Einleitung

Im Rahmen des Praktikums „Datenbanken und verteilte Systeme“ am Arbeitsbereich „Verteilte Systeme und Informationssysteme (VSIS)“ der Universität Hamburg wird am Beispiel eines überschaubaren Softwareprojektes unter Anleitung in Kleingruppen eine datenbankbasierte, verteilte Anwendung mit aktuellen Technologien entwickelt.

Im vorliegenden Praktikumsbericht der Gruppe bestehend aus Victor, Judentino, Philipp und Janine wird zunächst das Projektziel kurz und präzise dargestellt und der Projektbeginn beschrieben. Anschließend wird die Problemdefinition und Anwendungsidee formuliert. Die Funktionalitäten der Anwendung werden durch die Aufteilung in Grundfunktionalitäten, sowie Ideen für zusätzliche Funktionalitäten präzise dargelegt. Auf dieser Grundlage werden die eingesetzten Technologien und die Software Architektur veranschaulicht. Anschließend wird das Praktikum reflektiert, indem der Ablauf, sowie die Probleme des Projektmanagements beschrieben werden. Im letzten Kapitel erfolgen eine Auswertung des Projektes (Was lief gut? Was sollten wir beim nächsten Mal besser machen? Was haben wir gelernt?), eine Projektevaluierung (Waren wir erfolgreich? Haben wir die definierten Ziele erreicht?) sowie ein Ausblick.

Projektziel

Ziel unserer Projektarbeit ist es, bis zum 24.08.2017 ein funktionsfähiges, datenbankbasiertes, verteiltes Softwaresystem zu entwickeln. Des Weiteren soll das gruppeninterne Ziel, stets effektiv, zielorientiert, motiviert und kommunikativ zu arbeiten, verfolgt werden.

Projektbeginn

Am ersten Tag des Praktikums hat sich zunächst jedes Gruppenmitglied selbstständig in die erforderlichen Technologien eingearbeitet. Um den Umgang mit Baqend zu lernen, wurde hierfür hauptsächlich das interaktive Tutorial durchgearbeitet, um die API und das Backend-as-a-Service-Konzept zu verstehen und zu erlernen (<https://www.baqend.com/tutorial.html>).

Zusätzlich ist jeder den Quickstart im Dashboard durchgegangen um die Funktionen von Baqend näher kennen zu lernen und das gelernte Wissen zu vertiefen.

Damit jedes Gruppenmitglied Zugriff auf alle erforderlichen Projektmaterialien hat, wurde ein Git-Repository für die Gruppe angelegt.

Am zweiten Tag des Praktikums stand die Identifikation eines relevanten und gegenwärtigen Problems sowie einer Anwendungsidee im Vordergrund. Um diesen Prozess konstruktiv zu gestalten, haben wir uns der Methode des Brainstormings bedient. Hierbei wurden zunächst spontan Ideen in den Raum geworfen, die alle protokolliert wurden. Da sich die Gruppenmitglieder gegenseitig inspiriert haben, bauten die Ideen teilweise aufeinander auf. Die Ergebnisse wurden daraufhin sortiert und bewertet und somit die finale Anwendungsidee festgelegt. Diese wird im folgenden Abschnitt beschrieben.

Problemdefinition

Schon seit einigen Jahren geht der Trend von der Bewerbungsmappe zur Online Bewerbung. Somit ist die Online Bewerbung heutzutage das gängigste Instrument, um sich auf eine Stelle zu bewerben. Laut einer aktuellen Umfrage bevorzugen rund 75% der Unternehmen die E-Mail-Bewerbung gegenüber der Bewerbungsmappe. Auch Online-Bewerbungsportale großer Unternehmen sind immer öfter aufzufinden, da vorgegebene Felder die Eingabe persönlicher Daten vereinfachen. (<http://www.stepstone.de/Karriere-Bewerbungstipps/die-online-bewerbung_altbewaehrtes-und-neue-trends.cfm>)

Das Problem, welches jedoch weiterhin besteht, ist dass die Vorauswahl der Bewerber, vor allem bei großen Unternehmen, ein sehr zeitaufwendiger Prozess ist. Eine vollständige Bewerbung besteht meist aus Bewerbungsanschreiben, Lebenslauf, Zeugnissen und weiteren Anhängen wie Zertifikate oder Fortbildungen. Vor allem bei großen Unternehmen, die jährliche mehrere hunderttausend Bewerbungen bekommen, sollte das Personal-Recruitment in Richtung eines effizienten Bewerbungsprozesses durch Kurzprofile verbessert werden.

