Projektauftrag

TU-Wien: ASE SoSe2011

Projektleiter:	Joachim Kaiser	0555744	(J)
Technischer Architekt:	David Vallner	0725470	(D)
Testbeauftragter:	Emre Taha Diker	0527588	(E)
Dokumentenbeauftragter:	Patrick Neubauer	1028573	(P)
zweiter technischer Architekt:	Metin Ljapo	0427469	(M)
zweiter Testbeauftragter:	Aleksandar Sibincic	0727895	(A)

Inhalt

Rollenbeschreibung

Projektleiter (PL):

Ist für die organisatorischen Aspekte des Projektes zuständig und Dient als Ansprechpartner der Auftraggeber. Er ist auch der einzige, der mit den Auftraggebern direkt kommuniziert.
In den Aufgabenbereich des PL fällt auch die Organisation von Treffen und die Aufgabenverteilung innerhalb der Gruppe. Wobei bei Implementierungspaketen mit dem TA abzusprechen ist, welche Pakete realistisch schaffbar sind.
Der PL muss dafür sorgen, dass die Gruppenmitglieder die Stundenlisten aktuell halten und ist bei den Reviews für die Statusberichte verantwortlich. Andere Dokumente in seinem Verantwortlichkeitsbereich sind: Der Projektplan, Risikoanalysen & die Iceberglist.
Projektleiter ist Joachim Kaiser, Stellvertreter ist Patrick Neubauer.

Technischer Architekt (TA):

Der technische Architekt verfgt über fundiertes Wissen über die verwendeten Programmiersprachen und Tools. Er kümmert sich um die Auswahl geeigneter Tools und erstellt Richtlinien, die den anderen Gruppenmitgliedern helfen sollen, einheitlichen Sourcecode zu programmieren. Ebenfalls obliegt dem TA das konkrete Design der Programmarchitektur.
David Vallner ist der technische Architekt, Metin Ljapo ist sein Stellvertreter.

Dokumentenbeauftragter (DB):

Ist für die Vollständigkeit der benötigten Dokumente verantwortlich. Er erstellt Dokumentationsrichtlinien (verwendete Software, Dateiformate, Schriftarten und ähnlcihe Konventionen) und kümmert sich anschließend um die Einhaltung derselben. Vor den Deadlines ist er für die Kontrolle der Dokumente verantwortlich. Außerdem kümmert er sich um die Archivierung der Dokumente im Repository.
Patrick Neubauer ist Dokumentenbeauftragter, Joachim Kaiser sein Stellvertreter

Testverantwortlicher (TV):

Der TV ist verantwortlich für ausreichende Überprüfungen des Programms. Dazu erstellt er einen Testplan mit geeigneten Testfällen und eine Vorlage für Fehlerberichte. Er muss die Gruppenmitglieder immer wieder dazu anhalten, ihren Code rechtzeitig zu testen, wobei die Gruppenmitglieder für das Erstellen der Unittests selbst verantwortlich sind. Allerdings obliegt es dem TV, eine Testumgebung zu erstellen (Integration von JUnit, DB mit Testdaten, etc.)
Emre Taha Diker ist Testbeauftragter, Aleksandar Sibincic ist sein Stellvertreter.

Ausgangssituation

Gegenwärtig existieren keine Websiten, die es ermöglichen, Events im lokalen (frei festlegbaren) Umfeld des Users zu suchen. Es gibt zwar Portale, die Events auflisten (zB Volume.at) und diese nach Städten oder Orten gliedern, eine Suchfunktion, die einen geographisch flexibel festlegbaren Bereich absucht, ist allerdings nicht vorhanden.

Nachdem viele (vor allem junge) Leute an einem Wochenende oft nicht wissen, wie sie ihre Abende gestalten können, möchten wir mit dieser Applikation eine Möglichkeit bieten diese Entscheidung zu erleichtern.

Facebook bietet zwar Möglichkeiten, Events zu erstellen, jedoch kann man sie nicht bewerten (außer als Pinnwandeintrag) und Facebook empfiehlt auch keine Events nach Qualität. Von geographischer Suche ist bei Facebook ebenfalls nichts zu finden.

Darum soll diese Applikation umfassende Möglichkeiten enthalten, Events zu suchen und Facebook durch eine Anbindung zu unterstützen.

