

Netzwerkarchitekturen & Web Technologien (NAWT)

Projekt ImmoMobile



Dokumentation & Realisierung von C. Bieri, T. R. Jung,
S. Renggli, P. Weber

HSLU Departement Informatik

Version 1.0

Dozent: Prof. Dr. Martin Zimmermann

Inhaltsverzeichnis

1. Ausgangslage.....	1
2. Präzisierung Themenstellung.....	1
2.1 Suchprofil	1
2.2 Gemeindeinformationen.....	1
2.3 Immobilien	2
2.4 Anmerkung.....	2
3. Use-Cases	3
3.1 Use-Case-Diagramm.....	3
3.2 Use-Case-Beschreibungen – das Vorgehen.....	4
3.2.1 Use-Case-Beschreibungen – die Beschreibungen	4
4. Einrichtung Webserver.....	7
5. Herausforderungen.....	8
5.1 Use-Case-Diagramm.....	8
5.2 Use-Case-Beschreibungen.....	8
5.3 Local Storage	9
5.4 Suchprofil zurücksetzen	10
5.5 Übergabe Filterkriterien, Auswertung in PHP	10
5.6 Google Maps per PHP	10
5.7 Slideshow	11
6. Schlussfolgerung	12
7. Abbildungsverzeichnis.....	13

1. Ausgangslage

Die ImmoMobile AG ist ein Immobilienhändler mit Sitz in Meggen, Kanton Luzern. Schon seit der Firmengründung im Jahre 1878 hat sich die ImmoMobile AG auf exklusive premium Immobilien im Kanton Luzern und Schwyz spezialisiert. Dabei ist die ImmoMobile AG zu einer schweizweit bekannten Grösse im Immobilienmarkt geworden.

Um den ständig wachsenden Anforderungen der Digitalisierung gewachsen zu sein, hat sich die Geschäftsleitung dazu entschlossen, eine mobile Applikation für eine komfortablere, Zeitgerechte und jederzeit verfügbare Immobiliensuche zu entwickeln.

2. Präzisierung Themenstellung

Die mobile Applikation soll es dem Benutzer ermöglichen, käufliche Immobilien aufzuspüren. Dabei sollen interessierten Kunden folgende Möglichkeiten zur Verfügung gestellt werden:

- Bearbeitung / Erfassung / Wiederherstellung Suchprofil
- Immobiliensuche basierend auf erfasstem Suchprofil
- Abruf von Gemeindeinformationen
- Kontaktaufnahme per Mail mit unserem kompetenten Team

2.1 Suchprofil

Im Suchprofil können potenzielle Kunden angeben, für welche Objekte sie sich interessieren. Basierend auf der Suche werden die vorhandenen Immobilien ausgewertet und die gewünschten Angebote angezeigt. Im Suchprofil können dabei folgende Angaben gemacht werden:

- Angabe Gemeinde oder Region
- Angabe Objekttyp (Haus oder Wohnung)
- Angabe Mindestwohnfläche
- Angabe Maximalpreis
- Angabe Zimmeranzahl

2.2 Gemeindeinformationen

Damit sich die Kunden einen Überblick von der Umgebung machen können, besteht die Möglichkeit, Gemeindeinformationen abzurufen. Die Informationen beschränken sich dabei auf folgende Punkte:

- Gemeindename
- Postleitzahl, Ort und Kanton
- Anzahl Einwohner
- Gemeindebeschreibung
- Steuerlage
- Infrastruktur (Einkaufsmöglichkeiten, Spital, Kindergarten, Primarschule, Sekundarschule, Gymnasium, Hochschule, Universität)
- Gemeindebilder

2.3 Immobilien

Um die ermittelten Objekte genauer analysieren zu können, enthalten die exklusiven Immobilien folgende Angaben:

- Objektart (Haus oder Wohnung)
- Titel & Beschreibung
- Adresse (PLZ, Ort, Strasse, Hausnummer)
- Position (Längengrad, Breitengrad)
- Preis
- Anzahl Zimmer
- Wohnfläche
- Grundstücksfläche
- Verfügbarkeit
- Bilder
- Baujahr
- Ausstattung (Lift, Balkon, Garten)

2.4 Anmerkung

Weiterhin erachten wir es als wichtig anzumerken, dass die praktische Umsetzung unserer Projektarbeit für den Webbrowser «Chrome» ausgelegt und optimiert wurde.

