Fortgeschrittenenpraktikum

# Portierung der Bavarian App in Ionic

*Patrick Frömel, Tim Romonath*

*Unter Anleitung von Prof. Christoph Bockisch und PD. Dr. Hanna Fischer*

29. Oktober 2021

# Inhaltsverzeichnis

# 1 Kurzbeschreibung

Die App „Welcome to Bavaria“ in der Version 3.0 ist eine Portierung der gleichnamigen Version 2.0 auf das Ionic-Framework um diese auf verschiedenen Betriebssystemen - insbesondere iOS und Android - mit geringen Entwicklungsaufwand zur Verfügung zu stellen. Die App wurde dabei im Zeitraum von Juli bis Oktober 2021 Entwickelt.

Die App beruht auf dem gedrucktem Heft „Migraboarisch - Von uns fia eich. Ein Wegweiser durch den Dialekt“ welcher im Schuljahr 2016/17 von Kelheimer Schülern Entwickelt wurde. Ziel war es dabei Neuankömmlingen in Bayern den Bayrischen Dialekt näher zu bringen.

Die App wurde in der Version 1.0 von Marburger Studenten 2019/20 entwickelt und um verschiedene Sprachaufnahmen ergänzt. Für die Version 2.0 wurden weitere Sprachaufnahmen und ein primitives Archivement-System eingebracht.

## 1.1 Zur Portierung

Es wurde sich für Ionic mit dem Angular-Framework entschieden um die Vorteile der Webentwicklung bezüglich Ihrer Anpassungsfähigkeit in verhältnismäßig einfachen Frameworks zu nutzen und damit die Reichweite der App zu erweitern. Sogenannte „Web-Apps“ sind weit verbreitet und stellen - vereinfacht ausgedrückt - Websites in speziellen Browsern dar so dass der Quellcode auf unterschiedlichen Betriebssystemen stets der selbe bleiben kann. So müssen bei der Wartung nicht verschiedene Programme parallel Entwickelt werden.

Ionic mit Angular wird in dieser Kombination von quasi allen Seiten propagiert und wurde in unserer unbedarftheit dankend angenommen. Ionic stellt viele mehr oder minder komplexe Bausteine zur Webentwicklung bereit (von Textfeldern über Knöpfen bis hin zu Slidern mit Animation) während Angular zum einen HTML um viele nützliche „Funktionen“ erweitert (Loops, If-Else,...) und zum anderen komplexe Mechanismen im Hintergrund verwaltet so dass beispielsweise eine Website mit verschiedenen wechselnden Komponenten hoch dynamisch gestaltet werden kann.

In der Portierung wurde versucht sich an S.O.L.I.D-Prinzipien zu halten, auch wenn dies mit Type-Script als Programmiersprache bisweilen kompliziert gestaltete. Während die Datenbank-Mechanismen noch verhältnismäßig gut ohne Abhängigkeiten funktionieren und dank Schnittstellen (Interfaces) auch einfach ausgetauscht werden können sind die Seiten der UI sehr eng an verschiedene „Services“ gebunden welche wiederum untereinander stark abhängig sind. Dies ist aber keine Notwendigkeit und das refactoren dieser kann Gegenstand eines Nachfolgeprojekts sein. Kern der Bemühungen ist ein „Request-Response“-Pattern in welchen ein Request über ein Interactor gestellt wird welcher wiederum von den Datenbankmechanismen eine Response-Methode aufrufen. Sehr großer Vorteil ist hier das sich die asynchrone Programmierung in Grenzen hält - eine Aufgabe kann und wird auch erst dann zuende gebracht wenn eine Antwort vorhanden ist.

Alle Daten der App liegen als Exceltabellen vor. Um diese besser verarbeiten

zu können werden die Daten in einer Datenbank abgespeichert. In der App 2.0 wurde dies durch eine SQL-Datenbank realisiert. Daher war auch unsere erste Herangehensweise eine SQL-Datenbank zu verwenden, um die Modellierung der DB und der Abfragen von der Version 2.0 wiederverwenden zu können. Diese Idee wurde aber verworfen, da eine NoSQL-Datenbank (IndexedDB) schon in den Browser eingebaut ist und somit wesentlich einfacher zu erreichen und auch zu testen ist. Dies ruht daher, dass wir so auch die App in einem normalen Browser ausführen konnten, anstatt sie jedes mal für ein Mobilgerät neu „builden“ zu müssen. Des weiteren hat sich herausgestellt, dass die benötigten Anfragen auf die Datenbank eine relativ niedrige Komplexität aufwiesen.

