Inhaltsverzeichnis

Einleitung

Das Modul Webbasierte Anwendungen 2 handelt von der Erstellung einer REST-Architektur zwischen einem Dienstnutzer und Dienstgeber mit der Angabe von REST-Ressourcen sowie einer Browseransicht.

In meinem Projekt wird dieses am Beispiel eines Praktikumsanmeldungstools dargestellt. Dieses soll mehrere Probleme von Studenten und Dozenten lösen. Es soll auch die Anmeldung für alle Praktika vereinfachen.

Bitte beachten Sie, dass dieses Projekt eine 1-Woman-Arbeit ist und somit ein Workload von 75 Stunden nur notwendig sein wird.

Szenario

Fall 1

Der Student Oliver studiert allgemeine Informatik und hat im dritten Semester gemerkt, dass er sich für fünf verschiedene Praktika (Datenbanken Systeme 1, Kommunikationstechnik und Netze, Softwaretechnik 1, Paradigmen der Programmierung und Algorithmik) auf fünf verschiedene Seiten anmelden muss

Fall 2

Die viert Semsterin Jana hat sich das Wahlpflichtfach Softwaretechnik und agile Methoden am Dienstag Nachmitttag belegt. Auf diesem Termin fallen auch drei Termine ihres Softwaretechnikpraktikums und dazu kommt noch die Tatsache das zwei Praktika erst um 18 Uhr beginnen, sie muss also 6 Stunden auf den Beginn des Praktikums warten. Sie muss also mindestens dreimal den Termin tauschen und einmal 6 Stunden warten bis das Praktikum anfängt.

Fall 3

Der sechst Semester Tim nicht genervt er hat fast alles schon abgeharkt nur ein fast bestanden Betriebssystem Praktikum ist noch offen und sein Praxisprojekt ist auch schon so gut wie fertig. Jetzt muss er nur nach die Betriebssystem Klausur schreiben und bestehen. Dann kann er endlich mit seiner Bachelorarbeit anfangen. Jedoch stellt sich heraus, dass sich Tim ohne das bestandene Praktikum, nicht mal unter Vorbehalt seine Betriebssystemklausur schreiben, darf obwohl er keine Probleme bei Praktikum hat und den Ilias Test mit 20 von 20 Punkten bestanden hat.

Tim würde die Klausur sehr gerne sofort schreiben. Er muss als mit dem Professor reden um eine Sondergenehmigung zu bekommen und nicht mit den Angestellten aus dem ADV-Labor anzulegen. [1]

Fall 4

Die Studentin Julia ist hat keine Ahnung in welchen dem Raum Sie muss, nach dem Sie 10 Dokumente geöffnet und im Stundenplan geguckt weiß sie endlich wohin sie muss und kommt ein paar Minuten zu spät.

[1] **Anmerkung**: Die Szenarien 1, 2 und 4 sind während meiner Studienzeit passiert die Namen der Studenten wurden jedoch geändert. Das Szenario 3 ist in einer Abgewandelten Form passiert. Zwar ließen sich alle Probleme Lösung jedoch war es ein kleiner Aufwand verbunden um diese Termine zu verschieben, dadurch das neue Staffelpläne oder E-Mails wurde geschrieben.

Use Cases

Anwendungsfall1

Der Student Steffen möchte sich für alle seine Praktika einfach schnell und problemlos anmelden, dieses soll nach Möglichkeit über eine Webseite geschehen. Steffen will im Idealfall alle seine Fächer, Übungen, Wahlpflichtfächer und Praktika unter einem Hut bekommen und keine Doppelbelegungen von Praktika mit anderen Praktika, Übungen, Vorlesung oder Wahlpflichtfächer haben.