3. Use-Cases

3.1 Use-Case-Diagramm

Unser nachfolgendes Use-Case-Diagramm zeigt eine Übersicht über das Gesamtsystem der ImmoMobile AG. Dabei wird das Diagramm möglichst schlicht und einfach gehalten, um einen einfachen Überblick zu gewährleisten. Zu weit hergeholte und nicht bedeutsame Use Cases lassen wir aussen vor.

Die funktionalen Anforderungen sind strukturiert gehalten und zeigen das aktuelle Profil der ImmoMobile AG.

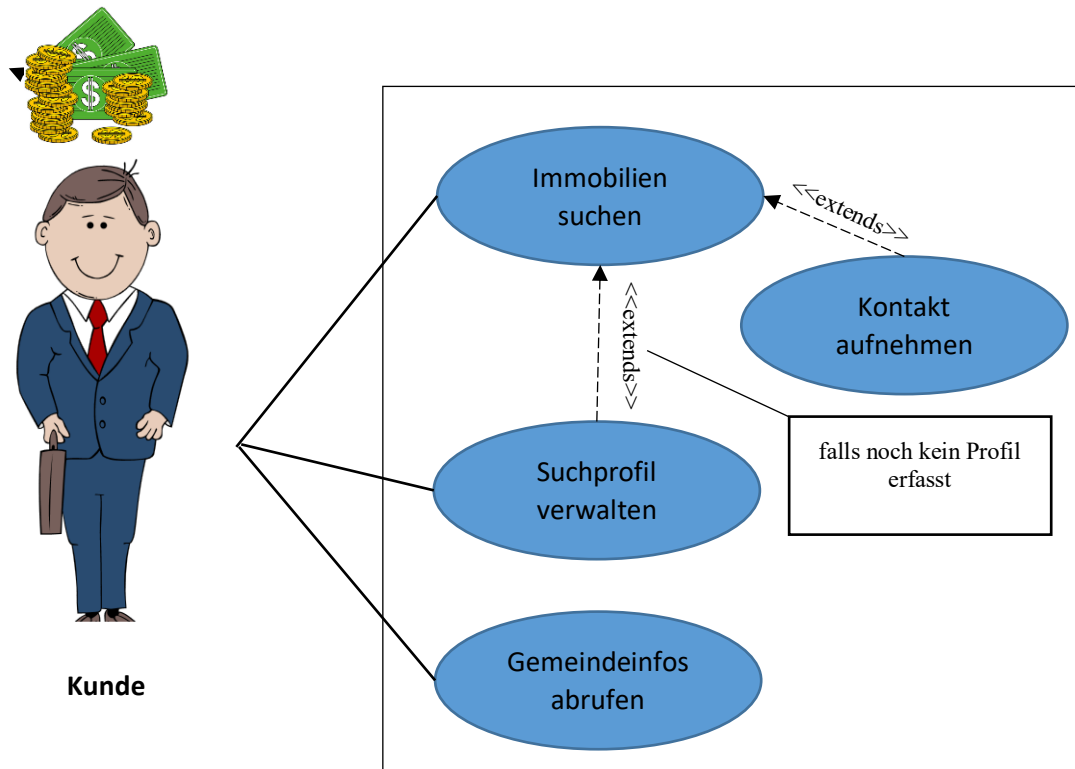


Abbildung 1 - Use-Case-Diagramm

Im Rahmen unserer Gruppenarbeit haben wir uns dazu entschlossen, vor allem Wert auf die Use Cases «Suchprofil verwalten» und «Immobilien Suchen» zu legen. Den Use Case «Gemeindeinfos abrufen» haben wir zusätzlich in die Immobiliensuche integriert.

3.2 Use-Case-Beschreibungen – das Vorgehen

Nachdem wir uns in der Gruppe für die im Use-Case-Diagramm enthaltenen Use Cases entschieden haben, konnten wir deren Beschreibung in Angriff nehmen. Dabei war es wichtig, dass jeder in der Gruppe die einzelnen Use Cases genauestens verstand und umschreiben konnte. Bei der Beschreibung einzelner Use Cases sind wir auch auf Probleme gestossen, die wir im Unterkapitel 5.2 beschreiben.