Schaut man sich die andere Seite an, müssen sich auch Bewerber durch eine große Anzahl an Jobbörsen wie beispielsweise monster.de, indeed.com oder stepstone.de klicken, um passende Stellenangebote zu finden. Auch dieser Prozess sollte effizient verbessert werden, indem sich Unternehmen mit ihren Stellenangeboten in einer Anwendung kurz und präzise vorstellen.

Anwendungsidee

Unsere Anwendungsidee greift genau diese Probleme auf. Im Vordergrund unserer Anwendung steht der „One-Sentence-Pitch“. Bewerber sollen sich in ihrem Kurzprofil neben relevanten Angaben wie Kernkompetenzen, Softskills und der absolvierten Ausbildung vor allem durch einen Satz vorstellen, der deutlich machen soll, warum sie der beste Kandidat für die gewünschte Stelle sind.

Im Gegensatz sollen auch die Unternehmen in ihren Stellenangeboten durch einen Satz verdeutlichen, warum man sich in ihrem Unternehmen für genau diese Stelle bewerben soll.

Die Funktionsweise unserer Anwendung soll sich an der App „Tinder“ (<https://www.gotinder.com/>) orientieren. Bewerber bekommen die passenden Stellenangebote angezeigt und können diese „liken“ beziehungsweise „disliken“. Dasselbe gilt für die Unternehmen. Sie „liken“ bzw. „disliken“ die Kurzprofile der verschiedenen Bewerber. Kommt es zu einem Match, also haben sich Unternehmen und Bewerber gegenseitig geliked, werden weitere Informationen angezeigt und die Kontaktaufnahme ermöglicht.

Auf die Funktionsweise unserer Anwendung wird im nächsten Kapitel näher eingegangen.

Funktionalitäten

Grundfunktionalitäten

* Registrierungsmaske für User (Unternehmen und Bewerber)
* Die User registrieren sich durch Angabe einer E-Mail- Adresse, welche als Username fungiert und eines Passwortes.
* Login Bereich für User (Unternehmen und Bewerber)
* Die User melden sich mit ihren Daten (Username und Passwort) an.
* Bewerber erstellen ihr Profil
* Die Bewerber geben alle relevanten Informationen zu ihrer Person an, wie Name, Adresse, Geburtsdatum, Telefonnummer und E-Mail-Adresse. Außerdem geben sie die, für die Bewerbung an sich relevanten Informationen wie One-Sentence-Pitch, Jobbezeichnung, Ausbildung, Kernkompetenzen, Softskills und Sprachen an. Außerdem können Profilbild, sowie Anhänge wie Lebenslauf und Zeugnisse hochgeladen werden.
* Unternehmen erstellen ihr Profil
* Die Unternehmen geben alle relevanten Informationen wie Name, Adresse, Telefonnummer, One-Sentence-Pitch, Homepage, Gründungsdatum, Mitarbeiteranzahl, und Branche an. Außerdem kann das Unternehmenslogo, sowie weitere Bilder hochgeladen werden.
* Unternehmen erstellen Stellenangebote
* Die Unternehmen geben alle relevanten Informationen zu den einzelnen Stellenangeboten an. Hierzu zählen die genaue Bezeichnung, Beschreibung, Vertragsart, Startdatum, Sprache und die Anforderungen für die Stelle.
* Erstellung der Kurzprofile
* Anhand der eingegebenen Informationen werden in der App Kurzprofile für die Stellenangebote und die Bewerber erstellt. MOCKUP??
* Swipen durch die Kurzprofile
* Das Swipen durch die Kurzprofile stellt die Kernfunktionalität unserer Anwendung dar. Bewerber swipen durch die, nach ihren Kriterien wie Berufsfeld, Vertragsart und Ort gefilterten, Stellenangebote der Unternehmen. Die Unternehmen swipen ebenfalls durch die nach ihren individuell festgelegten Kriterien gefilterten Bewerberprofile. Die einzelnen Kurzprofile können geliked oder gedisliked werden. Dies kann einerseits durch Klicken auf einen Daumen nach oben für like, und für einen Daumen nach unten für dislike, oder andererseits nach swipen des Kurzprofils nach rechts für like und swipen des Kurzprofiles nach links für dislike geschehen.
* It’s a Match
* Wenn sich ein Unternehmen und ein Bewerber im Laufe des Swipingprozesses gegenseitig geliked haben, wird ihnen dies mitgeteilt. Daraufhin werden weitere Informationen, die noch nicht im Kurzprofil aufgeführt wurden, ausgetauscht und die Kontaktaufnahme durch den E-Mail-Verkehr ermöglicht.

