Bachelorarbeit

Entwurf und Implementierung einer Webanwendung für die Jobsuche mit Unterstützung durch Testautomatisierung

zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

vorgelegt dem

Fachbereich Mathematik, Naturwissenschaften und Informatik

der Technischen Hochschule Mittelhessen

Yvan Richnel Tchiengue

im September 2023

Referent: Prof. Dr. Axel Schumann
Korreferent: Herr Manuel Groh

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig und unter ausschließlicher Verwendung der angegebenen Literatur und Hilfsmittel erstellt zu haben.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

>Ort, Datum< >Unterschrift<

Inhalt

Abbildungen

Tabellen

Listings

Abkürzungen und Formelzeichen

Kurzfassung

1 Einleitung

In einer sich ständig verändernden Welt ist die Arbeitssuche für viele Menschen auf der ganzen Welt zu einem wichtigen Anliegen geworden. Eine Stelle zu finden, die den eigenen Fähigkeiten, Interessen und beruflichen Zielen entspricht, ist eine Herausforderung, der sich viele Menschen stellen müssen. In diesem Zusammenhang spielen Jobsuchplattformen eine wesentliche Rolle, indem sie die Kontaktaufnahme zwischen Arbeitssuchenden und Arbeitgebern erleichtern.

Diese Bachelorarbeit konzentriert sich auf die Entwicklung einer Jobsuchplattform, ein leistungsstarkes Instrument, das den Prozess der Jobsuche für Einzelpersonen, die nach beruflichen Möglichkeiten suchen, erleichtern soll.

Kamerun ist ein aufstrebender Land mit einer steigenden Nachfrage nach Arbeitsplätzen. Mit einer jungen und dynamischen Bevölkerung ist es von entscheidender Bedeutung, innovative Lösungen anzubieten, um kamerunische Talente mit den verfügbaren Arbeitsmöglichkeiten zu verbinden. Diese Bachelorarbeit befasst sich speziell mit den Herausforderungen und Chancen, die mit der Schaffung einer Plattform für die Jobsuche in Kamerun verbunden sind, wobei die einzigartigen Merkmale des Lands berücksichtigt und geeignete Lösungen vorgeschlagen werden.

1.1 Motivation und Problemstellung

In Kamerun beobachten wir eine signifikante demografische Konzentration in zwei großen Städten, Yaoundé und Douala, die allein Millionen von Einwohnern beherbergen. Der schrittweise Ausbau des Internetzugangs in diesen Regionen hat die Qualität des täglichen Lebens zweifellos verbessert. Neben Yaoundé und Douala gibt es jedoch noch viele andere Städte, die über das riesige kamerunische Staatsgebiet von 475.445 km² verteilt sind und in denen der Rest der Bevölkerung lebt.

Es ist wichtig anzumerken, dass Einzelpersonen, die an Universitäten im Norden des Landes, wie der Universität Ngaoundéré, studiert und ihren Abschluss gemacht haben, Schwierigkeiten haben, einen Arbeitsplatz zu finden, obwohl es auch in anderen Teilen des Landes freie Stellen gibt. Dies ist vor allem auf den Mangel an digitalen Plattformen zurückzuführen, die Arbeitgeber und Arbeitssuchende miteinander verbinden. So kann es sein, dass es in Douala eine offene Stelle gibt, aber eine Person, die beispielsweise in Garoua wohnt, würde nicht von dieser Möglichkeit erfahren, da es sowohl umständlich als auch finanziell schwierig wäre, persönlich von Garoua nach Douala zu reisen, um sich zu bewerben.

Die Einrichtung einer zuverlässigen Online-Plattform für die Jobsuche würde es Einzelpersonen, unabhängig davon, ob sie im Norden oder im Süden des Landes leben, ermöglichen, sich auf Stellenangebote in jedem Ort zu bewerben. Diese Lö-

sung würde auch das Problem lösen, dass Menschen, die in derselben Stadt wohnen, in der es eine offene Stelle gibt, nichts davon wissen, da diese Möglichkeiten oft durch Mundpropaganda oder durch Aushänge an öffentlichen Orten bekannt gemacht werden.

Ich kenne Menschen in Kamerun, die auf der Suche nach einer Arbeitsstelle wiederholt von Tür zu Tür gehen mussten, in der Hoffnung, ein Unternehmen zu finden, das Arbeitskräfte sucht, aber ohne Erfolg. Gleichzeitig war ein kamerunischer Unternehmer gezwungen, sein Unternehmen für einige Zeit zu schließen, weil es an qualifizierten und geeigneten Arbeitskräften mangelte.

Die Einrichtung einer robusten digitalen Plattform, die eine effektive Verbindung zwischen Arbeitgebern und Arbeitssuchenden ermöglicht, ist daher von entscheidender Bedeutung, um diese Probleme zu beheben und ein besseres Gleichgewicht zwischen Angebot und Nachfrage nach qualifizierten Arbeitsplätzen im ganzen Land zu fördern.

1.2 Zielsetzung

Im Rahmen der Thesis über die Schaffung einer Plattform für die Stellensuche ist das Hauptproblem, das es zu lösen gilt, das bestehende Ungleichgewicht auf dem kamerunischen Arbeitsmarkt. Dieses Ungleichgewicht äußert sich in einer hohen Anzahl qualifizierter Arbeitssuchender, die nach beruflichen Möglichkeiten suchen, während es für Arbeitgeber schwierig ist, die richtigen Talente für ihre spezifischen Bedürfnisse zu finden. Dieses Problem führt zu einer Lücke zwischen Angebot und Nachfrage auf dem kamerunischen Arbeitsmarkt, was hohe Arbeitslosigkeit und eine unzureichende Nutzung von Qualifikationen zur Folge hat.