Zwei Gruppenmitglieder haben sich hierbei zusammengesetzt und das Vorgehen besprochen. Dabei wurden einige Deklarationen wieder gelöscht und neu erfasst. Schlussendlich konnten wir folgende Use-Case-Beschreibungen gestalten:

3.2.1 Use-Case-Beschreibungen – die Beschreibungen

Use Case Name:	Immobilie suchen
Auszulösender Akteur:	Kunde / Interessenten
Zweck / Ziel:	Beschreibt den typischen Ablauf bei der Suche nach einer Immobilie
Eingehende Information:	Suchkriterien aus Speicherraum
Ergebnis:	Immobilie gefunden & Ausgabe als Liste [optional: Weitere Immobilieninformationen anzeigen, Kontaktaufnahme via Mail]

Grundlegender Ablauf:

1. Kunde betätigt den Button «Immobilie Suchen».
2. System lädt die Suchkriterien aus dem Speicher. Falls noch kein Suchprofil vorliegt, muss dieses zuerst abgefüllt werden. (siehe Use Case «Suchprofil verwalten, Seite 7»).
3. System führt, basierend auf den Suchkriterien, die Immobiliensuche durch.
4. System listet die gefilterten Immobilien auf.
5. [optional] Kunde kann bei gefilterten Immobilien aktuelle Daten (Preis, Anzahl Zimmer, Wohnfläche, Grundstückfläche, Verfügbarkeit, Baujahr, Ausstattung (Lift, Balkon, Garten)) anzeigen lassen.
6. [optional] Kunde kann via Mail/Telefon Kontakt mit der ImmoMobile AG aufnehmen.

Use Case Name:	Suchprofil verwalten
Auszulösender	Kunde / Interessenten
Aktor:	
Zweck / Ziel:	Beschreibt den typischen Ablauf bei der Verwaltung des Suchprofils
Eingehende Information:	[optional: Gemeinde, Objekttyp (Haus / Wohnung), Anzahl Zimmer, Mindestwohnfläche, Maximalpreis]
Ergebnis:	Suchkriterien werden im Speicher gespeichert

Grundlegender Ablauf:

1. [optional] Kunde wählt die gewünschte Gemeinde aus.
2. [optional] Kunde entscheidet sich für das gewünschte Objekt (Haus oder Wohnung).
3. [optional] Kunde spezifiziert die Suche mit den gewünschten Anzahl Zimmer.
4. [optional] Kunde spezifiziert die Suche mit der gewünschten Mindestwohnfläche.
5. [optional] Kunde spezifiziert die Suche mit der Angabe des Maximalpreises.
6. System speichert sämtliche Angaben des Suchprofils im Speicher.
7. [optional] Kunde kann, wenn gewünscht, das Suchprofil mit dem Button «Suchprofil zurücksetzen» zurücksetzen.
8. [optional] Kunde kann bei einem erneuten Besuch der Webseite mit dem Button «Suchprofil laden» das via Speicher gespeicherte Suchprofil aufrufen.

Use Case Name:	Gemeindeinfos abrufen
Auszulösender Akteur:	Kunde / Interessenten
Zweck / Ziel:	Beschreibt den typischen Ablauf bei der Suche nach Gemeindeinformationen
Eingehende Information:	Gemeinde
Ergebnis:	Gewünschte Gemeinde wird mit den grundlegenden Informationen angezeigt.

Grundlegender Ablauf:

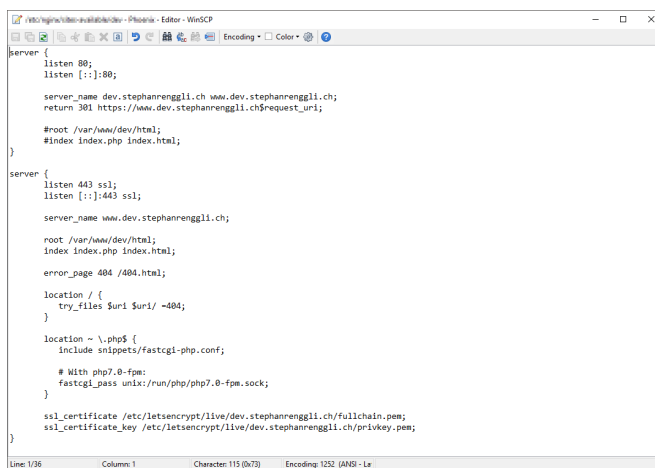
1. Kunde entscheidet sich für die gewünschte Gemeinde.
2. Kunde betätigt den Button «Gemeindeinfos anzeigen».
3. System führt die Suche durch.
4. System zeigt folgende Informationen über die Gemeinde an:
 - a. den Namen der Gemeinde
 - b. die Postleitzahl
 - c. den Kanton
 - d. die Anzahl Einwohner
 - e. eine Beschreibung zur gefilterten Gemeinde
 - f. die aktuelle Website der Gemeinde
 - g. den Steuerfuss
 - h. die Infrastruktur (Einkaufsmöglichkeiten, Spital, Kindergarten, Primarschule, Sekundarschule, Kantonsschule, Hochschule, Universität)
 - i. drei Bilder zur Gemeinde

4. Einrichtung Webserver

Für eine saubere Darstellung und Hands-on-experience haben die Teammitglieder beschlossen die programmierte Webseite auf einen Webserver hochzuladen. Es ermöglichte der Gruppe einfacheres Testen auf diversen Endgeräten und zeigte, welche Applikationen per Webserver bereitgestellt werden müssen. Der Upload auf den Server wurde via FTP (Client: WinSCP) getätigt. Der verwendete Webserver ist Nginx 1.10.3 mit PHP7.0. Um die sichere Übertragung unserer Kundendaten zu gewährleisten, haben wir die Website per SSL gesichert.

Das Endprodukt unserer Arbeit wird unter <https://www.dev.stephanrenggli.ch> bereitgestellt.

Nachfolgend sind das notwendige Nginx-Konfigurationsfile (Abb. 2) und das Testresultat eines SSL-Sicherheitstests von sslabs.com (Abb. 3) dargestellt.



```
server {  
    listen 80;  
    listen [::]:80;  
  
    server_name dev.stephanrenggli.ch www.dev.stephanrenggli.ch;  
    return 301 https://www.dev.stephanrenggli.ch$request_uri;  
  
    #root /var/www/dev/html;  
    #index index.php index.html;  
}  
  
server {  
    listen 443 ssl;  
    listen [::]:443 ssl;  
  
    server_name www.dev.stephanrenggli.ch;  
  
    root /var/www/dev/html;  
    index index.php index.html;  
    error_page 404 /404.html;  
  
    location / {  
        try_files $uri $uri/ ~404;  
    }  
  
    location ~ \.php$ {  
        include snippets/fastcgi-php.conf;  
  
        # With php7.0-fpm:  
        fastcgi_pass unix:/run/php/php7.0-fpm.sock;  
    }  
  
    ssl_certificate /etc/letsencrypt/live/dev.stephanrenggli.ch/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/dev.stephanrenggli.ch/privkey.pem;  
}
```

