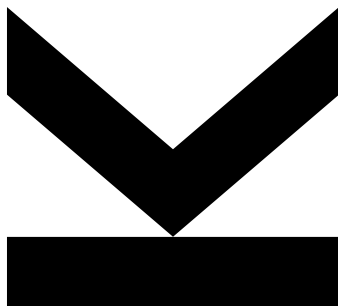


LERNVERTRAG – URL-SHORTENER



Progressive Web Apps

Gruppe 9

Projektleiter:

K11808821, Thomas Deutsch, deutsch-thomas@gmx.at

Teammitglieder:

K11818868, Stefan Bolda

K11818867, Stefan Hinterhölzl

Inhaltsverzeichnis

1. Einleitung und Zielsetzung.....	3
2. Frameworks.....	3
2.1. AngularJS.....	3
2.2. TypeScript.....	4
2.3. Materialize.....	5
2.4. Bootstrap.....	6
3. Wahl der Entwicklungsumgebung.....	7
3.1. Visual Studio	7
3.2. Eclipse.....	7
3.3. Visual Code.....	8
4. Wahl der API	8
4.1. SOAP-API	8
4.2. Chatter REST-API	9
4.3. Bulk-API	9
4.4. REST-API.....	9
5. Informationsinfrastruktur	10
5.1. Der Server.....	10
5.2. Anbindung an das World Wide Web	10
5.3. DDNS – Dynamische DNS Adresse	10
5.4. Port Forwarding – Routing im lokalen Netzwerk	11
5.5. Domain von OHV – www.shortly.at.....	11
5.6. Hosting der API – Nodemon.....	12
5.7. Node.js.....	12
5.8. Hosting der Angular Applikation.....	12
5.8.1. Firebase	12
5.8.2. IIS Hosting.....	12
5.8.3. Browsersync Hosting.....	12
6. Abbildungsverzeichnis	13
7. Literaturverzeichnis.....	14

1. Einleitung und Zielsetzung

Ziel ist es, durch Anwendung der praktischen und theoretischen Aspekte eine Website zu erstellen, welche eine URL verkürzt und bei Eingabe dieser im Browser den Benutzer auf die gewünschte Seite automatisch weiterleitet.

URLs (Uniform Resource Locators) sind heutzutage unvermeidlich lang, da darin beispielsweise notwendige Information für Search Querys gespeichert ist.

Beispiel:

https://www.google.com/search?q=javascript&rlz=1C1CHBF_enAT786AT786&oq=javascript+&gs=chrome..69i57j0l6j69i60.1597j0j7&sourceid=chrome&ie=UTF-8

An dieser Beispiel Google-Suche kann man erkennen, dass in dieser URL eingegebene Suchparameter (Wörter, Zeichen, Browserinfo etc.) eingebettet sind. Um solche unübersichtlichen URLs zu verkürzen, werden heutzutage Webanwendungen wie Bit.ly genutzt. Diese verkürzten Links helfen Nachrichten oder Beschreibungen kompakter zu gestalten. JavaScript Libraries bieten heutzutage schon Funktionen beziehungsweise Algorithmen, um dies zu implementieren.

Des Weiteren bieten manche URL-Shortener an, dass man zwischen dem Öffnen des Links und der Ziel-Website kurze Werbung geschaltet wird, um für den Nutzer der Anwendung Profit zu generieren. Ein weiteres Beispiel ist die Verlinkungen von beworbenen Produkten bei Youtube-Videos.

Unsere Webanwendung soll auf einem von uns selbst bereitgestellten dedizierten Server zur Verfügung gestellt werden. Die verkürzten Links werden in einer Datenbank gespeichert. Mögliche weitere optionale Funktionen sind das Anlegen von Nutzerkonten sowie die Schaltung von Werbeanzeigen. Bekannte Beispiele für solche Anwendungen sind zum Beispiel: Bitly.com, Shorturl.at, tiny.cc etc. (*Why Are URLs Full of So Many Garbage Characters?*, o. J.).

2. Frameworks

2.1. AngularJS



Abb. 1 Logo Angular

Für die Ausarbeitung unseres Projekts wird als Programmiersprache überwiegend JavaScript verwendet. Zudem bedienen wir uns an den Funktionalitäten des open-source front-end Web-Framework AngularJS.

Dieses Framework ist speziell für dynamische Webanwendungen entwickelt worden.