Die Arbeitssuche ist für Bewerber oft ein komplexer und zeitaufwändiger Prozess mit vielen Herausforderungen wie der Suche nach geeigneten Möglichkeiten, der Übereinstimmung mit den Anforderungen der Arbeitgeber, der effektiven Präsentation von Kompetenzen und dem Zugang zu relevanten Informationen über verfügbare Beschäftigungsmöglichkeiten. Darüber hinaus stehen Arbeitgeber vor der Herausforderung, ein breites Spektrum an qualifizierten Bewerbern zu erreichen, deren Eignung für die offenen Stellen zu bewerten und den Einstellungsprozess effektiv zu steuern.

Mit der Schaffung einer auf Kamerun zugeschnittenen Plattform für die Stellensuche soll dieses Problem gelöst werden, indem eine effiziente Kontaktaufnahme zwischen Arbeitssuchenden und Arbeitgebern erleichtert wird. Die Plattform wird einen zentralen Bereich bereitstellen, in dem Bewerber ihre Fähigkeiten präsentieren, nach relevanten Möglichkeiten suchen und sich auf Stellenangebote bewerben können, während Arbeitgeber gezielt Stellenangebote veröffentlichen, nach qualifizierten Bewerbern suchen und den Einstellungsprozess rationeller verwalten können.

1.3 Aufbau der Arbeit

2 Stand der Wissenschaft und Technik

Es ist wichtig zu erwähnen, dass die Forschung im Bereich der Entwicklung von Webanwendungen, insbesondere im Zusammenhang mit der Arbeitssuche, intensiv war. Es wurden zahlreiche Forschungsarbeiten und Artikel veröffentlicht, die eine solide Grundlage an Wissen und Techniken für diese Art der Entwicklung bieten. Zu den Schlüsselaspekten dieses Themas gehören nutzerzentriertes Design, Zugänglichkeit, Sicherheit, Effizienz und Zuverlässigkeit.

Die Testautomatisierung spielt bei der Entwicklung moderner Webanwendungen eine entscheidende Rolle und ist eine zunehmend gängige Praxis, um die Effizienz und Zuverlässigkeit von Software zu steigern. Die Testautomatisierung ist entscheidend, um sicherzustellen, dass die Anwendung wie geplant funktioniert, und ermöglicht es, Regressionen und Fehler schnell und zuverlässig zu identifizieren.[8]

Im Hinblick auf die Sicherheit veröffentlicht das Open Web Application Security Project (OWASP) regelmäßig eine Liste der zehn wichtigsten Sicherheitslücken in Webanwendungen, die eine wertvolle Ressource für Entwickler von Webanwendungen darstellt. Sicherheit ist für Anwendungen zur Stellensuche besonders wichtig, da sie häufig sensible Daten wie persönliche Informationen und Karrieredetails der Nutzer verarbeiten.[13]

Die Entwicklung einer Webanwendung für die Stellensuche, unterstützt durch automatisiertes Testen, ist ein etabliertes und sich ständig weiterentwickelndes Forschungsgebiet. Die bestehende Forschungsarbeit bietet eine reiche Informationsquelle über bewährte Verfahren, Herausforderungen und aktuelle Trends in diesem Bereich.

3 Grundlagen

3.1 Werkzeuge

3.1.1 Angular

Angular ist ein leistungsfähiges, robustes und zunehmend beliebtes Frontend-Framework, das von Google entwickelt wurde. Es wurde 2010 unter dem Namen AngularJS eingeführt, erfuhr mit Version 2 im Jahr 2016 eine wichtige Weiterentwicklung und hat seitdem einen exponentiellen Wachstumspfad eingeschlagen, der es zu einer beliebten Wahl unter Entwicklern für die Erstellung umfangreicher Webanwendungen gemacht hat.[1]

Eine der herausragenden Eigenschaften von Angular liegt in seiner komponentenbasierten Architektur, welche die Förderung der Modularität, Wiederverwendbarkeit und Wartbarkeit des Codes ermöglicht. Durch diese modulare Architektur haben Entwickler die Möglichkeit, Anwendungen inkrementell aufzubauen, wodurch die Verwaltung umfangreicher Entwicklungsprojekte erleichtert wird.[12]

Abb. 3.1: Angular Übersicht (aus [11])

3.1.2 git

Git ist ein verteiltes Versionskontrollsystem, das 2005 von Linus Torvalds, dem Schöpfer von Linux, entwickelt wurde[16]. Es wurde entwickelt, um alles, von kleinen bis zu sehr großen Projekten, schnell und effizient zu verwalten. Git ist leicht zu erlernen und hat einen winzigen Speicherfußabdruck bei blitzschneller Leistung.

Die Versionskontrolle ist ein System, das den Verlauf von Änderungen an einer Datei oder einer Gruppe von Dateien über die Zeit hinweg aufzeichnet, sodass spezifische Versionen später abgerufen werden können. Mit Git besitzt jeder Entwickler

eine vollständige lokale Kopie des Projektverlaufs, was eine außergewöhnliche Flexibilität ermöglicht und verschiedene Arbeitsmethoden unterstützt.