Preconditions	Der Student muss eine Matrikelnummer, eine CampusID und ein Passwort besitzen.
Success End Condition	Der Student hat sich für seine Praktika angemeldet.
Failed End Condition	Der Student konnte sich nicht anmelden.
Primary Actors	Student
Trigger	Der Student möchte sich für eine offenen Praktika anmelden.
Beschreibung	1. Der Student geht auf http://localhost:3000/ 2. Der Student trägt seine CampusID und Passwort in die dazugehörigen Felder. 3. Er trägt seine Daten Testfall CampusID: dgaeb Passwort: wba2 4. Er klickt mit der Maus auf dem Login Button loggt sich ein. 5. Der Student sucht seine offenen Praktika und meldet sich an. 6. Der Student schickt seine Anmeldung ab. 7. Der Student logt sich aus.
Extensions	Serververbindung schlägt fehl und Anmeldung wird angebrochen.

Anwendungsfall 2

Die Studentin Julia kann durch das Nutzen App auch im Offline Modus schnell nachgucken, in welchem Raum Julia um welche Zeit zu erscheinen hat. Sie hat es jetzt auf einem Blick alle Praktika nach Fach und Datum geordnet.

Preconditions	Der Stundet muss eine Matrikelnummer haben und eine CampusID besitzen und hat sich für Praktika angemeldet.
Success End Condition	Der Student weiß in welchem Raum er zu welcher Zeit mit sein muss.
Failed End Condition	Verbindung konnte nicht aufgebaut werden. Daten wurden nicht geladen.
Primary Actors	Student
Trigger	Der Student möchte wissen in welchem Raum und um welche Uhrzeit er welches Praktikum hat.
Beschreibung	1. Der Student geht auf http://localhost:3000/ 2. Der Student trägt seine CampusID und Passwort in die dazugehörigen Felder. 3. Er trägt seine Daten Testfall CampusID: dgaeb Passwort: wba2 4. Er klickt mit der Maus auf dem Login Button loggt sich ein. 5. Der Student sucht **mit seinen Augen** das betreffende Praktikum und sieht sich Ort, Zeit und Datum nach. 6. Der Student schließt das Fenster und geht in seinem Raum oder trägt sich es manuell in seinen Kalender ein.[2]
Extensions	Serververbindung schlägt fehl und Anmeldung wird angebrochen.

[2] Die Kalendereintragung kann keine Serverspezifikation und ist in deiesm Fall eher als Interfache zu verstehen, da der Stunden sich die Termine auf einem Endgerät (Handy, Tablet oder PC) oder altmodisch in einem Taschenkalender eintragen kann.

Anwendungsfall 3

Professoren können das Bestehen oder nicht Bestehen von Praktika direkt in eine Datenbank schreiben. Das schlussendliche Bestehen des Praktikums kann der Professor am Ende des Tages oder einer Woche direkt an das Prüfungsamt weiterleiten. Das bedeutet der Student muss nicht bis zur letzten Sekunde warten bis das Praktikum eingetragen ist.

Preconditions	Der Student muss sich für das Praktikum angemeldet und der Dozent oder wissenschaftliche Mitarbeiter muss vom Professor die Lizenz (oder Erlaubnis) der Abnahme von Praktika erhalten haben.
Success End Condition	Praktikum und Anwesenheit sind als bestanden markiert.
Failed End Condition	Praktikum konnte nicht markiert werden.
Primary Actors	Dozent, Professor oder wissenschaftlicher Mitarbeiter
Trigger	Abnehmer des Praktikums möchte die Anwesenheit und das Bestehen des Studenten in die Datenbank eintragen.
Beschreibung	1. Der Abnehmer des Praktikums geht auf http://localhost:3001/ 2. Der Abnehmer trägt seine Daten Testfall CampusID: dgaeb Passwort: wba2 3. Er klickt mit der Maus auf dem Login Button loggt sich ein 4. Der Abnehmer des Praktikums sucht die Gruppe, die er Abnehmen muss. 5. Der Abnehmer des Praktikums markiert die Anwesenheit der anwesenden Studenten in der Datenbank. 6. Abnehmer des Praktikums trägt das Bestehen nach erfolgreicher Abnahme der Praktikumsaufgaben in die Datenbank ein. [3]
Extensions	Serververbindung schlägt fehl. Praktikum wird nicht bestanden.

[3] Das Abnehmen des Praktikums ist keine Serverspezifikation. Das Praktikum wird vom Abnehmer des Praktikums einem persönlichen Gespräch mit dem Studenten angenommen.