Es eignet sich perfekt für sogenannte Single Page Applications (SPA) Projekte. Diese SPA's sind überwiegend Web-Applications die bei Interaktion mit dem Nutzer die aktuelle Seite dynamisch zu behandeln, anstatt eine komplett neue Webpage zu laden.

Als Front-End bezeichnet man Alles, das man visuell darstellt. In Verbindung mit HTML kann man bei AngularJS mit wenigen LoC (Lines of Code) einen großen sichtbaren Fortschritt feststellen.

Des Weiteren ist AngularJS eine Model-View-Controller Architektur, die leichten Einstieg bietet, um eine simple Webapplikation von Grunde auf zu erstellen (*AngularJS Tutorial—Tutorialspoint*, o. J.).



2.2. TypeScript

Abb. 2 Logo TypeScript

TypeScript ist ein sogenanntes Superset entwickelt von Microsoft für JavaScript. Da AngularJS ein Front-End Framework ist, nutzt es TypeScript für die Logik beziehungsweise dem Back-End. Jeder valide JavaScript Code ist auch valider TypeScript-Code, jedoch kommen bei TypeScript viele zusätzliche Funktionen hinzu.

JavaScript ist zwar eine flexible Programmiersprache, jedoch nicht objektorientiert. Diese Funktion wird durch TypeScript erweitert. Es können Klassen und Objekte erstellt werden (Eygi, 2019).

Ein Beispiel-Codefragment:

```
class Student {  
    private firstName: String;  
    private lastName: String;  
  
    constructor(firstName?: string, lastName?: string) {  
        this.firstName = firstName;  
        this.lastName = lastName;  
    }  
  
    getGrades() {  
        // some code  
    }  
}
```



2.3. Materialize

Abb. 3 Logo Materialize

Um eine moderne Oberfläche beziehungsweise ein modernes Userinterface zu bieten, entschieden wir uns für Materialize. Hierbei handelt es sich um ein responsive CSS (Cascading Style Sheet) Framework basierend auf Material Design von Google.

Um Materialize nutzen zu können, müssen hier die richtigen Dependencies gesetzt werden:

```
<!-- Compiled and minified CSS -->
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.cs
s">

<!-- Compiled and minified JavaScript -->
<script
src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js">
</script>
```

Anschließend müssen die Source-Files installiert werden. Hier muss lediglich im Terminal der genutzten IDE (Visual Code) folgender Command eingegeben werden:

```
npm install materialize-css@next
```

Nun lassen sich eine Vielzahl von Grafiken und Styles von Google verwenden, um einen modernen Look zu erreichen. Beispielsweise werden Eingabefelder in den Google-Stil konvertiert (*Getting Started—Materialize*, o. J.).

Beispiel:

First Name
Placeholder

Last Name

Disabled
I am not editable

Password

Email

Helper text

Abb. 4 Exemplarisches Beispiel des Materialize-Designs



2.4. Bootstrap

Abb. 5 Logo Bootstrap

Bei Bootstrap handelt es sich ebenfalls um ein responsive Front-end Framework zum Designen von modernen Websites und Applikationen.

Es werden verschiedenste CSS templates für User-Interfaces wie zum Beispiel Buttons und Eingabeformulare. Wir entschieden uns Bootstrap in Verbindung mit Materialize zu verwenden, um sich mit beidem vertraut zu machen.

Ein Vorteil von Bootstrap ist, dass sich die verschiedenen Elemente an unterschiedlichste Endgeräte wie Tablets, Desktops oder Smartphones anpassen. Beispielsweise lassen sich Navigationsbars sehr leicht durch Kopieren der Codesegmente in die Webapplikation einbauen („Bootstrap (front-end framework)“, 2020).

Example

TitleHomeLinkLink

```
1. <div class="navbar">
2.   <div class="navbar-inner">
3.     <a class="brand" href="#">Title</a>
4.     <ul class="nav">
5.       <li class="active"><a href="#">Home</a></li>
6.       <li><a href="#">Link</a></li>
7.       <li><a href="#">Link</a></li>
8.     </ul>
9.   </div>
10. </div>
```

Abb. 6 Beispiel Bootstrap

3. Wahl der Entwicklungsumgebung

Ein wichtiger Einflussfaktor für den Erfolg des Projekts ist die richtige Wahl der Entwicklungsumgebung. Um eine faktengestützte Wahl treffen zu können, werden nun 3 Entwicklungsumgebungen vorgestellt und verglichen

