus entsprechenden Datensätzen wiederum Objekte zu erzeugen. Beziehungen zwischen Objekten werden auf entsprechende Datenbank-Relationen abgebildet. [4]

Darüber hinaus bietet Hibernate Mechanismen zur Kompatibilität mit verschiedenen Datenbanken. Die zum Datenbankzugriff erforderlichen SQL-Statements werden nicht explizit in SQL programmiert, sondern von Hibernate in Abhängigkeit vom SQL-Dialekt der verwendeten Datenbank generiert. [4]

Anwendungsseitig kann Hibernate in Java-Applikationen und Servlet-Engines benutzt werden oder in einen Applikationsserver integriert werden. [4]

### Funktionalität

Zur Abfrage der persistierten Objekte gibt es die Möglichkeit über die SQL-ähnliche Abfragesprache Hibernate Query Language (HQL), mittels SQL-Statements oder objektorientiert mittels der Hibernate Criteria-API. Die Abfragen werden je nach verwendeter Datenbank mittels JDBC in den entsprechenden SQL-Dialekt übersetzt. Hibernate bietet für alle aktuellen relationalen Datenbanksysteme entsprechende Einstellungen, neue SQL-Dialekte können vom Benutzer selbst hinzugefügt werden. [4]

**Objektrelationale Abbildung:**

Hibernate ermöglicht eine transparente Persistenz von Plain Old Java Objects (POJOs). Einzig ein parameterloser Konstruktor muss gegeben sein. Da Hibernate mittels Reflection auf die Attribute der Klassen zugreift, sind keine öffentlichen Zugriffsfunktionen notwendig. [4]

Das Mapping von Java-Klassen auf Datenbanktabellen wird mittels XML-Datei(Mapping File) oder mit Java-Annotation durchgeführt. Bei der Verwendung von XML-Dateien können mittels der Hibernate Tools sowohl die passenden Java-Klassen als auch die Datenbanktabellen generiert werden, bei der Verwendung von Annotations ist letzteres auch möglich. [4]

Hibernate unterstützt Objektreferenzen (1:1-Relationen) und (typisierte) Java Collections (1:N- und N:M-Relationen), sowie auch reflexive Beziehungen auf die eigene Klasse. Hibernate kann auch so konfiguriert werden, dass Operationen wie Speichern oder Löschen auch über Relationen kaskadieren und somit die referentielle Integrität gewahrt bleibt. [4]

Hibernate unterstützt alle drei Arten der objektrelationalen Abbildung von Vererbungsbeziehungen (Tabelle pro Vererbungshierarchie, Tabelle pro Unterklasse und Tabelle pro konkrete Klasse), darüber hinaus auch Impliziter Polymorphismus als Spezialform von Tabelle pro konkrete Klasse. [4]

Ebenso ermöglicht Hibernate eine wahlfreie Abbildung von Java-Typen auf die von der Datenbank unterstützten SQL-Typen. Damit wird z. B. ermöglicht, Java-Enums analog zu normalen Integer-Properties abzubilden, oder einzelne Properties auf mehrere Tabellenspalten zu verteilen. [4]

Die wohl wichtigsten Hibernate Klassen sind SessionFactory, Session und Transaction. SessionFactory lädt die Konfiguration und die Mappings, und wird normalerweise nur einmal pro Anwendung erzeugt. Session ist das Bindeglied zwischen der Java-Applikation und den Hibernate-Diensten, und bietet Methoden für Insert-, Update-, Delete- und Query-Operationen. Transaction bildet JDBC- und JTA-Transaktionen ab. Geschachtelte Transaktionen werden nicht unterstützt. [4]

**Weitere Funktionalitäten:**

Neben der Kernfunktionalität umfasst Hibernate noch folgende Unterprojekte:

- ***Hibernate Annotations***

  OR-Mapping mittels Annotations

- ***Hibernate EntityManager***

  Umsetzung der Schnittstelle Java Persistence API

- ***Hibernate Shards***

  Zugriff auf horizontal partitionierte Datenbanktabellen

- ***Hibernate Validator***

  Definition von Daten-Integritäts- und Validierungsregeln in JavaBean-Klassen mittels Annotations

