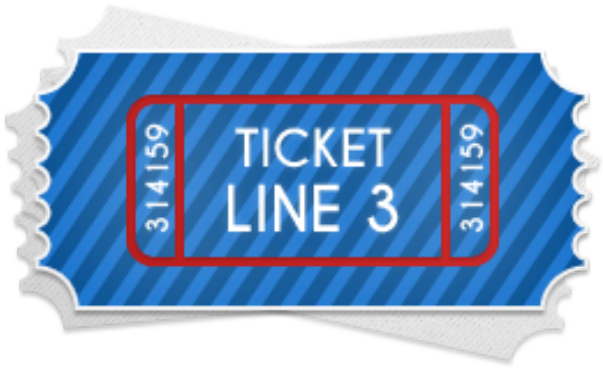

Software Engineering & Projektmanagement

Ticketline 3.0 - Szenario und User Stories

Version 3.0

(18. November 2016)

Dominik Moser, Wolfgang Gruber

Inhaltsverzeichnis

1 Projektbeschreibung

Ticketline ist ein österreichweit tätiges Unternehmen, das sich auf den Verkauf von Tickets für verschiedene Veranstaltungen (z.B. Kino, Theater, Opern, Konzerte, usw), sowie den Verkauf von dazugehörigen Merchandising-Artikeln (beziehungsweise Fan-Artikel) spezialisiert hat. In den Bundeshauptstädten gibt es hierzu Verkaufsstellen, in denen die Kunden sich informieren und Karten, sowie Fan-Artikel direkt kaufen können.

Um die Verkaufsaktivitäten zu unterstützen, wurde ein eigenes IT-System namens Ticketline Kassa entwickelt, die von den Verkäufern in den Verkaufsstellen benutzt wird. Es wird dabei eine Client-Server-Architektur genutzt. Den Server-Teil bildet ein zentrale Datenbank, die im Rechenzentrum des Unternehmens betrieben wird. Die Clients sind Rich Client-Applikationen, die auf den Computern der Verkäufer installiert sind und alle auf die zentrale Datenbank zugreifen.

Da die bestehende Client-Applikation auf einer veralteten Technologie basiert und die Wartung, sowie Weiterentwicklung zu hohe Kosten verursacht, hat sich das Management entschlossen, die Applikation zu ersetzen. Nach einer Evaluierung verschiedener Technologien, wurde die Wahl getroffen, Java SE, Spring und JavaFX als neue Plattform zu nutzen. Als Server soll ein Apache Tomcat verwendet werden, der die Ticketline-Funktionalitäten mittels REST-Interfaces zur Verfügung stellt. Wesentliches Ziel bei der Neuentwicklung ist dabei die Abläufe innerhalb der Applikation zu verbessern, um einen schnellen und reibungslosen Verkauf von Tickets und Artikeln zu gewährleisten.

2 User Stories

Die Anforderungen an das zu erstellende Projekt sind zu Beginn in Form von User Stories zu erstellen und zu priorisieren. Über den Projektverlauf sind die User Stories laufend zu aktualisieren. Details zu User Stories sind in der Prozessbeschreibung zu Scrum aufgeführt.

2.1 Basis User Stories

Im Rahmen der Anforderungsanalyse wurden nachfolgende User Stories identifiziert. Beachten Sie, dass es sich hierbei um die Minimalanforderungen an die Applikation handelt. Die User Stories decken daher nicht alle Funktionen ab, die innerhalb der Applikation benötigt werden.

2.1.1 Authentifizierung

Accountsperre nach falscher Passworteingabe

"Als Verkäufer möchte ich mich durch Eingabe des Benutzernamens und des Passwortes einloggen können. Bin ich eingeloggt, möchte ich mich auch wieder ausloggen können. Sollte ich fünf mal hintereinander ein falsches Passwort eingeben, soll mein Account aus Sicherheitsgründen gesperrt werden."

Entsperren von Accounts

"Als Administrator möchte ich gesperrte Benutzer anzeigen und entsperren können."

2.1.2 News

Aktuelle News anzeigen

"Als Verkäufer möchte ich die aktuellen internen Informationen (Unternehmensnews) nach erfolgreicher Authentifizierung angezeigt bekommen. Dafür möchte ich direkt nach der Authentifizierung eine Seite sehen, auf der mir die Informationen präsentiert werden. Es sollen dabei nur die Informationen angezeigt werden, die ich noch nicht gesehen habe. Auf der Übersichtsseite soll zumindest der Titel und das Datum sowie eine Kurzzusammenfassung der News angezeigt werden."

Detailansicht der News

"Als Verkäufer möchte ich eine Detailansicht eines Newseintrags sehen. Dieser soll den kompletten Text, falls vorhanden, ein angezeigtes Bild und alle weiteren Details des Newseintrags enthalten."

Frühere News anzeigen

"Als Verkäufer möchte ich die Möglichkeit haben, bereits 'gelesene' News erneut anzuzeigen."