3.1. Visual Studio

Visual Studio ist eine von Microsoft entwickelte Entwicklungsumgebung. Den Fokus setzte Microsoft hierbei auf die Unterstützung zur Entwicklung von Desktop-Applikationen, Web-Services sowie auf Webdevelopment. Plattformen wie Windows API, Windows Presentation Foundation, Windows Forms, Windows Store und Microsoft Silverlight und viele viele weitere Plugins und Frameworks sind verfügbar welche alle für unser Projekt in Frage kommen würden, um den Entwicklungsprozess zusätzlich zu unterstützen. Visual Studio bietet einen „source-level Debugger“ als auch einen „machine-level Debugger“ was das Fehler finden und beheben deutlich beschleunigt. Weiters unterstützt diese IDE annähernd alle verbreiteten Programmiersprachen wie: C, C++, C++/CLI, VB.NET, C# und F# von natur aus, kann aber durch language service installationen um annähernd jede erweitert werden wie z.B. Python, Ruby, HTML/XHTML, XML/XSLT, JavaScript, TypeScript und CSS um nur ein Paar zu nennen (*Visual Studio 2019 | Kostenloser Download, o. J.*).

3.2. Eclipse

Eclipse bietet eine Entwicklungsumgebung, welche auf den gesamten Life-Cycle eines Produktes ausgelegt ist. Die größte Verbreitung findet Eclipse für Programmiersprachen wie Java, C/C++, Javascript und PHP. Doch ähnlich wie Visual Studio ist auch Eclipse über den Marketplace stark erweiterbar. Grundsätzlich besteht Eclipse aus über 60 Open-source Projekten, welche sich in 6 Hauptkategorien entwickelt haben:

- *. Enterprise Development
- *. Embedded and Device Development
- *. Rich Client Platform
- *. Rich Internet Applications
- *. Application Frameworks
- *. Application Lifecycle Management (ALM)
- *. Service Oriented Architecture (SOA)

Da die Entwicklungsumgebung Open-source ist, ist auch das verwenden dieser Kostenlos. Durch unsere Ausbildung an der JKU in dieser Entwicklungsumgebung und der Möglichkeit darin Web-Applications zu entwickeln ist Eclipse deshalb eine mögliche Wahl für die Durchführung unseres Projektes und eine unserer Favoriten (Foundation, o. J.).



Abb. 7 Logo Visual Code

3.3. Visual Code

Als Entwicklungsumgebungen entschieden wir uns für Visual Code. Dieser Source-Code-Editor von Microsoft ist Open-Source und kostenlos. Des Weiteren bietet er sehr leichte Handhabung mit unserem Git-Repository auf GitHub.

Visual Code ist heutiger Standard in der JavaScript-Programmierung, da er im Vergleich zu Microsoft Visual Studio sehr leichtgewichtig ist. Durch nutzen des Terminals lassen sich leicht Frameworks und Bibliotheken installieren.

(„What is Visual Studio Code and its advantages Webner Blogs—ELearning, Salesforce, Web Development & More“, 2018).

4. Wahl der API

Eine API (“Application-Programming-Interface”) wird genutzt um einfach und schnell zwischen Server, Datenbank und Website/Applikation zu kommunizieren. Aufgabe der API ist es die benötigten Daten zur Verfügung zu stellen, ohne dabei unnötigen Datentransfer zu erzeugen. Eine API ist grundsätzlich sehr stark individualisierbar und an jedes Projekt adaptierbar. So kann zum Beispiel eine spezielle Anfrage nötig sein bevor hochgradig sensible Daten übertragen werden, und dies auf Wunsch nur verschlüsselt (*Which API Do I Use?*, o. J.).

Jede nutzende Applikation muss durch den Einsatz einer API nur noch einen Befehl senden anstelle alle Funktionen und Logischen-Abläufe selbst durchführen zu müssen um einen Dateneintrag aus einer Datenbank zu erhalten. Viele APIs überlassen auch die Wahl welche Kommunikationssprache verwendet wird. Da uns persönlich asynchrone Kommunikation in einer State-of-the-art Single-Page-Application wichtig ist, ist die Einsetzbarkeit von JSON eine Grundvoraussetzung für uns (*Which API Do I Use?*, o. J.).

