Vorprojekt-Entwurfsbeschreibung

Yannik Voelker

8. April 2015

Inhaltsverzeichnis

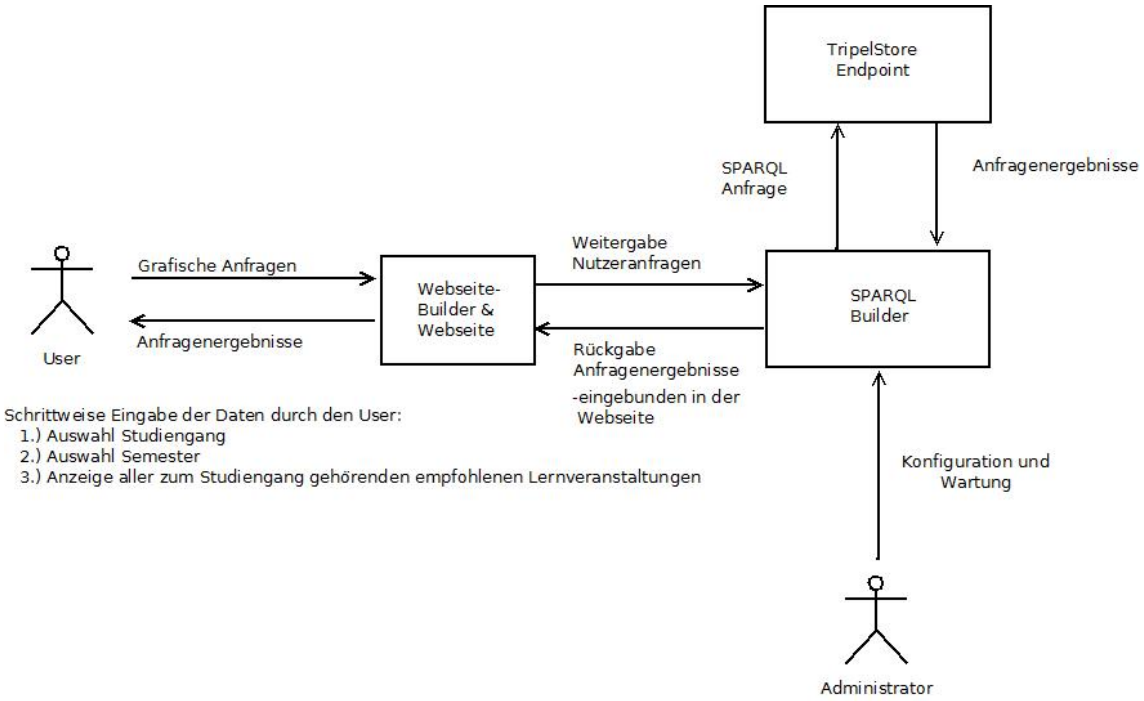
1 Allgemeines

Die jetzige Situation an der Universität Leipzig ist, dass man mit der Belegung der Module die Termine der einzelnen Veranstaltung teilweise von unterschiedlichen Quellen erhält. Zudem sind die existierenden Veranstaltungsübersichten (bspw. die der Fakultät für Informatik[1]) häufig zu unübersichtlich, sodass auch hier eine Stundenplan-Zusammenstellung aufwendig ist.

Das Ziel unseres Projektes ist deswegen die Erstellung personalisierter Stundenpläne auf Basis der Open-Data-Schnittstelle od.fmi.uni-leipzig.de. Diese Schnittstelle hält die Informationen zu den Veranstaltungen der Fakultät für Mathematik und Informatik als RDF-Datenbank. Diese wollen wir nutzen, um eine einfachere Möglichkeit der Stundenplanerstellung zu bieten.
Für das Vorprojekt werden die Informationen der Kurse und Seminare für ein aktuelles Semester aus der od-Schnittstelle ausgelesen und visuell dargestellt. Realisiert wird dies als Webanwendung, welche gleichzeitig Grundgerüst für das gesamte Projekt ist.