Die Lösung

Die Lösung dieser Probleme ist ein einheitliches Praktikumsanmeldungstool mit Berücksichtigung des individuellen Stundenplans des Studenten, sowie einer Funktion die bestandenen oder nicht bestandenen Praktika in eine Datenbankspeichert schreibt.

Das System

Jeder Student kann sich für alle Praktika anmelden, die es in seinem Studiengang im diesem Semester (Sommer- oder Wintersemester) gibt. Jeder Student kann sich seinen Stundenplan zusammenstellen, sodass er keine Vorlesung, Übung oder Wahlpflichtsfach verpassen muss und „unnötige“ Wartezeit vermeiden kann.

Es wird zwischen Studenten die in Gummersbach und nicht in Gummersbach wohnen unterschieden. Der Student kann mehrere Wunschpartner oder eine Gruppe eingeben und gegebenenfalls ein Wunschthema (für das Mathe 2 Praktikum notwendig).

Das Programm wird als App mit dargestellt, da es so auch einen offline Modus geben soll in dem der Student eine Termine einsehen kann. Alle Informationen werden die mit dem Praktikum in Verbindung stehen werden über diese Anwendung übertragen und als Push Nachricht gesendet.

Der Student kann seine Praktikumstermine einsehen mit Datum, Uhrzeit, Fach und Raumnummer. Der Dozent und der Professor kann die Termine ändern und einzelne Praktikumstermine oder das ganze Praktikum als Bestanden markieren. Der Dozent kann nur die Praktika seiner Fächer einsehen.

Die Anmeldung erfolgt per Campus ID, da man später das bestandene Praktika direkt an das PSSO weiterleiten könnte. Desweitern ist hat jeder Mitarbeiter und Student eine CampusID.

REST-Ressourcen

Ressource Path	Meth-ode	Semantik	Content-Type Request	Content-Type Response
/	GET	Login-Funtion für Studenten, Professoren und wissenschaftlichen Mitarbeiter.	Text/html	Text/html
/UserErstellen	POST	Anmeldung eines neuen User.	application/ json	application/ json
/UserErstellen/:id/	GET	Übersicht über die User	text/html	text/html
/auswahl	Get	Zum Suchen der passenden ansicht	Text/html	Text/html
/praktikaErtsellen	POST	Professor erstellt ein neues Praktikum. Mit Anzahl der Termine, Fach, Gruppengröße	Text/html	Text/html
/praktikaErtsellen	GET	Übersicht über alle Praktika	text/html	text/html
/anmeldenPraktika	POST	Studentmeldet sich für seine Praktika an.	text/html	text/html
/PraktikumViewProf	POST	Professor oder Wissenschaftlicher Mitarbeiter kann Student im Praktikum als anwesend oder abwesend und bestanden oder noch offen	application/ json	application/ json
/praktikumsViewStudenten	GET	Student kann seine Termine und sein Status vom Praktika einsehen.	text/html	text/html

Installation des Clients und des Servers

Laden Sie die beiden Module neuerClient und neuerServer von git herunter und speichern Sie diese auf ihrem Rechner ab.

Installation des Servers

Um den Server zu installieren benötigen Sie die aktuelle nodeJS Version und die Module faye, express, body-parser und redis. Bitte gehen Sie über das Terminal mit dem Befehl

```
cd …/Dienstanbieter [4]
```

in dem Ordner. Bitte installieren Sie dort die zusätzlichen Module mit dem Terminalbefehlen:

Nach der Installation kann der Server mit dem Befehl `node server.js` gestartet werden. Der Server läuft nun auf dem Port `3000.`

Installation des Clients

Nun installieren Sie den Client. Sie benötigen genauso wie beim Server die aktuelle nodeJS Version und die Module express, body-parser sowie ejs und fs. Bitte gehen Sie über das Terminal mit dem Befehlt

```
cd …/Dienstnutzer [4]
```

in dem Ordner.

Nach der Installation kann der Client mit dem Befehl `node client.js` gestartet werden. Der Server läuft nun auf dem Port `8000.`

Starten der Anwendung im Browser

Um die Applikation im Browser zu starten, öffnen Sie den Browser Ihrer Wahl und geben Sie die Adresse http://localhost:3000/ in die Adresszeile ein.