Abb. 3.2: Git Übersicht (aus [3])

3.1.3 Html

HTML (HyperText Markup Language) ist die grundlegende Säule jeder Webentwicklung. Sie wurde Anfang der 1990er Jahre eingeführt und dient als Skelett für alle Webseiten. Sie ermöglicht es Entwicklern, Inhalte zu strukturieren und Informationen organisiert und verständlich darzustellen.[15]

HTML5, die neueste Version von HTML, die 2014 eingeführt wurde, brachte eine Reihe von Verbesserungen und neuen Funktionen mit sich, die zur Vereinfachung und Standardisierung der Webentwicklung beigetragen haben. Zu diesen Neuerungen gehören semantische Elemente wie <article>, <section> und <nav>, die es Entwicklern ermöglichen, aussagekräftigere und zugänglichere Webseitenstrukturen zu erstellen.[14]

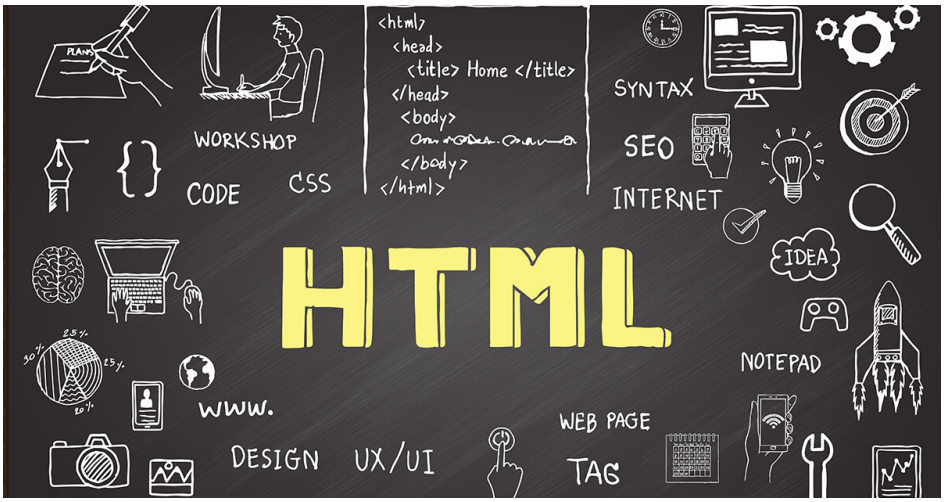

Abb. 3.3: Html Übersicht (aus [5])

3.1.4 TypeScript

TypeScript ist eine höhere Programmiersprache, die 2012 von Microsoft eingeführt wurde. Es ist eine typisierte Obermenge von JavaScript, was bedeutet, dass es JavaScript um Funktionen erweitert, einschließlich eines statischen Typensystems. Alle gültigen JavaScript-Programme sind auch gültige TypeScript-Programme, was die Übernahme von TypeScript für bestehende JavaScript-Entwickler einfacher macht.[19]

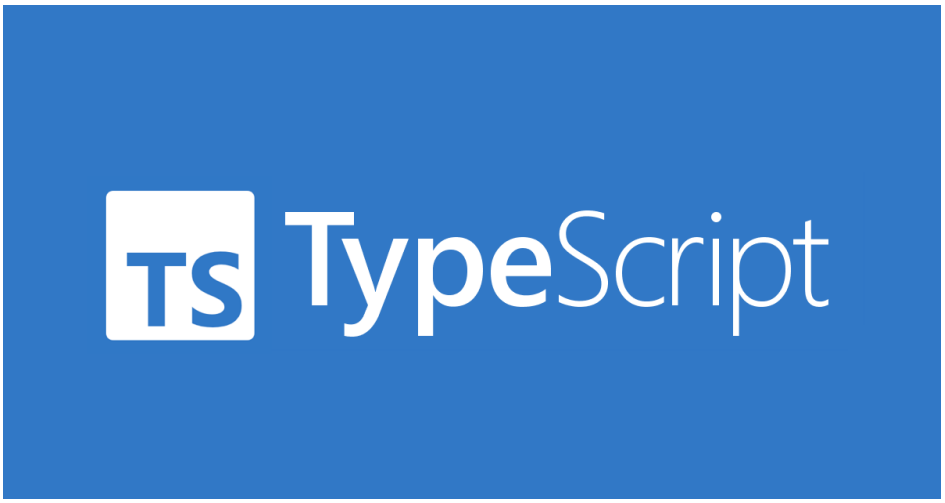

Abb. 3.4: TypeScript Übersicht (aus [20])

3.1.5 JavaScript

JavaScript ist eine Programmiersprache, die 1995 von Brendan Eich entwickelt wurde. Sie wurde ursprünglich für den Netscape Navigator entwickelt, um dynamische Interaktionen in Webseiten zu ermöglichen, und entwickelte sich zur am weitesten verbreiteten Programmiersprache für die clientseitige Webentwicklung. Im Laufe der Jahre hat sich JavaScript weiterentwickelt und mit der Einführung von Plattformen wie Node.js auch auf die serverseitige Entwicklung ausgedehnt.[7]

Abb. 3.5: JavaScript Übersicht (aus [6])

Trotz vieler Kritikpunkte hat JavaScript den Test der Zeit bestanden und ist nach wie vor eine der beliebtesten und meistgenutzten Programmiersprachen der Welt[2]. Seine Allgegenwärtigkeit in der Webentwicklung, seine ständige Weiterentwicklung und seine Flexibilität machen es zu einem wichtigen Thema für jede Studie über Softwareentwicklung.