- ***Hibernate Search***

  Transparente Erstellung eines Volltextindexes und Volltextsuche mit Lucene

- ***Hibernate Tools***

  Entwicklungstools für Eclipse und Ant

- ***NHibernate***

  Hibernate für das .NET Framework. Die aktuelle Version 3.3.3 ist im August 2013 erschienen. NHibernate ist mehr als eine reine C# Portierung, sondern nutzt die zusätzlichen Möglichkeiten von C# wie z.B. Properties. Zu NHibernate gibt es bereits eine Reihe von Unterprojekten. So kann mit Fluent NHibernate die Erstellung der xml-Dateien für das Mapping vermieden werden. Das Automapping erlaubt es, auf eine erneute Aufzählung einzelner Attribute zu verzichten. Für die Konfiguration macht Fluent NHibernate intensiven Gebrauch von Lambda-Ausdrücken. [4]

### Java Persistance API

Dient zum Mappen von Relationalen Daten in Java Applikationen. [2]

**4 Bereiche:**

- Java Persistance API
- Java query language
- Java Persistance Criteria API
- Object/relational mapping metadata [2]

### Entities

Die Entitys representieren Tabellen in einer relationalen Datenbank. [2]

Entity instance = Datensatz / Spalte in einer Tabelle [2]

**Voraussetzungen für Entity-Klassen:**

- Klasse muss „javax.persistence.Entity“ beinhalten
- Klasse muss mind. einen leeren Konstruktor haben
- Klasse darf nicht als „final“ deklariert werden

### Primary-Keys in Entities

Jede Entity hat einen Unique Object identifier(=Primary-Key). [2]

### Managing Entities

Wird verwendet zur Verwaltung der Entity Insances. [3]

- Createn und Removen von Entity Instances
- Finden von Entites mittels Primary Key
- allows queries to be run on entities

### Mapping Annotations

**2 Arten:**

- Logical Mapping --> describing the object model, the association between two entities etc.
- Physical Mapping --> describing the physical schema, tables, columns, indexes, etc. [3]

**Grundlegende Annotationen: [3][5]**

- @Entity - Deklariert die Klasse als Entity --> (Klasse wird alls Tabelle gemapped)
- @Table - Zum setzten von: Name, Schema, Unique Constraints und Catalog der Tabelle
- @Id - Zur Identifikation der Tabelle (In der Art Primary-Key)
- @Column - Zum setzten der Eigenschaften einer Spalte (z.B.: Name, Length, Nullable, Unique)
- @GeneratedValue - Zum Festlegen der Strategie zur Erzeugung des Primary-Keys
- Annotations für Beziehungen zwischen Tabellen:
    - @OneToOne
    - @OneToMany
    - @ManyToOne
    - @ManyToMany

### Konfigurationen in der Datei "hibernate.cfg.xml"

Damit das Mapping zustandekommt, benötigt man die Datei „hibernate.cfg.xml". In dieser Datei befinden sich wichtige Konfigurationen für das Hibernate-Mapping.

Die Daten des Datenbnak-Users müssen darin angegeben werden (dieses Beispiel bezieht sich auf eine MySQL-Datenbank):

```
<!-- Database connection settings -->
<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
<!-- Nach der Erstellung eines Users in der MySQL-Datenbank muessen wie folgt die User-Daten angegeben werden -->
<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/westbahn</property>
<property name="hibernate.connection.username">hibernateUser</property>
<property name="hibernate.connection.password">hibernatePasswd</property>
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
```

Eine komplette Beispiel-Konfiguration von „hibernate.cfg.xml" sieht wie folgt aus:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<!--
    Beschreibung:   CONFIG-File fuer Hibernate
    Autor:          Osman Oezsoy, Christian Bobek
    Version:        2014-04-27
-->