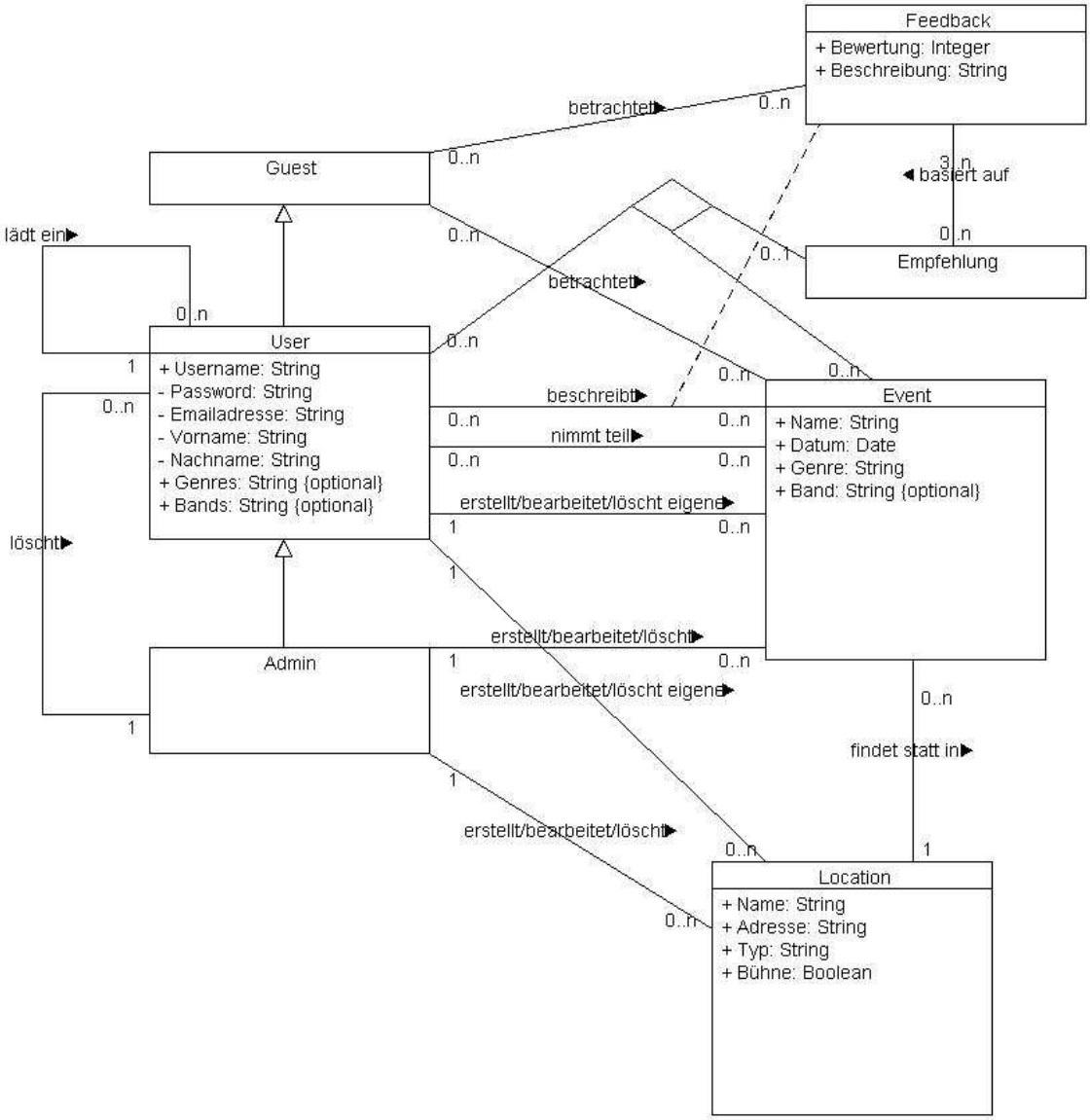
Projektbeschreibung

Das Ziel des Projektes ist die Entwicklung einer dynamischen, datenbankgestützten Website, die als Web-Anwendung auf einem Applikationsserver läuft. Die Website soll in der Lage sein verschiedene Ressourcen (Events, Locations, etc.), Benutzer und Kommentare (Feedbacks) zu verwalten.