[1] `https://www.informatik.uni-leipzig.de/ifi/studium.html` (01.04.2015)

2 Produktübersicht



Der Nutzer wird zuerst auf die Startseite gelangen, auf der im späteren Hauptprojekt eine kurze Erklärungen zur Bedienung des Stundenplans steht. Als erstes wird nach dem Studiengang gefragt, danach nach dem Semester, welches man gerade belegt. Da für das Vorprojekt das grobe Design-Layout der Webseite feststehen sollte, sind die nächsten Seiten der Webseite zunächst nur zur Veranschaulichung mit Dummy-Elementen gefüllt.

Man sieht zunächst die Seiten zur Auswahl der Module für ein Semester, die später ausgewählt werden können. Auf der nächsten Webseite befinden sich (ebenfalls zunächst als Dummy-Elemente) die einzelnen Veranstaltungen der gewählten Module.

Schließlich wird, anstatt den Stundenplan und die Downloadfunktion zu bieten, die vorherige Auswahl von Studiengang und Semester ausgewertetet und die dazugehörigen empfohlenen Veranstaltungen in Listenform angezeigt.

Ebenfalls noch nicht implementiert ist die Nachkonfiguration des Stundenplans, dies wird wie der eigentliche Erstellungsvorgang realisiert, ohne dass der Nutzer ein Konto anlegen oder sich einloggen muss.

3 Grundsätzliche Struktur- und Entwurfsprinzipien

In diesem Projekt wird als Programmiersprache Python benutzt. Weiterhin verwenden wir Flask als minimalistisches Webframework. Bei Auswahl von Modulen und Veranstaltungen werden stets SPARQL-Anfragen an den Triple-Store `od.fmi.uni-leipzig.de` gesendet (vgl. Datenmodell).

Für das Vorprojekt sind die einzelnen Module des Programmes in vier separaten Dateien ausgelagert. Sie ist gegliedert in Hauptmodul, welche die Webseite aufbaut, ein SPARQL-Modul, welches die Datenbank abfragt, ein Modul zur Bestimmung von relevanten Zeiten, sowie ein Modul zur Bestimmung der Forms. Zusätzlich sind für das Flask-Framework HTML-Templates erstellt worden, welche die verarbeiteten Daten darstellen.

4 Struktur- und Entwurfsprinzipien einzelner Pakete

Im folgenden wird der genauere Aufbau der einzelnen Module erläutert. Dabei wird in Python-Module und Module für die HTML-Templates unterschieden.

4.1 Python-Module

Das Modul `app.py` bildet den Kern des Programms. Die Inhalte der einzelnen Webseite werden hier durch die Abfrage anderer Module (`SPARQL-Modul` und `Zeit-Modul`) generiert und anschließend in die Forms, sowie in die Webseite angebunden. Jede Funktion bildet dabei jeweils eine HTML-Seite. Zusätzlich werden hier später, benötigte Informationen in die Browser-Session eingebunden.

Das Erscheinungsbild der Formulare/Forms wird in dem Modul `forms.py` festgelegt. Da für die Webseite verschiedene Varianten für die Forms sinnvoll sind, bspw. eine einfache Liste für die Auswahl des Studiengangs, sind hier verschiedene Forms vorhanden, die vom Hauptmodul aufgerufen werden. Dies ermöglicht die Darstellung von Checkfeldern und DropDown-Listen für die verschiedenen HTML-Templates. Jede der verschiedenen Form-Varianten bildet dabei eine eigene Klasse.

Die Abfragen an die Datenbank `od.fmi.uni-leipzig.de` finden im SPARQL-Modul

`sparql_requests.py` statt. Hier befinden sich für die verschiedenen benötigten Abfragen jeweils eine Funktion, die die Datenbank abfragt und die resultierenden Daten im JSON-Format ausgibt. Dies ermöglicht es, die Daten effizient und leicht lesbar an das Hauptmodul weiterzugeben. Weiterhin ist hier das Zeit-Modul eingebunden, da dies für die korrekte Zuordnung des aktuellen Semesters notwendig ist.