<hibernate-configuration>
    <session-factory>

        <!-- Database connection settings -->
        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <!-- Nach der Erstellung eines Users in der MySQL-Datenbank muessen wie folgt die User-Daten angegeben werden -->
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/westbahn</property>
        <property name="hibernate.connection.username">hibernateUser</property>
        <property name="hibernate.connection.password">hibernatePasswd</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

        <!-- JDBC connection pool {use the built-in} -->
        <property name="connection.pool_size">1</property>

        <!-- Echo all executed SQL to stdout -->
        <property name="show_sql">true</property>

        <property name="current_session_context_class">thread</property>
        <property name="hibernate.transaction.factory_class">org.hibernate.transaction.JDBCTransactionFactory</property>
        <property name="hibernate.cache.provider_class">org.hibernate.cache.HashtableCacheProvider</property>

        <!-- Drop and re-create the database schema on startup -->
        <property name="hibernate.hbm2ddl.auto">create</property>

        <!-- Names the annotated entity class -->
        <mapping class="westbahn.model.Bahnhof"/>
        <mapping class="westbahn.model.Benutzer"/>
        <mapping class="westbahn.model.Einzelticket"/>
        <mapping class="westbahn.model.Preisstaffelung"/>
        <mapping class="westbahn.model.Sonderangebot"/>
        <mapping class="westbahn.model.Strecke"/>
        <mapping class="westbahn.model.Ticket"/>
        <mapping class="westbahn.model.Zeitkarte"/>
        <mapping class="westbahn.model.Zug"/>

        <!-- Die Klasse Reservierung wird mittels XML-Mapping gemappt -->
        <mapping resource="westbahn/model/Reservierung.hbm.xml" />

