

# Semesterarbeit NDS HF Applikationsentwicklung

## TRIP PLANNER

D.Biedermann / R.Kaufmann / 08.01.2017

## Inhaltsverzeichnis

<b>Inhaltsverzeichnis</b>	<b>1</b>
<b>Termine</b>	<b>1</b>
<b>Beschreibung</b>	<b>1</b>
<b>Komponenten</b>	<b>3</b>
<b>Funktionale Anforderungen / User Stories</b>	<b>3</b>
<b>Nicht funktionale Anforderungen</b>	<b>12</b>
<b>Klassendiagramm</b>	<b>13</b>
<b>Branch Konzept</b>	<b>14</b>
<b>Scrum</b>	<b>17</b>
23.12.2016 Stand Sprint 1	17
07.01.2017 Stand Sprint 1	18
<b>Datasets</b>	<b>19</b>
<b>Libraries</b>	<b>19</b>

## Termine

- 08.01.2017 Abgabe Projektskizze / Anforderungsdokumentation
- 26.01.2017 Präsenzveranstaltung zur freien Arbeit an der Semesterarbeit
- 31.01.2017 Präsenzveranstaltung zur freien Arbeit an der Semesterarbeit
- 29.01.2017 Abgabe Git Repository
- 02.02.2017 Präsentation der Semesterarbeit

## Beschreibung

### Version 1, per 2.2.2017

Mit der Trip Planner Applikation kann ein Benutzer seine individuelle Reise planen und grafisch darstellen.

Dazu werden vom Administrator CSV Datenfiles mit Orten und Point-of-Interest (POI) in Datenbank-Tabellen eingelesen.

Grundsätzlich werden diese CSV Files immer wieder aktualisiert



müssen vom Administrator auf periodischer Basis neu eingelesen und automatisch mit der Tabelle abgeglichen werden. Für den CSV Import nutzt der Administrator ein eigenes Admin-GUI.

Der Benutzer loggt sich ein und beginnt mit der Erstellung einer neuen Reise. Danach sucht er sich aus den Orte/POI Tabellen die für seine Reise interessanten Orte und Sehenswürdigkeiten aus und speichert diese in seiner Reiseaktivitätenliste. Diese wird ihm tabellarisch und auf einem GoogleMap-Panel grafisch angezeigt. Diese Aktivitäten können mittels Pfeilen sortiert werden.



#### Version 2, ab 2.2.2017 (Nach Projekt)

Die Bearbeitung von Reise und Reiseaktivitäten sowie auch eine Registrierungsmöglichkeit für neue Benutzer. Allenfalls auch noch eine Exportfunktion der gesamten Reise als PDF.

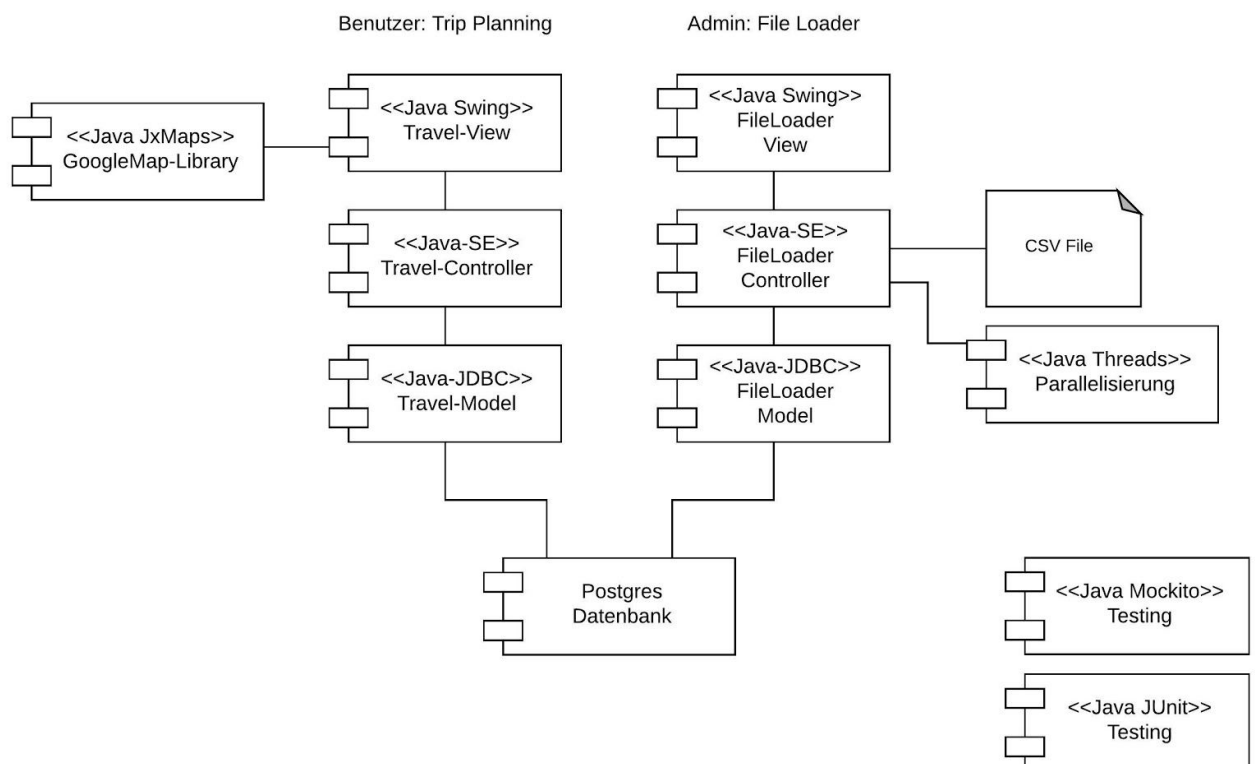
User Stories aus der Version 2 können gegebenenfalls in Version 1 einfließen, falls Version 1 frühzeitig fertig implementiert ist. Der Product Owner definiert in diesem Falle welche User Stories noch umgesetzt werden sollen.

# Komponenten

UML Werkzeug für das Komponentendiagramm:

<https://www.lucidchart.com/documents/edit/243b6d9d-7d1b-4da7-8696-394787cbf1bb>

Die Applikation basiert auf den im ersten Semester erlernten Komponenten mit Java. Dies beinhaltet im Moment auch ein Swing GUI. Natürlich könnte dieses GUI später auch durch ein Web-basiertes Java GUI, wie zum Beispiel "Vaadin", ersetzt werden.



## Funktionale Anforderungen / User Stories

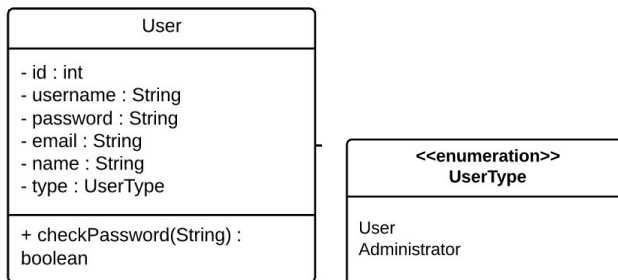
ID	Titel	Story Points
	Beschreibung	
	Akzeptanzkriterien	
IBW1-11	Als Benutzer möchte ich mich einloggen können, damit ich meine personalisierten Reisen erfassen und bearbeiten kann.	2
	Eine Maske zum Anmelden mit Username und und Passwort wird benötigt. Username und Passwort muss mit bereits gespeicherten User Daten abgeglichen werden. Mockup für Login Form:	

**Login**

UserId

Password

UML Klassendiagramm für den User:



- \* Eingabe Username/Passwort
- \* Einloggen des Users
- \* Erkennungsmerkmal ob Benutzer oder Admin ersichtlich

IBW-12	Als Administrator möchte ich ein Kategorien CSV-File mittels GUI hochladen können, damit die Point of Interest (POI) Kategorien importiert werden.
--------	--

5

Es braucht ein GUI, in welchem das CSV File ausgewählt und das CSV File importiert werden kann.  
Der Import soll neue Zeilen in der Tabelle CATEGORY einfügen und bereits vorhandene Zeilen mutieren.  
Für den Upload braucht es eine File Auswahl, Auswahl des File Typs (Category oder Location Data) und die Auswahl des Trennzeichens.  
Mockup für die Upload Form:

## Tripp planner administration

Upload a new CSV OSM File to update the category or Point of interest

File content:      ☒ Category      ☐ Location Data

Delimiter:      ☐ ;      ☒ |      ☐ ,

- \* Auswahl eines CSV Files in einem "File öffnen"-Dialog
- \* Alle Zeilen des Files werden in die Tabelle CATEGORY geschrieben.