Außerdem wurde das Zeit-Modul `time-conversion.py` eingerichtet, welches die Zeiten für den Beginn des aktuellen Semesters, Sommer/Wintersemester und die vorlesungsfreie Zeit anhand des derzeitigen Datums berechnet. Einige der Funktionen werden jedoch erst im weiteren Verlauf des Projektes benötigt.

4.2 HTML-Templates

Die verschiedenen HTML-Templates wurden erstellt, um den Aufbau der Webseite zu beschreiben. Die Basis fast aller weiteren HTML-Templates bildet dabei die `base.html`. Hier werden die Kernelemente des HTML, wie z.B. die `<html>`- und `<head>`-Tags, sowie das Stylesheet geladen. Dies ermöglicht, dass die weiteren Templates übersichtlicher erscheinen und sich Wiederholung von Code minimiert.

Ein weiteres Hilfstemplate ist die `_formhelpers.html`. Sie erzeugt die HTML-Formulare mit den Elementen in Listenform. Auch hier ist die Hauptintension das Vermeiden von Wiederholung.

Schließlich bilden die weiteren Templates jeweils eine Webseite ab. Wie bereits erwähnt werden die vorher erwähnten Module importiert, sofern benötigt. Es ist zu erwähnen, dass im Vorprojekt, wie besprochen, nur die Studiengang- und Semesterwahl, sowie die die Anzeige der Lernveranstaltungen (`lvshow.html`) dynamisch aufgebaut sind, während die weiteren Templates statisch zur Veranschaulichung eingebaut wurden.

5 Datenmodell

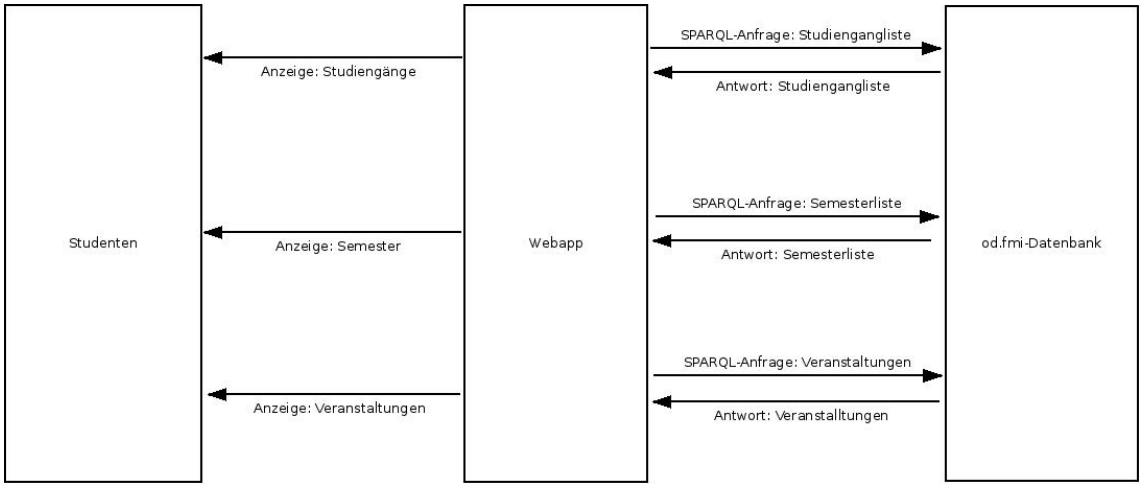


Abbildung: Datenmodell

Nach dem ersten Besuch der Seite stellt die Webanwendung eine SPARQL-Anfrage an die od.fmi-Datenbank und erhält eine Liste der Studiengänge. Der Besucher der Seite wählt seinen Studiengang, woraufhin erneut eine SPARQL-Anfrage zur Ermittlung der möglichen Semester gestellt wird. Die Webanwendung verarbeitet die Auswahl des Semesters und Studiengangs und schickt eine letzte SPARQL-Anfrage. Die Webanwendung breitet die Informationen auf und präsentiert sie schließlich dem Nutzer.