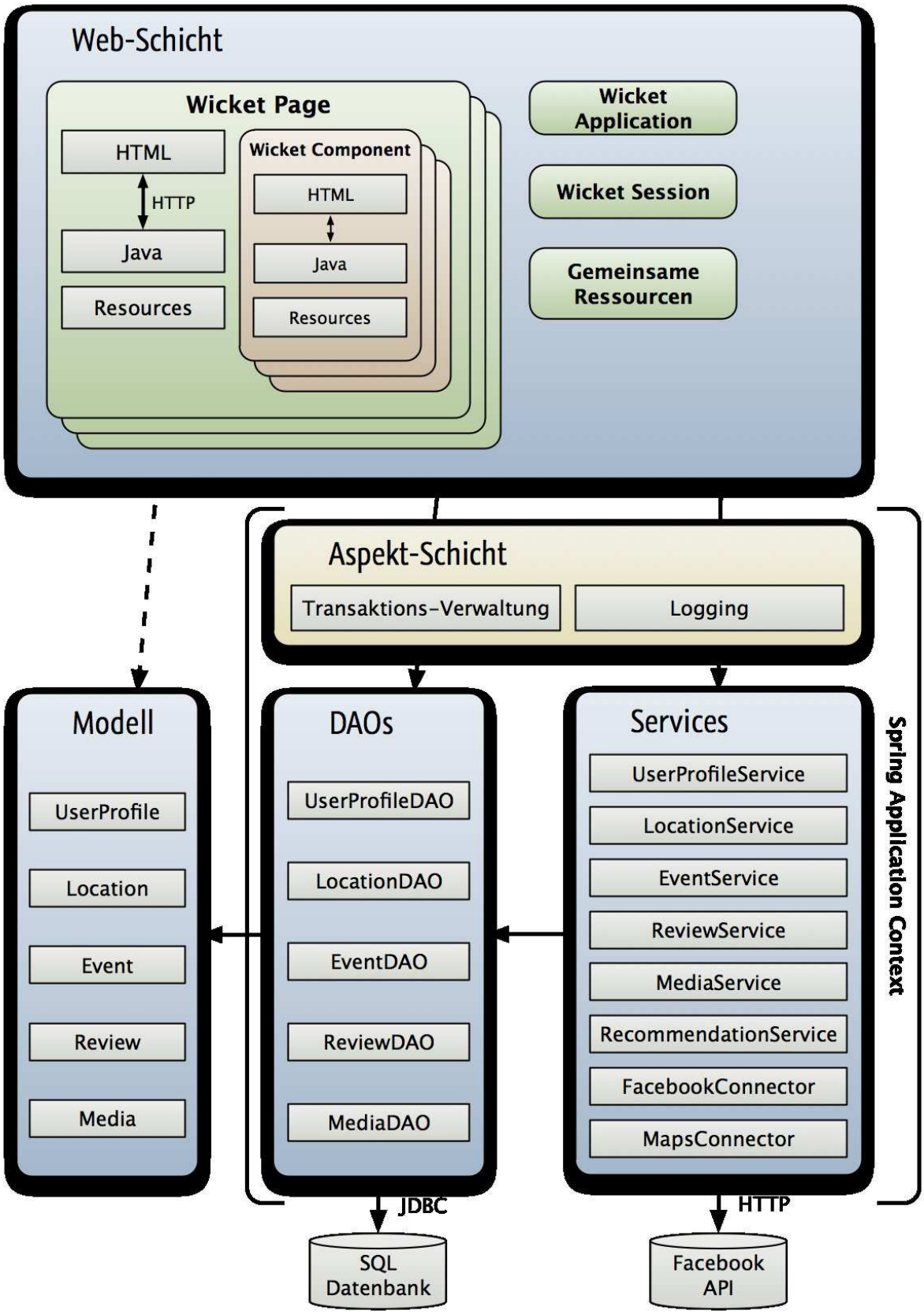
Kernkonzept der Applikation ist es, den Nutzern die Möglichkeit zu bieten, Feedbacks von anderen Benutzern einzusehen, um sich einen Eindruck von der Qualität des Events machen zu können. Auf diese Weise können Benutzer ihre persönlichen Erfahrungen (in Form eines Feedbacks) für eine bestimmte Veranstaltung hinterlassen, die andere Nutzer lesen können. Dies dient als Entscheidungshilfe, ob es sich lohnt an dieser Veranstaltung teilzunehmen.

Die Suche nach Veranstaltungen und ihren Bewertungen ist öffentlich zugänglich, also haben auch nicht registrierte Nutzer das Recht, Events zu betrachten. Eine Einschränkung ist, dass nur registrierte Benutzer Feedbacks für Veranstaltungen hinterlassen und Manager Veranstaltungen erstellen und bearbeiten können.

Domänenmodell



Komponentendiagramm



Architekturübersicht

Die Komponenten der Applikation werden mit Hilfe des *Spring Framework* zusammengestellt. Dieses Framework stellt auch die Aspekt-Schichte bereit, mit der eine transparente Transaktionsverwaltung und ähnliche Anwendungsdienste implementiert werden.

Persistenz von Objekten des Datenmodells (Datenentitäten) wird mit Hilfe von *JPA 2.0* erreicht. *Hibernate Entity Manager* und *Hibernate Annotations* wurden als Implementierung ausgewählt.

Das Backend folgt einer üblichen Aufteilung auf DAOs und Services. DAOs implementieren die CRUD- und Suchoperationen auf Entitäten je eines Datentyps. In den Services werden Tabellenübergreifende Operationen und die Anwendungslogik implementiert. Für Leseoperation ist ein direkter Zugriff auf DAOs aus der Web-Schicht erlaubt. Im Backend können sich auch Utility-Klassen, die keine vollwertigen Services sind, befinden.

Das Frontend, bzw. die Web-Schicht wird auf der Server-Seite mit dem Framework *Apache Wicket* realisiert. Auf der Client-Seite werden die *jQuery* und *jQuery UI* Javascript-Bibliotheken verwendet, sowie der CSS-Framework *Blueprint*.

Die Entwicklung wird so weit wie möglich lokal ablaufen, es wird also kein Entwicklungserver bereitgestellt. Als SQL Datenbank wird *H2* im Embedded-Modus verwendet, und direkt in die Anwendung integriert.

Maven wird für mehrere Zwecke verwendet: als Build- und Dependency Management Tool; um die Projektreports und Javadocs zu generieren; und um die Anwendung bei der Entwicklung und Demonstration zu starten.

Andere verwendete Bibliotheken sind JUnit für Unit Tests und SLF4J / Logback für die Protokollierung. Der Source Code wird mit Git auf Github verwaltet; es wird auch der mit SCM integrierte Issue Tracker, und der Code Review Tool verwendet.

Projektabgrenzung

Im Zug des Projekts wird keine API entwickelt, die einen externen Zugriff auf die Datenbank möglich macht. allerdings wäre diese Funktion für weitere Iterationsschritte sehr interessant.

Im Lauf der Entwicklung wird das Projekt nicht online gestellt, daher entfallen das Einrichten eines Web- und Datenbankservers und die Miete für diese.

Ein Forum oder eine ähnliche Kommunikationsfunktion für die User wird vorerst ebenfalls nicht integriert, hier wird auf bereits bestehende Communities verwiesen.