IBW1-13	Als Administrator möchte ich ein POI CSV-File mittels GUI hochladen können, damit die POI's importiert werden.
---------	--

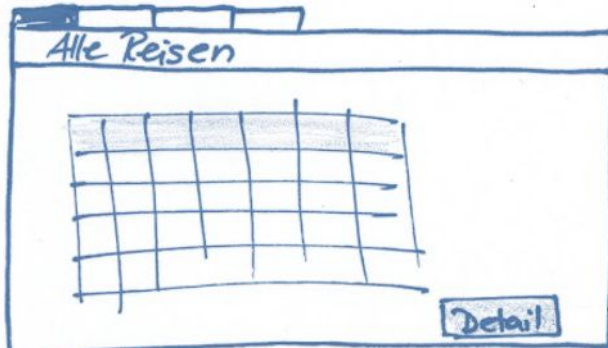
3

Es braucht ein GUI, in welchem ein File ausgewählt werden kann. Dieses File muss dann in die POI Tabelle gespeichert werden (bzw. vorhandene Einträge müssen überschrieben werden).

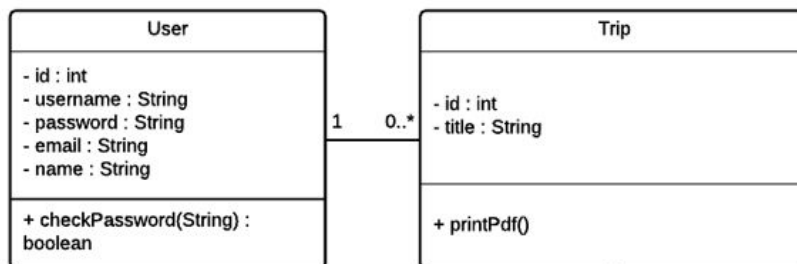
Für den Upload braucht es eine File Auswahl, Auswahl des File Typs (Category oder Location Data) und die Auswahl des Trennzeichens. (Es soll das gleiche GUI wie für das Kategorien File Upload verwendet werden.)

UML Klassendiagramm für den File Upload:

	<div><div><div><div><div>ImportController</div><div>- rowQueue : LinkedList&lt;String&gt; - rowQueueCount : long - counter : long</div></div><div><div>DatabaseProxy</div><div>+ prepareStatement(String) : PreparedStatement + setAutoCommit(Boolean) : void + Commit() : void</div></div></div><div><div>FileReader</div><div>+ run() : void</div></div><div><div>ImportProgress</div><div>+ run() : void</div></div><div><div>CategoryConsumer</div><div>+ run() : void</div></div><div><div>DatabaseImport</div><div>+ insertMultiValuePois(ArrayList&lt;String[]&gt;) : void + insertMultiValueCategories(ArrayList&lt;String[]&gt;) : void</div></div></div></div> <div><ul style="list-style-type: none"><li>* Auswahl eines CSV Files in einem "File öffnen"-Dialog.</li><li>* Alle Zeilen des Files werden in die Tabelle geschrieben.</li><li>* Wenn die CATEGORY Tabelle noch leer ist, muss ein Fehler ausgegeben werden.</li><li>* Wenn es die Kategorie für die Zeile noch nicht in der CATEGORY Tabelle gibt, kann diese Zeile nicht geschrieben werden.</li></ul></div>	
IBW1-30	<div><p>Als Administrator möchte ich den Fortschritt beim Import sehen können, damit ich sehe wie lange es dauert.</p><p>Im Administrator GUI braucht es einen Abschnitt für die Anzeige des Fortschritts. Es sollen die Anzahl verarbeitete Zeilen, das Total Zeilen und die benötigte Zeit angezeigt werden.</p><p>Mockup der Fortschrittsanzeige:</p><div><div>Progress</div><div><div>Total Rows: 12345</div><div>Rows processed: 455</div><div>Rows with Errors: 12</div><div>elapsed time: 1:04</div></div><div></div></div></div> <div><ul style="list-style-type: none"><li>* Anzahl verarbeitete Zeilen und Anzahl Total Zeilen anzeigen</li><li>* Die bereits benötigte Zeit in Sekunden und Minuten anzeigen</li><li>* Den Fortschritt in einem Balken (Progress bar) anzeigen</li></ul></div>	2
IBW1-15	<div><p>Als Benutzer möchte ich alle meine Reisen in einer Liste sehen, damit ich eine Übersicht über aktuelle und vergangene Reisen bekomme.</p><p>Die Liste soll im GUI die bereits erfassten Reisen mit Name, Anzahl der Reiseaktivitäten sowie Start- und Enddatum anzeigen.</p><p>Mockup für Reiseliste Form:</p></div>	2



UML Klassendiagramm für die Reiseliste



\* Alle Reisen des eingeloggten Users werden korrekt angezeigt.

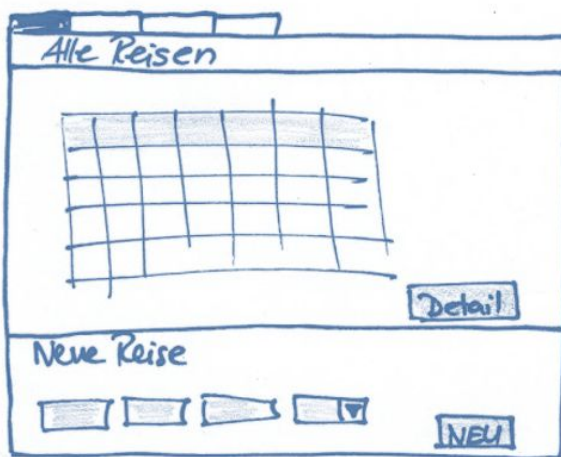
IBW1-16

Als Benutzer möchte ich eine neue Reise erfassen können, damit ich diese für meine Reiseplanung verwenden kann.

3

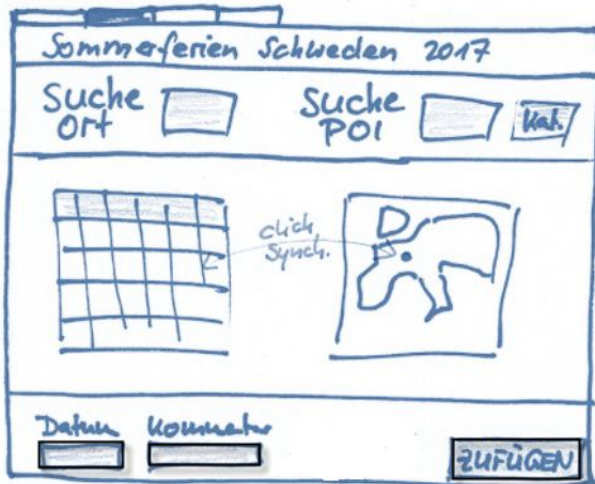
Eine Reise beinhaltet zunächst eine Bezeichnung (und später eine Liste von POIs). Die Bezeichnung möchte ich bei der Erstellung der Reise eingeben und später auch mutieren können.

Mockup für Reise-Neuerfassung (Neue Reise soll unterhalb der Reiseliste direkt erfasst werden können):

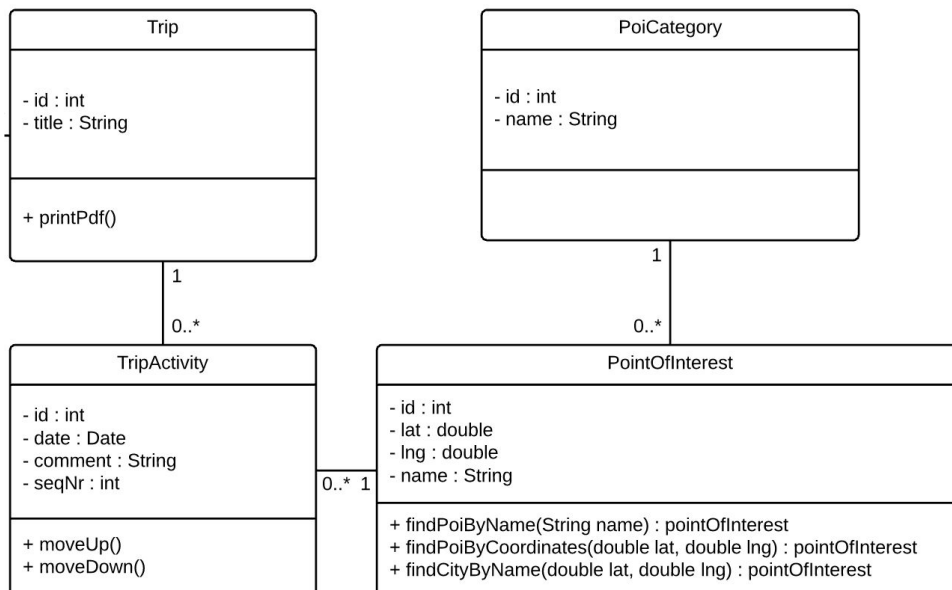


UML Klassendiagramm für die Reiseerfassung:

	<div data-bbox="256 235 1037 490"> <pre> classDiagram     class User {         -id : int         -username : String         -password : String         -email : String         -name : String         +checkPassword(String) : boolean     }     class Trip {         -id : int         -title : String         +printPdf()     }     User "1" -- "0..*" Trip           </pre> </div>	
	<p>* Eine neue Reise kann mit Titel erfasst werden.</p>	
IBW1-17	<p>Als Benutzer möchte ich Reisen löschen können, damit diese nicht mehr in der Liste aufscheinen</p> <p>Die Reise muss aus der Datenbank gelöscht werden. Der Benutzer soll nochmal gefragt werden, ob er die Reise wirklich löschen möchte.</p> <p>Mockup für Reise-Löschung Form (Reise soll innerhalb der Reiseliste direkt gelöscht werden können):</p> <div data-bbox="215 768 807 1227"> </div> <p>UML Klassendiagramm für die Reiselöschung:</p> <div data-bbox="256 1319 1008 1565"> <pre> classDiagram     class User {         -id : int         -username : String         -password : String         -email : String         -name : String         +checkPassword(String) : boolean     }     class Trip {         -id : int         -title : String         +printPdf()     }     User "1" -- "0..*" Trip           </pre> </div>	2
	<p>* Die Reise und die dazugehörigen Aktivitäten werden korrekt gelöscht.</p>	
IBW1-18	<p>Als Benutzer möchte ich Aktivitäten mit Ort und POI zur Reise hinzufügen können, damit die Reiseplanung fortschreitet.</p> <p>Neue Reiseaktivität (Speichern von POIs) mit Datum- und Kommentareingabe (in Liste)</p> <p>Mockup für Erfassung Reiseaktivität:</p>	3



UML Klassendiagramm für die Reiseaktivitäten:



\* Datum und ein Kommentar muss in der Aktivität erfasst werden können.

IBW1-19

Als Benutzer möchte ich nach Ort für die Reiseplanung suchen können, damit ich die nachfolgende POI Suche einschränken kann.

Die Suche wird mit der Eingabe eines Ortes begonnen. So können die POIs auf den Ort (Umkreis wählbar durch Benutzer) eingegrenzt werden. Die gefundenen POIs werden auf der Karte und in einer Liste dargestellt und können markiert werden. Die Markierung erfolgt synchron in der Liste und auf der Karte.

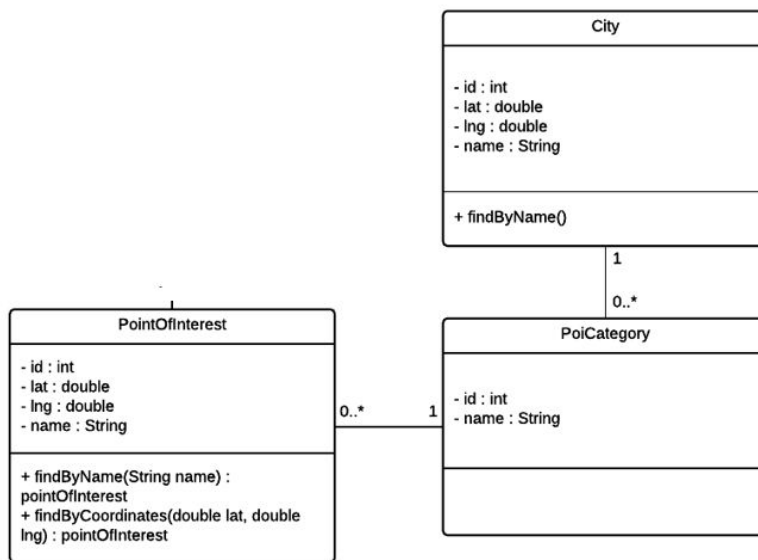
Mockup für Suche von Orten/POI's für Reiseaktivität:

5





UML Klassendiagramm für Orte / POI Suche:



- \* erste Suche zwingend nach Ort
- \* Angabe des Such-Umkreises (5 km, 10 km, 20 km, 50 km)
- \* Die Suchresultate sollen mittels MAP grafisch angezeigt werden
- \* Die Suchresultate sollen in Liste dargestellt werden
- \* Klick auf Eintrag in Liste markiert in Maps und umgekehrt

IBW1-31 Als Benutzer möchte ich nach POI Name und Kategorie im Umkreis des ausgewählten Orts suchen können, damit ich meine relevanten Dinge finde

- \* zweite Suche nach POI innerhalb oder im Umkreis vom Ort (Filter)
- \* Die Suchresultate sollen mittels MAP grafisch angezeigt werden
- \* Die Suchresultate sollen in Liste dargestellt werden
- \* Klick auf Eintrag in Liste markiert in Maps und umgekehrt

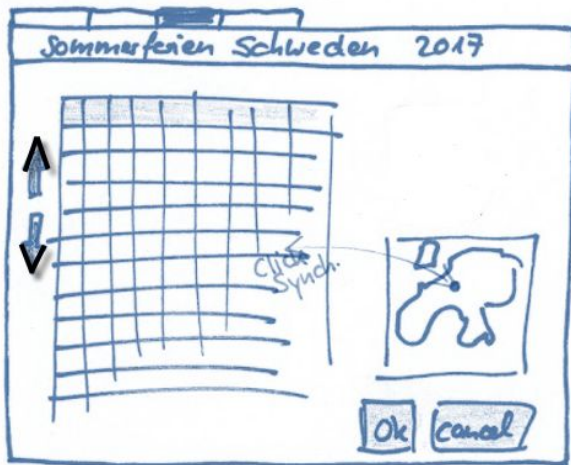
- \* Es werden alle POIs angezeigt, die mit dem eingegebenen Namen oder der Kategorie übereinstimmen.
- \* Die POIs werden nur im Umkreis des ausgewählten Ortes angezeigt.

IBW1-20 Als Benutzer möchte ich eine Aktivitätenliste zur Reise sehen, damit ich diese neu Ordnen kann.

Die einzelnen POIs müssen nach Datum aufgelistet werden. Der Benutzer kann das Datum der einzelnen Aktivitäten ändern und so die Reihenfolge anpassen.  
Mockup für Aktivitätenliste mit Sortierfunktion:

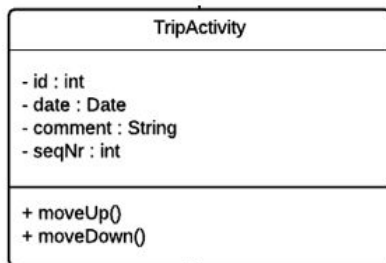
8

5



plus NEU

UML Klassendiagramm:



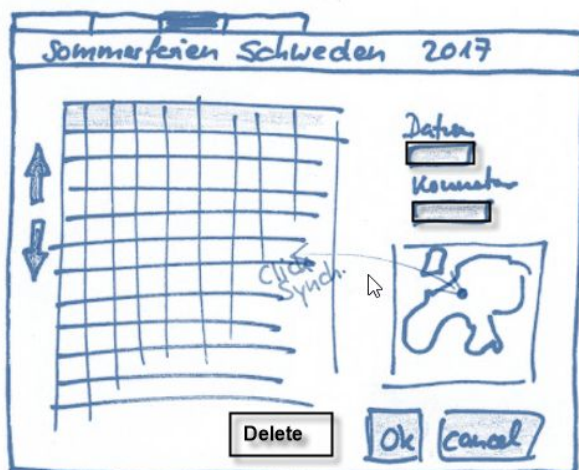
- \* Reiseaktivitätenliste (Gespeicherte POIs als Liste anzeigen)
- \* Reiseaktivitätenliste (Gespeicherte POIs in MAP anzeigen)
- \* Synch Liste und MAP

IBW1-21

Als Benutzer möchte ich bereits erfasste Aktivitäten bearbeiten können, damit ich diese korrigieren kann.