Weiters bietet der Einsatz einer API einen erheblichen Schutz, so werden zum Beispiel nie Login-Daten von der API an den User gesendet, sondern nur die Antwort ob die Eingaben richtig waren. Auch die Datenbank ist nicht angreifbar, da die API den Zugang von ihren Methoden streng ausschließen kann und somit zum Beispiel nur die IP-Adresse des Webserver Anfragen senden darf. Somit können sensible Daten wie Passwörter oder Kreditkarteninformationen sicher verwaltet und gespeichert werden (*Which API Do I Use?*, o. J.).

4.1. SOAP-API

SOAP bietet die Möglichkeit, Datensätze zu erstellen, abzurufen, zu aktualisieren oder zu löschen. SOAP API kann auch Suchvorgänge ausführen oder in ein ERP- und Finanzsystem integriert werden. Jede Sprache, die der Webdienst unterstützt, kann verwendet werden. Das Datenformat lautet XML, das Protokoll SOAP (WSDL) und die Kommunikation ist synchron. Durch die Persönlichen Kompetenzen und die erlernten Werkzeuge durch andere Kurse werden wir jedoch keine SOAP API wählen (*Which API Do I Use?*, o. J.).

4.2. Chatter REST-API

Chatter REST API eignet sich gut für mobile Anwendungen. Es bietet Chatter-Feeds, Gruppen, Benutzer, Themen, Empfehlungen und Benachrichtigungen. Das Protokoll basiert auf REST und das Datenformat ist entweder JSON oder XML. Fotos werden in diesem Fall asynchron verarbeitet, wobei alle anderen Kommunikationen synchron verarbeitet werden. Würden wir eine eigenständige Mobile Applikation in Kombination zu unserer Website vorsehen, wäre dies eine sehr gute Wahl, doch die extra Funktionalität würde von uns nicht ausgeschöpft werden (*Which API Do I Use?*, o. J.).

4.3. Bulk-API

Diese API eignet sich für die Verarbeitung von großen Datensätzen, auch als „Batch“ bezeichnet. Funktionen wie query, queryAll, insert, update, upsert oder delete können verwendet werden. Die Datensätze werden im Hintergrund verarbeitet. Bulk API wird dann verwendet, wenn einige Tausend bis Millionen Datensätze verarbeitet werden. Andernfalls wird in der Wirtschaft häufig auf die SOAP-API oder REST-API zurückgegriffen, da diese für die Echtzeitanwendung optimiert sind. Das Protokoll ist REST, die verarbeiteten Datenformate sind CSV, JSON und XML und die Kommunikation ist asynchron. Da diese API für sehr viele Anfragen ausgelegt ist was wir noch nicht benötigen werden wir auch diese API nicht wählen da die REST API schneller antwortet und wir nicht Millionen von Anfragen haben wie es Börsen oder ähnliches haben (*Which API Do I Use?*, o. J.).

4.4. REST-API

Der REST API ist eine einfache Integrations- und Entwicklungs-API, welches sich hervorragend für Webprojekte eignet, da es schnell aufgesetzt und funktionsfähig ist. Es ist jedoch nicht die beste Wahl, wenn viele Datensätze verarbeitet werden müssen. Dazu eignet sich die Bulk API besser, welches ein ähnliches Grundprinzip wie REST hat, jedoch für große Datenmengen optimiert ist. Das Datenformat von Rest ist entweder JSON oder XML. Die Kommunikation ist bei XML synchron, durch den Einsatz von JSON jedoch asynchron möglich, was uns für eine gute Userexperience wichtig ist. Da die Persönlichen Kompetenzen bei dieser API am stärksten eingesetzt und gefestigt werden können als auch der technischen und projektbezogenen Eignung, ist diese API unser Favorit und wird die Grundlage unserer Kommunikation werden (*Which API Do I Use?*, o. J.).

5. Informationsinfrastruktur

5.1. Der Server

Der Shortly Webservice wird auf einem bestehenden Server von Herrn Hinterhölzl, der zuhause im Heimnetzwerk integriert wurde, gehostet. Als Betriebssystem wurde auf spezielle Server OS verzichtet, da es für unsere Anwendung nicht zwingend notwendig ist. Der PC läuft stattdessen auf normalen Windows 10 Pro. Der Server verfügt über einen Intel Core i5 der dritten Generation und über 16 GB DDR3 Ram. Um eine schnelle Übertragung zu gewährleisten ist der Server über Cat 6e Lan Kabel direkt mit dem Router verbunden.