    </session-factory>
</hibernate-configuration>
```

## Vaadin

Vaadin ist ein freies Webanwendungs-Framework für Rich Internet Application (RIA). Im Gegensatz zu JavaScript-Bibliotheken und auf Browser-Plugins basierenden Lösungen bietet es eine serverseitige Architektur, was bedeutet, dass der Großteil der Programmlogik auf dem Server läuft. Auf Client-Seite baut Vaadin auf dem Ajax-Framework Google Web Toolkit auf und kann damit erweitert werden. [6]

Die Programmiersprach von Vaadin ist Java und das Betriebssystem ist somit plattformunabhängig. [6]

### Erweiterungen und Werkzeuge

Neben einer Reihe von OpenSource-Erweiterungen von Vaadin [7] vermarkten die Hersteller von Vaadin auch folgende kommerzielle Erweiterungen und Werkzeuge:

**Vaadin TouchKit**

Eine Erweiterung für Vaadin Applikationen, die es ihnen ermöglicht auf mobilen Endgeräten derart zu laufen, dass sie wie native Applikationen wirken. Dabei laufen diese Applikationen mit HTML5, JavaScript und CSS auf dem Endgerät in einem Browser im Vollbildmodus. Mittels der Möglichkeiten von HTML5 und den Apache Cordova APIs können dabei Fähigkeiten mobiler Endgeräte wie Beschleunigungssensor, Kamera, Kompass undGeolokation angesprochen werden. [6][8]

**Vaadin TestBench**

Vaadin TestBench ist ein auf Selenium2 aufbauendes Werkzeug für die Erstellung automatisierter Oberflächentests. Mit einem Recorder können die Testfälle aufgenommen werden und als JUnit Tests abgelegt und verändert werden. Diese Tests können dann lokal oder remote laufen, die Testergebnisse werden in Dateien gespeichert. [6][9]

**Vaadin Charts**

Eine Bibliothek von visuellen Komponenten für die Darstellung von animierten und interaktiven Diagrammen in Vaadin Applikationen. [6][10]

## Glassfish-Server

GlassFish ist ein Open-Source-Anwendungsserver-Projekt für Java EE, das von Sun Microsystems gestartet wurde und seit 2010 von der Oracle Corporation gesponsert wird. GlassFish ist eine freie Software. [11]

**Das GlassFish-Projekt**

Das GlassFish-Projekt betreut die Weiterentwicklung des früheren Sun GlassFish Enterprise Server (SGFES) innerhalb der Firma Oracle und der Open-Source-Community. Mit der Übernahme von Sun Microsystems durch Oracle im Februar 2010 wurde das GlassFish-Projekt in Oracle GlassFish Server umbenannt. [11]

Das Sun-GlassFish-Enterprise-Server-Projekt (SGFES) startete am 8. Juni 2005. Es gilt seit Erscheinen der Java-EE-5-Spezifikation als deren Referenzimplementierung. Suns Java-EE-5-Paket ohne Updates vom 16. Mai 2006 basierte auf der Codebasis des Sun-Java-System-Application-Server 9. Seit 2006 ersetzt SGFES das Java Web Services Development Pack. Am 17. September 2007 erschien SGFES Version 2 zeitgleich mit SJSAS 9.1. Am 19. Dezember 2007 wurde das erste Update für Version 2 mit neuen Funktionen und Fehlerkorrekturen veröffentlicht. Seit März 2008 gibt es auch ein SGFES Paket mit einer MySQL-Datenbank. [11]

Im Dezember 2009 erschien GlassFish in der Version 3, welches die Referenzimplementierung der neuen Java-EE-6-Spezifikation ist. In Version 3.1 wurde Clustering wieder Bestandteil von GlassFish.[12] GlassFish bedient sich des Open-Source-Persistenzframeworks EclipseLink (bis 2.0 TopLink) sowie Grizzly als Servletcontainer bzw. Java-Webserver-Schicht (web-tier), um Webinhalte zu liefern. Seit Februar 2010 heißt GlassFish (SGFES bzw. SJSAS) offiziell „Oracle GlassFish Server“. [13] Im Zuge des Java EE Panel auf der W-JAX 2013 gab Oracle bekannt den kommerziellen Support für GlassFish einzustellen. [14][11]

## PostgreSQL

PostgreSQL ist ein freies, objektrelationales Datenbankmanagementsystem (ORDBMS). Seine Entwicklung begann in den 1980er Jahren, seit 1997 wird die Software von einer Open-Source-Community weiterentwickelt. [15]

PostgreSQL ist weitgehend konform mit dem SQL-Standard ANSI-SQL 2008, d. h. der Großteil der Funktionen ist verfügbar und verhält sich wie definiert. PostgreSQL ist vollständig ACID-konform und unterstützt erweiterbare Datentypen,Operatoren, Funktionen und Aggregate. Obwohl sich die Entwicklergemeinde sehr eng an den SQL-Standard hält, gibt es dennoch eine Reihe von PostgreSQL-spezifischen Funktionalitäten, wobei in der Dokumentation bei jeder Eigenschaft ein Hinweis erfolgt, ob dies dem SQL-Standard entspricht, oder ob es sich um eine spezifische Erweiterung handelt. Darüber hinaus verfügt PostgreSQL über ein umfangreiches Angebot an Erweiterungen durch Dritthersteller, wie z. B.PostGIS. [15]

### Eigenschaften [15]

- Umfassendes Transaktionskonzept, das Multiversion Concurrency Control unterstützt
- Ermöglicht komplexe Abfragen mit Unterabfragen (Subselects), auch geschachtelt
- Referentielle Integrität
- Mengenoperationen
- Maximale Datenbankgröße nur durch zur Verfügung stehenden Speicher begrenzt.
- Views, die mit Hilfe von Regeln auch schreibfähig sein können
- Schnittstellen zu vielen Unix-Plattformen, ab Version 8.0 auch unter Microsoft Windows
- Export und Import sowohl von Daten, als auch von Datenbankstrukturen (Schemata)
- Erweiterbarkeit durch Funktionen, selbstdefinierte Datentypen und Operatoren
- Asynchrone und synchrone Replikation

### Einbinden einer PostgreSQL-Datenbank in Hibernate

In der Datei „**hibernate.cfg.xml**“ müssen folgende Datenbank-Verbindungseinstellungen hinzugefügt/geschrieben werden, um eine Verbindung mit einer PostgreSQL-Datenbank herzustellen (Hier wurde phpPgAdmin verwendet: **URL:** `localhost:8080/phpPgAdmin`):