News erstellen

"Als Administrator möchte ich neue News erstellen können. Zu jedem Newseintrag möchte ich mindestens ein Bild anhängen können, das in der Detailansicht der News gezeigt wird."

2.1.3 Kunden

Kunden anlegen

"Als Verkäufer möchte ich neue Kunden anlegen können. Der Kunde soll nach dem speichern permanent im System erfasst sein."

Kunde bearbeiten

"Als Verkäufer möchte ich von bestehenden Kunden die Daten ändern können. Die Änderungen sollen permanent im System erfasst sein."

Mehrsprachigkeit

"Als Verkäufer möchte ich die Möglichkeit haben, die Anwendung zur Laufzeit und ohne Neustart sowohl auf deutsch, als auch auf englisch einstellen zu können."

2.1.4 Veranstaltungen

Top Ten Veranstaltungen

"Als Verkäufer möchte ich mir die Top Ten Veranstaltungen nach verkauften Tickets und Kategorien in einem Monat anzeigen lassen. Die Darstellung soll grafisch in einem Chart erfolgen. In weiterer Folge möchte ich als Verkäufer dem Kunden Tickets zu diesen Veranstaltungen verkaufen oder reservieren können."

Suchen/Filtern nach Künstlern

"Als Verkäufer möchte ich nach Veranstaltungen eines Künstlers suchen können. Dabei möchte ich nach den Kriterien Vorname und Nachname des Künstlers filtern können. Die Künstler, die den Suchkriterien entsprechen, sollen mir in einer Liste angezeigt werden. Wenn ich einen Künstler auswähle, möchte ich seine Veranstaltungen angezeigt bekommen. In weiterer Folge möchte ich dem Kunden zu einer Aufführung des Künstlers Tickets verkaufen oder reservieren können."

Suchen/Filtern nach Orten

"Als Verkäufer möchte ich nach Aufführungsorten suchen können. Dabei möchte ich nach den Kriterien Bezeichnung, Strasse, Ortname, Land oder PLZ des Ortes filtern können. Die gefundenen Orte sollen mir in einer Tabelle angezeigt werden. Wenn ich einen Ort auswähle,

möchte ich die Aufführungen des Ortes angezeigt bekommen. In weiterer Folge möchte ich dem Kunden Tickets zu einer Aufführung an dem Ort verkaufen oder reservieren können."

Suchen/Filtern nach Veranstaltungen

"Als Verkäufer möchte ich nach Veranstaltungen suchen können. Dabei möchte ich einen Suchbegriff eingeben können, der nach den Kriterien Bezeichnung der Veranstaltung, Typ, Dauer (+/- 30 Minuten Toleranz) oder dem Inhalt filtert. Nach der Suche sollen mir die gefunden Veranstaltungen in einer Tabelle angezeigt werden. Wenn ich eine Veranstaltung auswähle, möchte ich die zugehörigen Aufführungen angezeigt bekommen. In weiterer Folge möchte ich dem Kunden Tickets zu einer Aufführung der Veranstaltung verkaufen oder reservieren können."

Suchen/Filtern nach Zeit

"Als Verkäufer möchte ich nach Aufführungen suchen können. Dabei möchte ich nach den Kriterien Datum/Uhrzeit, Preis (+/- Sinnvolle Toleranz), Veranstaltung und Säle filtern können. Nach der Suche sollen mir die gefunden Aufführungen in einer Tabelle angezeigt werden. In weiterer Folge möchte ich dem Kunden Tickets zu einer gefundenen Aufführung verkaufen oder reservieren können."

Veranstaltungsarten

"Als Verkäufer möchte ich Tickets für unterschiedliche Arten von Veranstaltungen verkaufen. Zumindest sind Veranstaltungen mit Platzwahl (mit der Auswahl von Reihe und Sitzplatz) und Veranstaltungen ohne Platzwahl (mit der Auswahl von unterschiedlichen Sektoren) notwendig. Dabei sollen unterschiedliche Sektoren und Reihen jeweils unterschiedliche Preiskategorien haben können."

2.1.5 Tickets

Saalplan

"Als Verkäufer möchte ich eine grafische Darstellung des Saalplans aufrufen können. Hierbei sollen Reihennummer und Platznummer sowie Preiskategorien sichtbar unterschiedlich hervorgehoben werden. Aus der Darstellung soll klar hervorgehen, welche Plätze für eine Veranstaltung noch frei sind und welche bereits belegt sind. Außerdem soll eine Darstellung unterschiedlicher Saal Layouts möglich sein. (zum Beispiel: unterschiedliche Platzanzahl pro Reihe)"

Ticketverkauf

"Als Verkäufer möchte ich einem neuen, bestehenden oder anonymen Kunden Tickets für eine Aufführung verkaufen. Dabei soll der Saalplan für die Platzauswahl grafisch erfolgen und eine Mehrfachauswahl von Sitzplätzen ermöglichen."