3.1.6 CSS

Cascading Style Sheets (CSS) ist eine Stylesheet-Sprache, die zur Beschreibung des Layouts eines in HTML oder XML geschriebenen Dokuments verwendet wird. CSS wurde vom World Wide Web Consortium (W3C) entwickelt und 1996 zum ersten Mal veröffentlicht. Es gehört neben HTML und JavaScript zu den Grundpfeilern der Webentwicklung.[23]

3.1.7 GitHub

3.1.8 WebStorm

3.2 HTTP

3.3 JSON

3.4 API

3.5 NodeJS

3.6 NPM

3.7 Webanwendung

3.8 Mobilanwendung

3.9 Unterschied zwischen Web- und Mobilanwendung

3.10 Website

3.11 Unterschied zwischen Website und Webanwendung

4 Methodik

4.1 Schritte zur Entwicklung einer Webanwendung

4.1.1 Analyse der Bedürfnisse und Ziele

In diesem Schritt werden die Anforderungen und Ziele des Projekts sowie die Bedürfnisse der Endbenutzer eingehend analysiert. Dieser erste Schritt ist von entscheidender Bedeutung, um den Umfang und die Funktionen der Anwendung klar abzugrenzen und damit eine solide Grundlage für den gesamten Entwicklungsprozess zu schaffen. Sie bedeutet, dass die Anforderungen und Ziele des Projekts gesammelt, verstanden und genau definiert werden müssen, bevor die Entwicklung beginnt. Diese Phase ist von entscheidender Bedeutung, da sie den Rahmen für das Projekt vorgibt, die Entscheidungsfindung beim Design lenkt und sicherstellt, dass die entwickelte Anwendung die Erwartungen der Nutzer und die festgelegten Geschäftsziele erfüllt.

4.1.2 das Design der Anwendung

Die Designphase ist bei der Erstellung einer Webanwendung von entscheidender Bedeutung. Ihr Ziel ist es, eine umfassende Vision der Anwendung zu entwickeln, indem ihre Architektur, Struktur, Benutzeroberfläche (UI) und Benutzererfahrung (UX) genau definiert werden. Als wesentlicher Bestandteil des Prozesses spielt diese Phase eine entscheidende Rolle für den Erfolg des Projekts, da sie dafür sorgt, dass die während der Analyse ermittelten Anforderungen und Ziele in einen konkreten und kohärenten Plan umgewandelt werden, der als Grundlage für die Entwicklung dient.

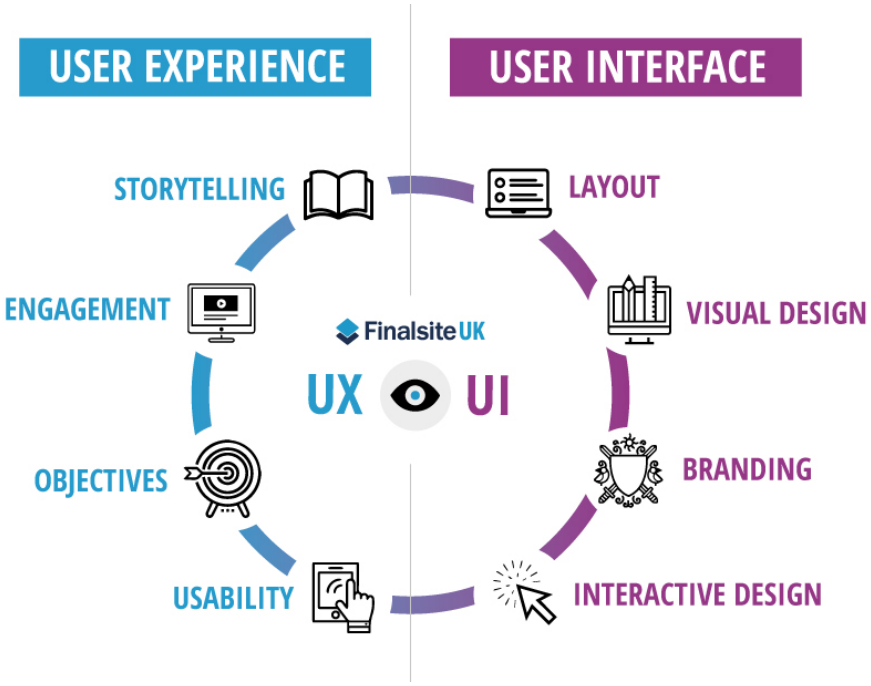

Abb. 4.1: User Experience, User Interface(aus [21])

4.1.3 die Back-End-Entwicklung

Die Backend-Entwicklung ist die Phase bei der Erstellung einer Webanwendung, in der die für den Benutzer unsichtbaren, aber dennoch für das reibungslose Funktionieren der Anwendung wichtigen Prozesse gesteuert werden. Diese Phase betrifft den Server, die Datenbank und die Serveranwendungen (d. h. die API), die die Funktionen der Anwendung versorgen.