Ideen für zusätzliche Funktionalitäten

* Im Zuge der näheren Kontaktaufnahme nach einem Match könnte in unserer Anwendung eine eigene Chat-Funktion zur Verfügung gestellt werden, anstatt den weiteren Bewerbungsprozess auf den E-Mail-Verkehr auszulagern. Dies vereinfacht den Bewerbungsprozess, da der komplette Prozess bis zum Erstgespräch über unsere Anwendung läuft. Dies erhöht die Qualität, sowie die Nutzeranzahl unserer Anwendung.
* Unsere Anwendung ist zum Zeitpunkt der Abgabe am 24.08.2017 lediglich in der Sprache Deutsch verfügbar. Um die Nutzeranzahl der Anwendung zu erhöhen, sollte die Sprache individuell einstellbar sein. Priorität hätten hierbei vor allem die Einstellungen für Englisch, Spanisch, Französisch, Russisch und Chinesisch.
* Für Benutzer sollte es die Möglichkeit geben, sich mit vorhandenen Konten von Facebook oder LinkedIn registrieren und anmelden zu können. Dadurch können persönliche Informationen übernommen werden und die Erstellung des Profils verkürzt werden.
* Momentan gibt es lediglich die Möglichkeit, einen Job in einer bestimmten Stadt durch Angabe des Stadtnamens zu suchen. Um diese Suche flexibler zu gestalten, sollte eine Umkreissuche durch Einbindung von Google Maps ermöglicht werden. So gibt ein Benutzer beispielsweise ein, dass er einen Job in Hamburg und in einem Umkreis von 50 km sucht, um zum Beispiel auch Stellenangebote aus Buchholz angezeigt zu bekommen. Dieses Feature ist vor allem bei kleineren Städten mit wenigen Stellenangeboten sinnvoll.
* Im Hinblick auf unsere Einnahmequellen könnte insbesondere für die Unternehmen die Möglichkeit eines kostenpflichtigen Premium-Angebotes geschaffen werden. Dieses bietet ihnen die Möglichkeit, ihre Stellenangebote besonders hervorzuheben und um somit mehr Bewerber zu erreichen.  
  Speziell für Bewerber könnte ein Premium-Angebot zur Verfügung gestellt werden, welches es ihnen ermöglicht, ihre Kompetenzen verifizieren zu lassen und ihr Profil somit authentischer wirken zu lassen.
* Im Laufe des weiteren Refactorings sollte auch die Datenvalidierung überarbeitet werden. Anstatt jedes einzelne Feld aufwendig zu überprüfen, könnte das von Angular zur Verfügung gestellte „Validating Template-Driven Forms“ verwendet werden, um den Quellcode übersichtlicher zu gestalten.