Außerdem wird vorerst auf einen komplexen Eventkalender verzichtet.

Zielgruppen

Zielgruppen der Applikation sind einerseits Leute, die sich nur informieren möchten, andererseits auch solche, die bereit sind einen aktiven Beitrag zu leisten und solche, die selbst Events verwalten.

Suchende Personen, die ein ansprechendes (Abend-) Programm in ihrer Umgebung suchen, aber noch keine konkreten Pläne haben. Diesen Leuten soll bei der Entscheidung geholfen werden. Genutzt werden wird das Programm vermutlich vor allem von jüngeren Menschen, da diese eher für derartige Hilfsmittel zu begeistern sind.

Reviewer Die Teilmenge der Anwender, die bereit sind qualitativ hochwertige Reviews zu erstellen und aktiv in der Community tätig sein wollen.

Promoter Personen, die entweder Events organisieren, Bands managen oder Locations betreiben. Diese tragen Events ein und verwalten das laufende Programm ihrer Locations.

Featureliste

- Eventsuche & Feedbacksuche: Jede Benutzer kann Events nach verschiedenen Kriterien (Genre, Location, Entfernung, Empfehlungen, Rating, attending Friends, etc.) suchen und dazu Reviews anderer Nutzer betrachten. Benutzer können auch besonders hilfreiche Reviews anderer Benutzer als solche bezeichnen - diese werden dann bei der Anzeige hervorgehoben.
- Event / Feedback/ Location Verwaltung: eingeloggte User können Events, Reviews und Locations erstellen und ihre eigenen Komponenten bearbeiten und wieder löschen. Admins können auch Events, die sie nicht selbst erstellt haben, bearbeiten und gegebenenfalls löschen. Zusätzlich können Bilder und Videos zu Events, Locations hinzugefügt werden.
- Facebook-Login: Eine Verknüpfung mit Facebook, die eine erleichterte Form des Login darstellt soll neben dem klassischen Login implementiert werden.
- Recommendation System: Zur Erleichterung der Auswahl der verfügbaren Events können registrierte Benutzer Empfehlungen erhalten. Benutzer erhalten Empfehlungen beispielsweise nach den Bewertungen der anderen User. Eine andere Möglichkeit ist eine Funktion, die Events empfiehlt, die von befreundeten Usern besucht werden. Befreundete User können über die Verlinkung mit Facebook gefunden werden. Auch wäre hier denkbar, dass das System Events empfiehlt, die von Leuten mit einem ähnlichen Musikgeschmack gut bewertet wurden, vorschlägt. Der Recommender soll also nicht nur rein auf numerischen

Bewertungen basieren, sondern auch über komplexe Möglichkeiten verfügen, dem User behilflich zu sein.

- Anbindung an Google Maps: Registrierte Benutzer,die ihre Adresse in das System eingegeben haben, können Locations in ihrer Umgebung aufspüren und auf einer Karte anzeigen. Dabei können für die anzuzeigenden Locations verschiedene Kriterien angegeben werden (Entfernung, Genre, Art der Location, etc.)
- Invitations: Registrierte User sollen die Möglichkeit haben, andere User entweder über das Portal oder per Email zu einem Event einzuladen. Die Einladung per Email kann auch an nicht registrierte User erfolgen. User haben auch die Möglichkeit, Events in ihrem Facebook-Profil zu teilen.

Funktionale Anforderungen

Login/Logout

Beschreibung: Der Benutzer kann sich entweder über ein Konto anmelden, das er zuvor angelegt hat, oder er benötigt ein Facebook-Konto, um sich einloggen zu können. Er gibt den Benutzernamen und das Passwort seines Kontos ein. nach erfolgtem Login kann der User das Portal gemäß seinen Berechtigungen nutzen.

Vorbedingung: Um sich einzuloggen, der Benutzer darf nicht schon eingeloggt sein. Er kann sich nur ausloggen, wenn er eingeloggt ist.

Eventsuche

Beschreibung: Jeder Benutzer kann Events nach verschiedenen Kriterien (Genre, Adresse, Entfernung, Empfehlungen, Rating, etc.) suchen. Die Suchergebnisse werden nach ihrer Qualität in einer Tabelle dargestellt. Die Tabelle enthält Spalten mit allen wichtigen Informationen für diese Veranstaltung(Name, Ort, Genre, Datum, Bewertung, etc.).

Neue Veranstaltung erstellen

Beschreibung: User können neue Veranstaltungen erstellen, die dann zum System hinzugefügt werden. Beim Erstellen einer neuen Veranstaltung müssen alle wichtigen Informationen für diese Veranstaltung angegeben werden. (Name, Adresse, Genre, etc.).

Vorbedingung: Um eine neue Veranstaltung zu erstellen, muss der User eingeloggt sein.

Veranstaltung bearbeiten

Beschreibung: User können Name, Adresse, Genre oder Datum ihrer Veranstaltungen bearbeiten. Neue Daten werden im System gespeichert. Administratoren können alle Veranstaltungen bearbeiten.

Vorbedingung: Um eine Veranstaltung zu bearbeiten, muss der User beziehungsweise der Administrator eingeloggt sein.

Veranstaltung löschen

Beschreibung: User können ihrer Veranstaltungen löschen. Diese werden dann aus dem System entfernt. Administratoren können alle Veranstaltungen löschen.

