Pflichtenheft

*UrVent*

*A02*

*07.10.2019*

*Vor- oder Endversion*

***Wichtige Hinweise:***

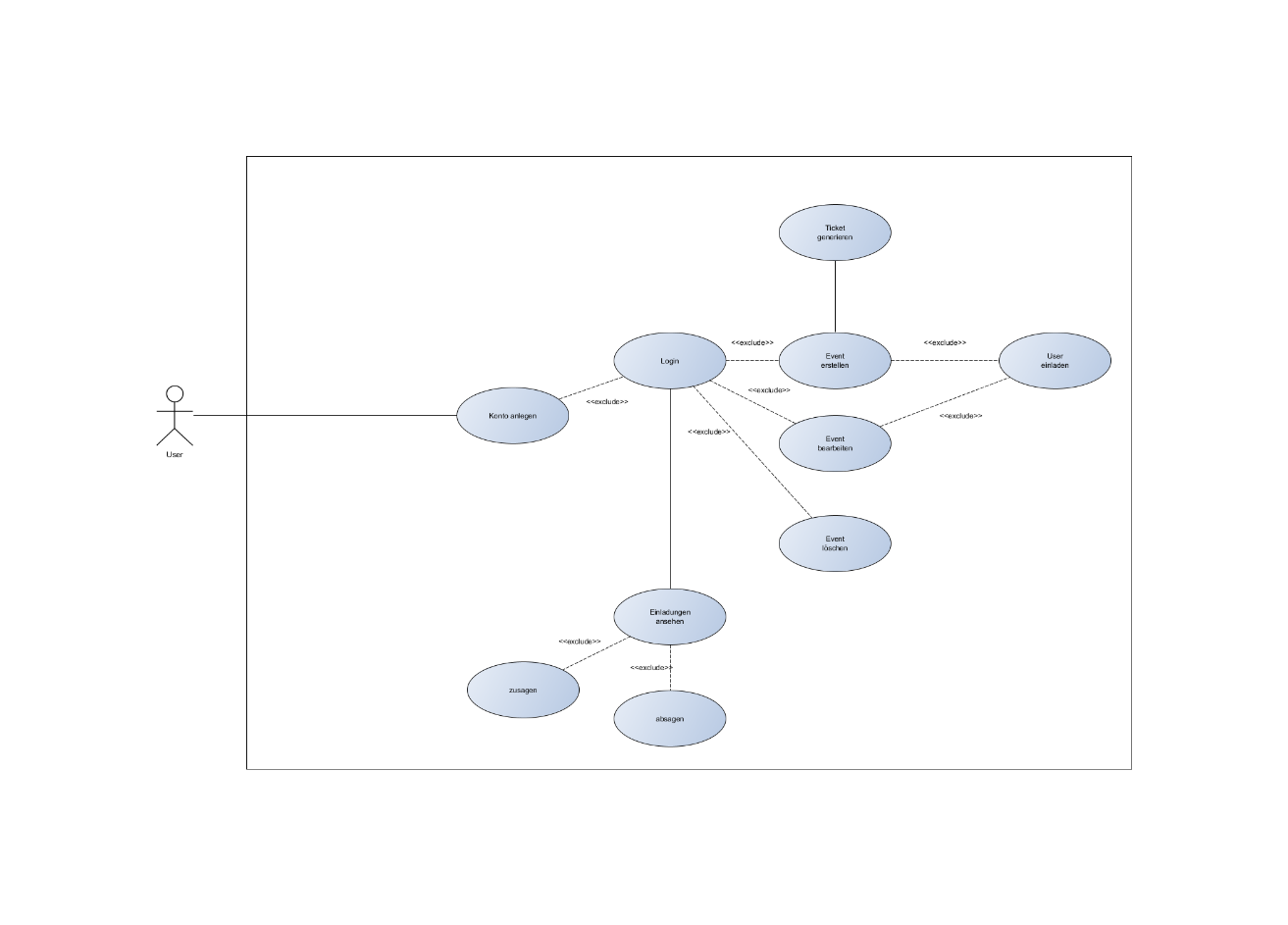
* ***Die in diesem Dokument aufgeführten Beschreibungen in Kursivschrift (außer einigen Abschnittüberschriften) sind beispielhaft und erläuternd und müssen aus dem fertiggestellten Bericht entfernt werden.***
* ***Kennzeichnen Sie welcher Abschnitt dieses Berichts von welchem Teammitglied erstellt wurde.***