Die Backend-Entwicklung beinhaltet hauptsächlich die Erstellung einer API (Application Programming Interface), die die Kommunikation zwischen dem Front-End der Anwendung und dem Server ermöglicht. Dies kann mithilfe einer Vielzahl von Programmiersprachen wie JavaScript (Node.js), Python, Ruby, Java, PHP, .NET u. a. geschehen. Die API empfängt Anfragen vom Frontend, interagiert bei Bedarf mit der Datenbank und gibt die angeforderten Daten an das Frontend zurück.[17]

Die Verwaltung der Datenbank ist ein weiterer wichtiger Aspekt der Backend-Entwicklung. Datenbanken speichern die für die Anwendung benötigten Informationen, wie z. B. Benutzerdetails, Nachrichten, Dateien usw. Backend-Entwickler müssen den richtigen Datenbanktyp auswählen (z. B. SQL wie PostgreSQL oder NoSQL wie MongoDB) und die Struktur der Datenbank so gestalten, dass effiziente Abfragen erleichtert werden.

4.1.4 die Front-End-Entwicklung

Dieser Teil der Entwicklung befasst sich mit der Benutzeroberfläche und allem, was für den Benutzer sichtbar ist. Frontend-Entwickler verwenden Programmiersprachen wie HTML, CSS und JavaScript, um das Design, die Benutzeroberfläche und das interaktive Verhalten einer Webanwendung zu gestalten.

Die Frontend-Entwicklung stellt eine anspruchsvolle Aufgabe dar, die eine harmonische Verbindung von Programmier-, Design- und UX-Kompetenzen erfordert. Sie spielt eine entscheidende Rolle bei der Schaffung einer positiven Nutzererfahrung und ist direkt verantwortlich für das Erscheinungsbild, das Gefühl und die Interaktivität einer Webanwendung.

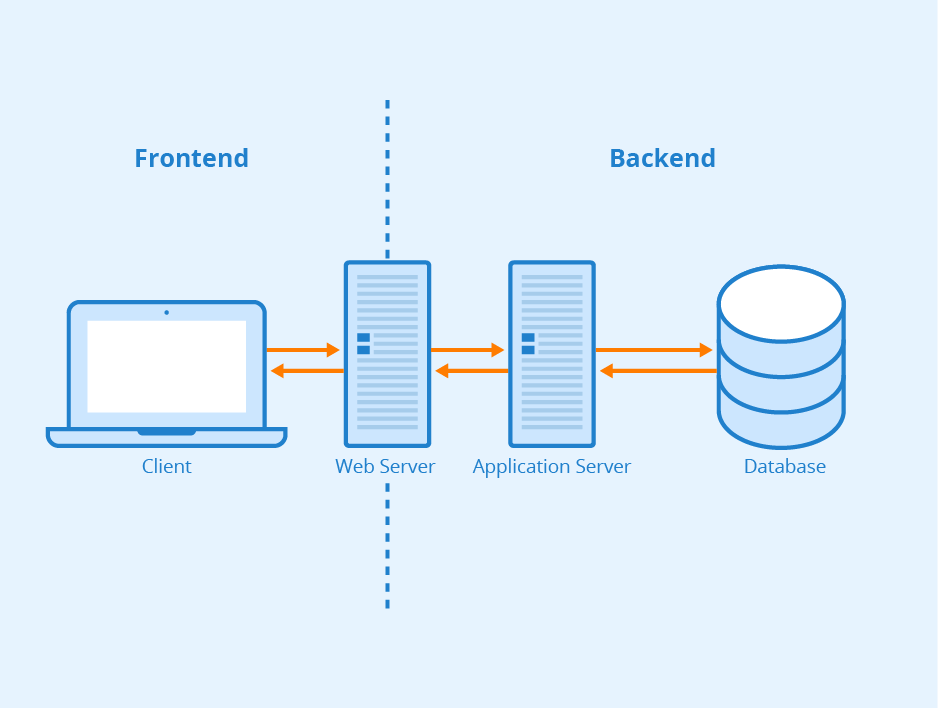

Abb. 4.2: Back-End vs. Front-End Development(aus [18])

4.1.5 Die Durchführung der Tests

Diese Phase stellt sicher, dass die Anwendung wie geplant funktioniert, bevor sie in Produktion geht, und ermöglicht es, Fehler zu erkennen und zu beheben, bevor sie sich auf die Endbenutzer auswirken.

Tests finden in der Regel auf mehreren Ebenen statt.

- **Unit-Tests** zum Beispiel überprüfen, ob die einzelnen Komponenten der Anwendung richtig funktionieren. Sie konzentrieren sich auf isolierte Teile des Codes, um sicherzustellen, dass sie unter verschiedenen Bedingungen richtig funktionieren.[10]
- **Integrationstests** prüfen, ob die einzelnen Komponenten der Anwendung richtig zusammenarbeiten. Sie können z. B. testen, wie die verschiedenen Teile der Anwendung mit der Datenbank oder anderen externen Diensten interagieren.[9]
- **End-to-End-Tests (E2E)** simulieren die gesamte Benutzererfahrung, um sicherzustellen, dass der gesamte Interaktionsfluss wie erwartet funktioniert.

4.1.6 Deployment der Webanwendung

Zu diesem Zeitpunkt wird die Anwendung, die entworfen, entwickelt, getestet und freigegeben wurde, schließlich für die Endbenutzer im Internet verfügbar gemacht.

Eine der ersten Entscheidungen, die bei der Bereitstellung getroffen werden müssen, ist die Wahl des Hostings. Dabei kann es sich um einen dedizierten Server, einen gemeinsam genutzten Server oder eine Umgebung in der Cloud handeln. Jede Option hat ihre eigenen Vor- und Nachteile in Bezug auf Kosten, Leistung, Sicherheit und Flexibilität[4].

Hosting-Anbieter wie Amazon Web Services[5], Google Cloud[6] und Microsoft Azure[3] werden häufig verwendet, um moderne Webanwendungen zu hosten.