Vorbedingung: Um eine Veranstaltung zu löschen, muss der User beziehungsweise der Admin eingeloggt sein.

Videos und Bilder hinzufügen

Beschreibung: User können Videos und Bilder zu einer bestimmten Veranstaltung hinzufügen, die dann zum System hinzugefügt werden. Diese werden dann auf der Event-Seite angezeigt.

Vorbedingung: Um neue Videos und Bilder hinzufügen, muss der User eingeloggt sein.

Neues Feedback erstellen

Beschreibung: Ein eingeloggter Benutzer kannein Feedback zu einer Veranstaltung verfassen. Dieses wird gespeichert, und einer bestimmten Veranstaltung zugeordnet.

Vorbedingung: Um ein neues Feedback zu erstellen, muss der User eingeloggt sein.

Feedback bearbeiten

Beschreibung: User können ihre Feedbacks bearbeiten. Änderungen werden dann im System gespeichert. Administratoren können Feedbacks immer bearbeiten.

Vorbedingung: Um ein Feedback zu bearbeiten, muss der User oder Admin eingeloggt sein.

Feedback löschen

Beschreibung: User können ihre Feedbacks löschen. Dies wird dann aus dem System entfernt. Admins könne jedes Feedback löschen

Vorbedingung: Um ein Feedback zu löschen, muss der User beziehungsweise Admin eingeloggt sein.

Empfehlungen erhalten

Beschreibung: Zur Erleichterung der Auswahl der verfügbaren Events können User Empfehlungen erhalten. Benutzer erhalten Empfehlungen beispielsweise aufgrund von Bewertungen anderer User. Eine andere Möglichkeit ist eine Funktion, die Events empfiehlt, die von befreundeten Usern besucht werden. Befreundete User können über die Verlinkung mit Facebook gefunden werden. Auch wäre hier denkbar, dass das System Events empfiehlt, die von Leuten mit einem ähnlichen Musikgeschmack gut bewertet wurden.

Vorbedingung: Der Benutzer muss eingeloggt sein.

Anbindung an Google Maps

Beschreibung: Registrierte Benutzer, die ihre Adresse in das System eingegeben haben, können Locations in ihrer Umgebung aufspüren und auf einer Karte anzeigen. Dabei können für die anzuzeigenden Locations verschiedene Kriterien angegeben werden (Entfernung, Genre, Art der Location, etc.)

Andere User einladen
Beschreibung: Benutzer sollen die Möglichkeit haben, andere User entweder über das Portal oder per Email zu einem Event einzuladen. Die Einladung per Email kann auch an nicht registrierte User erfolgen. User haben auch die Möglichkeit, Events in ihrem Facebook-Profil zu teilen.
Vorbedingung: Um andere Benutzer einzuladen, muss der User eingeloggt.

Nichtfunktionale Anforderungen

Benutzbarkeit: Die Benutzeroberfläche muss so gestaltet sein, dass sich die Benutzer möglichst ohne Einarbeitungszeit zurechtfinden und nach kurzer Einarbeitungszeit alle Funktionen verstanden haben und sie auch nutzen können. Die Benutzerhinweise müssen in Form von kurzen und prägnanten Sätzen ausgedrückt werden. Auf Fehleingaben des Benutzers muss mit einer entsprechenden Fehlermeldung reagiert werden.

Änderbarkeit / Wiederverwendbarkeit: Es sollen, soweit möglich, beim Design der Applikation standardisierte Software Patterns benutzt werden.

Aussehen und Handhabung: Das User Interface muss schlicht und übersichtlich gestaltet sein. Auf „Überladung“ mit Features wird verzichtet. Das User Interface muss ein konsistentes Design für alle Benutzerinteraktionen bereitstellen.

Sicherheitsanforderungen: Für reguläre Benutzer sollte es nicht möglich sein, mit Hilfe von Manipulation von URLs oder Formparametern Einträge anderer Benutzer zu editieren oder zu löschen. Außerdem soll es nicht möglich sein, auf interne Daten der Applikation (zB. Benutzerrollen anderer Benutzer) zuzugreifen.

Performance: Die Anwendung sollte keine gravierende Performance- bzw. Skalierungsanomalien aufweisen. Beim Deployment auf dem Laptop eines Teammitgliedes, und simulierter Belastung über Computer der anderen Mitglieder, soll die Anwendung ohne auffällige Verzögerung reagieren.