5.2. Anbindung an das World Wide Web

Da der Server in einem gewöhnlichen Home Internet läuft ist eine statische IPV4 Adresse keine Option. Der Internetanbieter, in diesem Fall T-Mobile, bietet dies auch nicht gegen Bezahlung. Als Lösung wird für diesen Server Dynamisches DNS verwendet. Dieser Service ist gratis von Anbietern wie No-Ip erhältlich

5.3. DDNS – Dynamische DNS Adresse

Dynamisches DNS ist eine Technik, um Domains automatisch zu aktualisieren. Hintergrundgedanke ist für einen PC oder, in unserem Fall Router, nach dem Wechsel seiner public IP-Adresse automatisch und schnell den dazugehörigen Domaineintrag zu ändern. Durch diese Vorgehensweise ist der Router immer unter der gleichen Domain erreichbar, obwohl sich die IP-Adresse des Gerätes im Laufe der Zeit immer ändert. Die Aktualisierung erfolgt über http oder https („Dynamisches DNS“, 2018).

In unserem Fall wurde die DDNS shortly.ddns.net erstellt. Diese wurde im Router hinterlegt. Dadurch verweist die Domain auf den Router im Heimnetzwerk.

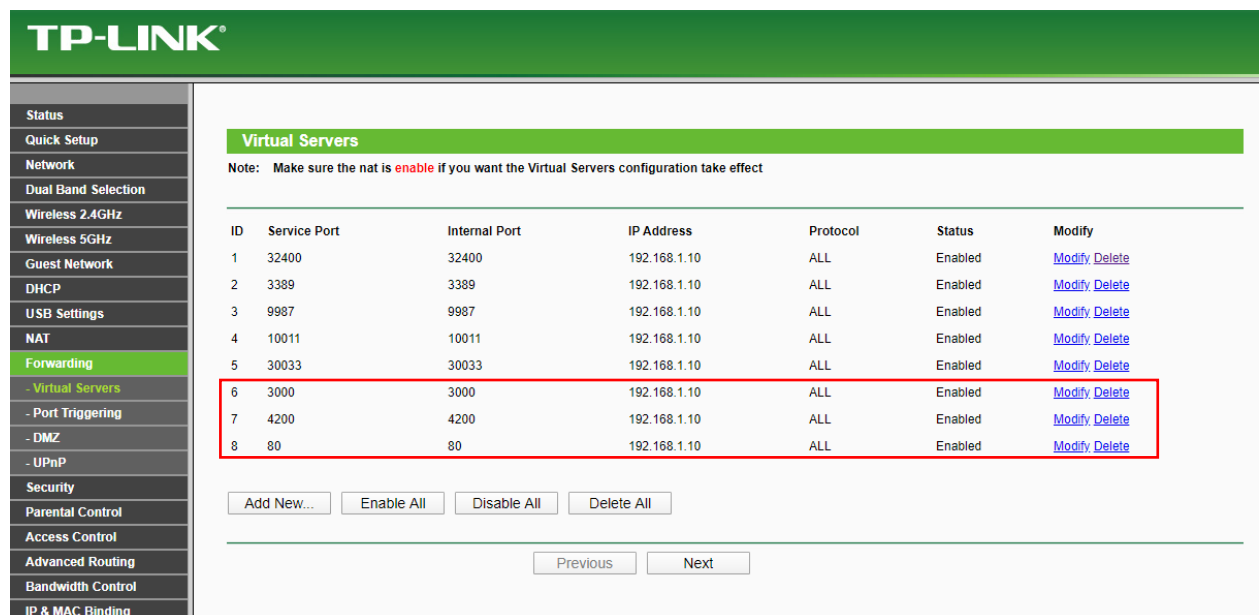
The screenshot shows the TP-LINK router's web interface. On the left is a sidebar menu with various settings like Status, Quick Setup, Network, Dual Band Selection, Wireless 2.4GHz, Wireless 5GHz, Guest Network, DHCP, USB Settings, NAT, Forwarding, Security, Parental Control, Access Control, Advanced Routing, Bandwidth Control, IP & MAC Binding, Dynamic DNS (highlighted), and IPv6 Support. The main content area is titled 'DDNS' and contains the following configuration details:

- Service Provider:** No-IP (www.no-ip.com) with a link to 'Go to register...'
- User Name:** stefan_hinter
- Password:** masked with asterisks
- Domain Name:** shortly.ddns.net
- Enable DDNS:** checked
- Connection Status:** Succeeded!
- Buttons:** Login, Logout, and a large Save button at the bottom.

