Pflichtenheft

*UrVent*

*A02*

*13.10.2019*

*Vorversion*

***Wichtige Hinweise:***

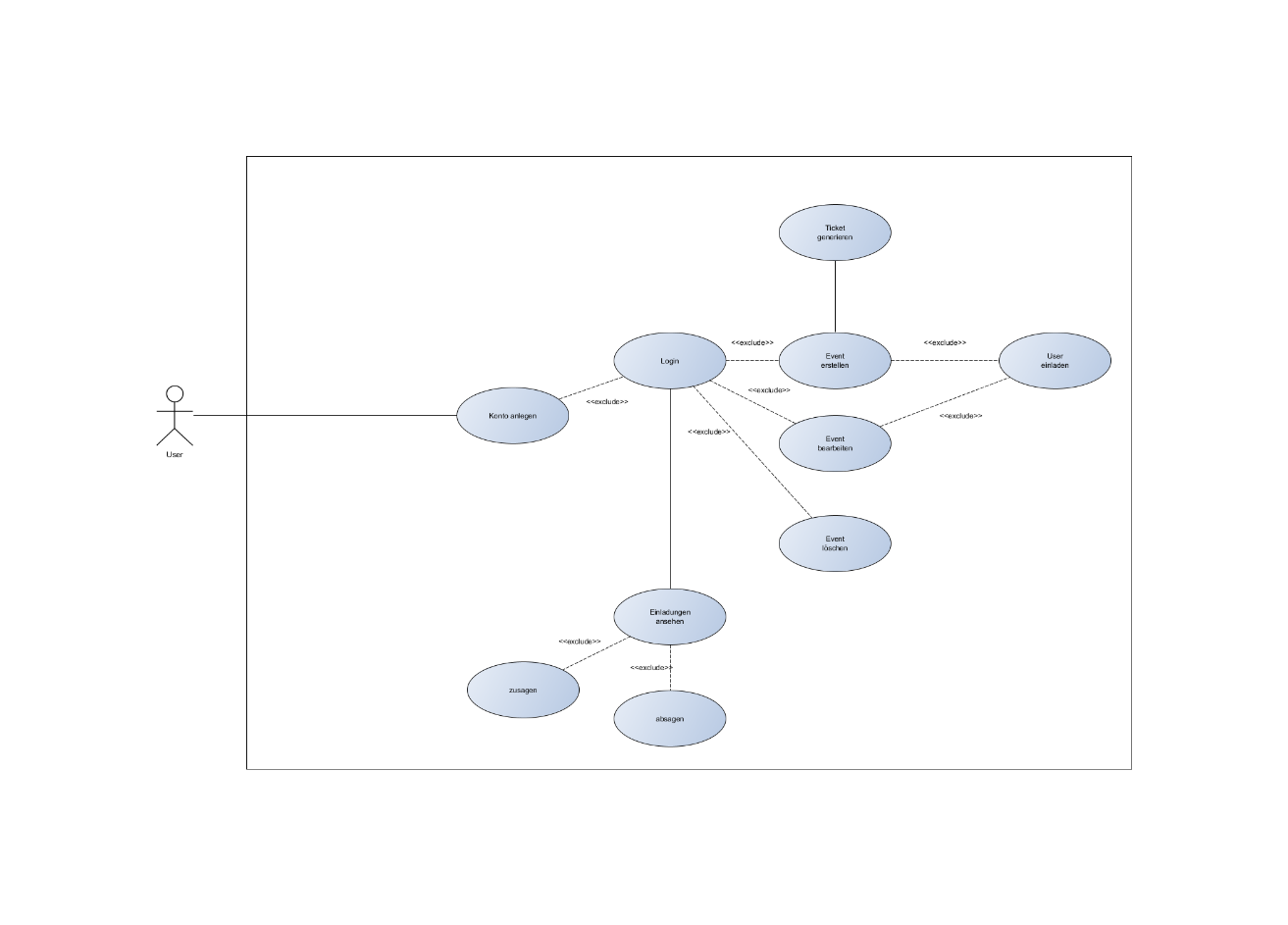
* ***Die in diesem Dokument aufgeführten Beschreibungen in Kursivschrift (außer einigen Abschnittüberschriften) sind beispielhaft und erläuternd und müssen aus dem fertiggestellten Bericht entfernt werden.***
* ***Kennzeichnen Sie welcher Abschnitt dieses Berichts von welchem Teammitglied erstellt wurde.***

# Aufgabenstellung und Zielbestimmung

Der Anwendungsfall des Projekts stellt sich wie folgt da: Als Veranstalter eines Events ist es möglich in „UrVent“ ein Event zu erstellen, um so Werbung für eine Veranstaltung zu machen. Außerdem kann man als Veranstalter die Teilnehmeranzahl einsehen, um so eine Übersicht zur potenziellen Besucheranzahl zu bekommen.

