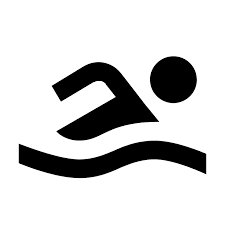
# My Swim Results



Im Auftrag eines Linzer Schwimmvereins soll eine App entwickelt werden, die die Abfrage von Schwimm-Bestzeiten über die Homepage des OSV (Österreichischen Schwimmverbands) erleichtern soll.

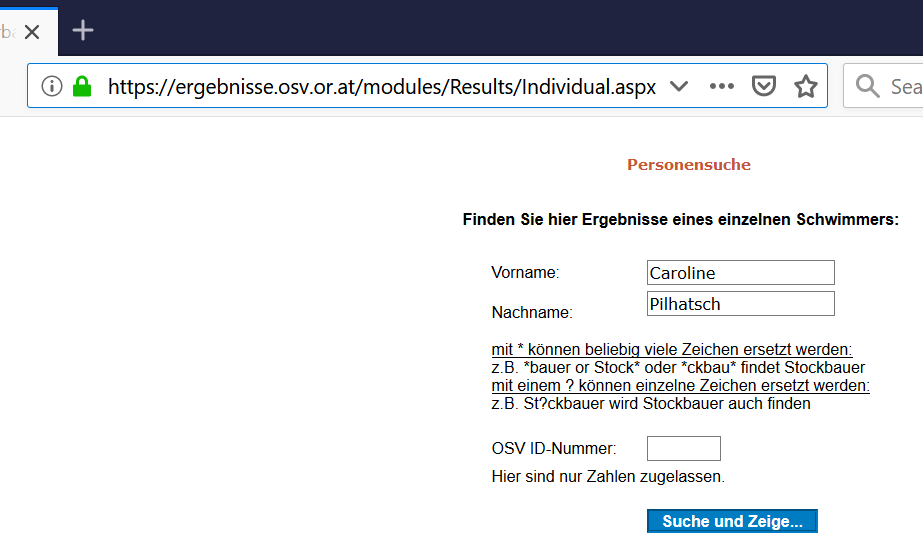
**Im Rahmen dieser Aufgabe soll die erste Ausbaustufe** für das Hinzufügen und Entfernen von SchwimmerInnen anhand deren Homepage-internen *SwimmerID* mithilfe einer lokalen Room-Datenbank implementiert werden.

In der Endversion sollen dann auf Knopfdruck die jeweiligen Bestzeiten aller von der App verwalteten SchwimmerInnen von der OSV-Webseite in die lokale Datenbank synchronisiert werden, um daraus individuelle Ranglisten erstellen zu können.

## OSV-Website

Leider bietet die OSV-Website keine REST-Schnittstellen zur Schwimmerdatenbank an, sondern es müssen stattdessen die HTML-Seiten, die von einem ASP.Net-Server erstellt werden, geparst werden. Dazu ist es notwendig, die interne *SwimmerID* zu kennen. Diese lässt sich über das Web-Formular Personensuche herausfinden: <https://ergebnisse.osv.or.at/modules/Results/Individual.aspx?Lang=de-DE>

Zum Testen werden hier bereits die internen IDs von einigen österreichischen SpitzenschwimmerInnen bereitgestellt:



|  |  |
| --- | --- |
| 40096 | Birgit Koschischek |
| 369527 | Caroline Pilhatsch |
| 242511 | Lena Kreundl |
| 50309 | Markus Rogan |
| 41575 | Bernhard Reitshammer |
| 373202 | Alexander Trampitsch |

Wie man sieht, ist die interne ID numerisch mit einer Länge von entweder 5 oder 6 Ziffern.

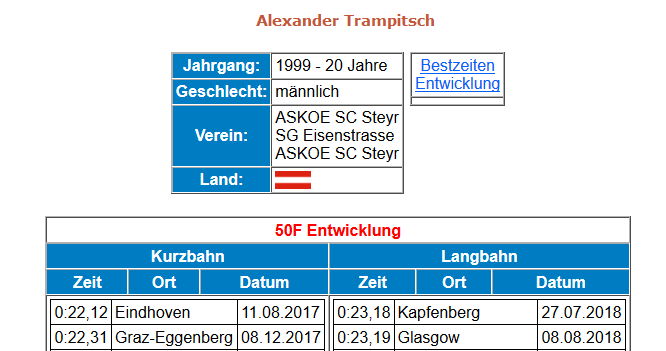
Da die Suchformulare auf der OSV-Website *hidden fields* mit Zustandsinformationen an den ASP.Net-Server mitsenden, gibt es leider nur eine Suchanfrage nach Schwimmergebnissen, die zustandslos funktioniert – nämlich die Suche nach der Bestzeiten-Entwicklung für eine SwimmerID und eine spezifische Disziplin.