Icebergliste

ID	Feature	Anwendungsfälle	Priorität	Version	Aufwand(in Tagen)	Verantwortlichkeit
1.1	EventSuche	Kriterien für EventSuche wird eingeben	H	1	2	D
1.2	EventSuche	Such Ergebnisse anzeigen	H	1	2	E
1.3	EventSuche	Reviews Betrachten	H	1	1	M
2.1	Event Verwaltung	Event erstellen	H	1	2	A
2.2	Event Verwaltung	Event bearbeiten	M	1	2	J
2.2	Event Verwaltung	Event löschen	M	1	1	P
2.3	Event Verwaltung	Videos/Bilder hinzufügen	N	1	3	D
3.1	Location Verwalten	Location erstellen	H	2	2	P
3.2	Location Verwalten	Location bearbeiten	H	2	1	A
3.3	Location Verwalten	Location löschen	M	2	1	J
3.4	Location Verwalten	Videos/Bilder hinzufügen	N	2	2	A
4.1	Feedback Verwalten	Feedback erstellen	H	2	1	M
4.2	Feedback Verwalten	Feedback bearbeiten	M	2	2	E
4.3	Feedback Verwalten	Feedback löschen	N	2	1	J
5.1	Facebook-Login	Verknüpfung mit Facebook	H	2	3	D
6.1	Recommendation System	Statistik von Event Bewertungen	H	3	4	J
6.2	Recommendation System	Event empfehlen durch Bewertung	H	3	4	D
6.3.1	Recommendation System	Event empfehlen durch besuchte Events von freunde	H	3	3	E
6.3.2	Recommendation System	Event empfehlen durch ähnlicher Musikgeschmack	N	3	3	M
7.1	Anbindung an Google Maps	Adresse speichern	H	4	4	J
7.2	Anbindung an Google Maps	Umgebung aufspüren	H	4	3	P
7.3	Anbindung an Google Maps	auf Karte anzeigen	H	4	3	D
7.4	Anbindung an Google Maps	Angegebene Kriterien speichern	M	4	3	A

8.1	Invitations	Registrierte User einladen	H	4	3	M
8.2	Invitations	Nicht Registrierte User einladen	M	4	2	E

Lieferumfang & Abnahme

Bei den Milestone-Abgaben und dem Abschluss des Projektes werden folgende Artefakte (entsprechend den Meilensteindefinitionen) übergeben:

WAR-Archiv: Die Anwendung, archiviert als WAR-Datei, zum Einsatz auf einem Java EE –konformen Application Server geeignet. Der gewählte Technologie-Stack verlangt nur Unterstützung der Servlet Spezifikation in der Version 2.5.

Datenbank-Schema: In der Anwendung wird das Hibernate Framework das Datenbank-Schema generieren und verwalten. Um ein Übersicht über dem Relational-Modell zu bieten wird aber der aktuelle Zustand des Schema exportiert und übergegeben.

Demo-Datensatz: Für Demonstrationszwecken wird eine, mit Testdaten befüllte, H2 Datenbank-Datei bereitgestellt. Für Betrieb ohne Testdaten sollte keine Datenbankinitialisierung notwendig sein.

Source Code: Ein Schnappschuss des vollständiges Quelltextes, inklusive vollständige Historie (Git Repository), der Applikation, wie sie beim Milestone vorgeführt wird.

Projektdokumentation: ein Ausdruck der Projektdokumentation entsprechend der Meilensteindefinition. Unter anderem sind ist das die Dokumentation zum Entwurf der Applikation (Domänenmodell, Use Cases usw.), die zugehörige Diagramme, Testplan, Testfälle und Testberichte.

Abgrenzung

Folgendes wird **nicht** Teil des Lieferumfangs:

Einsatz in Produktiv-Umgebung: Konfigurationsdateien, die den Einsatz in einer anderen Umgebung als einer Demonstration auf einem beliebigem Servlet Container ermöglichen, werden nicht bereitgestellt. Die Anwendung wird trotzdem so strukturiert, dass die umgebungsabhängige Konfiguration isoliert bleibt und möglichst einfach zu ändern ist.

Vollständige Historie der Dokumentation: Um Kollaboration zu erleichtern, wird die Projektdokumentation mit Hilfe von Google Docs verwaltet und laufend aktualisiert. Deswegen werden nur Schnappschüße davon versioniert.

Meilensteine

Bei jedem Meilenstein werden die Artefakte des Projektmanagements abgegeben. Es wird immer auch eine ergänzte und aktualisierte Version der Dokumentation, die bei den vorherigen Meilensteinen beschrieben ist, abgegeben.

MS-1

1. Projektauftrag (dieses Dokument).

IR-1

1. Pre-Alpha-Version der Applikation: der Technologie-Stack ist konfiguriert und integriert, und wird für die Implementierung eines “hello-world” Prototyps angewendet.
2. Entwurf:

a. Funktionale Anforderungen
b. Komponentendiagramm
c. UI-Skizzen
d. Deployment Diagramm
e. Datenmodell

Testartefakte

. Testplan
a. Akzeptanztests

MR-2

1. Software: Alpha-Version. Features sind zu 40% implementiert und die DAO Tests sind vollständig.
2. Zu den Testartefakten kommt auch der Testbericht, der den Zustand zum MR-2 beschreibt.
3. Zum Entwurf kommt das Klassendiagramm hinzu.

IR-2

1. Software: Beta-Version. Features sind zu 70% vollständig, und die Servicetests sind vollständig.

MR-3 - Projektabnahme

1. Software: Release Candidate. Implementierung des Produktes ist vollständig und bereit zum QA-Einsatz.
2. Präsentation, die der Produkt vorstellt.