Abbildung 2 - Nginx-Konfigurationsfile

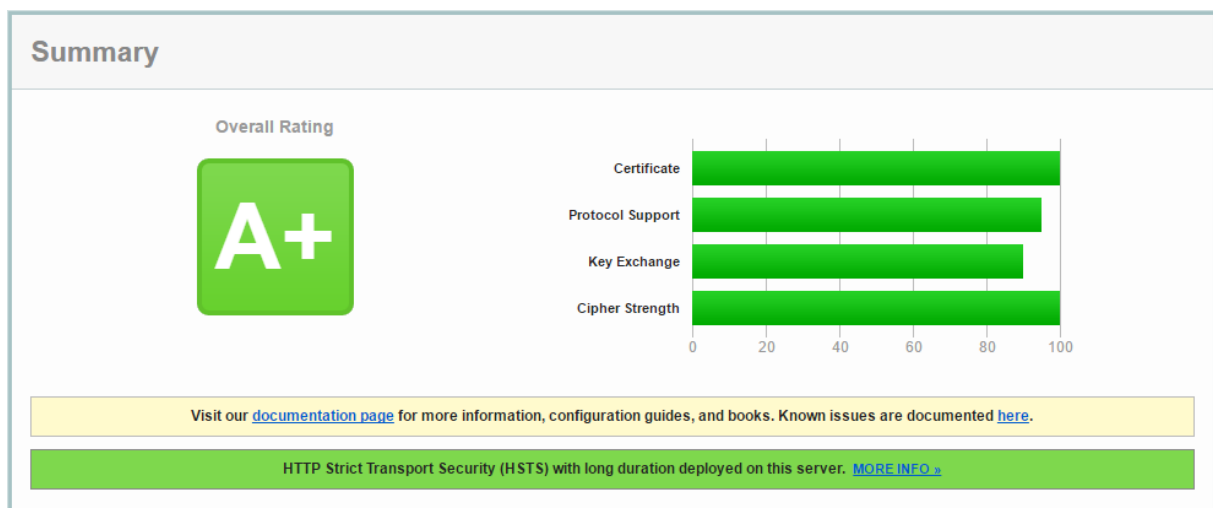


Abbildung 3 - SSL Testresultat von sslabs.com

5. Herausforderungen

Beinahe jede Projektarbeit beinhaltet Herausforderungen und Probleme, manche bereiten einem mehr Mühe, andere können schneller beseitigt werden.

Auch bei unserer Arbeit an der Umsetzung der mobilen Applikation von «ImmoMobile AG» standen wir teilweise vor kniffligen Situationen. Durch genaue Recherche und in Zusammenarbeit mit dem Team konnten wir diese jedoch erfolgreich lösen.

5.1 Use-Case-Diagramm

Nachdem uns die Aufgabenstellung mitgeteilt wurde, entschieden wir uns in der Gruppe, dass jeder auf die darauffolgende Woche Überlegungen zu den Use Cases und dem Diagramm machen würde.

Bei der Besprechung der Diagramme standen wir vor der Herausforderung, jenes Diagramm nicht zu kompliziert darzustellen. Manche Gruppenmitglieder griffen eindeutig zu weit, viel zu viele Use Cases wurden ins Diagramm eingebaut. Die Herausforderung hierbei war es, aus der Aufgabenstellung ein einfaches Diagramm zu erstellen. Ein Diagramm, das auch ein potenzieller Kunde nachvollziehen kann und davon angesprochen wird. Wir sind durchaus zufrieden, wie wir anschliessend vorgegangen sind. Nachdem wir bei Herrn Zimmermann diesbezüglich nachgefragt haben, konnte wir das Use Case schlussendlich um einiges übersichtlicher gestalten. Abschliessend entschieden wir uns dazu, lediglich drei Use Cases im Diagramm darzustellen. Dies ist zum einen übersichtlich und ansprechend.

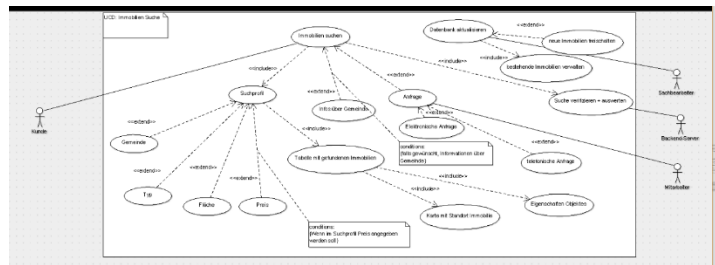


Abbildung 4 - kompliziertes Use-Case-Diagramm

5.2 Use-Case-Beschreibungen

Tatsächlich hatten wir zu Beginn etwas Mühe. Besonders der Use Case «Immobilien suchen» empfanden wir schwierig in Worte zu fassen. So vermuteten wir in der Startphase, dass dieser genannte Use Case die einzelnen Suchkriterien (Gemeinde, Objekt, Anzahl Zimmer, Mindestwohnfläche, Preis) beinhalten sollte. Dies ist jedoch nicht der Fall. So tätigt der Kunde / Interessent lediglich den Button «Immobilien suchen». Die Kriterien werden nun aus dem Speicherraum (Local Storage) geladen.