Abb. 8 Screenshot aus Router-Konfiguration: Dynamic DNS

5.4. Port Forwarding – Routing im lokalen Netzwerk

Um sicherzustellen, dass einerseits die REST Aufrufe an die API aber auch der Aufruf der angular Applikation, den Server erreichen, muss ein internes Routing und Port Forwarding eingerichtet werden. Dazu muss für den Server im Netzwerk eine fixe IP Adresse vergeben werden. In unserem Fall loggt sich der Server automatisch mit der Adresse 192.168.1.10 im Netzwerk ein. Es ist unbedingt notwendig die lokale IP nicht durch einen DHCP Server zu vergeben, da so kein Port Forwarding möglich wäre. Die Ports für unseren Service sind 80 für die angular Applikation, was im allgemein der Standardport für Web-Services ist und 3000 für die REST-API. Diese zwei Ports werden in der Routerkonfiguration an den Server weitergeleitet. Wir leiten TCP und UDP Anfragen mit diesem Port weiter, wobei zur Kommunikation TCP verwendet wird.



TP-LINK

Virtual Servers

Note: Make sure the nat is **enable** if you want the Virtual Servers configuration take effect

ID	Service Port	Internal Port	IP Address	Protocol	Status	Modify
1	32400	32400	192.168.1.10	ALL	Enabled	Modify Delete
2	3389	3389	192.168.1.10	ALL	Enabled	Modify Delete
3	9987	9987	192.168.1.10	ALL	Enabled	Modify Delete
4	10011	10011	192.168.1.10	ALL	Enabled	Modify Delete
5	30033	30033	192.168.1.10	ALL	Enabled	Modify Delete
6	3000	3000	192.168.1.10	ALL	Enabled	Modify Delete
7	4200	4200	192.168.1.10	ALL	Enabled	Modify Delete
8	80	80	192.168.1.10	ALL	Enabled	Modify Delete

Abb. 9 Screenshot aus Router-Konfiguration: Port Forwarding

5.5. Domain von OHV – www.shortly.at

Beim Anbieter OHV wurde von uns die Domain www.shortly.at erworben. In der OHV Cloud hat man die Möglichkeit eine Redirecting der Domain auf eine beliebige Adresse oder IP Adresse einzustellen. Wir haben damit die Domain auf die DDNS <http://shortly.ddns.net/80> weitergeleitet. Bei dem Redirecting hat man die Wahl einer sichtbaren und unsichtbaren Weiterleitung. Bei der sichtbaren Weiterleitung wird die Adresse sichtbar in der URL Leiste ersetzt. Bei der unsichtbaren ist dies nicht der Fall. Jedoch muss für eine unsichtbare Weiterleitung die Webpage die Anzeige durch einen IFrame unterstützen.

5.6. Hosting der API – Nodemon

Die API wurde in Javascript mit Node.js programmiert. Node.js ist eine sehr mächtige Plattform für Javascript, welches durch die npm (Node Package Manager) Befehle ein sehr leichtes erweitern der Module in Node.js ermöglicht. Das Hosting des Servers wird durch Nodemon zur Verfügung gestellt. Dieses Modul kann einfach durch den Befehl `npm install -g nodemon` installiert werden. Durch den Befehl `npm start` wird die node.js Applikation zur Verfügung gestellt. Darüber hinaus wird der Server bei Änderungen im Code automatisch neu gestartet (Nodemon, o. J.).

5.7. Node.js

Node.js ist eine serverseitige Plattform, welche für die Entwicklung von Netzwerkanwendungen verwendet wird. Insbesondere Webserver lassen sich sehr gut in Node.js realisieren. Node.js bietet die Möglichkeit JavaScript Code außerhalb eines Web Browser auszuführen. Damit lassen sich Server-seitig Inhalte für eine dynamische Webpage erstellen, welche dann zum User gesendet werden. Aus diesem Grund repräsentiert Node.js ein „JavaScript everywhere“ Paradigma. Node.js ist eines der wenigen Systeme, welche das Entwickeln von Frontend und Backend in einer Sprache ermöglicht. Während Standardmodule in das Binary von Node.js inkludiert sind, kann das System durch den Node Package Manager (npm) über die Konsole um verschiedenste Module erweitert werden. Node.js wurde ursprünglich von Ryan Dahl im Jahr 2009 geschrieben. Node.js wird von namenhaften Firmen wie Microsoft, IBM, Netflix und Paypal benutzt („Node.js“, 2020).