Projektstrukturplan

ID	Arbeitspaket	Anfang	Ende	Aufwand (h)	Verantwortlicher
MS 0	Kick Off & Projektvorschlag	15.Mär	18.Mär	10	
1.1	**Projektauftrag**	19.Mär	25.Mär		alle
1.1.1	Featureliste zu Icebergliste verfeinern	19.Mär	25.Mär	5	A,E,M
1.1.2	Ausgangssituation & Beschreibung aus P verfeinern	19.Mär	25.Mär	1	J
1.1.3	funktionale Anforderungen & User Stories	19.Mär	25.Mär	5	A,E,M
1.1.4	Rollenbeschreibungen	19.Mär	25.Mär	1	J
1.1.5	Domänennodell	19.Mär	25.Mär	1	J
1.1.6	Projektabgrenzung	19.Mär	25.Mär	2	P
1.1.7	nichtfunktionale Anforderungen	19.Mär	25.Mär	1	A,E,M
1.1.8	Lieferumfang & Abnahme	19.Mär	25.Mär	1	D
1.1.9	Projektstrukturplan	19.Mär	25.Mär	3	J
1.1.10	Risikoabschätzung	19.Mär	25.Mär	2	J
1.1.11	Informationswesen	19.Mär	25.Mär	1	J
1.1.12	Komponentendiagramm & Architekturübersicht	19.Mär	25.Mär	2	D
1.2	**Stundenlisten**	19.Mär	25.Mär		alle
1.3	**Statusbericht**	19.Mär	25.Mär	1	J
1.4	**Präsentation**	19.Mär	25.Mär	2	
1.5	**Protokolle**	19.Mär	25.Mär	1	J, P
MS 1	Projektauftrag	19.Mär	25.Mär		
MR 1	erstes Managmentreview				
2.1	**Projektauftrag verfeinern**	26.Mai	14.Apr		alle

2.1.1	Anwendungsfälle fertigstellen	26.Mai	14.Apr	3	A,E,M
2.1.2	Akteurhierarchie	26.Mai	14.Apr	1	
2.1.3	Kompononentendiagramm verfeinern	26.Mai	14.Apr	1	D
2.1.4	UI-Skizzen & Beschreibung	26.Mai	14.Apr	5	
2.1.5	Verteilungsdiagramm	26.Mai	14.Apr	2	
2.2	**Prototyp**	26.Mai	14.Apr	150	alle
2.2.1	Repository einrichten	26.Mai	14.Apr		D
2.2.2	DB-Schema erstellen und erste Daten einfügen	26.Mai	14.Apr		
2.2.3	Persistenzschicht & DAOs erstellen	26.Mai	14.Apr		
2.2.4	Interaktionsschicht erstellen	26.Mai	14.Apr		
2.2.5	erste Tests	26.Mai	14.Apr		
2.3	**Stundenlisten**	26.Mai	14.Apr		alle
2.4	**Protokolle**	26.Mai	14.Apr	1	J, P
2.5	**verfeinerte Risikoabschätzung**	26.Mai	14.Apr	2	J
2.6	**Statusbericht**	26.Mai	14.Apr	1	J
2.7	**Strukturplan verfeinern**	26.Mai	14.Apr	2	J
2.8	**Testplan**	26.Mai	14.Apr		E
2.8.1	Testplan erstellen	26.Mai	14.Apr	3	E
2.8.2	erste Testfälle erstellen (funktionale Tests)	26.Mai	14.Apr	2	E
2.9	**DB Beschreibung & Datenbankdiagramm**	26.Mai	14.Apr	3	
2.10	**Statusbericht**	26.Mai	14.Apr	1	J
MS 2	Pre - Alpha Version				
IR 1	erstes internes Review				
3.1	**Stundenlisten**	15.Apr	09.Mai		alle
3.2	**Protokolle**	15.Apr	09.Mai	1	J,P
3.3	**Risikoabschätzung aktualisieren**	15.Apr	09.Mai	2	J
3.4	**Strukturplan abschliessen**	15.Apr	09.Mai	3	J
3.5	**Statusbericht**	15.Apr	09.Mai	1	J

3.6	**Testartefakte**	15.Apr	09.Mai		E
3.6.1	Tesplan fertigstellen	15.Apr	09.Mai	3	E
3.6.2	weitere funktionale Tests	15.Apr	09.Mai	3	E
3.6.3	Testberichte	15.Apr	09.Mai	2	ALLE
3.7	**Anwendungsfälle fertigstellen**	15.Apr	09.Mai	3	
3.8	**Klassendiagramme**	15.Apr	09.Mai	4	
3.9	**DB Beschreibung fertigstellen**	15.Apr	09.Mai	2	
3.10	**Alpha Version**	15.Apr	09.Mai		
3.10.1	Implementierung gemäß Icebergliste	15.Apr	09.Mai	150	ALLE
3.11	**UI Skizzen fertigstellen**	15.Apr	09.Mai	4	
3.12	**Verteilungsdiagramm fertigstellen**	15.Apr	09.Mai	2	
3.13	**Komponentendiagramm fertigstellen**	15.Apr	09.Mai	2	D
3.14	**Präsentation**	15.Apr	09.Mai	2	
MS 3	Alpha Version				
MR 2	zweites Managementreview				
4.1	**Stundenlisten**	10.Mai	30.Mai		ALLE
4.2	**Protokolle**	10.Mai	30.Mai	1	J,P
4.3	**Statusbericht**	10.Mai	30.Mai	1	J
4.4	**Risikoabschätzung verfeinern**	10.Mai	30.Mai	2	J
4.5	**Klassendiagramme erweitern**	10.Mai	30.Mai	2	
4.6	**Testartefakte**	10.Mai	30.Mai		E
4.6.1	funktionale Testfälle erweitern	10.Mai	30.Mai	3	E
4.6.2	Testberichte	10.Mai	30.Mai	5	ALLE
4.7	**Beta Version**	10.Mai	30.Mai		
4.7.1	Implementierung gemäß Icebergliste	10.Mai	30.Mai	150	ALLE
MS 4	Beta Version	10.Mai	30.Mai		
IR 2	zweites internes review	10.Mai	30.Mai		