```
<!-- Database connection settings -->
<property name="hibernate.connection.driver_class">org.postgresql.Driver</property>
<property name="hibernate.connection.url">jdbc:postgresql://localhost:5432/sigmascheduler</property>
<property name="hibernate.connection.username">postgres</property>
<property name="hibernate.connection.password">[redacted]</property>
<property name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>
```

## Arbeitsdurchführung

Als erstes haben wir uns die Aufgabenstellung durchgelesen und uns miteinander über die Aufgabenstellung unterhalten, um zu klären, was uns erwartet und was alles zu tun ist. Anschließend wurden die Rollen im Projektteamverteilt und mit den Designüberlegungen begonnen.

Bis zur Designabnahme wurden die folgenden Diagramme erstellt:

- UML-Klassendiagramm
- Statemachine-Diagram (Zustandsdiagramm)
- Use-Case-Diagramm

Zusätzlich wurde noch mittels eines GUI-Mockup-Tools ein Entwurf zu der grafischen Benutzeroberfläche unserer Webapplikation erstellt.

Die ganze Aufgabe haben wir nun in Teilbereiche aufgeteilt und jedem Projektteam-Mitglied einen Teilbereich zugewiesen.

Grundsätzlich unterscheiden wir zwischen drei großen Teilbereichen:

- Erstellen der Datenbank
- Erstellen der grafischen Benutzeroberfläche
- Anbinden der Datenbank mit der GUI

Für die Erstellung der Datenbank wurde Hibernate verwendet. Als Datenbank-Typ haben wir uns für PostgreSQL entschieden. Die Funktionalität der Datenbank wurde mittels Junit-Test-Cases getestet.

Für die Erstellung der grafischen Benutzeroberfläche wurde das Webframework „Vaadin“ verwendet. Dieses Framework bietet, wie in der Technologiebeschreibung beschrieben, sehr viele Funktionalitäten an und ist leicht anzuwenden.

Das Testing wurde so durchgeführt, dass jeweils ein Projektmitglied die einzelnen Funktionen implementiert hat und der andere zur gleichen Zeit diese implementierten Funktionen getestet hat.

Nachdem die Aufgabe fertiggestellt wurde, wurde die Funktionaliät der gesamten Webapplikation gemeinsam getestet und dokumentiert.

# Testbericht

## Manuelle Tests

| Testfall (Eingabe, ...) | erwartete Wert | erhaltener Wert |
|---|---|---|
| **RegisterUser(x,x,x,x)** | UserObjekt mit Parameter +id | UserObjekt mit Parameter + id |

**Resultat:** Worked

**Beachten:** Username nicht schon vergeben

| Testfall (Eingabe, ...) | erwartete Wert | erhaltener Wert |
|---|---|---|
| **LoginUser(name,password)** | UserObjekt mit Name und Password aus Parameter wird rausgelesen | UserObjekt mit Name und Password aus Parameter |

**Resultat:** Worked

| Testfall (Eingabe, ...) | erwartete Wert | erhaltener Wert |
|---|---|---|
| **CreateEvent(x,x,x,x,x)** | EventObjekt mit werten aus Parameter + id wird erstellen | EventObjekt mit Parameterwerten + id |

**Resultat :** Worked

| Testfall (Eingabe, ...) | erwartete Wert | erhaltener Wert |
|---|---|---|
| **CreateEvent(x,x,x,x,x,event)** | Das Eventobjekt im Parameter wird geupdated | Eventobjekt aus Parameter gefüllt mit anderen Parameterwerten |

**Resultat**: Worked

| Testfall (Eingabe, ...) | erwartete Wert | erhaltener Wert |