Ticketreservierungen

"Als Verkäufer möchte ich für einen neuen, bestehenden oder anonymen Kunden eine Reservierung für eine Aufführung durchführen. Dabei soll der Saalplan für die Platzauswahl grafisch erfolgen und eine Mehrfachauswahl von Sitzplätzen ermöglichen. Nach erfolgreicher Reservierung soll mir die Reservierungsnummer angezeigt werden."

Stornierung von Ticketreservierungen

"Als Verkäufer möchte ich reservierte Tickets stornieren. Dafür nennt mir der Kunde entweder die Reservierungsnummer oder seinen Namen und die Aufführung, zu der die Tickets reserviert wurden. Die stornierten Plätze sollen wieder frei gegeben werden."

Ticketverkauf reservierter Tickets

"Als Verkäufer möchte ich zuvor reservierte Tickets dem Kunden verkaufen. Dafür nennt er mir entweder die Reservierungsnummer oder seinen Namen mit der Aufführung, zu der die Tickets reserviert wurden. Die verkauften Plätze sollen danach als verkauft dargestellt werden. Auch soll der Kunde die Möglichkeit haben, nicht alle Plätze der Reservierung zu kaufen. Die nicht gekauften Plätze sollen wieder frei gegeben werden."

Stornierung von verkauften Tickets

"Als Verkäufer möchte ich verkaufte Tickets stornieren. Dafür nennt mir der Kunde seinen Namen und die Aufführung, zu der er Tickets erworben hat. Die stornierten Plätze sollen wieder frei gegeben werden."

2.1.6 Rechnungsdruck

PDF Druck von Rechnungen

"Als Verkäufer möchte ich für jeden Verkauf eine Rechnung im PDF Format erstellen können. Die Rechnung muss den in Österreich gültigen Bestimmungen zur Rechnungslegung entsprechen."

PDF Druck von Stornorechnungen

"Als Verkäufer ich möchte für jede Stornierung eine Stornorechnung drucken können. Dazu werden die Kundendaten aus der zu stornierenden Rechnung vorausgefüllt und der Kauf rückabgewickelt."

2.2 Erweiterte User Stories

Zusätzlich zu den Basis User Stories gibt es vier Kategorien von erweiterten User Stories: “Merchandise und Prämien”, “Zahlungsabwicklung”, “Benutzerverwaltung”, “Ticketverkauf und Dateneingabe”. Aus diesen vier Kategorien müssen Sie zwei wählen, die Sie umsetzen möchten.

2.2.1 Merchandise und Prämien

Verkauf von Merchandise

“Als Verkäufer möchte ich neuen, bestehenden oder anonymen Kunden Merchandise Artikel mit unterschiedlichen Zahlungsmöglichkeiten verkaufen. Dabei sollen mir die verfügbaren Merchandise Artikel angezeigt werden. Der Kunde soll unterschiedliche Artikel in unterschiedlichen Mengen kaufen können.”

Prämienübersicht

“Als Verkäufer möchte ich die verfügbaren Prämien sehen können. Als Prämien können bestimmte Merchandise Artikel gewählt werden.”

Prämienpunktekonto

“Als Verkäufer möchte ich einem registrierten Kunden den aktuellen Punktestand seines Accounts mitteilen können.”

Erhalt von Prämienpunkten

“Als Verkäufer möchte ich, dass Stammkunden bei jedem Kauf Prämienpunkte gutgeschrieben bekommen.”

Eintausch von Prämienpunkten

“Als Verkäufer möchte ich für einen registrierten Kunden die Punkte seines Accounts gegen eine verfügbare Prämie eintauschen können.”

2.2.2 Zahlungsabwicklung

Online Zahlungsanbieter

“Als Administrator möchte ich, dass Kunden mit verschiedenen Zahlungsmitteln bezahlen können. Hierzu soll die API eines Zahlungsdienstleisters (Stripe, PayPal, ...) angebunden werden. [Verwenden Sie für diese Anforderung unbedingt einen kostenfreien Testaccount des Zahlungsdienstleisters]”

Modulare Architektur

"Als Administrator möchte ich, dass es einfach möglich ist, das System um neue Zahlungsdienstleister zu erweitern, dazu sollen wenn überhaupt nur minimale Änderungen im SourceCode notwendig sein."

Stornierung

"Als Administrator möchte ich, dass alle Stornierungen über den entsprechenden Zahlungsdienstleister rückabgewickelt werden."

2.2.3 Benutzerverwaltung

Nutzer anlegen

"Als Administrator möchte ich neue Nutzer (Verkäufer/Administratoren) im System anlegen können."

Nutzer sperren

"Als Administrator möchte ich Nutzer aus dem System aussperren können. Sowie gesperrte Nutzer wieder aktivieren können. Dabei ist es wichtig, dass sich ein Administrator nicht selbst aus dem System aussperren kann."

Passwort zurücksetzen

"Als Administrator möchte ich die Passwörter von anderen Nutzern zurücksetzen können."