5.1	**Stundenlisten**	01.Jun	20.Jun		ALLE
5.2	**Protokolle**	01.Jun	20.Jun	1	J,P
5.3	**Endbericht**	01.Jun	20.Jun	2	J
5.4	**Risikoabschätzung fertigstellen**	01.Jun	20.Jun	2	J
5.5	**Testartefakte**	01.Jun	20.Jun		
5.5.1	funktionale Testfälle fertigstellen	01.Jun	20.Jun	2	E
5.5.2	Testberichte fertigstellen	01.Jun	20.Jun	5	ALLE
5.6	**Präsentation**	01.Jun	20.Jun	1	
5.7	**Release Candidate**	01.Jun	20.Jun		
5.7.1	Implementierung gemäß Icebergliste	01.Jun	20.Jun	150	ALLE
MS 5	Release Candidate Version				
MR 3	drittes Managementreview				

Risikoabschätzung

ID	Titel	Beschreibung	Typ	Priorität	WS	Verantwortlicher	Gegenstrategie
1	Ausfall	Ein Teammitglied fällt für längere Zeit oder auch für immer aus	immer	hoch	mittel	PL	neu Zuteilung der Arbeitspakete, ausgefallene Expertenrolle neu zuweisen.
2	Aufgabenverteilung	Es gibt teaminterne Probleme bei der Aufteilung der Arbeitspakete	immer	mittel	mittel	PL	PL muss die Aufgaben fair verteilen, Gruppenmitglieder müssen sich auch fügen können
3	Probleme mit Tools	ein GM hat Probleme mit der verwendeten Technologie	implementierung	hoch	mittel	GM	Das GM muss sich rechtzeitig in die Technologien einarbeiten, ein anderes GM kann dabei helfen
4	Zusammenführung vor Deadline	GM sind vor der Deadline nicht erreichbar und ihre Arbeitsteile nicht verfügbar	immer	hoch	mittel	GM	regelmäßiger Sync mit dem Repository & Onlinestellen der Dokumente
5	Kommunikation	die Kommunikation in der Gruppe funktioniert nicht wie gewünscht	immer	hoch	hoch	GM	regelmäßiges Beobachten der GGroup, wenn am PC in Skype verfügbar sein, Handynummern austauschen
6	unvollständige Dokumente	Dokumente fehlen oder sind nicht vollständig	immer	hoch	hoch	DB & PL	vor einer Deadline muss der DB od. PL die Dokumente auf Vollständigkeit prüfen
7	Verlust von Daten	Code oder andere Dokumente gehen durch einen Crash verloren	management	hoch	niedrig	GM & TA	das GM muss für seine Teile ein Backup führen, der TA soll regelmäßig Backups des Repository durchführen

8	zeitlicher Engpass	das rechtzeitige Fertigstellen ist in Gefahr (Ausfall von GM, techn. Schwierigkeiten, etc.)	implementierung	hoch	mittel	PL, TA, GM	die Arbeitspakete müssen angemessen verteilt werden, GM müssen genügend Zeit einplanen
9	Technologie nicht verfügbar	Ein Teil der benötigten Technologie ist nicht (frei) verfügbar	implementierung	mittel	niedrig	TA	passenden Ersatz finden oder das Feature entsprechend umfunktionieren
10	Codeverständnis	GM haben Probleme den Code anderer GM zu verstehen	implementierung	niedrig - mittel	hoch	GM	den Code angemessen kommentieren, bei Bedarf Erklärungen durch den Programmierer
11	Zeitverlust durch Bugs	Fehlerhafter Code und das Auffinden der Fehler verzögert das Vorankommen	implementierung	hoch	hoch	GM, TV	regelmäßige UnitTests durchführen und Code kommentieren

PL Projektleiter
TA technischer Architekt
DB Dokumentenbeauftragter
GM Gruppenmitglied
TV Testverantwortlicher

Informationswesen

Um die Kommunikation in der Gruppe zu gewährleisten, werden verschiedene Möglichkeiten verfolgt. Zum einen gibt es regelmäßige interne Treffen der Gruppe (Zeitpunkt wird jedesmal neu beschlossen). Weiters wird es regelmäßige Treffen mit dem Tutor und die verschiedenen Reviews mit der LVA-Leitung geben.

Als elektronische Kommunikationswege steht eine Google Group zur Verfügung, in die auch der Tutor eingeladen wurde. Außerdem gibt es eine Skype Gruppendiskussionen sowie individuelle Absprachen per Telefon.

Die Kommunikation mit der LVA-Leitung erfolgt über den Projektleiter in Form von Mails, wobei der Tutor in die schon erwähnte Google Group antwortet, damit alle Gruppenmitglieder das Feedback gleichzeitg erhalten und ein weiterer Zwischenschritt umgangen wird.

Für die Versionierung des Codes steht ein GitHub Repository zur Verfügung, auf den alle Gruppenmitglieder Schreibtrechte haben und der Tutor per Lesezugriff Zugang erhalten wird.