|---|---|---|
| **DeleteEvent(event)** | Das EventObjekt im Parameter ist leer(Inhalt null) | EventObjekt ohne Inhalt ( null) |

**Resultat :** Worked

## Automatische Tests mittels Selenium

## Bemerkungen

| Nr. | Bemerkung |
| --- | --- |
| | |

## Quellenangaben


| | |
|---|---|
| **[1]** | Titel: Moqups;<br>Autor: moqups.com;<br>Online/Quelle: https://moqups.com/;<br>geändert am: /<br>entnommen am: 08.05.2014; |
| **[2]** | Titel: The Java EE 6 Tutorial;<br>Autor: Oracle and/or ist affiliates;<br>Online/Quelle: http://docs.oracle.com/javaee/6/tutorial/doc/bnbpy.html;<br>geändert am: /<br>entnommen am: 09.05.2014; |
| **[3]** | Titel: Hiberate Annotations;<br>Autor: Red Hat Inc. And the various authors;<br>Online/Quelle: http://docs.jboss.org/hibernate/stable/annotations/reference/en/html/;<br>geändert am: /<br>entnommen am: 09.05.2014; |
| **[4]** | Titel: Hibernate Framework;<br>Autor: Wikipedia;<br>Online/Quelle: http://de.wikipedia.org/wiki/Hibernate_(Framework);<br>geändert am: /<br>entnommen am: 09.05.2014; |

**[5]** Titel: Important annotations used in Hibernate;
Autor: Sujith Mohan;
Online/Quelle: http://sujithforu.blogspot.co.at/2013/05/important-annotations-used-in-hibernate.html;
geändert am: /
entnommen am: 09.05.2014;

**[6]** Titel: Vaadin;
Autor: Wikipedia;
Online/Quelle: http://de.wikipedia.org/wiki/Vaadin;
geändert am: 07.05.2014;
entnommen am: 14.05.2014;

**[7]** Titel: Vaadin Erweiterungen;
Autor: Vaadin Ltd.;
Online/Quelle: https://vaadin.com/directory;
geändert am: /
entnommen am: 14.05.2014;

**[8]** Titel: Vaadin TochKit;
Autor: Vaadin Ltd.;
Online/Quelle: https://vaadin.com/add-ons/touchkit;
geändert am: /
entnommen am: 14.05.2014;

**[9]** Titel: Vaadin TestBench;
Autor: Vaadin Ltd.;
Online/Quelle: https://vaadin.com/add-ons/testbench;
geändert am: /
entnommen am: 14.05.2014;

**[10]** Titel: Vaadin Charts;
Autor: Vaadin Ltd.;
Online/Quelle: https://vaadin.com/add-ons/charts/;
geändert am: /
entnommen am: 14.05.2014;

**[11]** Titel: GlassFish;
Autor: Wikipedia;
Online/Quelle: http://de.wikipedia.org/wiki/GlassFish;
geändert am: 14.01.2014;
entnommen am: 14.05.2014;

**[12]** Titel: Clustering in GlassFish Version 3.1;
Autor: Tom Mueller, Bobby Bissett, Joe Fialli, Mahesh Kannan
Online/Quelle: https://glassfish.java.net/public/clustering31.html;
geändert am: Februar 2014;
entnommen am: 14.05.2014;

**[13]** Titel: Oracle GlassFish Server: Frequently Asked Questions;
Autor: Oracle Corporation;
Online/Quelle: http://www.oracle.com/us/products/middleware/application-server/oracle-glassfish-server-faq-071872.pdf;
geändert am: /
entnommen am: 14.05.2014;

**[14]** Titel: Todesstoß für GlassFish? Oracle stellt kommerziellen Support ein;
Autor: Hartmut Schlosser;
Online/Quelle: http://jaxenter.de/news/Todesstoss-fuer-GlassFish-Oracle-stellt-kommerziellen-Support-ein-168710;
geändert am: 06.11.2014;
entnommen am: 14.05.2014;

**[15]** Titel: PostgreSQL;
Autor: Wikipedia;
Online/Quelle: http://de.wikipedia.org/wiki/PostgreSQL;
geändert am: 21.05.2014;
entnommen am: 31.05.2014;