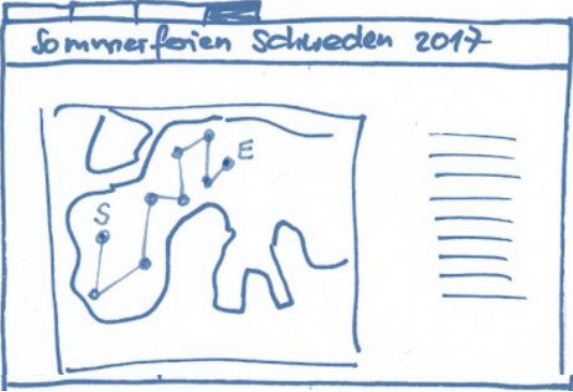

2

Mockup für Aktivitäts-Bearbeitung:



plus NEU

UML Klassendiagramm:

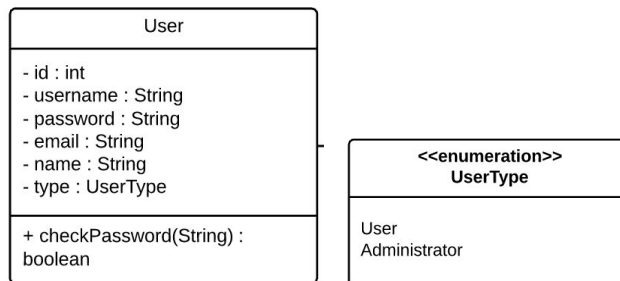
	<div><div><div><div>TripActivity</div><div><div><div>- id : int</div><div>- date : Date</div><div>- comment : String</div><div>- seqNr : int</div></div><div><div>+ moveUp()</div><div>+ moveDown()</div></div></div></div></div></div>	
	<div>* Reiseaktivitätenliste bearbeiten, z.B. Datum, Kommentar ändern, Reiseaktivität löschen</div>	
IBW1-22	<div>Als Benutzer möchte ich zur Reise die gesamte Reiseplanung sehen können, damit ich die einzelne Reise im Gesamten anschauen kann.</div> <div>Grosse Karte mit grafisch verbundenen Reiseaktivitäten Mockup Reiseübersicht:</div> <div></div>	2
	<div>* Auf der grossen Karte werden alle Aktivitäten korrekt verbunden angezeigt.</div>	
IBW1-23	<div>Ich als Benutzer möchte ich meine aktuelle Reiseplanung als PDF exportieren können, damit ich sie ausdrucken und versenden kann.</div> <div>Der PDF Export soll die Karte mit Reiseroute sowie eine Auflistung aller POIs mit Datum, Kommentar, Ort, etc. enthalten. Mockup für PDF Button:</div> <div></div>	3
	<div>* PDF File wird korrekt erstellt.</div>	
IBW1-10	<div>Als Benutzer möchte ich mich registrieren können, damit meine Reisen personalisiert werden können.</div>	2

Eine Eingabemaske zur Erfassung von User-Daten für die Erstanmeldung wird benötigt. Die erfassten Daten müssen gespeichert werden. Es wird zwischen Benutzern und Administratoren unterschieden.

**Registrierung neuer User**

UserId:   
 Password:   
 Name:   
 Email:   
 Type:  ▼

UML Klassendiagramm für den User:



- \* Eingabe von User Daten (Name, Email, Username, Passwort)
- \* Speichern der User Daten (insert)

## Nicht funktionale Anforderungen

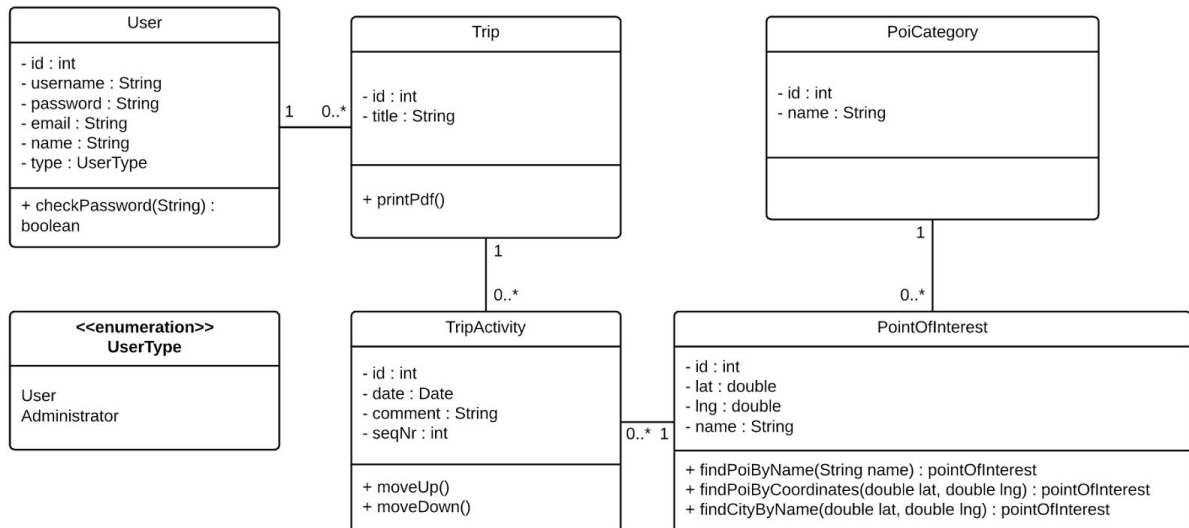
- „Das System muss für Benutzer den unautorisierten Zugriff auf die Reisen anderer Personen verhindern“
- „Das System muss für Administatoren das Laden von CSV Files in max. 30 Minuten ermöglichen“

# Klassendiagramm

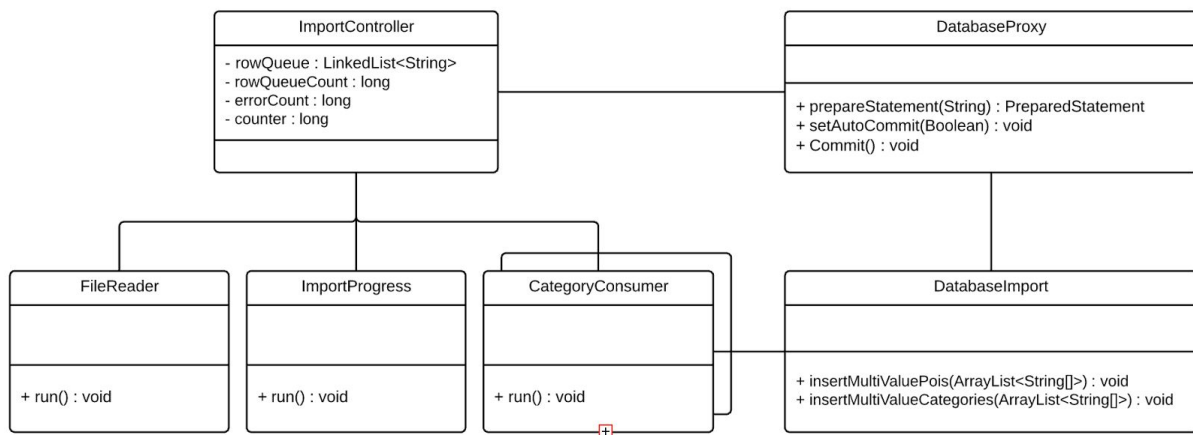
UML Werkzeug für Klassendiagramm:

<https://www.lucidchart.com/documents/edit/c8868b0a-ed08-4ce9-8cd1-a5870883d9d8>

Travel:



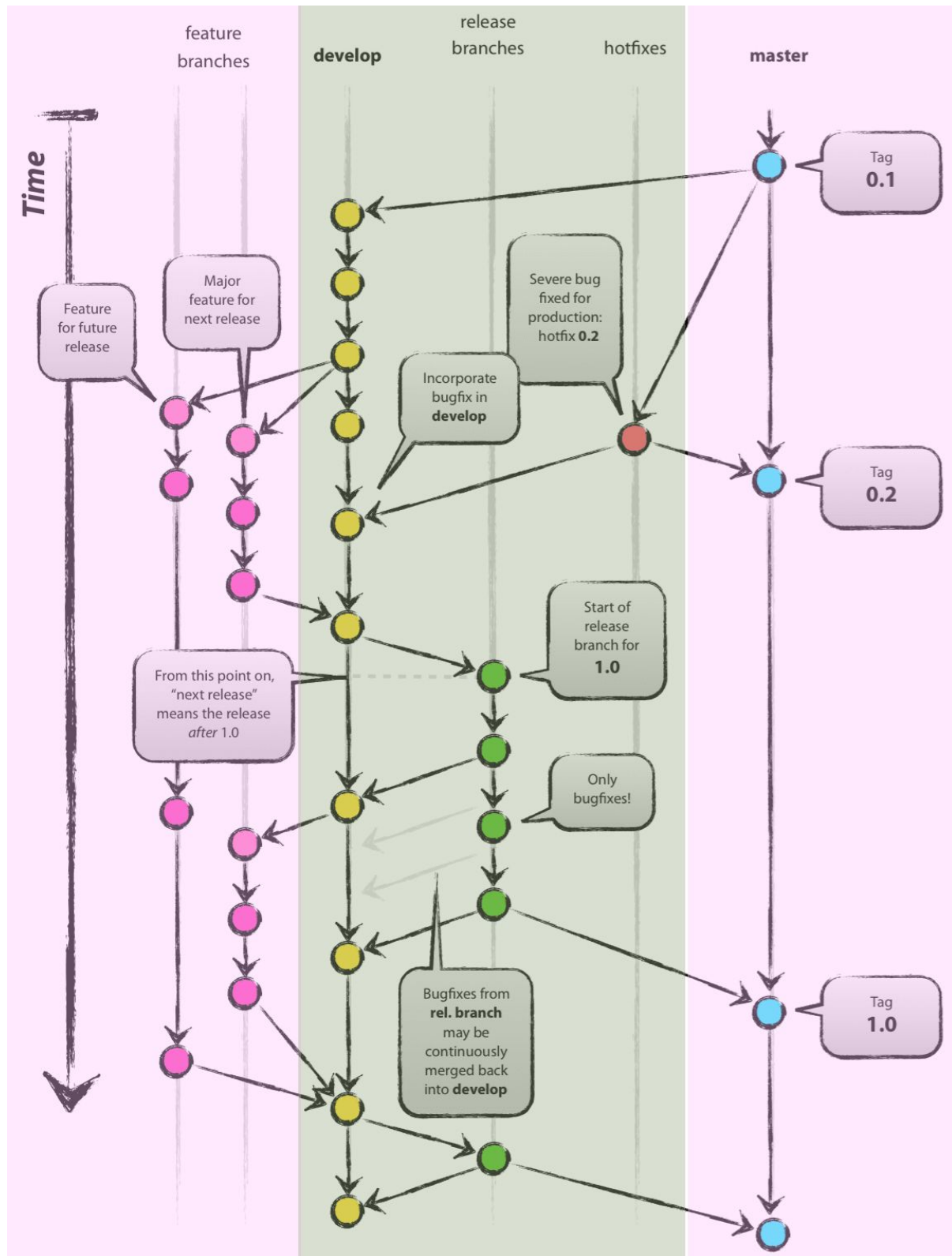
Admin:



# Branch Konzept

Wir verwenden wir einen Master-/Feature-Branch Konzept (siehe Diagramm unten, jedoch ohne den grauen Teil). Dies entspricht in im Grunde dem Konzept aus Git-Flow. Da wir aber zur zwei Entwickler sind und das Projekt eine sehr kurze Dauer hat, wenden wir ein reduziertes Setup an:

- Verzicht auf Hotfix, Release und Develop Branch
  - Verzicht auf die Nutzung von Git Flow als Werkzeug.
- Wir erstellen diese Branches via Standard-GIT Commands



Vor dem erstellen eines neuen Feature-Branch: Pull Changes vom Upstream Master

```
> git checkout master
```

```
> git pull
```

Erstellen neuer Featurebranch (wir stehen auf dem Develop)

```
> git checkout -b <featurename>
```

Vor dem Push des Feature-Branch zurück zum Develop dafür sorgen dass es auf dem Develop (in GitHub!) keinen Mergekonflikt gibt: Dafür ein Pull des Develop machen (neueste Änderungen der Anderen), dann den Develop in den Feature-Branch mergen. Damit erhalten wir allfällige Mergekonflikte auf dem Feature Branch (lokal). Nochmals alle Tests starten. Dann erst in den Master hochpushen.

```
> git commit -m "xx"
```

```
> git pull origin master
```

```
> git merge master --no-ff
```

```
> git push (falls das 1. Mal bei Neuerstellung des Branch: git push origin <featurename>)
```



# Scrum

Scrum Werkzeug: <https://www.vivifyscrum.com> mit Storypoints und Planning-Poker.

Sprint Plan:

- Sprint 1: 20.12.2016 bis 08.01.2017 (danach wöchentlich)
- Sprint 2: 09.01.2017 bis 15.01.2017
- Sprint 3: 16.01.2017 bis 22.01.2017
- Sprint 4: 23.01.2017 bis 29.01.2017
- Danach Präsentationsvorbereitung bis 02.02.2017

Definition Storypoint Referenz: Loginform sind 2 Storypoints

23.12.2016 Stand Sprint 1

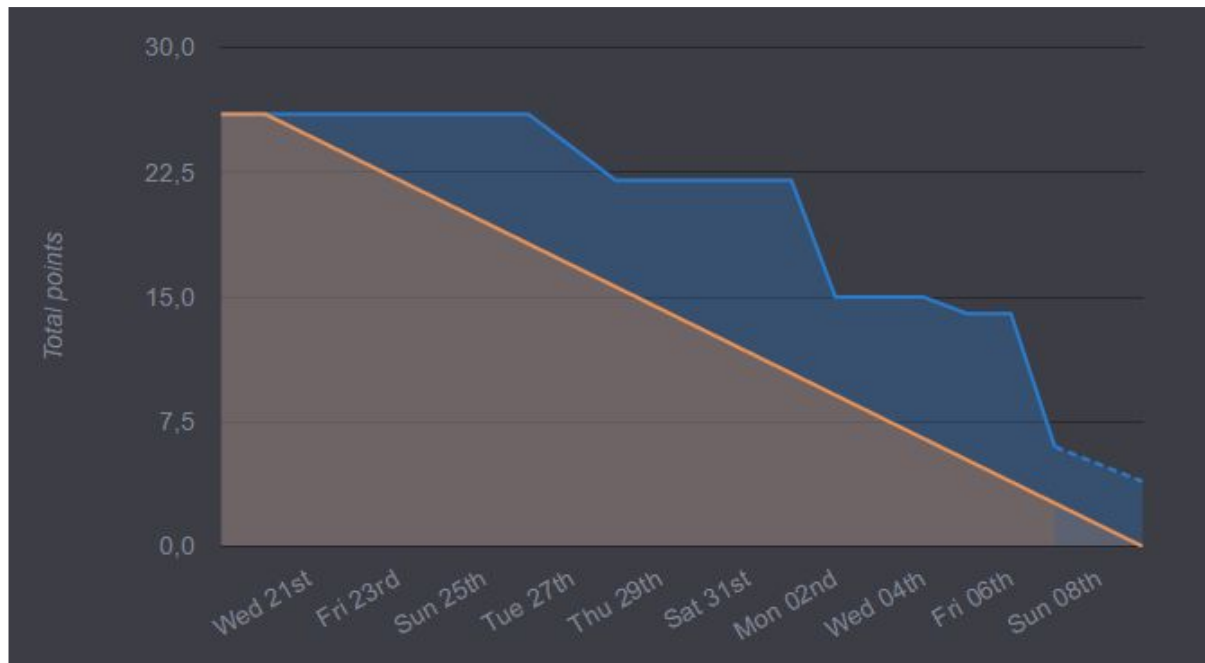
Aktuelles Product Backlog mit Priorisierung, aber noch ohne Storypoints.

The screenshot displays the Vivify Scrum web application interface. On the left, a sidebar contains navigation options: 'Create a new Board', 'Semesterarbeit 2016 Trip Planner' (selected), 'Product Backlog', 'Sprint Backlog', 'Burndown', 'Stats', 'Files', 'Configure Team', and 'Configure Board'. The main area is divided into two columns. The left column, titled 'Backlog' with a '13i 0p' indicator, lists 23 items (IBW1-1 to IBW1-23) with descriptions of user stories related to a trip planner. The right column, titled 'Sprint 1 - Projekt Setup' with a '6i 0p' indicator and dates 'Dec 20 - Jan 8', lists 6 items (IBW1-24 to IBW1-9) including technical tasks and a data model. Each item in the backlogs includes a priority icon (a person in a circle) and a status icon (a green checkmark).

## 07.01.2017 Stand Sprint 1

Product Backlog priorisiert und mit Storypoints versehen. Sprint 1 fast fertig.