Zerlegung in Teilaufgaben und Festlegung von Meilensteinen/Zeiträume zur Umsetzung

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 07.-08.08 | 09.-10.08 | 11.-14.08 | 15.-16.08 | 17.-18.08 | 21.-22.08 | 23.-24.08 |
| Ideenfindung |  |  |  |  |  |  |  |
| Anforderungsanalyse/ Funktionalitäten |  |  |  |  |  |  |  |
| Relevante Informationen für Profile identifizieren |  |  |  |  |  |  |  |
| Klären von Entwurfsfragen |  |  |  |  |  |  |  |
| Erstellen von Mockups |  |  |  |  |  |  |  |
| Festlegung der einzusetzenden Technologien |  |  |  |  |  |  |  |
| Datenbank Einpflegung |  |  |  |  |  |  |  |
| Registrierung Frontend/Backend |  |  |  |  |  |  |  |
| Login Frontend/Backend |  |  |  |  |  |  |  |
| Registrierungs-Email |  |  |  |  |  |  |  |
| Passwort Reset |  |  |  |  |  |  |  |
| Profilkonfigurationen |  |  |  |  |  |  |  |
| Profilkarten |  |  |  |  |  |  |  |
| Karten für Stellenangebote |  |  |  |  |  |  |  |
| File-API |  |  |  |  |  |  |  |
| Swipe-Frontend |  |  |  |  |  |  |  |
| Swipe-Backend |  |  |  |  |  |  |  |
| Matching-Algorithmus-Backend |  |  |  |  |  |  |  |
| Erzeugen von Testdaten |  |  |  |  |  |  |  |
| User-Konto löschen |  |  |  |  |  |  |  |
| Speichern der hochgeladenen Fotos in Datenbank |  |  |  |  |  |  |  |
| Speichern der Likes |  |  |  |  |  |  |  |
| Anzeige für keine Matches |  |  |  |  |  |  |  |
| Anzeige für keine passenden Angebote |  |  |  |  |  |  |  |
| Refactoring |  |  |  |  |  |  |  |
| Datenvalidierung |  |  |  |  |  |  |  |
| Design überarbeiten |  |  |  |  |  |  |  |
| Erstellung des 1. Vortrages |  |  |  |  |  |  |  |
| Erstellung des 2. Vortrages |  |  |  |  |  |  |  |
| Erstellung des 3. Vortrages |  |  |  |  |  |  |  |
| Erstellung Praktikumsbericht |  |  |  |  |  |  |  |

Eingesetzte Technologien (Was wird wofür genutzt? /Warum haben wir uns dafür entschieden?)

Angular

Baqend

Google Material Design

Bootstrap

Datenbankschema

Unsere Datenspeicherung läuft über Baqend. Im Folgenden wird unser Datenbankschema beschrieben.

* Tabelle User
* Allgemeine Tabelle, in welcher der Username gespeichert wird.
* Tabelle Unternehmen
* Erbt von Tabelle User. In ihr werden alle spezifischen Daten zum Unternehmen gespeichert.
* Tabelle Benutzer
* Erbt von Tabelle User. In ihr werden alle spezifischen Daten zum Benutzer gespeichert.
* Tabelle Stellenangebot
* Ein Stellenangebot wird von einem Unternehmen erstellt. Hier werden alle spezifischen Daten zu einem Stellenangebot gespeichert.
* Tabelle Unternehmenlikes
* Speichert die Bewerber, welche von einem Unternehmen geliked werden.
* Tabelle Bewerberlikes
* Speichert die Stellenangebote, welche von einem Bewerber geliked werden.
* Tabelle Match
* Speichert die Stellenangebote (Unternehmen) und die Bewerber, die sich gegenseitig geliked haben.
* Tabelle Berufsfeld
* Speichert vordefinierte Berufsfelder, aus denen Unternehmen und Bewerber auswählen.
* Tabelle Vertragsart
* Speichert vordefinerte Vertragsarten, aus denen Unternehmen und Bewerber auswählen.