Wenn der Server vorbereitet und konfiguriert wurde, wird dann der Code der Anwendung auf den Server übertragen. Dies kann manuell geschehen, aber es ist üblich, Werkzeuge für kontinuierliche Integration/continuous deployment (CI/CD) zu verwenden, um diesen Prozess zu automatisieren[1]. Diese Tools, wie Jenkins[2] oder GitLab CI/CD[3], stellen die Anwendung automatisch bereit, wenn der Quellcode geändert wird.

Als nächstes muss die Domäne der Anwendung konfiguriert werden. Dies bedeutet in der Regel, dass Sie einen Domainnamen[4] von einem Registrar kaufen und diesen dann so konfigurieren, dass er auf die IP-Adresse des Servers verweist.

Es ist auch wichtig, ein SSL-Zertifikat einzurichten, um die Kommunikation zwischen der Anwendung und den Nutzern zu sichern. Dadurch werden die ausgetauschten Daten verschlüsselt, was das Abfangen durch böswillige Akteure erheblich erschwert.[22]

4.2 Kommunikation zwischen Fron-End und Back-End

4.3 Testautomatisierung

5 Bearbeitung und Ergebnisse

5.1 Bearbeitungsphase

5.1.1 Verwendete Technologien

5.1.2 Verwendete Bibliotheken

5.1.2.1 cors

5.1.2.2 jwt

5.1.2.3 express

5.1.2.4 bcript

5.1.3 Implementierung der wichtigsten Funktionen

5.1.3.1 Back-End

```
11 const express = require('express');
12 const app = express();
13 const port = 3000;
14
15 const bodyParser = require('body-parser');
16 const bcrypt = require('bcrypt');
17 app.use(bodyParser.json());
18 const jwt = require('jsonwebtoken');
19 const cors = require('cors');
20 app.use(cors());
21
22 var mysql = require('mysql');
23
24 var con = mysql.createConnection({
25     host: "localhost",
26     user: "root",
27     password: "tonyonline*123",
28     database: 'jobSearch'
29 });
```

Listing 5.1: Konfiguration des Express-Servers

Der gezeigte Codeausschnitt ist ein entscheidender Teil der Implementierung des Backends der Anwendung. Er behandelt die Konfiguration des Express-Servers, die Verwendung des Moduls "body-parserßur Analyse von JSON-Anfragen, die Verwaltung der Passwortverschlüsselung mithilfe der Bibliothek "bcryptünd die Erstellung und Überprüfung von JSON Web Tokens (JWT), um die Authentifizierung der Benutzer zu gewährleisten.
Die Verbindung zur MySQL-Datenbank wurde mithilfe des Moduls "mysql"hergestellt, sodass Daten über Benutzer, Stellenangebote und andere für die Anwendung wichtige Informationen gespeichert und abgerufen werden können.
Darüber hinaus wurde mithilfe der "corsBibliothek Zugriffskontrollmechanismen implementiert, die Cross-Origin-Anfragen zulassen und eine sichere Kommunikation zwischen dem Frontend und dem Backend der Anwendung gewährleisten.
Durch die Implementierung dieser Funktionen ist die Anwendung in der Lage, Benutzerkonten einzurichten, die Informationen jedes Benutzers sicher zu speichern, Stellenangebote zu verwalten und Arbeitssuchenden die Möglichkeit zu geben, sich auf bestimmte Stellen zu bewerben.

```
11 con.query('CREATE DATABASE IF NOT EXISTS JobSearch;', (err, result) => {
12     if (err) throw err;
13     console.log("Database created");
14 
15     con.changeUser({database : 'JobSearch'}, function(err) {
16         if (err) throw err;
17 
18         con.query(`
19     CREATE TABLE IF NOT EXISTS Employer (
20         id INT AUTO_INCREMENT PRIMARY KEY,
21         namee VARCHAR(255),
22         adress VARCHAR(255),
23         email VARCHAR(255),
24         telephone VARCHAR(255)
25     )`, (err, result) => {
26             if (err) throw err;
27             console.log("Employer table created");
28 
```

Listing 5.2: Erstellung der Datenbank JobSearch

Der Code zeigt einen Teil unserer Datenbank-Implementierung. Zunächst wird die Datenbank 'JobSearch' erstellt, falls sie nicht bereits existiert, und dann wird diese Datenbank ausgewählt, um die weiteren Operationen durchzuführen. Dann wird die Tabelle 'Employer' erstellt, die die wichtigsten Informationen über die Arbeitgeber enthält, wie Name, Adresse, E-Mail-Adresse und Telefonnummer. Die Kennung 'id' wird als Primärschlüssel definiert und automatisch inkrementiert, wenn ein neuer Arbeitgeber hinzugefügt wird.
Wir verwenden das Modul 'mysql', um eine Verbindung mit der MySQL-Datenbank herzustellen, und verwenden dann die Methode 'query', um unsere SQL-Abfragen auszuführen. Wenn beim Erstellen der Datenbank oder der Tabelle ein Fehler auftritt, verwenden wir die Struktur 'if (err) throw err;', um einen Fehler zu melden und die Ausführung des Programms zu unterbrechen.
Dieser Teil des Codes bildet die Grundlage für die Verwaltung von Arbeitgebern

in unserer Anwendung. Mit diesen Datenbanktabellen können wir die wichtigsten Informationen über Arbeitgeber speichern und die Interaktionen mit ihnen verwalten, z. B. wenn sie Stellenangebote veröffentlichen oder eingegangene Bewerbungen einsehen.

```
11 con.query('USE JobSearch', (err, result) => {
12     if (err) throw err;
13 
14     con.query('ALTER TABLE Employer ADD COLUMN type_of_account VARCHAR(255);', (err, resu
15         if (err) throw err;
16         console.log("Column type_of_account added to Employer table");
17     });
18 
19     con.query('ALTER TABLE JobSeeker ADD COLUMN type_of_account VARCHAR(255);', (err, res
20         if (err) throw err;
21         console.log("Column type_of_account added to JobSeeker table");
22     });
23 });
```