Burndown Chart: Sprint 1 müsste sich vom Aufwand her gerade noch ausgehen



## 18.1.2017 Stand Sprint 3

3i 10p	5i 13p	9i 35p	11i 24p
<b>Backlog</b>	<b>Sprint 3 - Aktivitäten und Test</b> Jan 16 - Jan 22	<b>Sprint 2 - Admin und Reisen</b> (finished at 2017-01-16 10:12:44)	<b>Sprint 1 - Projekt Setup</b> (finished at 2017-01-13 12:18:28)
IBW1-32 Org-Task: Projektdoku, Präsentation, Ins...	IBW1-20 Als Benutzer möchte ich eine Aktivitätsliste zur Reise sehen, damit ich diese neu...	IBW1-13 Als Administrator möchte ich ein PointOfInterest CSV-File mittels GUI...	IBW1-9 Ich als Benutzer benötige eine Grundmaske damit ich zwischen den...
IBW1-23 Ich als Benutzer möchte meine aktuelle Reiseplanung als PDF exportieren...	IBW1-21 Als Benutzer möchte ich einzelne Aktivitäten bearbeiten können, damit ich...	IBW1-11 Als Benutzer möchte ich mich einloggen können, damit ich meine personalisierte...	IBW1-28 Org-Task: Userstories erstellen
IBW1-10 Als Benutzer möchte ich mich registrieren können, damit meine Reisen personali...	IBW1-22 Als Benutzer möchte ich zur Reise die gesamte Reiseplanung sehen können,....	IBW1-30 Als Administrator möchte ich den Fortschritt beim Import sehen können...	IBW1-12 Als Administrator möchte ich ein Kategorien CSV-File mittels GUI...
	IBW1-33 Tech-Task: GUI Framework finalisieren	IBW1-18 Als Benutzer möchte ich Aktivitäten mit Ort und POI zur Reise hinzufügen können,....	IBW1-7 Tech-Task: Es muss ein PointOfInterest...
	IBW1-34 Tech-Task: Aufsplittung in Unit- und Inte...	IBW1-19 Als Benutzer möchte ich nach Ort für die Reiseplanung suchen können, damit ich...	IBW1-6 Technical Task: Es muss ein Ort-CSV Fil...
		IBW1-15 Als Benutzer möchte ich alle meine Reisen in einer Liste sehen, damit ich eine...	IBW1-8 Tech-Task / PoC: Google Maps Library o...
		IBW1-17 Als Benutzer möchte ich Reisen löschen können, damit diese nicht mehr in der...	IBW1-25 Tech-Task: Datenmodell
		IBW1-16 Als Benutzer möchte ich eine neue Reise erfassen können, damit diese für die...	IBW1-27 Tech-Task Postgresql DB anlegen
		IBW1-31 Als Benutzer möchte ich nach POI Name und Kategorie im Umkreis des...	IBW1-24 Technical Task / PoC: Import von einer S...
			IBW1-26 Tech-Task: GitHub Repository mit MD R...
			IBW1-29 Org-Task: Projekt Dokumentation vervoll...

# Datasets

Städte und Point of Interests:

<http://planet.osm.org/>

Städte:

<https://raw.githubusercontent.com/datasets/un-locode/master/data/code-list.csv>

<https://www.maxmind.com/de/free-world-cities-database>

<http://download.geonames.org/export/dump/>

Flughäfen:

<http://openflights.org/data.html>

POI/Tour:

<http://tour-pedia.org/about/datasets.html>

Hotels:

<https://datahub.io/dataset/wikivoyage-listings-as-csv>

Mapping von OSM Data nach CSV:

<https://github.com/MorbZ/OsmPoisPbf>

# Libraries

JxMaps als Open-Source Lizenz.

Beispielcode:

<https://github.com/TeamDev-Ltd/JxMaps-Examples/tree/master/src/com/teamdev/jxmaps/examples>

# PROJEKTPRÄSENTATION, NEU 14.01.2017

## Präsentation/Diskussion der Semesterarbeit

Was erwartet Corsin (habe ich sortiert, also Vorschlag für die Präsentation):

- 1) Vorstellung der Projektidee
- 2) Reflexion der Arbeit → was ist hier gemeint, WIE wir es angegangen sind?
- 3) Ausgewählte Technische Aspekte → demnach WAS wir implementiert haben
- 4) Demonstration der lauffähigen Applikation

Slides:

- Inhalt
  - Projektidee
  - Projektvorgehen
  - Demo
  - Technische Aspekte
- Projektidee
- 

## Handouts

- Projektidee
- Projektorganisation (Scrum, ...)
- Klassendiagramm
- Statistik
- MVC, Testing, Maven
- Technische Aspekte (Producer/Consumer, Swing Framework)
- Beschreibung und 1-2 kleine Screenshots aus Userstory
- Vivify Ansicht Storyboard (full)
- Burndown Chart
- Featurebranche je UserStory
- Code Review

Doku, was ist erwähnenswert (technische Aspekte)

- Producer/Consumer Pattern, Threads beim Importer (Dieter)
  - Producer: FileReader liest File und füllt Queue
  - Consumer: CategoryConsumer / PoiConsumer lesen die Queue und schreiben die Daten mit Hilfe der DatabaselImport Klasse in die Datenbank
  - DatabaselImport verwendet Multi Value Insert Statements, welche dynamisch erstellt werden.
  - Für das Update wird ein Trigger verwendet, damit im Programm keine zusätzliche Abfrage auf die Datenbank notwendig ist und beim

Schreiben eine Unterscheidung zwischen Update/Insert gemacht werden muss. (Geschwindigkeit Optimierung)

- Swing Framework (Dieter) mit MainController , Vor/Zurück, Fehleranzeige, ...
  - MainTripPlanner Klasse: Ein kleines Swing Framework, damit GUIs schneller und einfacher entwickelt werden können.
  - ViewInfo Klasse speichert die Informationen/Optionen zu einer View (Content) ab.
  - Mit der FormPanel Klasse haben wir ein einheitliches Design für Formulare bereitgestellt. Diese Klasse enthält auch Hilfsmethoden zum schnellen erstellen von Buttons, RadioButtons, TextFields, etc.
  - Beispiel Button:  
`addComponentToPanel(createButton("Open file", "open_file", adminController));  
addPanelWithLabel("File:", true);`
  - Beispiel RadioButton:  
`addComponentToPanel(fileTypeCategory = createRadioButton("Category",  
"category", true, fileTypeGroup));  
addPanelWithLabel("Type:", true);`
- Interface für die SearchView (Dieter)
  - Da wir zwei Suchen (City und Poi) haben und diese Teils die gleichen Funktionen bereitstellen, haben wir für die Suche ein Interface erstellt. Durch dieses Interface ist es möglich den gleichen Controller für die beiden Suchen zu verwenden.
- Pair Klasse mit den generischen Datentypen (Dieter)
  - Als Hilfsklasse für die Map. Dort müssen wir die Marker mit den Pois verknüpfen.
- MVC konsequent
- Testing unterscheidet UnitTest (@Category({ UnitTest.class }) ) und Integrationstest
- Mockito Controller- und View-Test  
<https://github.com/ibwgr/TripPlanner/blob/master/src/test/java/controller/comm on/LoginControllerTest.java#L65>  
<https://github.com/ibwgr/TripPlanner/blob/master/src/test/java/model/admin/FileReaderTest.java#L27>

## Doku Vorgehen

- Scrum mittels Tool, Aufzeigen Sprintplanung, Backlogs
- Branching mit Pull-Request und somit Code Review (Travis Pre-Check)

## Doku, was haben wir gelernt

- Immer direkt Maven Projekt machen, dann stimmt Struktur. Travis kann dann problemlos die Tests via GitHub durchführen

- Vielen freie Werkzeuge (Google Docs, Vivify Scrum, Lucidcharts, GitHub, Travis) ideal für Collaboration
- Etwas über Lizenzen
- 
- 

Was könnten Fragen von Corsin sein

- DB Trigger?
- Views weshalb? URKA: Layer, keine redundante Queries
- 

Handout:

- Datenmodell

Git "Code-Zeilen" Statistik per 27.01.2017:

Language	files	blank	comment	code
Java	68	1189	941	5035
SQL	1	53	102	173
Maven	1	5	12	111
Markdown	1	13	0	60
XML	1	0	0	6
YAML	1	1	4	6
SUM:	73	1261	1059	5391

Test Coverage per 27.01.2017:

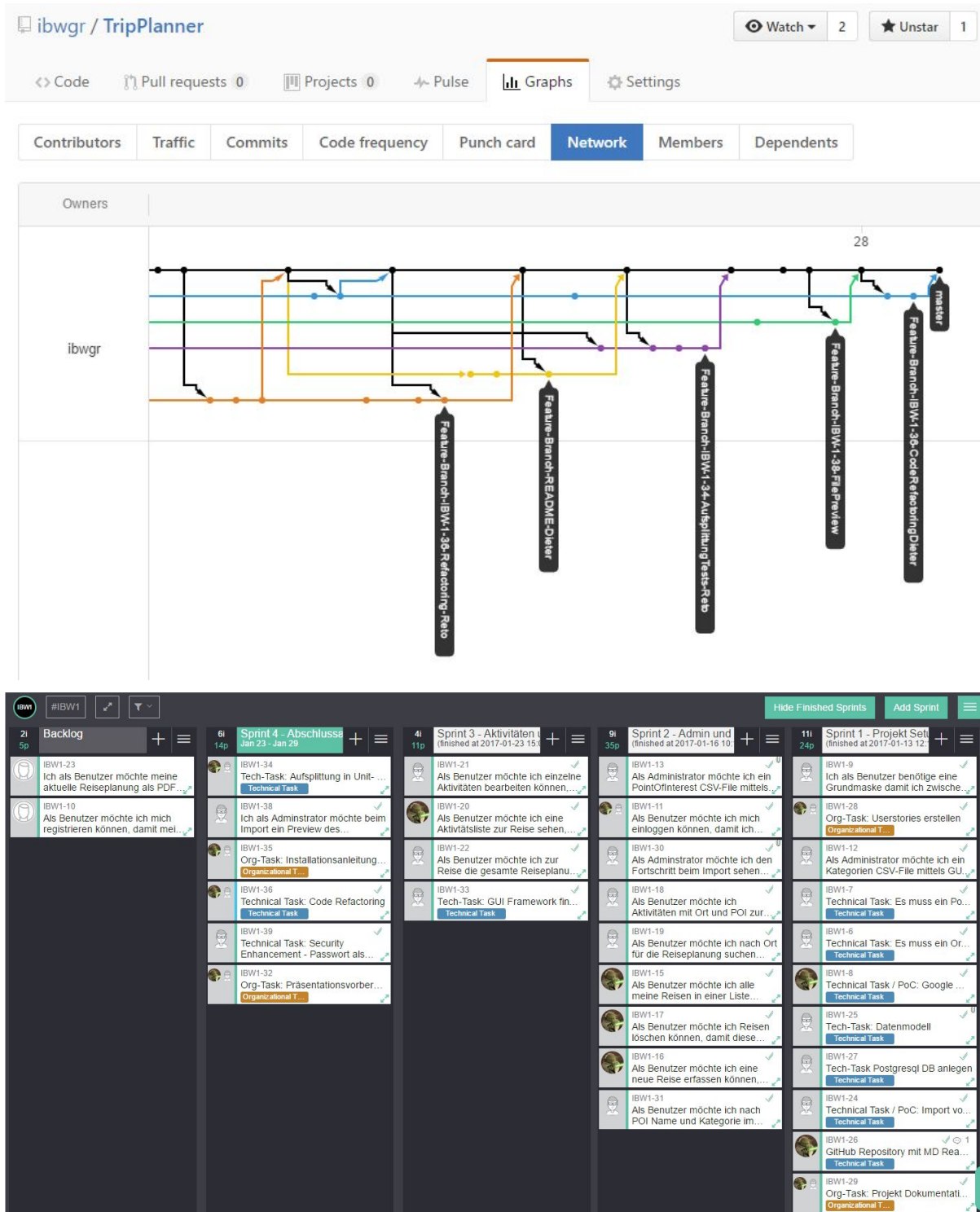
Package	Class, %	Method, %	Line, %
all classes	71.7% (38/ 53)	49.2% (174/ 354)	48.6% (1024/ 2107)

#### Coverage Breakdown

Package ▲	Class, %	Method, %	Line, %
<empty package name>	0% (0/ 1)	0% (0/ 2)	0% (0/ 3)
controller.admin	100% (3/ 3)	63.6% (14/ 22)	44.4% (56/ 126)
controller.common	100% (2/ 2)	67.9% (19/ 28)	62.2% (97/ 156)
controller.travel	66.7% (2/ 3)	22.6% (7/ 31)	17.3% (33/ 191)
model.admin	80% (4/ 5)	72.7% (8/ 11)	44.8% (69/ 154)
model.common	100% (10/ 10)	64.7% (55/ 85)	54.8% (218/ 398)
model.travel	100% (2/ 2)	48.8% (20/ 41)	50.8% (134/ 264)
view.admin	66.7% (2/ 3)	40% (6/ 15)	57.9% (44/ 76)
view.common	100% (6/ 6)	69.6% (32/ 46)	80.9% (225/ 278)
view.travel	38.9% (7/ 18)	17.8% (13/ 73)	32.1% (148/ 461)



## 19 Branches, 84 Pull Requests in den Master Ein Featurebranch jeweils pro Scrum Userstory









---

ALT:

### Funktionsliste und GUI Skizzen zum Reiseplaner:

- Registrierung
  - Eingabe von User Daten
  - Speichern der User Daten (insert)
- Login
  - Eingabe User/Passwort
  - Eingaben prüfen (select)
  - wenn OK:
    - bei Admin: Admin View anzeigen
    - bei User: Reiseplaner View anzeigen
- Admin (upload und Fortschrittsanzeige, Fehler in Log)
  - File öffnen, parsen und insert/update
    - alle Tabellen mit CSV Datei abfüllen (Category, Location Data)
    - delimiter auswählen
    - auswahl des daten typs: location oder category
    - Location data nach Kategorie (city, poi) aufteilen und in eigenen Threads verarbeiten (bulk insert in postgres ?) -> COPY, INSERT VALUES()
      - best possible way of inserting with multiple rows was using 100 rows per statement, 100 statements per transaction, using prepared statements – it did it's work in 37.73 seconds.
      - `INSERT INTO "table" ( col1, col2, col3) VALUES ( 1, 2, 3 ) , ( 3, 4, 5 ) , ( 6, 7, 8 );`
      - `ON CONFLICT DO UPDATE`
      - oder mit check auf table und update queue erstellen?
      - mit ON CONFLICT DO UPDATE und POI/CITY in einer Tabelle braucht es nur eine queue!
    - Fast File Reading: MappedByteBuffer oder Stream (am Schnellsten)
    - das CSV in ZIP einpacken, wird dadurch Platz gespart?
    - Alle Rows (Stream.lines) werden je nach category in die city queue oder poi queue geschrieben
    - Threads verarbeiten die queues und erstellen pro 100 rows ein SQL Statement (multi values insert / prepared statement)
    - falls id bereits in tabelle vorhanden ist, wird die row in die update queues gestellt
    - eigene Threads verarbeiten die update queue und erstellen ein SQL Statement mit prepared statements. Mit batch:
      - `ps.addBatch();`
      - `ps.executeBatch();`
    - alle Indizes/Triggers vor der Verarbeitung deaktivieren und danach wieder aktivieren

- transaction handling:  
<http://docs.oracle.com/javase/tutorial/jdbc/basics/transactions.html>
- synchronization der Queue Threads muss noch angeschaut werden
- total anzahl queues (LinkedList): insertPoiQueue, insertCityQueue, updatePoiQueue, updateCityQueue
- mit einem Thread wird der aktuelle Status ermittelt und im GUI angezeigt (Total Rows, Rows für City/Poi, inserted city/poi, updated city/poi)
- falls mit inserts zu langsam, kann mit copy dies auch gelöst werden (fortschritt kann angezeigt werden, bis file eingelesen ist, danach kommt postgres (hier kann nur eine sanduhr angezeigt werden))
  - <https://www.postgresql.org/message-id/3a0028490809301807j59498370m1442d8f5867e9668@mail.gmail.com>
- Log Ausgabe mit anzahl inserts/updates/etc.
- anzeigen aller rows für poi, city und category in jtable, mit einfachem Filter?
- (alle POIs anzeigen/suchen)
- (POIs in Tabelle speichern (insert / update)) -> insert/update nur über CSV

### Trip planner administration


Upload a new CSV OSM File to update the category, city or Point of interest

File content:    ☒ Category    ☐ Location Data

Delimiter:    ☐ ;    ☒ |    ☐ ,

Progress

Total Rows: 1234556  
 City Rows: 1234    (Inserts: 123 / Update 555)  
 POI Rows: 455    (Inserts: 123 / Update 555)  
 Rows finished: 1231