Die Loginseite für das Praktikumstool öffnet sich darauf und mit den Login Daten

User: dgaeb
Passwort: wba2

Kann sich der User einloggen.

[4] … → Bitte geben Sie hier ihren Path ihrer Ordner Struktur ein.

Aufgaben des Praktikumsanmeldetools

Das Praktikumsanmeldetool soll das Anmelden für die Studenten und die Betreuer erleichtern. Es ist einheitlich und kann für alle Fakultäten eingesetzt werden.

Datenhalten

Die Datenhaltung erfolgt in dem Praktikumsanmeldetool mit der NoSQL-Datenbank *Redis*. In dieser Datenbank werden die Vor- und Nachnamen und Martrikelnummer der Studenten sowie Studiengang und alle Praktika. Die CampusID und das dazugehörige Passwort befinden sich ebenfalls in dieser Datenbank. darstellen.

Die zweite Datenbank ist die Datenbank des Praktikumsanmeldetools in dieser stehen die Matrikelnummer, Vor- und Nachname des Studenten, das Praktika und die Termine (Anwesenheit und erfolgreiche Abnahme). In dieser Datenbank wird auch das Datum, die Uhrzeit und die Raumnummer des Praktikums angeben. Der Schlüssel ist hier Matrikelnummer und das Praktikum.

Aufgaben des Dienstgebers

Der Dienstgeber ist in diesem Fall der Server des Praktikumsanmeldetools und das Herzstück dieses Projektes.

Des weiteren wird durch eine Nutzereingebe der individuelle Stundenplan erstellt und die offenen Praktika angezeigt. Die Anmeldung erfolgt durch eine Nutzeraktion.

Danach werden die Praktika Termine mit dem individuellen Stundenplan und den Terminen des Praktikums abglichen. Die freien Stunden werden gespeichert und die Praktika möglichst in diese Freistunden gelegt, diesen erfolgt manuell. Die Praktika Termine werden dann in einer Liste mit Datum, Uhrzeit und Fach und Raumnummer angezeigt.

Das Praktikumsanmeldetool soll zwei Personengruppen bedienen, einerseits die Studenten und andererseits die Dozenten und Praktikumsbetreuer. Der Praktikumsbetreuer kann die bestandenen Praktika eintragen. Wenn das Praktikum endgültig bestanden wurde, wird eine Nachricht an das PSSO gesendet und direkt im Notenspiegel vermerkt. So kann ein Student direkt alle eine Klausur zum nächst möglichen Termin eintragen.

Anwendungslogik des Dienstgebers

Die Anwendungslogik Dienstgeber ist in der Lage aus dem HoPS und dem PSSO die wichtigen Daten zu laden und die Daten direkt aufzubereiten.[5] Die Erstellung des Stundenplans und die

[5] Das PSSO wird hier mit Redis statt mit SQL nachgebaut. Das HoPS wird nach ein zwei Dimensonales Array ersetzt einfachere Umsetzbarkeit.

Anmeldung sind ebenfalls senden des bestandenen Praktikum an das PSSO ein Teil der Anwendungslogik.

Aufgaben des Dienstnutzers

Der Dienstnutzer sind in diesem Fall Studenten, Dozenten, Praktikumsbetreuer und Professoren.

Der Professor bestimmt die Spielregeln des Praktikums, wie beispielsweise die Länge der einzelnen Termine, die Anzahl der Termine und die Gruppengröße. Diese Angaben werden benötigt um die Termine bilden zu können.

Jeder Student meldet sich mit seiner Campus ID und seinem Passwort an. Somit liegen die Daten wie Semester, Studiengang und bestandene Praktika vor. Danach muss der Student seinen Stundenplan eintragen[6] um und sich für die Praktika seiner Wahl anmelden.

Die Betreuer und die Dozenten tragen dann die Anwesenheit und das Bestehen oder das nicht bestehen eines Praktikums Termin ein.