Hingegen ist das Suchprofil und deren Kriterien im Use Case «Suchprofil verwalten» massgebend. Hier spezifiziert der Kunde die gewünschte Suche mit der Angabe der einzelnen Kriterien. Dabei mussten wir uns zuerst Gedanken machen, ob diese Kriterien als obligatorisch zu betrachten sind oder nicht. Da wir der Meinung sind, dass eine Suche auch ohne Angabe der Kriterien erfolgen kann, entschlossen wir uns dazu, diese Spezifikationen als optional einzufügen. Demzufolge haben wir die Beschreibung des Use Case auch zu grössten Teilen als optional gekennzeichnet. Der Kunde kann zum Beispiel das Suchprofil mit der Angabe einer Gemeinde spezifizieren, er muss allerdings nicht. Die gewählten Kriterien werden automatisch im Speicher (Local Storage) abgespeichert.

Dieses geschilderte Vorgehen war uns anfangs nicht klar, weswegen wir auch hier auf Herrn Zimmermann angewiesen waren. Mit hilfreichen Formulierungen half uns Herr Zimmermann auf die Sprünge. Die Beschreibungen konnten wir somit zufriedenstellend abschliessen.

5.3 Local Storage

Nachdem wir die optische Gestaltung des Suchprofils via Datei «index.html» gestaltet haben, nahmen wir den Local Storage in Angriff. Die Umsetzung des Local Storage beanspruchte einiges an Zeit und auch Nerven.

Neben der Anforderung eines Local Storage in der Aufgabenstellung erachten wir es auch selber als sinnvoll, dass ein Suchprofil automatisch gespeichert wird. Angenommen, der Kunde hat es gerade eilig und muss erzwungenermassen den Browser schliessen. Beim Verlassen der Seite werden die Suchkriterien im Cache des Browsers automatisch gespeichert. Falls der Kunde die Seite nun wieder aufrufen möchte, kann er das im Local Storage gespeicherte Profil mit dem in der «index.html» erstellten Button «Suchprofil laden» erneut aufrufen. Die Kriterien werden nun aus dem Local Storage geladen.

Eine der Herausforderung war es hierbei, eine geeignete Funktion für die Speicherung der Werte zu programmieren. Nachdem wir die Variablen in der Datei «controller.js» deklariert und mit den ID's von «index.html» verknüpft haben, schafften wir es, die Werte via «setItem» zu speichern und via «getItem» im Local Storage anzuzeigen.

Die rechts in Abbildung 3 ersichtliche Funktion konnten anschliessend für alle Suchkriterien programmieren. Die vom Kunden getätigte Spezifikation im Suchprofil wird nun automatisch im Local Storage abgespeichert.

```
$(function () {
  selectGemeinde.change(function () {
    localStorage.setItem('gemeinde', this.value);
  });

  if (localStorage.getItem('gemeinde')) {
    selectGemeinde.val(localStorage.getItem('gemeinde'));
  }
});
```

Abbildung 5 - Funktion zur Speicherung im Local Storage

Eine weitere Herausforderung waren die «Slider», mit der die Suche via Mindestwohnfläche und dem Preis der Immobilie spezifiziert werden kann. Hierbei mussten wir im Internet recherchieren, um eine geeignete Funktion ausfindig zu machen. Auch dies konnten wir allerdings erfolgreich umsetzen.

```
function setUpEventHandlersPreis() {
  $('#sliderPreis').change(function() {
    localStorage[this.id] = $(this).val();
  });
}
```

Abbildung 6 - Funktion zur Speicherung "slider"-Werte

Um einiges problematischer war es, die vom Kunden gewählten Kriterien wieder anzeigen zu lassen, sobald das Suchprofil geladen wird. Wir mussten somit eine Funktion programmieren, mit der die Werte aus dem Local Storage wieder im Suchprofil angezeigt werden. Die bereits genannten Funktionen zur Speicherung der Werte im Local Storage konnten wir beinahe eins zu eins in die Funktion «loadProfil()» kopieren. Aber ein entscheidendes Merkmal war es, die Funktion via «.trigger('change');» abzuschliessen. Mit dieser Eigenschaft werden die Werte aus dem Local Storage automatisch ins Suchprofil geladen.

```
function loadProfil() {
  $(function () {
    selectGemeinde.change(function () {
      localStorage.setItem('gemeinde', this.value);
    });

    if (localStorage.getItem('gemeinde')) {
      selectGemeinde.val(localStorage.getItem('gemeinde')).trigger('change');
    }
  });
}
```

Abbildung 7 - Funktion loadProfil()

5.4 Suchprofil zurücksetzen

Ziel des Buttons «Suchprofil zurücksetzen» ist es, dass der Kunde jederzeit seine gewählten Spezifikationen (Gemeinde, Typ, Anzahl Zimmer, Mindestwohnfläche, Preis) resettet kann. Im Team entschieden wir uns dazu, diese Funktion einzubauen, da sie als hilfreich angesehen werden kann.