## 1.2 Das Team

### Leitung

Hanna Fischer

### Koordination und Betreuung

Hanna Fischer, Christoph Bockisch, Peter Kaspar

### Systementwicklung

Version 3.0: Patrick Frömel, Tim Romonath
Version 2.0: Lea Fischbach
Version 1.0: Benedikt Batton, Lester Brühler, Lea Fischbach, Hannah Greß, Anh-Thy Truong, Qi Yu

### Mitarbeit

Milena Gropp

### Sprachaufnahmen

Melanie Bajrami – Regensburg (Nordmittelbairisch)
Christina Böhmländer – Zwiesel (Nordmittelbairisch)
Felicitas Erhardt – Walleshausen (Schwäbisch-Mittelbairisches Übergangsgebiet)
Christian Ferstl – Regensburg (Nordmittelbairisch)
Edith Funk – Krumbach (Schwäbisch)
Markus Kunzmann – Mühldorf am Inn (Mittelbairisch)
Barbara Neuber – Altenstadt an der Waldnaab (Nordbairisch)
Michael Schnabel – Bayreuth (Ostfränkisch)

### Übersetzung

Arabisch: Clara Mikhail
Englisch: Jeffrey Pheiff
Tschechisch: Andrea Königsmarková

### Grafiken

Udo Butler

### Beteiligte Lehrerinnen und Schülerinnen

Peter Kaspar, Daniela von Schultz
Karam Alayoubi, Sali AlBoulad, Yazan Aljleilati, Ahmad AlShalabi, Mustafa AlShalabi, Arreeya Banlo, Anita Blaha, Nicolas Bruckmaier, Florian Fahle, Nico Fichte, Lukas Fischer, Tina Föhre, Nico Horn, Aisa Jusic, Alexej Kobzev, Karina Kluger, Max Kohlmeier, Samuel Konschelle, Damian Kowalski, Andreas Lening, Laura Lohwasser, Markus Müller, Felix Neuhauser, Franziska Nigl, David Nguyen, Samuel Noy, Daniel Rißmann-Toledo, Lena Ritzinger, Anna Roidl, Joshua-David Roland, Heba Sandafi, Julian Schanderl, Andreas Stemmer, Wahyu Susilo, Liviu Vasiu, Mihai Vasiu, Maria Waldhier, Markus Weber, Stefanie Weber, Amadeus-Daniel Wiselka, Burak Yılmaz, Andreas Zellner, Tobias Zinkl, Thomas Zirmer

# 2 Entwicklungsprozess

Der Entwicklungsprozess selbst wurde zunächst keiner komplexen Systematik unterworfen. Es gab die Trennung, dass Patrick Frömel die Datenbankmechaniken und Interfaces inkl. Interactor entwickelt während Tim Romonath sich in Ionic und Angular einarbeitet und das User-Interface entwickelt.

Die ersten 5 Wochen haben wir an den Wochentagen in den Semesterferien uns morgens getroffen, den aktuellen Stand besprochen und dann 6 - 8 Stunden entwickelt. Anfangs haben wir mit Agantty (Webtool zur Aufgabenplanung) gearbeitet, aber schnell festgestellt dass dies nicht unseren Bedürfnissen entsprach. Danach folgte eine Pause von 4 Wochen in welcher wir anderen Dingen nachgehen mussten. Nach dieser Pause haben wir nur schwer wieder den Einstieg in das Projekt gefunden. Zur Unterstützung haben fortan mit Trello gearbeitet, ein einfaches Webtool für Kanban-Boards.

## 2.1 Agiles vorgehen

Unser vorgehen kann am ehesten noch als „agil“ beschrieben werden. Wir haben die zugrundeliegende App Stück für Stück nachgebaut - zunächst einen Funktionierenden Kern (Datenbankmechaniken, auslesen und übertragen dieser in die UI - Soundwiedergabe, grundlegendes Routing usw.), dann wichtige Kernfeatures (Bilder, Fortschritt, Archievements, usw.) und nach und nach alle anderen implementierten Features (Dialektauswahl, App zurücksetzen, usw.).

Allerdings ist unser Vorgehen nicht besonders inkrementell und iterativ gewesen - eher modular chaotisch. Wir haben im Grunde gemacht was gerade anfiel und für den Großteil des Projekts auf Projektmanagement verzichtet. Das hat ob der geringen Komplexität und einer verhältnismäßig guten Codequalität funktioniert - ist aber für zukünftige Projekte nicht ratsam. Ein wirkliches, agiles Vorgehen kam erst am Ende mit Trello auf - dort wurden Aufgaben in einem Backlog gesammelt und nach und nach fertig gestellt. Allerdings gab es zu diesem Zeitpunkt nur noch ein Zwischenziel: Fertig werden.