Listing 5.3: Aktualisierung von der Struktur der Datenbank

Um den spezifischen Anforderungen unserer Webanwendung für die Stellensuche gerecht zu werden, haben wir uns vorgenommen, die Struktur unserer JobSearch-Datenbank zu aktualisieren. Ziel dieser Aktualisierung ist es, eine neue Spalte type-of-account in die Tabellen Employer und JobSeeker aufzunehmen.
Die Spalte type-of-account spielt eine wesentliche Rolle bei der Unterscheidung zwischen den beiden Arten von Nutzern unserer Anwendung: Arbeitgeber, die Stellenangebote anbieten, und Arbeitssuchende, die sich auf diese Stellenangebote bewerben. Durch das Hinzufügen dieser Spalte können wir nun den Kontotyp jedes Nutzers in der Datenbank speichern und so den Prozess der Verwaltung der verschiedenen Rollen in der Anwendung vereinfachen.
Der Code veranschaulicht den Prozess der Aktualisierung der Datenbankstruktur. Wir verwenden die SQL-Abfragen ALTER TABLE, um die neue Spalte typeofaccount zu den Tabellen Employer und JobSeeker hinzuzufügen, wobei wir den Datentyp VARCHAR angeben, um den Kontotyp zu speichern. Diese Aktualisierung stellt sicher, dass wir nun über eine effektive Möglichkeit verfügen, die Kontotypen der Benutzer zu verfolgen und zu verwalten.

```
11 app.post('/api/account-creating', async (req, res) => {
12     console.log("voici les données récuperées du front-end: ", JSON.stringify(req.body),
13     const user = req.body;
14 
15     // Hash the password before storing it in the database
16     const salt = await bcrypt.genSalt(10);
17     user.password = await bcrypt.hash(user.password, salt);
18     console.log(user.password);
19     console.log(user);
20 
21 
22     let tableName;
23     if(user.type_of_account === 'jobseeker') {
24         tableName = 'JobSeeker';
```

```
    } else if(user.type_of_account === 'employe') {
        tableName = 'Employer';
    } else {
        res.status(400).json({ error: 'Invalid account type' });
        return;
    }

    const query = `INSERT INTO ${tableName} SET ?`;
    con.query(query, user, (err, result) => {
        if (err) {
            console.error(err);
            res.status(500).json({ error: 'Error when creating an account' });
            return;
        }
        console.log('creation de compte ok');
        res.json({ success: true });
    });
});
```

Listing 5.4: Implementierung der Kontoerstellung

Der Code ist ein Teil des Backend-Servers, der mit Node.js und Express entwickelt wurde. Wenn ein Benutzer über die Frontend-Schnittstelle ein Formular zur Erstellung eines Kontos absendet, werden die Daten als HTTP POST-Anfrage an das Backend gesendet. Der oben stehende Code verarbeitet diese Anfrage, extrahiert die Daten aus der Anfrage und verwendet sie, um einen neuen Datensatz in der entsprechenden Tabelle zu erstellen, entweder JobSeeker oder Employer, je nachdem, welchen Kontotyp der Benutzer ausgewählt hat.
Wenn die Benutzerdaten vom Frontend empfangen werden, verwenden wir die bcrypt-Bibliothek, um das Passwort zu verschlüsseln, bevor es in der Datenbank gespeichert wird. Dadurch wird sichergestellt, dass die Passwörter der Nutzer im Falle einer Sicherheitsverletzung geschützt sind.
Der Code verwendet außerdem SQL-Abfragen, um die Benutzerinformationen in die entsprechende Tabelle, "JobSeekeröder Ëmployer", einzufügen, je nach angegebenem Kontotyp.
Im Erfolgsfall wird eine JSON-Antwort mit dem Schlüssel ßuccessän das Frontend zurückgegeben, die angibt, dass das Konto erfolgreich erstellt wurde.
Diese wichtige Funktion der Kontoerstellung stellt sicher, dass die Nutzer sich anmelden und auf ihr personalisiertes Konto zugreifen können, um die Funktionen der Anwendung voll auszuschöpfen.

```
app.post('/api/authentification', (req, res) => {
    console.log("voici les credentials recus du front-end: "+JSON.stringify(req.body));
    const { email, password } = req.body;

    // Check JobSeeker table
    con.query('SELECT * FROM JobSeeker WHERE email = ?', [email], async (error, jobSeekerRe
        if (error) {
            return res.status(500).json({ error: 'Database error' });
        }

```

```
        // If there is a JobSeeker with this email
        if (jobSeekerResults.length > 0) {
            const user = jobSeekerResults[0];
            const isMatch = await bcrypt.compare(password, user.password);

            if (!isMatch) {
                return res.status(400).json({ error: 'Incorrect email address or password
            }

            const token = jwt.sign({ id: user.id, type: 'JobSeeker' }, 'secretKey', { exp

            //return res.json({ token });
            return res.json({ token, userType: user.type_of_account });
        }
```