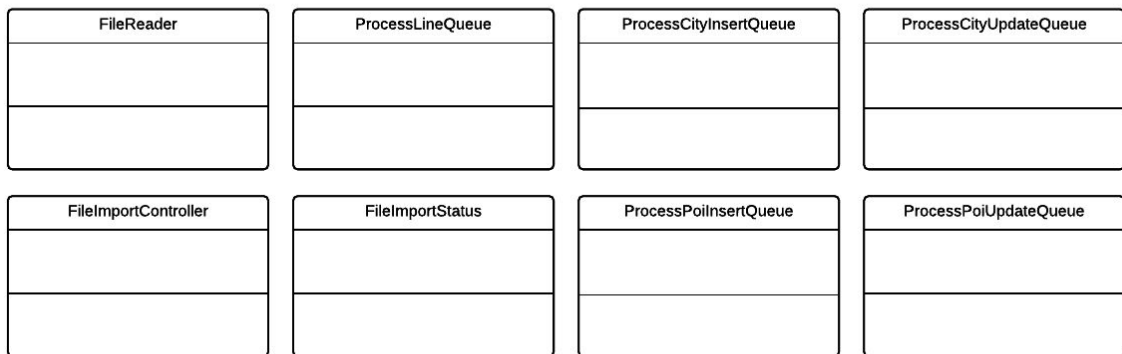
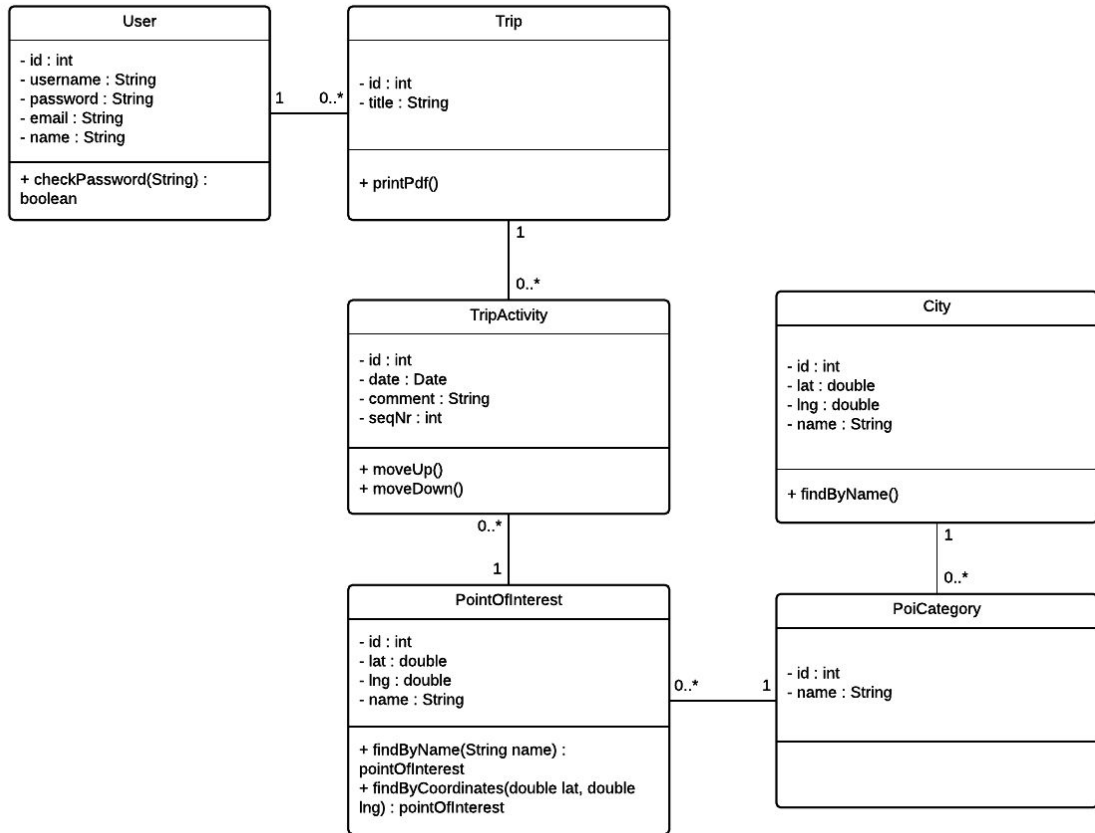
- Liste aller meiner Reisen
- Neue Reise
- Reiseaktivitäten
  - Suche nach Stadt
  - Suche von POIs
  - anzeige in Google Maps und in Ergebnisliste
    - Klick auf Eintrag in Liste markiert in Maps und umgekehrt
  - Neue Reiseaktivität (Speichern von POIs) mit Datum- und Kommentareingabe (in Liste)
  - Reiseaktivitätsliste (Gespeicherte POIs als Liste und auf Map anzeigen)

- Reiseübersicht (gesamte Reise) mit grosser Map und PDF Export-Button
- Allenfalls Export Reise als PDF

Falls zu wenig Zeit, könnte man Panel 4 auch weglassen oder versuchen es in Panel 3 zu integrieren (da eigentlich nur der Reiseplan mittels Verbindungen dargestellt wird. Das könnte man auch auf der Map in Panel 3 machen)

- Weitere generelle/technische ToDo-Tasks:
  - CSV Daten besorgen (allenfalls aus Openstreetmap)
  - Google Maps Library besorgen
    - oder Openstreetmaps?
  - Scrum Tool auswählen/aufsetzen
  - UML Modellierungswerkzeug auswählen/aufsetzen
  - Swing Grundgerüst (Haupt-Window) mit Tabs, für gemeinsame Nutzung
  - Swing Superklasse für Panel, für gemeinsame Nutzung
- Allgemeines zum Projekt, Planung, Doku
  - Definition of Done definieren (z.B. Testcoverage)
  - Burndown Charts während den Sprints kopieren (Screenshot für Doku)

Alt:



Das Projekt enthält folgende Methoden und Technologien:

- Objektorientierte Programmierung
- JUnit / Mockito
- Exception Handling

- Parallele Programmierung (Threads)
- Datenbank
- Swing
- File I/O
- Scrum mit online Board (siehe weiter unten)

Reiseliste

Alle Reisen


Detail

Neue Reise

NEW

Suche / Erfassung  
Aktivitäten

Sommerferien Schweden 2017

Suche Ort  Suche POI

Click Suche

Datum  Kommentar  Sortierung

Aktivitätsliste  
Datenmutation

Sommerferien Schweden 2017

Click Suche

Datum  Kommentar

Ok Cancel

Übersicht/  
Reiseroute

Sommerferien Schweden 2017

PRINT

PDF

Registrierung

Name

Vorname

E-Mail

User

PW

Login

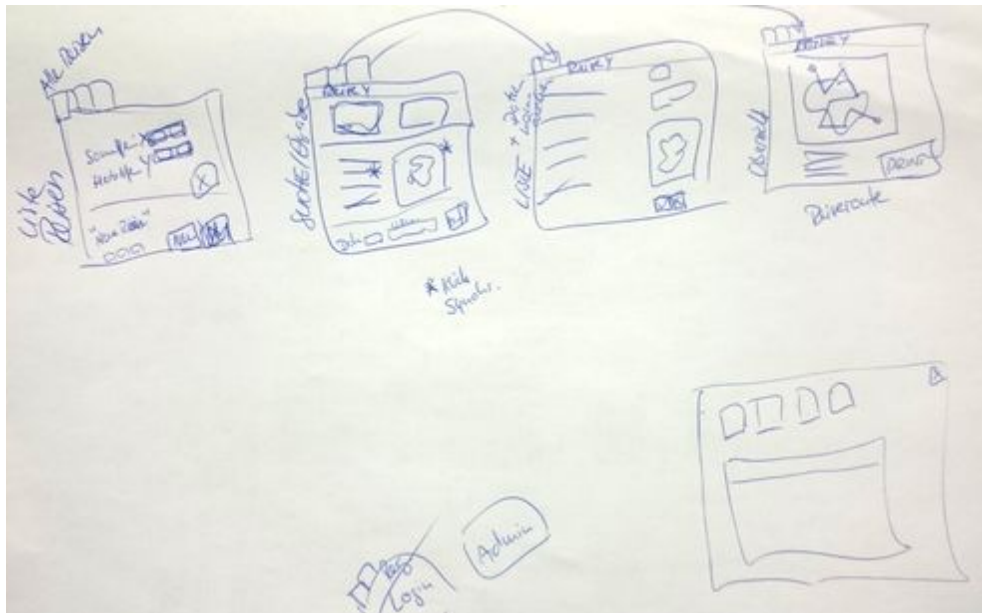
User

PW

OK

Admin/Upload

CSV File



Weitere betrachtete Scrum Werkzeuge:

<https://www.vivifyscrum.com> (Github /BitBucket Integration) → **recht gut!**

<http://www.scrumdesk.com>

<https://www.flying-donut.com> (Github /BitBucket Integration) → **unübersichtlich, unhandlich!**

<http://www.scrumdo.com>

<http://scrumfortrello.com> → **nicht so gut, enthält nur Storypoints, sonst Basis-Trello**

[http://scrum-project-management-tool.targetprocess.com/try-out-now/?utm\\_source=Agilescout&utm\\_medium=Listing&utm\\_campaign=Agilescout-Scrum-Listing](http://scrum-project-management-tool.targetprocess.com/try-out-now/?utm_source=Agilescout&utm_medium=Listing&utm_campaign=Agilescout-Scrum-Listing)

hier gibt es noch tausend andere: <http://agilescout.com/best-agile-scrum-tools>

Weitere Projektideen:

- CSV Import/Export
- Restaurant Programm, man gibt die Bestellungen des Kunden ein und am Schluss kann man eine Rechnung drucken.



- Nur Daten aus deinen CSV Files einlesen und mit Threads den Fortschritt darstellen, danach können Daten durchsucht werden
- Irgendwas mit Buchhaltung
- Mathe Quiz Applikation
- Shop:
  - User Verwaltung
  - Admin (Produkte Upload/Bearbeitung)
  - Login
  - Produkte suchen
  - Warenkorb