2.2.4 Ticketverkauf und Dateneingabe

Ablauf reservierter Tickets

"Als Verkäufer möchte ich, dass reservierte Tickets mindestens 30 Minuten vor Beginn der Vorstellung abgeholt werden. Falls nicht müssen die Tickets automatisch wieder freigegeben werden."

PDF Druck von Tickets

"Als Verkäufer möchte ich, dass für jeden Ticketverkauf ein Ticket ausgedruckt werden kann. Das Ticket muss alle relevanten Informationen der Veranstaltung enthalten. Überlegen Sie ob Ihnen Sicherheitsmaßnahmen einfallen, die das Ticket "fälschungssicher" machen."

Anlegen von Orten und Saalplänen

"Als Administrator möchte ich neue Aufführungsorte und Saalpläne anlegen können. Das Erstellen und Editieren der Saalpläne muss grafisch erfolgen."

3 Nicht-funktionale Anforderungen

Nicht-funktionale Anforderungen beziehungsweise Randbedingungen, die die Architektur der Applikation beeinflussen. Wichtig dabei ist, dass die Nicht-funktionalen Anforderungen auch messbar sind.

3.1 Anforderungen an Usability

Für die Usability sollen Sie eigene Anforderungen ausarbeiten.
Ein Beispiel für eine schlecht formulierte Anforderung wäre: "Das Programm soll einfach zu bedienen sein."
Ein Beispiel für eine gut formulierte Anforderung wäre: "Jede Kernfunktionalität (Verkaufen eines Tickets, Anlegen eines Kunden, ...) soll mit maximal fünf Klicks abgeschlossen werden."

3.2 Anforderungen an Performance

Generell müssen Sie für die Performancetests ein realistisches Set an Testdaten verwenden. Dieses umfasst mindestens 1000 Kunden, 1000 Verkäufe sowie 200 Veranstaltungen an mindestens 25 Orten. (Die Testdaten müssen dabei keine "sinnvollen" Daten sein.)

Weitere Anforderungen an Performance sollen Sie selbst ausarbeiten.
Ein Beispiel für eine schlecht formulierte Anforderung wäre: "Die Anwendung muss schnell reagieren."
Ein Beispiel für eine gut formulierte Anforderung wäre: "Jede Operation muss in maximal drei Sekunden zu einem für den User sichtbaren Ergebnis führen."

3.3 Anforderungen an Security

SQL-Injection

"Die Anwendung muss gegen SQL-Injection geschützt sein."

Berechtigungen

"Jeder User darf nur die seiner Rolle zugänglichen Operationen ausführen können."

Kryptographie

"Weder Passwörter noch Zahlungsinformationen dürfen im Klartext gespeichert werden."

3.4 Organisatorische und rechtliche Rahmenbedingungen

Diese können im Rahmen der Lehrveranstaltung außer Acht gelassen werden. In einem realen Softwareprojekt zählen Sie natürlich zu den Nicht-funktionalen Anforderungen.

4 Präsentation

Jede Präsentation ist, bis zu einem gewissen Grad, als "Verkaufsgespräch" mit dem Kunden zu sehen. Dabei muss jedes Teammitglied seinen Teil präsentieren, dabei muss auch auf die Verantwortlichkeiten der jeweiligen Rolle geachtet werden.

4.1 Projektauftakt (MR1)

Zum Projektauftakt wird das Projekt dem Tutor und dem Assistenten präsentiert.

Die **Projektpräsentation** ($\sim$ 30min): Das Projektteam sowie die Rollen jedes Teammitglieds werden vorgestellt. Danach wird das Projekt selbst vorgestellt, dabei werden auch alle geplanten Features durchbesprochen. Sollten zu dem Zeitpunkt bereits Mockups vorhanden sein, so werden diese präsentiert.

Im Rahmen des Projektauftakts werden die Anforderungen (Funktional und Nicht-funktionale) abgesteckt.

4.2 Zwischenpräsentation (MR2)

Die Zwischenpräsentation gliedert sich in zwei Teile:

Die **Projektpräsentation** ($\sim$ 30min): Sie zeigt den Projektverlauf, gibt eine Retrospektive über das Projekt (Was wurde umgesetzt, wie ist das Projekt, aus Management Sicht, bisher gelaufen, sowohl positives als auch negatives) und zeigt ausgewählte Features, die besonders gut umgesetzt wurden. Zudem ist es hier besonders wichtig zu klären wie der aktuelle Stand des Projekts ist und ob das Projekt auf Kurs ist.
Die **Produktpräsentation** ($\sim$ 30min): Eine "Verkaufspräsentation" mit Live-Vorführung des aktuellen Produktzustands. Das Review stellt eine gute Möglichkeit dar, Feedback vom Assistenten einzuholen.