# 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 die Veranstaltung zu machen. Außerdem kann ich als Veranstalter eine Teilnehmerzahl einsehen, um so eine Übersicht zur Besucherzahl zu bekommen.*

*Als “Endkunde“ kann ich Veranstaltungen in meiner Nähe sehen und auf diese zugreifen. Dort ist es dann möglich eine Beschreibung ein zu sehen und an einer Veranstaltung teil zu nehmen. 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 bekommen.*

*Als User möchte ich mich mit meiner E-Mail-Adresse und meinem Passwort Anmelden können*

*Als User möchte ich eine Veranstaltung anlegen können, um ein Event zu planen und zu veranstalten, dabei 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 Events bearbeiten können, um Änderungen vornehmen zu können.*

*Als User möchte ich von mir erstellte Events löschen können, um Events wieder abzusagen.*

*Als User möchte ich Events sehen können, zu denen ich eingeladen bin, um mich über Einladungen 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 |  |

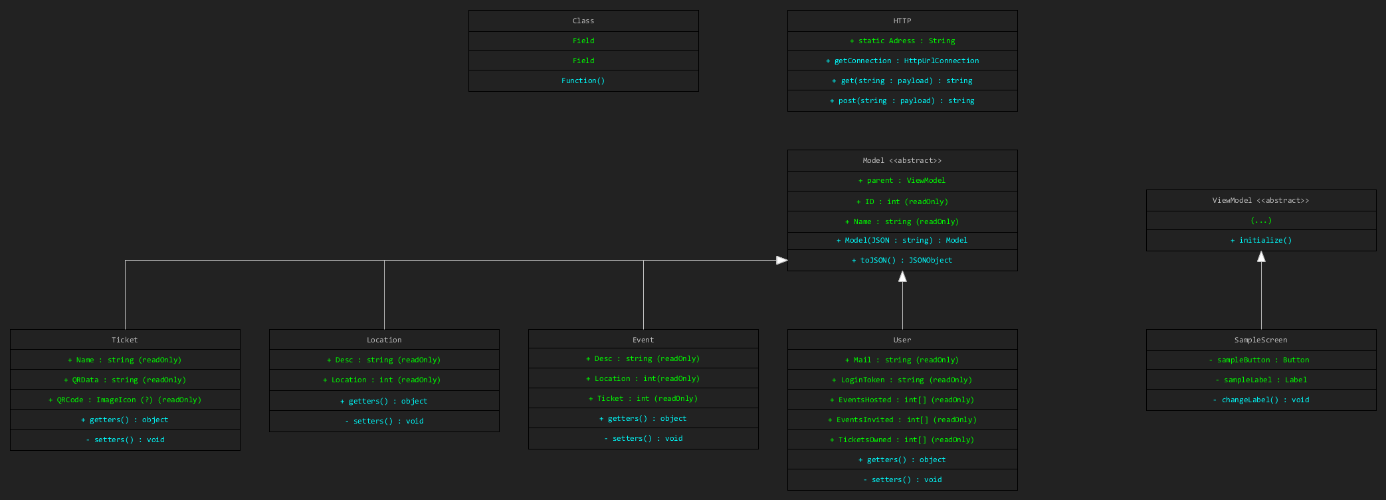
Tabelle 1: Qualitätsanforderungen

# 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.*

# Übersicht der Beziehungen zwischen den Klassen

*Gliedern Sie hier die Übersicht der Beziehungen zwischen den Klassen wieder. Die Darstellung muss noch nicht als Klassendiagramm erfolgen.*



(WIP)

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 Constructor überschreiben, und es ermöglichen, ein Objekt ohne weitere Umwandlung, aus einem JSON-Objekt zu erzeugen um ein simple und standardisierte „Sprache“ zwischen API und Client zu ermöglichen und 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 modfizieren. 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 dem wiederum Objekte vom Typ Model (User, Event usw.) erzeugt werden.

# Grobbeschreibung der Bedienung bzw. des Ablaufs

*Hier erfolgt eine Beschreibung, wie der Ablauf und die dazugehörige Bedienung erfolgt. Z. B. Ein Trainer möchte einen Trainingsplan für ein Mitglied erstellen. Dazu ruft er zuerst über die*

*Benutzeroberfläche die Mitgliederverwaltung auf und erhält das Profil des Mitglieds mit*

*allen nötigen Daten. Danach erstellt er einen Plan bestehend aus mehreren Trainingsterminen …*

# 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.*