Als „Endkunde“ kann man Veranstaltungen in der Nähe einsehen und auf diese zugreifen. Dort ist es dann möglich eine Beschreibung der Veranstaltung auszurufen und an einer Veranstaltung teilzunehmen bzw. zuzusagen. Außerdem ist der Ticketerwerb und das Beschauen von Bildern der vergangenen Veranstaltungen möglich.

# Anwendungsfall-Diagramm



# Funktionale Anforderungen

* Als User möchte ich mit meiner E-Mail-Adresse und einem Passwort ein Konto anlegen können, um Zugang zu „UrVent“ zu erhalten.
* Als User möchte ich mich mit meiner E-Mail-Adresse und meinem Passwort anmelden können.
* Als User möchte ich eine Veranstaltung anlegen können, um eine Veranstaltung zu planen und zu veranstalten. Hierbei soll automatisch ein Ticket mit QR-Code generiert werden. Dabei kann ich sowohl private, als auch öffentliche Veranstaltungen erstellen.
* Als User möchte ich von mir erstellte Veranstaltungen bearbeiten können, um Änderungen vorzunehmen und um ggf. Informationen hinzufügen zu können.
* Als User möchte ich von mir erstellte Veranstaltungen löschen können, um Veranstaltungen wieder abzusagen.
* Als User möchte ich alle Veranstaltungen einsehen können, zu denen ich eingeladen worden bin, um mich über diese Einladungen und Veranstaltungen zu informieren.
* Als User möchte ich auf eine Einladung mit einer Zu- oder Absage reagieren können, um den Veranstalter zu informieren.
* Als User möchte ich andere User zu einer von mir erstellten Veranstaltung einladen können.

# Qualitätsanforderungen

*Qualitätsziele anhand einer Tabelle bestimmen, wie unten angeführt:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Systemqualität | Sehr gut | Gut | Normal | Nicht relevant |
| Funktionalität |  | X |  |  |
| Zuverlässigkeit | X |  |  |  |
| Benutzbarkeit | X |  |  |  |
| Effizienz |  | X |  |  |
| Wartbarkeit |  |  | X |  |
| Portabilität |  |  | X |  |

Tabelle 1: Qualitätsanforderungen

# Abnahmekriterien

*Legen Sie hier die Kriterien fest, die bei Abnahme das Produkt auf Realisierung/Erfüllung der Anforderungen prüfen. Sie können hier u.a. Testfälle angeben oder darauf verweisen, die die Erfüllung Ihrer Anforderungen überprüfen. Definieren Sie diesen Abschnitt möglichst* ***vor*** *der Implementierung.*

**Folgt noch.**

# Übersicht der Beziehungen zwischen den Klassen

# 

(Work in Progress)

Um einzelne Objekte, die in der Datenbank abgelegt sind wie z.B. einen User, als Java-Objekte zu modellieren, sollen alle derartigen Klassen von einer gemeinsamen Parent-Klasse erben (Model), die allgemeine Methoden und Felder der einzelnen Modellklassen zur Verfügung stellt.  
  
Die von Model erbenden Klassen sollen den default-Konstruktor überschreiben und es ermöglichen ein Objekt ohne weitere Umwandlung aus einem JSON-Objekt zu erzeugen, um eine simple und standardisierte „Sprache“ zwischen API und Client zu ermöglichen und um das Projekt möglichst modular aufzubauen.  
  
Models haben ein Feld „parent“ vom Typ ViewModel. ViewModels modellieren den Zustand des User Interfaces und stellen Funktionen zur Verfügung um „Screens“ zu verwalten. Dabei soll die Datenbindung von JavaFX verwendet werden, um beispielsweise Labeltexte zu modifizieren. Über die Beziehung Model.parent 🡨🡪 ViewModel wird die Kommunikation zwischen Objekten dieser Klasse ermöglicht.  
  
Die Klasse HTTP stellet statische Methoden zur Verfügung, die von anderen Klassen aufgerufen werden sollen, um mit der API zu kommunizieren. Dabei sollen Requests an die API im JSON-Format gestellt werden und die API-Objekte aus der Datenbank als JSON-String zurückgeben, der jeweils in den Methoden daraus ein JSON-Objekt erzeugt. Aus jenem wiederum Objekte vom Typ Model (User, Event, usw.) erzeugt werden.

# Grobbeschreibung der Bedienung bzw. des Ablaufs

(Work in Progress)

Der Nutzer muss sich zunächst über das Registrierfenster/den Registrierbereich registrieren, um anschließend sich mit diesen Daten im Anmeldebereich einloggen zu können. Daraufhin kann der Nutzer die komplette Funktionalität der Software über das User Interface nutzen, indem er auf den von ihm gewünschten Bereich klickt. Um eine Liste für Veranstaltungen in der Nähe des Nutzer einzusehen, muss dieser beispielsweise…

# Literatur

*Wenn Sie Literatur oder andere Quellen verwendet haben, dann führen Sie diese in diesem Abschnitt auf und verweisen an entsprechender Stelle in diesem Dokument darauf.*

**Folgt ggf. noch.**