Listing 5.5: Anmeldung

Wenn ein Benutzer seine Anmeldeinformationen über die Frontend-Schnittstelle übermittelt, werden die Daten als HTTP POST-Anfrage an das Backend gesendet. Der oben stehende Code verarbeitet diese Anfrage, überprüft die Anmeldeinformationen des Benutzers in den Tabellen "JobSeekeründ Ëmployer"der Datenbank und generiert ein sicheres Zugangstoken, wenn die Informationen gültig sind.
Dieser Backend-Code führt eine Überprüfung des Benutzers durch, indem er in den Tabellen "JobSeekeründ Ëmployer"der Datenbank nach der angegebenen E-Mail-Adresse sucht. Wenn die Anmeldeinformationen des Benutzers gültig sind, wird mithilfe der Bibliothek jsonwebtoken ein sicheres Zugriffstoken generiert. Dieses Token wird dann in einer JSON-Antwort zusammen mit dem Kontotyp des Benutzers an das Frontend zurückgeschickt.
Die sichere Authentifizierung stellt sicher, dass nur autorisierte Benutzer auf die geschützten Funktionen der Anwendung zugreifen können, wie z. B. die Verwaltung von Stellenangeboten oder die Anzeige von Bewerbungen.

5.1.3.2 Front-End

5.2 Ergebnissbewertung

6 Zusammenfassung und Ausblick

Literatur

[1] *Angular*. URL: https://en.wikipedia.org/wiki/Angular_(web_framework) (siehe S. 14).

[2] David Curry. *TypeScript Fastest Growing Programming Language, JavaScript Most Popular*. Feb. 2023. URL: https://www.clouddatainsights.com/typescript-fastest-growing-programming-language-javascript-most-popular/ (siehe S. 17).

[3] *Git*. URL: https://www.20i.com/blog/beginners-guide-git-all-you-need-to-know/ (siehe S. 15).

[4] *How to choose a Web Hosting Provider*. Apr. 2019. URL: https://www.exposure.com/blog/how-to-choose-a-web-hosting-provider/ (siehe S. 22).

[5] *Html*. URL: https://www.ionos.de/digitalguide/websites/web-entwicklung/html-tags/ (siehe S. 15).

[6] *JavaScript*. URL: =%20https://www.pngwing.com/en/free-png-ngqmz (siehe S. 17).

[7] *JavaScript History*. URL: https://www.w3schools.com/js/js_history.asp (siehe S. 16).

[8] David Farle Jez Humble. *Continuous Delivery: Reliable Software Releases through Build, Test and Deployment Automation*. 1. Addison-Wesley Professional, Juli 2010. URL: https://continuousdelivery.com/ (siehe S. 13).

[9] Javier Calvarro Nelson Jos van der Til Martin Costello. *Integration tests in ASP.NET Core*. März 2023. URL: https://learn.microsoft.com/en-us/aspnet/core/test/integration-tests?view=aspnetcore-7.0 (siehe S. 21).

[10] Oliver Moradov. *Unit testing: definition, Examples, and Critical Best Practices*. Mai 2022. URL: https://brightsec.com/blog/unit-testing/ (siehe S. 21).

[11] Gaurav Mukherjee. *Angular 16 is huge*. Apr. 2023. URL: https://itnext.io/angular-16-is-huge-67288a3ff58b (siehe S. 14).

[12] Adekola Olawale. *What is Angular? Understanding Angular's Modular structure*. Aug. 2023. URL: https://www.freecodecamp.org/news/front-end-javascript-development-react-angular-vue-compared/ (siehe S. 14).

[13] OWASP. *OWASP Top 10 Web Application security Risks*. URL: https://owasp.org/www-project-top-ten/ (siehe S. 13).

[14] Priya Pedamkar. *Versions of Html, 6.html5*. Juli 2023. URL: https://www.educba.com/versions-of-html/ (siehe S. 15).

[15] Rinki. *Hyper Text Markup Language*. Mai 2023. URL: https://www.c-sharpcorner.com/article/the-importance-of-html-in-web-development/ (siehe S. 15).

[16] Robert Sheldon. *The history of Git*. Feb. 2023. URL: https://www.techtarget.com/searchitoperations/definition/Git (siehe S. 14).

[17] Joe Silk. *THE BASICS OF BACKEND DEVELOPMENT, APIS*. Juli 2022. URL: https://www.startechup.com/blog/back-end-development/ (siehe S. 20).

[18] Vinayak Tekade. *Getting Started with Backend Development*. März 2021. URL: https://medium.com/coders-capsule/getting-started-with-backend-development-8ce55585e860 (siehe S. 21).

[19] *TypeScript*. URL: https://en.wikipedia.org/wiki/TypeScript (siehe S. 16).

[20] *TypeScript*. URL: https://javascript.plainenglish.io/how-to-start-a-blank-typescript-project-1d260f7e2aa8 (siehe S. 16).

[21] *User Experience, User Interface*. URL: https://www.finalsite.com/blog/p/~board/b/post/what-is-user-experience (siehe S. 20).

[22] *What is an SSL Certificate?* URL: https://www.digicert.com/what-is-an-ssl-certificate (siehe S. 22).

[23] Wikipedia. *CSS*. URL: https://en.wikipedia.org/wiki/CSS (siehe S. 17).