Anwendungslogik des Dienstgebers

Die Anmeldung des Dienstgeben ist relativ wenig. Der Student oder Dozent kann sich einloggen. Der Student kann seine Praktika anmelden und einen Stundenplan eingeben. Der Dozent und der Professor können die Anwesenheit und die bestehen Praktika.

[6] Die Anmeldung für die Praktika findet meistens in der ersten und zweite Vorlesungswoche statt, demnach steht bei viele schon fest welche Praktika, welche Vorlesung und welche Übung besucht werden sollen auch die aus anderen Semestern.

Noch nicht umgesetzte Features

Zu den noch nicht umgesetzten Features gehören das Erstellen des Stundenplans und das Anmelden für die Praktika. Diese werden in dem zweiten Iterationsschritt in den Semesterferien erweitert.

Nicht umgesetzt Features

Erstellung des individuellen Stundenplanes

Das erste Feature das noch nicht umgesetzt wurde ist die Erstellung des individuellen Stundenplanes und dem Abgleich der freien Stunden und mit den Terminen des Praktika könnte wie folgt aussehen.

Stundenplan erstellen.

```
Function meinenStundenplanErstellen(){
      ausgabe (Aller Fächer und Wahlpflichtfächer);
      var meinStundenplan = [stunden][tagen];
      while(eigenenFächer == fertig){
            eingabe: eigenenFächer

            switch(eigenenFächer){
                  case “Fach1“:
                  fach1 = fach1[Tag][stunde];
                  break;
                  case “Fach1“:
                  fach2 = fach2[Tag][stunde];
                  break;
                  case “Fach1“:
                  fach3 = fach3[Tag][stunde];
                  break;
                  …
                  case “Fach1“:
                  fachN = fachN[Tag][stunde];
                  break;
                  case “fertig“;
                  break;
            }
      }
}
```

Stundenplan mit dem Praktikumstermine

```
function stundenpanAbgleichen (){
      for(var i = 0; i <= 15; i++){
            for(var j = 0 <= 1; j++ )
            if(fach1Praktikumstermin1[x][y] == meinStundenplan[i][j] &&
               fach1Praktikumstermin[x][y] == meinStundenplan[i][j]){
                  fügeDenStundenenAufDieListeAuf++
                  mengeDerTermineAnDemDerStudentKann++
               }
            if(fach1Praktikumstermin1[x][y] == meinStundenplan[i][j] &&
               fachNPraktikumstermin[x][y] == meinStundenplan[i][j]){
                  fügeDenStundenenAufDieListeAuf++
                  mengeDerTermineAnDemDerStudentKann++
               }
            }
      }
}
```

Automatische Verteilung

Das Features das ich nicht mehr umsetzen konnte war eine algorithmische Verteilung auf die Praktikumsgruppen, durch Zeitmangel und die noch nicht vorhandenen JavaScript Kenntnisse die dazu nötig wäre.

Ein Lösungsansatz im Pseudocode wäre gewesen:

```
Erstelle Listen mit allen Studenten die an einem Termin für alle Listen;
Sortiere nach(mengeDerTermineAnDenenDerStudentKann); //
var mengeDerTermineAnDenenDerStudentKann;
if(matrikelnummer ist in einer Gruppe){
      Lösche die eingetragenen User aus den anderen Listen && füge das Praktikum im
Stundenplan hinzu;
}
else{
      alleStudentenDieNichtInDerGruppeSind;
      mengeDerTermineAnDenenDerStudentKann -1;
}
```

Features die verbessert werden könnten, ist in jedem Fall die Datenbank. Da das offizielle PSSO mit der relationalen Datenbank SQL arbeitet. Wäre es hier Sinnvoller auch SQL zu nutzen.[7] Durch die Nutzung von SQL kann man von der PSSO Datenbank ein Select über den Name, Matrikelnummer, CampusID, Passwort, und die offenen Praktika schreiben. Dieses Datenbankfragment wird mit einem Join mit den einzelnen Terminen verbunden. Durch die Verwendung eines natürlichen Schlüssels (Matrikelnummer und Praktikum) können die Daten schneller auf der Datenbank gefiltert und abgefragt werden.

[7] Es wurde vorgeschrieben die NoSQL Datenbank zu nehmen Redis nehmen.