Ein wirkliches Problem hatten wir hierbei allerdings nicht. Teilweise war es mühsam, die Parameter auf aktuellem Stand zu halten. Via «setItem» werden die Werte auf den Anfangswert zurückgesetzt. Mit der rechts dargestellten Funktion werden die Werte aus dem Local Storage gelöscht, aber die «DropDown's» bleiben noch mit den aktuellen Werten bestehen. Mittels einer einfachen Funktion werden aber auch diese Werte zurückgesetzt.

```
function resetProfil() {  
    //reset localStorage  
    localStorage.setItem('gemeinde', 'all');  
    localStorage.setItem('immobilieTyp', 'all');  
    localStorage.setItem('anzahlZimmer', 'all');  
    localStorage.setItem('sliderFlaeche', '100');  
    localStorage.setItem('sliderPreis', '1');
```

Abbildung 8 - Funktion resetProfil()

5.5 Übergabe Filterkriterien, Auswertung in PHP

Eine vorgängige Testprogrammierung mittels Dynamic Collapsible aus der JQuery Datenbank im JavaScript lieferte uns erste Testresultate in mehreren Containern. Schnell hat sich jedoch gezeigt, dass sich Collapsibles im Kontext von ImmoMobile nicht als tauglich erweisen.

Eine weitere Problematik stellte sich bei der "Denkweise" für die Filterung der Objekte heraus. Es war den Autoren unklar, ob dies in einem Statement herausgelesen werden konnte, oder dies mittels mehreren Iterationen erfolgen sollte. Nach einiger Einarbeitung in PHP kristallisierte sich allmählich ein Algorithmus, der gezielt nicht gefilterten Objekte aus einem kopierten Array herauslöscht und dann das gefilterte Array dann in einem neuen Tab anzeigt.

Bezüglich der Weitergabe des Local Storage, wurde die Methode GET verwendet, so dass die URL auch in einen anderen Webbrowser eingesetzt werden kann und das Ziel erreicht. Bei dieser Thematik bedurfte es an Einarbeitungszeit der zu senden Parameter mittels GET oder POST.

Die Anzeige der Resultate erfolgt vollständig per PHP. Es wird für jede Immobilie dynamisch eine "Karte" erstellt, die alle wichtigen Informationen anzeigt. Zudem gibt es entsprechende Links um direkt einen Telefonanruf tätigen zu können oder ImmoMobile per E-Mail zu kontaktieren (inkl. Betreff und Standardnachricht).

5.6 Google Maps per PHP

Da wir null bis maximal 13 Ergebnisse erhalten können, je nach Suchparameter, mussten wir eine Lösung finden um auch die Google Maps dynamisch generieren zu können. Leider sind wir hier auf eine Limitierung der Google Maps API gestossen, anscheinend können wir nicht mehrere Karten gleichzeitig über dieselbe Funktion anfordern (der entsprechende Code ist jedoch auskommentiert in der Datei search.php einsehbar).

Als Alternative haben wir uns für einen Button entschieden, der beim Klick auf eine neue Seite verweist, auf welcher die Karte (inkl. Marker und Route) dargestellt wird. Die nötigen Positionsdaten werden wiederum per PHP GET-Methode übermittelt.

5.7 Slideshow

Die Slideshow auf der Startseite wurde per SwiperSlider(<http://idangero.us/swiper/>), die Slideshows auf der Ergebnisseite per SlickSlider(<http://kenwheeler.github.io/slick/>) realisiert. Beide verfügen über gute Dokumentation, die Einbindung war entsprechend einfach.

6. Schlussfolgerung

Die Projektarbeit im Modul «Netzwerk Architekturen & Web Technologien – Theorie» empfanden wir als eine spannende und herausfordernde Abwechslung. Auf der einen Seite konnten wir das erlernte theoretische Wissen in die Praxis umsetzen, andererseits empfinden wir auch den Themenbereich um die Web-Applikationen als spannend.