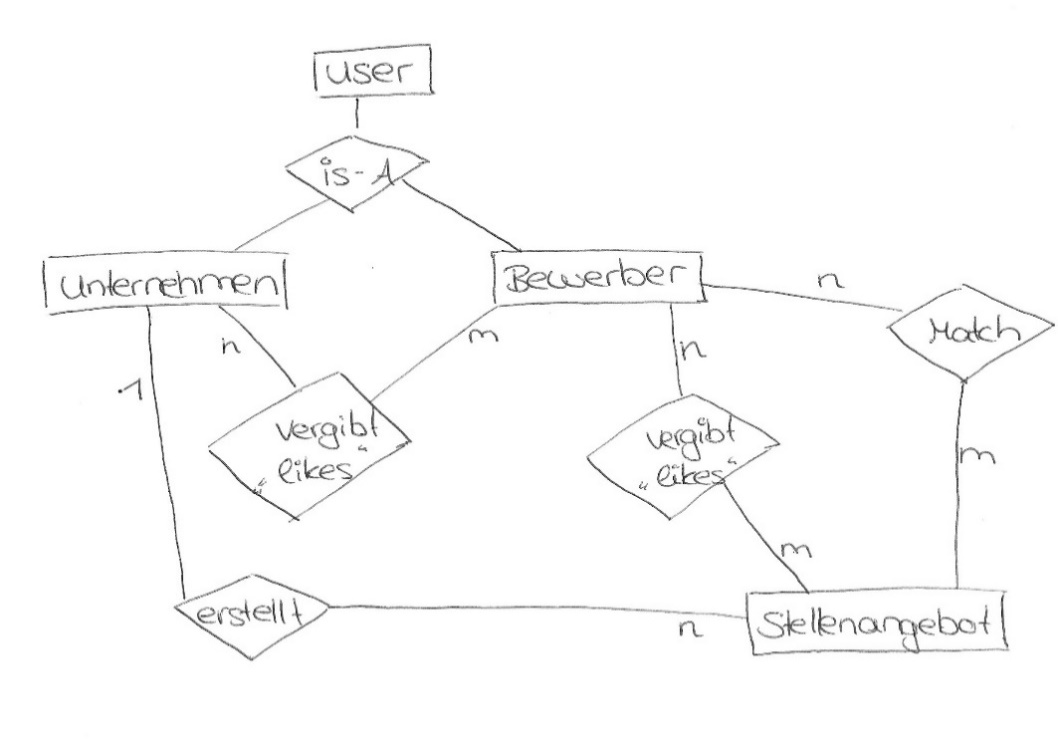
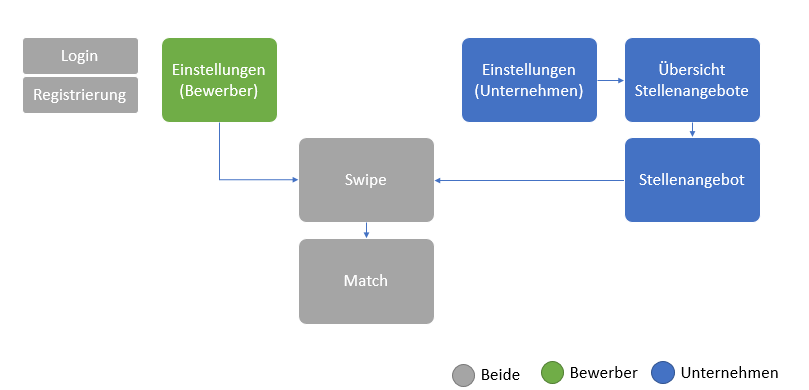