Der Nutzer überspringt die im Vorprojekt ausschließlich als Dummieelement vorhandenen Funktionen der Modulauswahl und Ergänzungen sowie der Generierung des nutzerspezifischen Hashes. Im fertigen Programm wird außerdem optional durch klicken des Herunterladen Buttons ein ical-Format generiert und heruntergeladen.

6 Testkonzept

Da in unserem Vorprojekt das Vertrautmachen mit der Triplestore-Schnittstelle, von Flask, sowie von Python im Allgemeinem im Vordergrund stand, ist das Testen der Anwendung zweitrangig. Hinzu kommt, dass einige Funktionen innerhalb des Projektes sich auf Daten des Triplestores beziehen und sich die Test-Automatisierung dieser Funktionen außerordentlich schwierig gestaltet. Daher wird größtenteils auf Tests verzichtet.

Lediglich die Tests für die Zeit-Konversionen, die später für das Hauptprojekt eine wichtigere Rollen spielen, wird aufgrund der Einfachheit von doctest mit diesem Tool geschrieben. Dabei wird darauf geachtet, dass möglichst 100% der einzelnen Funktionen abgedeckt wird.

7 Glossar

***.py** Dateiendung für Programme in der Programmiersprache Python

doctest Methode in Python, zur einfachen Erstellung von Modultests, der eigentliche Test befindet sich dabei in einem Docstring.

Dropdown-Liste Eine Dropdown-Liste ist ein grafisches Bedienelement, welches beim Aktivieren dem Nutzer mehrere Auswahlmöglichkeiten in einer Liste bietet

Dummy-Element Ein Element, welche lediglich als Platzhalter dienen und ansonsten keine Funktion haben.

Flask ein auf Python basierendes Web-Framework

Forms ist ein Element von HTML, welches Formulare darstellt. Formulare können verschiedene Formate haben (bspw. mit Radiobuttons, Checkboxen, etc.)

Hash eine Prüfsumme, die dazu dient, Datensätze auf Gleichheit zu prüfen

iCalendar ein Datenformat zum Austausch von Kalenderinhalten

JSON-Format kompaktes Datenformat in einer einfachen lesbaren Textform zum Zweck des Datenaustauschs zwischen Datenanwendungen.

open data bedeutet die freie Verfügbar- und Nutzbarkeit meist öffentlicher Daten

Python Programmiersprache

Radiobutton Ein Radiobutton ist ein Bedienfeld einer grafischen Oberfläche, welche von einem Nutzer eine Auswahl verlangt

RDF Tripel von Daten, welche Beziehungen zwischen repräsentieren

SPARQL ist eine graph-basierte Abfragesprache für RDF

Stylesheet ist ein Element von Webseiten, welches die Gestalt der Webseite bestimmt

Tag bezeichnet zumeist ein in Klammern eingeschlossenes Element, welches dazu dient, dazwischenliegende Textelemente zu kennzeichnen, bspw. den Titel einer HTML-Seite

Template ist eine Vorlage für die HTML-Seiten, die mit Hilfe von Flask zu vollständigen Webseiten zusammengesetzt werden

Triplestore (Virtuoso) ist eine RDF-Datenbank, die eine SPARQL-Schnittstelle zur Verfügung stellt. Hier werden die Daten, die für die Stundenplan-Generierung nötig sind, abgefragt

URL (auch Uniform Resource Locator) identifiziert und lokalisiert eine Website oder auch andere Rescourcen in einem Computernetzwerk