## 2.2 Verwendete Tools

### Agantty

Agantty ist ein Webtool um Projektmanagement zu unterstützen. Es implementiert dabei ein Gantt-Diagramm, welches zeitliche Aktivitäten als Balken darstellt. Wir haben Agantty sehr schnell verworfen, da unser prinzipiell agiles Vorgehen nicht gut mit dieser Art Diagram - welches besser zu fest geplanten Projekten mit klaren Zeiten und Zwischenzielen passt (vgl. Wasserfallmodell).

### Gittea Kanban-Boards

Vor der Etablierung von Trello haben wir uns erkundigt wie die Kanban-Boards von Gittea genutzt werden können. Diese sind im Grunde eine grafische Organisationsmöglichkeit von Issues was durchaus viele Vorteile mit sich bringt - wie beispielsweise alle bekannten Gittea-Tools zum Monitoring oder die Aufgabenverteilung direkt im Repository - aber auch einen großen Nachteil weswegen wir

uns dagegen entschieden haben: Für jedes Issue bekomme Beobachter des Repositorys eine Email. Beim Sammeln der fehlenden Features wäre eine regelrechte Email-Flut auf die betreuenden Personen losgebrochen.

#### Trello

Wir haben zum Ende des Projekts mit Trello gearbeitet - eine einfache Implementation eines Kanban-Boards. Zu dem Zeitpunkt der Einführung von Trello existierte bereits ein grober Kern der App in Form von Datenbankmechaniken, dem grundlegenden User-Interface, ordentlichen Interfaces für Datenklassen und einer Schnittstelle zwischen Datenbank und User-Interface. Das Vorgehen Fehlende oder unvollständige Features zusammen mit Dokumentationsaufgaben in einem Backlog zu sammeln welches auch weiterhin mit Bugs und anderen Auffälligkeiten gefüllt wurde hat sehr zur Übersichtlichkeit des Projekts beigetragen. Wir haben mit 3 Bereichen gearbeitet: Dem Backlog - einem Pool an noch zu erfüllenden Aufgaben, einem Bereich für Aufgabe an denen gerade gearbeitet wird und einem für Aufgabe wo die Weiterarbeit gerade nicht möglich ist.

## 2.3 Verlauf

Die erste, „modular chaotische“ Phase war durch hohe Arbeitsbereitschaft geprägt. Durch ständiges und langes arbeiten am Projekt waren wir stets auf der Höhe des Projekts und hatten damit keine größeren Probleme mit fehlender Organisation. Wir haben uns direkt zu Beginn auf bestmögliches Einhalten der S.O.L.I.D.-Prinzipien geeinigt und daher viel Wert auf sinnvolle Interfaces gelegt. Wir haben zwar versucht mit Agantty eine Art Projektmanagement zu betreiben, dies ist aber am falschen Tool gescheitert.

Die zweite, „optimistische“ Phase trat nach einer arbeitsbedingten, mehrwöchigen Pause ein. Wir hatten den Überblick über das Projekt verloren könnten diesen aber mit Trello zurückgewinnen. Leider waren wir beide auch in dieser Zeit im Broterwerb eingespannt wodurch das Projekt nun zwar organisiert, aber langsam voran ging.

Die dritte, „crunchige“ Phase war von der eiligen und unsauberen (im Sinne von „schlecht wartbar“ Implementierung letzter Features und dem fixen einiger Bugs geprägt. Nebenbei musste noch dieser Bericht geschrieben werden und es stellte sich eine gewisse Müdigkeit ein.

## 2.4 Fazit

Es wäre sehr viel Sinnvoller gewesen sich gleich zu Beginn des Praktikums mit Projektmanagementmethoden zu beschäftigen um Vorzüge etablierter Herangehensweisen zu nutzen. Mit Trello zu beginn des Projekts und von SCRUM adaptierten Sprints wäre das Ziel ein wenig klarer gewesen.

Die lange Pause zwischen der ersten und zweiten Projektphase war ein großer Fehler - wenn auch kein vermeidbarer. Es fiel uns sehr schwer wieder in dem Projekt Fuss zu fassen. Dabei garnicht so sehr vom Fachlichen her sondern eher rein von der Motivation. Das Projekt wirkte bereits so fern und so erledigt was

ein großer Trugschluss war. Wir sind überzeugt mit 6 oder 7 Wochen am Stück mehr erreicht haben zu können als wir schlussendlich erreicht haben.