4.3 Abschlusspräsentation (MR3)

Die Abschlusspräsentation gliedert sich in zwei Teile:

Die **Projektpräsentation** ($\sim$ 15min): Sie zeigt den Projektverlauf, gibt eine Retrospektive über das Projekt (Was wurde umgesetzt, wie ist das Projekt, aus Management-Sicht, gelaufen, sowohl positives als auch negatives) und zeigt ausgewählte Features, die besonders gut umgesetzt wurden. Zudem ist es hier besonders wichtig, zu klären, ob alles was geliefert werden soll auch fertiggestellt wurde.
Die **Produktpräsentation** ($\sim$ 45min): Eine "Verkaufspräsentation" mit Live-Vorführung des erstellten Produkts, bei der die Entwickler den Tutor und den Assistenten von ihrem Produkt überzeugen sollen. Hierbei wird auch überprüft, ob die Anforderungen (funktional und Nicht-funktionale) korrekt umgesetzt wurden.

5 Dokumente

Um die Projektdokumentation einheitlich zu gestalten, sind Dokumentationsrichtlinien und Dokumentenvorlagen zu erstellen.
Die Richtlinien legen Regeln im Zusammenhang mit der Dokumentation fest, so zum Beispiel, dass direkt nach einem Meeting ein Meetingprotokoll zu erstellen ist, oder dass alle Dokumente im Wiki verwaltet werden.
Vorlagen sind für immer wiederkehrende Dokumente zu erstellen, damit sie nicht jedes Mal neu erstellt werden müssen. Die gilt zum Beispiel für Protokolle oder Testberichte.

5.1 Benutzergruppenanalyse

Es sollen alle Benutzergruppen, die mit der Applikation arbeiten, identifiziert werden. Dazu muss für jede Benutzergruppe eine detaillierte Analyse für deren Benutzer durchgeführt werden. Diese Analyse beinhaltet unter anderem folgende Fragen:

- Über welche Kenntnisse verfügen die Benutzer in fachlicher und technischer Hinsicht?
- Wie oft und wie lang arbeiten die Benutzer mit der Applikation?
- Welche Nicht-Funktionalen Anforderungen haben die Benutzer an die Applikation, u.a. hinsichtlich Bedienkomfort, Hilfestellung innerhalb der Applikation, Unterstützung und Vereinfachung der Daten-Eingabe, Ausgabe von Meldungen der Applikation?
- Was sind die Kernfunktionen der Benutzergruppe?
- Wie können die Kernfunktionen für die Benutzergruppe optimiert werden?

5.2 Szenarien

Ausgehend von der Benutzergruppen-Analyse und den User Stories sollen Szenarien entwickelt und dokumentiert werden, wie die Benutzer mit der Applikation arbeiten.
Ein Szenario stellt einen komplexeren Ablauf dar, wie der Benutzer mit dem System arbeitet und eine bestimmte Aufgaben mit Hilfe der Applikation löst. Es kann dabei mehrere User Stories beinhalten beziehungsweise diese miteinander kombinieren.
Die Szenarien sollen textuell beschrieben und mittels Diagrammen (zB UML-Aktivitätsdiagramm) modelliert werden. Die wichtigsten User Interfaces, die in den Szenarien zur Anwendung kommen, sollen skizziert beziehungsweise im Rahmen des UI Prototyps (Mockups) ausgearbeitet werden. Es soll beschrieben werden, wie der Benutzer mit dem User Interface interagiert und welche Möglichkeiten er hat, um die Aufgabe zu lösen. Die Szenarien können Beispiele für Eingaben des Benutzers und mögliche Fehlerszenarien enthalten.
Es sind drei Szenarien, die die wichtigsten Kernfunktionen der Applikation abdecken, zu identifizieren und zu beschreiben.

5.3 Stundenübersicht

Sämtliche Tätigkeiten müssen in Redmine als Ticket erfasst werden um eine durchgehende Aufzeichnung des entstandenen Aufwands zu gewährleisten.
Achten Sie darauf, den richtigen Tracker zu verwenden und erstellen Sie auch Tickets für Meetings.

5.4 Meetingprotokolle

Nach jedem Meeting (egal ob mit/ohne Tutor, on-/offline, Reviews, ...) ist ein Protokoll zu erstellen, um den Inhalt und die Entscheidungen des Meetings auf Dauer festzuhalten. Ein Protokoll hat zumindest folgenden Inhalt:

- Beginn
- Ende
- Ort
- Anwesende
- Abwesende
- getroffene Entscheidungen
- Ergebnis des Meetings oder des Reviews

5.5 Risikomanagement

Die Risikoanalyse, bestehend aus den drei Schritten Risikoidentifikation, Risikobewertung und Risikomanagement. Sie wird mittels einer Liste von Projektrisiken durchgeführt.
Identifizieren Sie mögliche Risiken für das Projekt. Die Risiken sollen realistisch und sinnvoll sein. Bewerten Sie die Risiken nach Eintrittswahrscheinlichkeit und Schadensschwere, wenn das Risiko tatsächlich eintrifft und bilden Sie das Produkt aus den beiden Werten. Entwickeln Sie für die vier Risiken mit dem höchsten Wert entsprechende Gegenmassnahmen und beziehen Sie die Gegenmassnahmen in die Projektplanung ein.
Jedes identifizierte Risiko hat dabei zumindest folgenden Inhalt:

- Nummer des Risikos
- Bezeichnung
- Beschreibung des Risikos und wie es identifiziert werden kann
- Eintrittswahrscheinlichkeit: Angabe in Prozent in 5 oder 10-Prozent-Schritte (zum Beispiel: 5%, 10%, usw)

- Schadensschwere: 1 (geringer Schaden) - 5 (schwerer Schaden)
- Risiko: Produkt aus Eintrittswahrscheinlichkeit & Schadensschwere
- Gegenmassnahme: Was ist zu tun, wenn das Risiko eintritt
- Verantwortlicher, der das Risiko überwacht und die Gegenmassnahmen einleitet
- Kategorisierung in projektbezogene und projektunabhängige Risiken

Es ist wichtig, sowohl projektbezogene als auch projektunabhängige Risiken zu identifizieren. Außerdem muss der Eintritt eines Risikos regelmäßig überwacht werden um die entsprechenden Gegenmassnahmen rechtzeitig einzuleiten.

5.6 Gantt Chart

Mit einem Gantt Chart wird die zeitliche Abfolge von Projektaktivitäten grafisch in Form eines Balkendiagramms dargestellt.
Das von Redmine generierte GANTT Chart muss kommentiert werden können! Weiters sind die Ticketeigenschaften derart zu pflegen, dass ein möglichst aussagekräftiges GANTT Chart durch Redmine erstellt werden kann.

5.7 Userinterface Prototyp (Mockup)

Der Userinterface Prototyp (Mockup) ist ein grafischer Entwurf des Userinterfaces. Er vermittelt erste Eindrücke des Designs der Applikation. So können vor der Implementierung bereits Probleme im Design gefunden und behoben werden. Auch können erste Usability Tests anhand der Prototyps durchgeführt werden.
Der UI Prototyp kann entweder mit einem GUI Builder erstellt oder (sauber!) händisch gezeichnet werden. Im fertigen Zustand soll er die UI Masken der 10 wichtigsten User Stories umfassen und einen ersten Eindruck über die Applikation, die Menüführung und die Usability vermitteln.

5.8 Testplan

Der Testplan beschreibt die Strategie, mit der sichergestellt werden soll, dass das entwickelte Softwaresystem fehlerfrei ist und den Anforderungen entspricht. Insbesondere soll im Testplan festgehalten werden, welche Elemente des Softwaresystems zu welchen Zeitpunkten im Entwicklungsprozess getestet werden sollen (Testscope) und wie organisatorisch mit den Ergebnissen eines Testlaufes umgegangen werden soll (Testverantwortlichkeiten). Darüber hinaus soll im Testplan beschrieben werden, wie beim Testen verschiedene Teststufen zum Einsatz kommen:

- **Unit Tests:** Hierbei handelt es sich um Tests auf der tiefsten Ebene. Testgegenstand sind hierbei einzelne Klassen oder Module.
- **Komponententests:** Diese Tests überprüfen die Funktion einer Komponente des Software-Systems, die aus mehreren Klassen besteht.
- **Integrationstest:** Integrationstests testen die Zusammenarbeit mehrerer voneinander abhängiger Komponenten des Systems.
- **Systemtest**: Bei einem Systemtest wird das gesamte System gegen die gesamten Anforderungen getestet. Dies passiert entweder manuell oder mit einem GUI Testtool (zum Beispiel: TestFX).

5.9 Manuelle Systemtests

Dieses Dokument stellt eine tabellarische Auflistung aller manuellen Systemtests dar.
Zu jedem Testfall sollen folgende Eigenschaften aufgeführt sein:

- Nummer des Testfalls
- Typ des Testfalls:
  - Normalfall (NF)
  - Fehlerfall (FF)
  - Sonderfall (SF)
- Kurzbeschreibung
- Vorbedingungen
- Eingaben
- Erwartetes Ergebnis (Programmausgabe/-zustand)

5.10 Testprotokolle und Testberichte

Zu jedem manuellen Test muss während des Testlaufs ein Testprotokoll erstellt werden. Das Testprotokoll enthält folgende Informationen:

- Name des Testers
- Datum und Uhrzeit
- Dauer des Testlaufs

- Revisionsnummer (Commit-Nummer) beziehungsweise Versionsnummer des getesteten Systems

Außerdem sollen zu jedem Testfall aus dem Dokument “Testfälle“ in tabellarischer Form folgende Aspekte festgehalten werden:

- Nummer des Testfalls
- Erwartetes Ergebnis (Programmausgabe/-zustand)
- Tatsächliches Ergebnis (Programmausgabe/-zustand)
- Testergebnis: (Eine entsprechende Hintergrundfarbe erleichtert die Lesbarkeit des Dokuments)
- Fehlerfrei (Grün)
- Fehlerhaft (Rot)
- Test blockiert, weil ein vorhergehender Test fehlschlägt (Gelb)
- Test noch nicht vorhanden (keine Farbe)