Abbildung 1: Skizze unseres Datenbankschemas als Entity-Relationship-Modell

Software – Architektur

Abbildung 2: Skizze unserer Software-Architektur

Software-Architektur – Daten für das Matching

Die für das Matching verwendeten Daten liegen überwiegend als Auswahllisten vor, sodass die Menge der unterschiedlichen Werte beschrönkt ist. Bei freien Benutzereingaben, hätte man das Problem, dass schon minimale Abweichungen wie zum Beispiel („Software-Entwickler“ vs. „Software Entwickler“) ein Match verhindern, bzw. eine komplizierte Suche erfordern. Daher haben wir uns entschieden nur anhand von Feldern zu matchen, bei denen eine vorgegebene Wertemenge sinnvoll ist. Hierzu zählen die gesprochenen/geforderten Sprachen (Deutsch, Englisch, …), die verschiedenen Berufsfelder (IT, Gastronomie, Rechtswesen, …) und verschiedenen Vertragsarten (Teilzeit, Ausbildung, Minijob, …).  
Außerdem matchen wir anhand des Monatsgehalts, der Eigenschaft befristet/unbefristet und des Arbeitsortes (trotz des oben angesprochenen Problems aktuell exaktes Matching, Umkreissuche ist für eine spätere Version vorgesehen).

Die Auswahllisten für die Sprachen, etc. haben wir als eigene Tabellen in der Datenbank hinterlegt, die dann referenziert werden, sodass auch wirklich nur die vorgegebenen Werte verwendet werden können.

Software-Architektur – Services

Die von den Komponenten benötigten Daten werden von Services zur Verfügung gestellt. So liefert zum Beispiel der Vacancy-Service die Stellenangebote des eingeloggten Unternehmens.  
Eine kleine Besonderheit stellt der DropDownDataService dar, der die Daten nur einmalig lädt und dann speichert. Dies ist möglich, da die DropDownDaten wie zum Beispiel die Sprachen nur angezeigt, aber nie verändert werden.

Software-Architektur – Benutzer-Berechtigungen

Um die Daten der Benutzer zu schützen, sind die Berechtigungen an den Datenbank-Tabellen so streng wie möglich.   
Für das Matching ist es nötig, dass Bewerber lesenden Zugriff auf die Unternehmensdaten haben und die Unternehmen entsprechnend auf die Bewerberdaten. Dafür haben wir die beiden Rollen Bewerber/Unternehmen definiert, von denen jeder User bei der Anmeldung der entsprechenden Rolle zugewiesen wird.  
Der Zugriff auf die Bewerber-Tabelle ist zum Beispiel so geregelt, dass die Unternehmen-Rolle lesenden Zugriff (Load und Query) hat und Bewerber vollen Zugriff. Die Berechtigung für Bewerber sind dann aber auf Objekt-Ebene auf den eigenen Datensatz beschränkt, da eine Interaktion zwischen Usern des gleichen Typs nicht vorgesehen ist.

Software-Architektur – Schutz von Komponenten durch CanActivate-Guards

Wird eine bestimmte URL aufgerufen, z.B. durch einen Link in der Anwendung oder durch direktes eingeben in den Browser, so leitet der Router zu der angeforderten URL weiter und lädt die entsprechende Komponente. Um zu vermeiden, dass jemand so auf eine Seite gelangt, die er nicht erreichen soll (z.B. ein Bewerber die Seite zum Anlegen von Stellenangeboten oder ein User, der sein Profil noch nicht vervollständigt hat, die Seite zum Swipen), haben wir CanActivate-Guards definiert.

Neben dem vordefinierten Guard DBLoggedIn haben wir die beiden Guards IsBewerber und IsCompany hinzugefügt, die dafür sorgen, dass man nicht auf Seiten navigieren kann, die für den jeweils anderen Benutzertyp vorgesehen sind. Stattdessen wird man auf die entsprechende Seite für den eigenen Typ weitergeleitet. In der App sind stets nur die richtigen Links verfügbar, z.B. steuern wir im Hauptmenü die Anzeige der Menüpunkte über den Usertyp, den ConfigStatus und den LoggedIn-Status. Aber bei Kenntnis der Linkadressen könnte man die falschen URLs dennoch aufrufen.  
Außerdem haben wir noch einen DBNotLoggedIn-Guard eingefügt, der verhindert, dass ein eingeloggter User die Seiten für Login oder Signup erreichen kann, mit denen er eh nichts anfangen könnte.

Wesentliche Entscheidungen im Projekt

Reflektion des Praktikums

Ablauf und Probleme

Projektauswertung und Projektevaluierung

Ausblick