Kritische Betrachtung

Während des Projektes sind einige Problem aufgetreten, diese werden hier näher erklärt.

Das größte Problem in diesem Projekt ist das Fehler von Literatur in der die Vorgehensweise genaustens erläutert wird. In dem Buch von Sebastian Springer[8], welches 560 Seiten umfasst, befassen sich nur über 14 Seiten mit dem Thema REST-Server und 5 Seiten über Redis. Die Module Faye und EJS finden keine Erwähnung.

Auch die Voraussetzungen die in dem Fach Webbasierte Anwendungen 2 existieren, sind sehr interessant. JavaScript wird mit guten bis sehr guten Kenntnissen vorausgesetzt, obwohl diese Skriptsprachen im Rahmen von Webbasieren Anwendungen 1 nur an einem Workshop Tag mit 2 mal 90 Minuten Vorlesung vorgestellt wurde und mit sehr wenige Informationen weitergegeben wurde. Auf meine Frage, ob eine SQL-Datenbank verwendet werden darf, wurde mit der Begründung: „wir können nicht voraussetzen, dass Sie SQL Kenntnisse besitzen“ abgelehnt.[9]

Das zweite Problem ist das die Aufgabenstellung nicht eindeutig war, es war nicht klar was genau gemacht werden soll. Die Vorgehensweise und wie ein guter REST-Server implementiert und aufgebaut wird, wurde nicht dargestellt. Ich, beispielsweise, dachte bis zum vierten Workshops, dass eine REST-Architektur bedeutet der Dienstgeber ist Webseite oder Datenbank, die ggf. real existiert (in meinem Fall das Prüfungs- und Studierendenservice Online, PSSO, und das Hochschulplanungssystem HoPS dieser TH Köln). Informationen für meinen Webserver, den Dienstnutzer, bereitgestellt. Webseite oder Applikation lädt die für den User aufbereiteten Daten herrunter.

Ein weiteres Problem war die Zeit. Es ist unter dem Semester und zum Ende des Semesters relativ schwierig sich intensiv genug mit dem Thema auseinander zu setzen. Zumal die Workshops wenig hilfreich waren, da diese nicht nur zu kurz, ungenaue sind und die benutzten Module auch nicht ausreichend erklärt werden.

Wenig hilfreich waren auch die Feedback Gespräche, da der Raum (MI-Pool) anfangs sehr voll und laut war. Angenehmer wären 10 Minuten Slots in den man sich Eintragen muss, ähnlich wie im Audiovisuellen Medienprojekt oder in EIS.

[8] Springer, Sebastian, Node.js – Das Praxisbuch erschienen im Rheinwerk Computing 2. Aktualisierte Auflage.

[9] SQL wird im Rahmen des Faches Datenbank im dritten Semester sehr ausführlich behandelt. Während NoSQL Datenbanken in Datenbanken 1 nur erwähnt wird.

Fazit

Node ist praktisches kleines Tool schnell um einen kleinen aber stabilen Server aufzusetzen. Durch die zusätzlichen Module kann Node auch schnell erweitert werden und weitere Funktion, wie ein Datenbank oder eine Templet Engine erweitert werden.

Die REST-Architektur ist durchaus nützlich um großen System ohne Performanceproblemen diese auf kleinen Endgeräten anzuzeigen. Allerdings ist Node.js sehr neu und in der Kombination mit REST ist mit der Tatsache kaum vorhandenen JavaScript und nur für Workshops zwischen 30 – 60 Minuten kaum umsetzbar.

Quellen

Krause, Jörg, Node Einführung in node.js, textor 2015

Tilkov, Steffen; Eigenbrodt, Martin; Schreier, Siliva und Wolf, Oliver; REST und HTTP Entwicklung und Integration nach dem Architekturstil des Web dpunkt.verlag, 3 aktualisierte und erweiterte Auflage, 2015.

Springer, Sebastian, Node.js – Das Praxisbuch erschienen im Rheinwerk Computing 2. Aktualisierte Auflage.

Galo Roben Node.js & Co erschienen im dpunkt.verlag, 9. Auflage, 2015.