Der Testbericht enthält eine Zusammenfassung des Testprotokolls. Er enthält einfache Metriken, unter anderem:

- Gesamtanzahl der Testfälle
- Durchgeführte Testfälle
- Bestandene beziehungsweise fehlerhafte Testfälle
- Im Testprotokoll vermerkten Fehler

Diese Fehler werden einer Fehleranalyse unterzogen:

1. Feststellen des Fehlers: Handelt es sich wirklich um einen tatsächlichen Fehler der Software oder handelt es sich um einen fehlerhaften Testfall, eine falsche Testausführung, usw.
2. Ist der Fehler bereits aus vorhergehenden Tests bekannt?
3. Fehlerklassifikation:
    - **Klasse 1:** Fehlerhafte Spezifikation
    - **Klasse 2:** Systemabsturz
    - **Klasse 3:** Wesentliche Funktionalität ist fehlerhaft

- **Klasse 4:** Funktionale Abweichung beziehungsweise Einschränkung
- **Klasse 5:** Geringfügige Abweichung
- **Klasse 6:** Schönheitsfehler

4. Priorisierung:

- **Stufe 1:** Sofortige Behebung
- **Stufe 2:** Behebung in nächster Version
- **Stufe 3:** Korrektur erfolgt bei Gelegenheit
- **Stufe 4:** Korrekturplanung ist noch offen

Basierend auf der Fehleranalyse wird abschließend im Bericht das Ergebnis des Testlaufs festgehalten der folgendes enthält:

- Wie ist der Zustand der Software zu beurteilen?
- Wurden die Qualitätsziele erreicht?
- Welche Konsequenzen werden aus dem aktuellen Zustand gezogen, unter anderem: wie können zukünftig Fehler vermieden werden, wie kann der Entwicklungsprozess verbessert werden?

5.11 Entwicklungsrichtlinien

Die Entwicklungsrichtlinien sollen die Entwickler unterstützen, einen möglichst einheitlichen Code zu schreiben, in dem sie unter anderem Vorgaben zur Strukturierung und Benennung machen.
Definieren Sie Benennungsschemata für Packages, Interfaces, Klassen, Methoden und Variablen (statische, lokale, usw.).
Verwenden Sie die Code Conventions für die jeweilige Programmiersprache, die Sie nutzen, zum Beispiel: Code Conventions for the Java Programming Language

Hinweis: Oracle wartet die Java Code Convetions leider nicht mehr. Sie basieren grundsätzlich auf der Java Language Specification[1]*. Die Code Conventions sind durchaus noch gültig, decken neuere Paradigmen wie sie in Java8 verwendet werden aber nicht mehr ab.*

Wenn Sie von diesen Konventionen abweichen, halten Sie diese Abweichungen schriftlich fest. Moderne IDEs bieten außerdem die Möglichkeit, die Formatierungsregeln für den Source Code festzulegen. Diese Formatierungsregeln können exportiert und von der gesamten Gruppe genutzt werden.

[1] `http://docs.oracle.com/javase/specs/`

5.12 Designdokument

Das Design Dokument setzt direkt an die Software Architektur an und verfeinert die in der Software Architektur getroffenen Entscheidungen. Der Übergang zwischen Software Architektur und Software Design ist dabei fließend. Im Gegensatz zur Software Architektur werden beim Software Design auch Entscheidungen getroffen, die sich auf die Implementierung beziehen.

Modellierung der Kernstrukturen der Applikation:
Modellieren Sie wichtige wiederkehrende Strukturen der Applikation in Form eines Klassendiagramms. Wenn Sie ein Framework einsetzen, dokumentieren Sie das Zusammenspiel und die Integration des Frameworks in die Applikation.

Eingesetzte Patterns:
Welche Patterns werden wie innerhalb der Applikation eingesetzt?

Domänenmodell:
Das Domänenmodell umfasst die Klassen und Objekte der Domäne, jedoch keine technischen Klassen wie DAOs oder UI-Klassen. Es kann in Form eines Klassen- oder EER-Diagramms modelliert werden.

Exception-Handling:
Beschreiben Sie das Exception-Handling-Konzept der Applikation: Welche Arten von Exceptions werden eingesetzt (Checked oder Runtime Exceptions)? Welche applikationsspezifische Exceptions werden verwendet?

REST Spezifikation:
Definieren Sie die URLs unter denen die einzelnen Serverfunktionen erreichbar sind. Zusätzlich sollten Sie auch ein Mapping zwischen URLs und den Methoden der REST-Controller definieren.

Logging:
Welche Log-Levels werden eingesetzt? Definition welche Log-Nachrichten auf welchem Log-Level ausgegeben werden.

Security-Aspekte:
Wie erfolgt Authentifizierung und Authorisierung innerhalb der Applikation? Welche Techniken werden eingesetzt, um Sicherheitslücken in der Applikation zu verhindern?