5.8. Hosting der Angular Applikation

5.8.1. Firebase

Da Angular von Google entwickelt wird, bietet sich der Google Hosting Service Firebase sehr gut an. Dieser Service kommuniziert sehr gut mit Angular, die Website kann direkt über das Terminal deployed werden. Firebase ist bei niedrigerem Traffic kostenfrei, erst bei verstärktem Zugriff auf die Webpage wird der Dienst kostenpflichtig.

5.8.2. IIS Hosting

In Windows eingebettet findet man die Information Internet Services, welche ebenfalls die Möglichkeit bieten einen Webdienst zu hosten.

5.8.3. Browsersync Hosting

Durch das Terminal und der npm Funktion kann das Framework Browsersync installiert werden. Dieses Modul bietet neben der Hosting Funktion auch andere Features. Unter anderem die Möglichkeit offenen Fenster der Webpage zu synchronisieren. Somit kann man das Verhalten der Website gleichzeitig auf mehreren Geräten überprüfen werden und muss die Website nur auf einem Gerät benutzen. Diese Funktionalität erleichtert das Styling der Website erheblich.

6. Abbildungsverzeichnis

Abb. 1 Logo Angular	3
Abb. 2 Logo TypeScript	4
Abb. 3 Logo Materialize	5
Abb. 4 Exemplarisches Beispiel des Materialize-Designs	5
Abb. 5 Logo Bootstrap	6
Abb. 6 Beispiel Bootstrap.....	6
Abb. 7 Logo Visual Code	8
Abb. 8 Screenshot aus Router-Konfiguration: Dynamic DNS	10
Abb. 9 Screenshot aus Router-Konfiguration: Port Forwarding	11

7. Literaturverzeichnis

AngularJS Tutorial—Tutorialspoint. (o. J.). Abgerufen 19. April 2020, von

<https://www.tutorialspoint.com/angularjs/index.htm>

Bootstrap (front-end framework). (2020). In *Wikipedia*.

[https://en.wikipedia.org/w/index.php?title=Bootstrap_\(front-end_framework\)&oldid=951512080](https://en.wikipedia.org/w/index.php?title=Bootstrap_(front-end_framework)&oldid=951512080)

Dynamisches DNS. (2018). In *Wikipedia*.

https://de.wikipedia.org/w/index.php?title=Dynamisches_DNS&oldid=175715071

Eygi, C. (2019, September 17). *Understanding TypeScript Basics*. Medium.

<https://codeburst.io/understanding-typescript-basics-e003dbad2191>

Foundation, E. (o. J.). *Eclipse IDE 2020-03 | The Eclipse Foundation*. Abgerufen 19. April 2020, von

<https://www.eclipse.org/eclipseide/>

Getting Started—Materialize. (o. J.). Abgerufen 19. April 2020, von [https://materializecss.com/getting-](https://materializecss.com/getting-started.html)

[started.html](https://materializecss.com/getting-started.html)

Node.js. (2020). In *Wikipedia*. <https://en.wikipedia.org/w/index.php?title=Node.js&oldid=951937299>

Nodemon. (o. J.). Abgerufen 19. April 2020, von <https://nodemon.io/>

Nov 20/05, M. M. (o. J.). *About the Eclipse Foundation | The Eclipse Foundation*. Abgerufen 19. April

2020, von <https://www.eclipse.org/org/>

Visual Studio 2019 | Kostenloser Download. (o. J.). Visual Studio. Abgerufen 19. April 2020, von

<https://visualstudio.microsoft.com/de/vs/>

What is Visual Studio Code and its advantages Webner Blogs—ELearning, Salesforce, Web

Development & More. (2018, Juni 20). *Webner Blogs - ELearning, Salesforce, Web*

Development & More. <https://blog.webnersolutions.com/visual-studio-code/>

Which API Do I Use? (o. J.). Abgerufen 19. April 2020, von

https://help.salesforce.com/articleView?id=integrate_what_is_api.htm&type=5

Why Are URLs Full of So Many Garbage Characters? (o. J.). Gizmodo. Abgerufen 19. April 2020, von

<https://gizmodo.com/why-are-urls-full-of-garbage-characters-1719538363>