Wir erachten es als eine gute Idee, wenn uns der theoretische Teil in den ersten Blöcken des Moduls nähergebracht wurde. Somit konnten sich Neulinge zuerst einmal in den Themenbereich einarbeiten, bevor die praktische Arbeit folgte.

Während einer Projektarbeit ist das Team natürlich fundamental wichtig. Daher war es sehr förderlich, dass wir auch die Gruppen selber bilden konnten.

Das Vorgehen in unserem «Team» hat von Anfang an gepasst. Zuerst hat sich jeder selber mit der Aufgabenstellung auseinandergesetzt und basierend auf den Fragen sein eigenes Use-Case-Diagramm erstellt. In der Woche darauf haben wir diese zusammen besprochen. Dies war sinnvoll, da nun jedem die Aufgabe in vollem Umfang klar war. In diesem Meeting konnten wir auch gleich die Aufgabenverteilung für das Projekt festlegen. Hier haben wir darauf geachtet, dass jeder gemäss seinen Fähigkeiten eingesetzt wurde. Dies ist in unseren Augen ein wichtiger Schritt, da dies ebenfalls zu einer Projektarbeit gehört – das «managen» der Aufgabenverteilungen. Demzufolge haben wir uns nun immer in den darauffolgenden Wochen versammelt und die getätigten Aufgaben besprochen. Selbstverständlich haben wir auch die heutzutage gängigen elektronischen Kommunikationswege (WhatsApp, Skype) benutzt, um unklare Sachverhalte zu klären.

Bei der Umsetzung des Use-Case-Diagramm haben vor allem Patrice Weber und Timothy Jung sehr weit gegriffen, wie oben in der Abbildung 2 ersichtlich ist. Dabei deklarierten die Gruppenmitglieder sehr viele komplizierte Use Cases, die wir schlussendlich nicht beachtet haben. Zudem bauten wir die Deklaration «include» ein, welche ebenfalls nicht obligatorisch war. Wir fanden es wichtig, das Diagramm übersichtlich und einfach zu gestalten. Dies veranlasste uns auch dazu, lediglich drei Use Cases im Diagramm einzubauen. Somit ist das Diagramm zum einen übersichtlich gehalten, zum anderen aber auch mit der nötigen Klarheit versehen. Bei den Beschreibungen der einzelnen Use Cases mussten wir unsere Deklarationen auch teilweise erneut hinterfragen, besonders der Use Case «Immobilien suchen» bereitete Mühe.

Weiterhin konnten wir auch beim praktischen Teil Erfahrung sammeln, wie zum Beispiel beim Local Storage. Die Grundkenntnisse vom Programmieren haben sicherlich viel geholfen, doch teilweise mussten wir einige Recherchen im Internet durchführen. Durch hilfreiche Forenbeiträge konnten wir uns «schlau machen» und das Gelernte einbeziehen. Es war besonders motivierend, wenn wir stundenlang an einem Problem gekaut haben, dieses aber dann doch erfolgreich lösen konnten. Als Beispiel können wir hier die Funktion «loadProfil()» nennen, bei der die im Local Storage gespeicherten Suchkriterien durch diese genannte Funktion wieder in das Suchprofil eingelesen werden. Nach stundenlangen ausprobieren schafften wir dies durch eine kleine Änderung des Codes.

Abschliessend konnten wir alle sehr viel Erfahrung und Kenntnisse in diesem Projekt sammeln. Nicht nur Wissen im Bereich des Programmierens, sondern auch in der Kommunikation. Wir sind allesamt sehr zufrieden und stolz auf unsere abgeschlossene Arbeit.

7. Abbildungsverzeichnis

Abbildung 1 - Use-Case-Diagramm	3
Abbildung 2 - Nginx-Konfigurationsfile	7
Abbildung 3 - SSL Testresultat von sslabs.com	7
Abbildung 4 - kompliziertes Use-Case-Diagramm	8
Abbildung 5 - Funktion zur Speicherung im Local Storage	9
Abbildung 6 - Funktion zur Speicherung "slider"-Werte	9
Abbildung 7 - Funktion loadProfil()	9
Abbildung 8 - Funktion resetProfil()	10