5.13 Softwarearchitektur

Die Software Architektur legt die Struktur der Applikation fest und teilt sie in verschiedene Module. Die Architektur gibt einen groben Überblick über die gesamte Applikation, geht jedoch nicht auf Implementierungsdetails ein. Verwenden Sie zur Darstellung standardisierte UML Diagramme, wie zB. ein Komponenten- oder Verteilungsdiagramm!

Definition des Architekturstils:
Welcher Architekturstil wird eingesetzt? z.B. Client-Server, Web Applikation, Drei-Schicht-Architektur, Service-Orientierte Architektur, usw.

Strukturierung der Applikation:
Horizontale und vertikale Strukturierung der Applikation. Vertikal: Welche Layer werden verwendet? Horizontal: Aus welchen Modulen setzt sich die Applikation zusammen?

Eingesetzte Technologien:
Programmiersprache, Application Server beziehungsweise Web Server, Frameworks, Bibliotheken, Datenbank

Datenhaltung:
Wie werden die Daten gespeichert? Relationale Datenbank, Objekt-Orientierte Datenbank, NoSQL-Datenbank, Filestorage (XML, CSV)

5.14 Projektendbericht

Der Projektendbericht enthält eine Zusammenfassung des Projektverlaufs. Er kann als Retrospektive über das gesamte Projekt gesehen werden und soll mögliches Verbesserungspotential für weitere Projekte aufzeigen.

Sprints:
Wieviele Sprints wurden durchgeführt?
Konnten die Ziele der Sprints erreicht werden (Vergleich Planung zu Retrospektive)?
Wie sind die einzelnen Sprints verlaufen (erfolgreich / nicht erfolgreich / Abbruch)?
Wenn ein Sprint abgebrochen wurde, was war der Grund dafür?

Probleme:
Welche Probleme sind während des Projekts aufgetreten?
Wie wurde darauf reagiert?
Wurden sie bei den Projektrisiken identifiziert?
Wurde das Risiko richtig eingeschätzt?

Zusammenarbeit:
Wie ist die Zusammenarbeit im Team, dem Tutor und dem Senior verlaufen?

Bewertung:
Was waren erfolgreiche Aspekte des Projekts?
Was war negativ?
Was wurde gelernt?
Was könnte man in zukünftiger Projektarbeit verbessert werden?

6 Student SCRUM

6.1 Sprinttagebuch

Das Sprinttagebuch dokumentiert den Verlauf bereits abgeschlossener Sprints und bietet eine Rückschau auf den bisherigen Prozessverlauf. Es wird am Ende des gerade laufenden Sprints aktualisiert, indem ein zusätzlicher Sprinteintrag dem Dokument hinzugefügt wird.

Key Elements eines Sprinteintrages:
Kurzes Update zum Produktstatus. Was wurde im letzten Sprint eingebaut?

Ergebnisse der Sprint Retrospective:
Auflistung der Dinge, die schlecht funktioniert haben und verbessert werden müssen. Auflistung der Managemententscheidungen, die ausgeführt werden müssen um die Probleme und Ägernisse im vorherigen Sprint zu lösen/abzuschwächen.

6.2 Sprint Retrospective

Die Sprint Retrospective wird am Ende jedes Sprints durchgeführt. Das Team bespricht offen was beim Sprint gut funktioniert hat und welche Dinge in Zukunft verbessert werden müssen um den Sprint für das Team angenehmer und produktiver zu gestalten.
Es soll eine Prozessverbesserungen am Ende jedes Sprints sowie eine kontinuierliche Verbesserung der Arbeitsweise des Projektteams gewährleisten.

Die Retrospective ist ein kollaborativer Prozess zwischen allen Teammitgliedern und dem Scrum Master. Alle Teammitglieder identifizieren was gut funktioniert hat und was verbessert werden muss. Der Teamkoordinator priorisiert “actions” und “lessons-learned” basierend auf dem Team-Feedback. Das Team sucht gemeinsam Lösungen für die Probleme.

Die Retrospective sollte für diesen Scrum mit maximal einer Stunde “time-boxed” sein.
Aufgrund der Ergebnisse der Sprint Retrospective kann es dazu führen, dass Artefakte wie

Riskikoanalyse, Sprintplanung, Product Backlog eventuell angepasst werden müssen.

Das Resultat der Sprint Retrospective ist ein Protokoll!

6.3 Executeable / Deliverable

Die Executeable ist der aktuell implementierte Status der Applikation. Das Ergebnis jedes Sprints soll ein theoretisch auslieferbares, getestetes Inkrement sein (siehe Prozessbeschreibung).
Mit jedem Sprint sind weitere User Stories zu implementieren. Am Ende des letzten Sprints soll die Applikation fertig und bereit für ein Rollout sein. Bitte beachten Sie bei ihrer Planung, dass die User Stories einen unterschiedlichen Realisierungsaufwand haben.
Im Rahmen der Lehrveranstaltungen werden Sie etwa fünf bis sechs Sprints mit einer Dauer von jeweils etwa zwei Wochen umsetzen.