# 3 Verwendete Technologien

## 3.1 Node.js

## 3.2 Ionic

## 3.3 Angular

## 3.4 Cordova

## 3.5 Plugins

**NativeAudio**

**sprachdings**

## 3.6 TODO

## 3.7 TODO

## 3.8 TODO

# 4 Architektur

## 4.1 User-Interface

## 4.2 Services

## 4.3 Interfaces der Datenobjekte

bisschen kram zu den Category, VocabularyWord usw. Interfaces

## 4.4 Datenbank und Excel

wie werden die exceltabellen asugelesen und in die datenbank gebracht

## 4.5 S.O.L.I.D und Interactor-Requester

dein sehnlichst erwarteter Architekturabschnitt

# 5 Bekannte Probleme und Anregungen

# 6 Anhang

## 6.1 Script der Vorstellung

Die Bavarian App ist ein Lernprogramm um verschiedene Bayrische Dialekte ausgehend von unterschiedlichen Sprachen wie Deutsch, Arabisch oder Tschechisch zu lernen. Es gibt zwei Modi, einen um den ausgewählten Dialekt zu lernen und einen um sich diesem Dialekt in einem Quiz zu stellen. Ein Dialekt ist in verschiedene Kategorien aufgeteilt welche wiederum 8 Leveln untergeordnet sind. Wenn man alle Quizes eines Dialekts erfolgreich abgeschlossen hat - erfolgreich bedeutet hier mehr als 90% der Antworten sind richtig - bekommt man ein Archievment welches sich in den Einstellungen betrachten lässt.

Die voliegende App wurde mit Ionic und Angular in Typescript und HTML geschrieben. Ionic ist dabei ein Framework das verschiedene Elemente wie Buttons oder Slider zur verfügung stellt und Angular erweitert HTML zum einen um viele nützliche Elemente wie Loops oder If/Else-Abfragen und setzt zum anderen eine völlig neue Abstraktionsebene auf, indem mit Komponenten gearbeitet wird welche dynamisch auf einer HTML-Seite ausgetauscht werden können. Ziel war dabei die Bavarian App mit neuer Technologie in einfacher Entwicklung sowohl für iOS als auch Android zur verfügung stellen zu können.

Im Kern funktioniert die App bereits - so wird im Hintergrund eine lokale Datenbank verwaltet die Vokabeln aus Exceltabellen beinhaltet, Daten persistiert und Assets zuordnet. Im Vordergrund wird die Datenbank abgefragt, die UI befüllt und Eingaben des Users wieder an die Datenbank zurückgegeben.

Die noch zu Erfüllenden Aufgaben teilen sich auf vier große Bereiche auf:
Zuallererst fehlen noch einige Features wie zum Beispiel der Share-Button oder die Gratulationen am Ende eines Kurses welche im Detail dem Praktikumsbericht entnommen werden können.
Dann sind die Anforderungen lose der zugrundeliegenden „Bavarian App“ entnommen. Wir haben uns im Grunde angeguckt was die kann und uns zuletzt ein Backlog angelegt mit Fehlenden oder unvollständigen Features welche wir sukzessive Implementiert haben. Es wäre Vorteilhaft die Anforderungen der App einmal schriftlich in einem Pflichtenheft zu erfassen und dies mit Systemtests zu überprüfen.
Außerdem hat die UI noch einige Schwächen. So sind die Bilder nicht immer gleich groß, allgemein sind die einzelnen Größen nicht unbedingt an Smartphones angepasst und gelegentlich wird der User über die Vorgänge in der App nur unzureichend Informiert (Mit Toasts, Alerts oder auch mit ungeschickter Menüführung). Für diesen Punkt muss die App selbstständig analysiert werden.
Schlußendlich kann mit ein wenig Erfahrung in Angular ein umfassendes Refactoring der Dienste (Services) der UI erfolgen. Wir haben die UI und die Datenbank strikt getrennt und versucht uns soweit möglich an S.O.L.I.D-Prinzipien zu halten - die Dienste aber haben sich zum Schluss immer mehr verknotet. Hier Abhägigkeiten zu eleminieren ist wichtig wenn die App in späteren Projektphasen noch erweitert werden soll.

Der schwierigste Teil dieses Praktikums wird das Einarbeiten zum einen in

Ionic mit Angular sowie Angular selbst und zum anderen in unser Projekt sein. Da bereits alle Mechanismen bereit stehen ist eine Weiterentwicklung der Datenbank-Mechanismen vermutlich nciht unbedingt notwendig. Gegebenenfalls kann der Interactor als Schnittstelle zwischen den Datenbankmechanismen und dem Front-End erweitert werden. Das schön dabei ist, dass alles unangetastete dann mit dem alten Interface einfach weiterfunktionieren kann. Da der Fokus in diesem Projekt jetzt auf Qualitätssicherung liegt wird der Großteil der Aufgaben „theoretischer“ Natur sein - soll heißen viel Analyse der zugrundeliegenden Programmen und weniger Implementieraufwand.