Hier ein Beispiel für die Suche nach den 50m Freistil-Bestzeiten von Alexander Trampitsch:

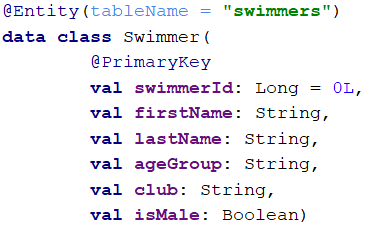
<https://ergebnisse.osv.or.at/modules/Results/Stroke.aspx?SwimmerID=373202&EventKey=50F>

Glücklicherweise enthält diese HTML-Seite alles, was sowohl für die Verwaltung der SchwimmerInnen in der App als auch für den späteren Download von Bestzeiten benötigt wird.

## Datenmodell

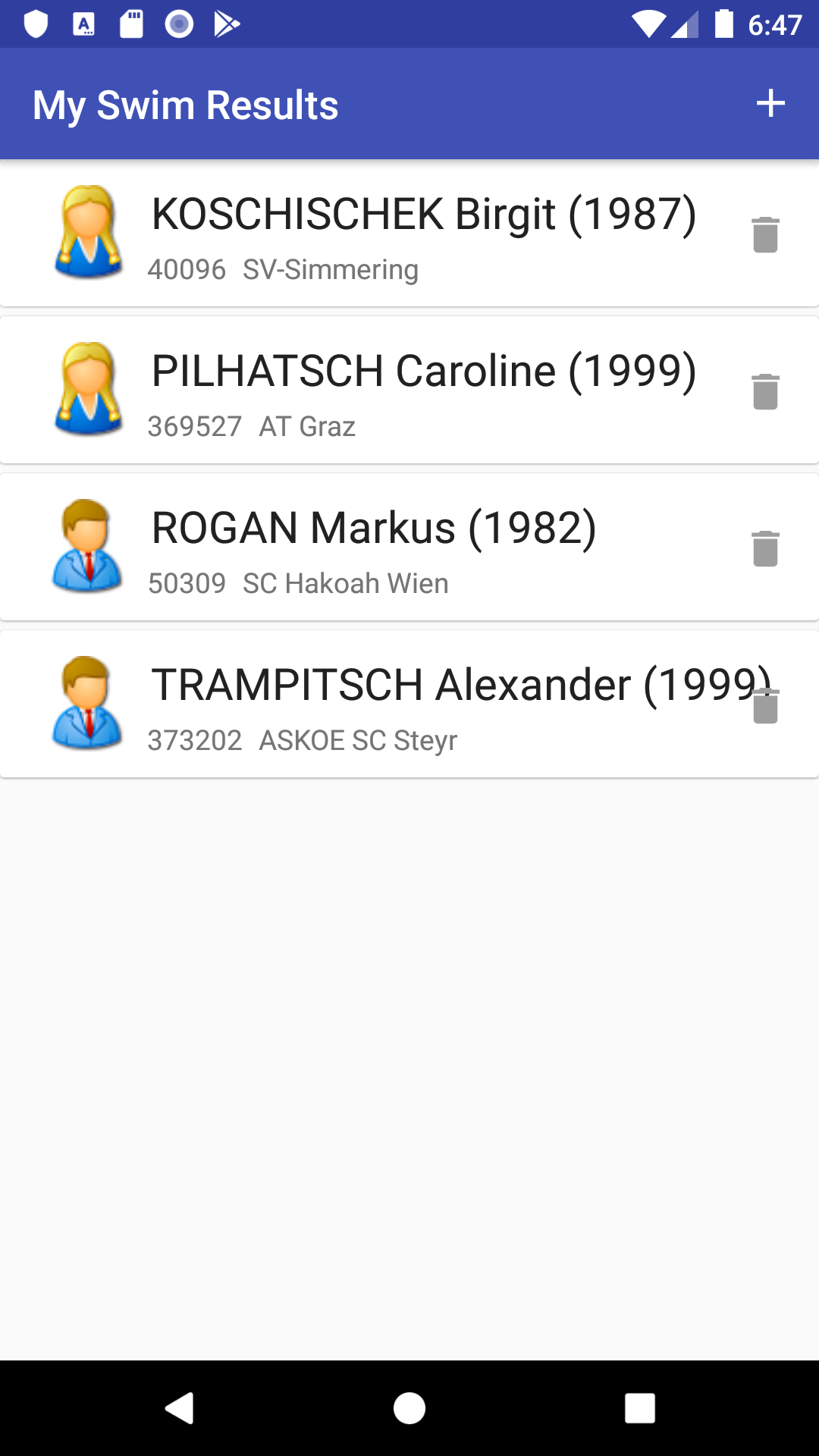


Für die erste Ausbaustufe reicht diese Entität:



**Achtung**! Bei mehreren Vereinen ist nur der erste Verein aus der Table-Column zu übernehmen! Für AgeGroup ist nur der Jahrgang (ohne Alter) relevant.

## GUI: SwimmerListFragment

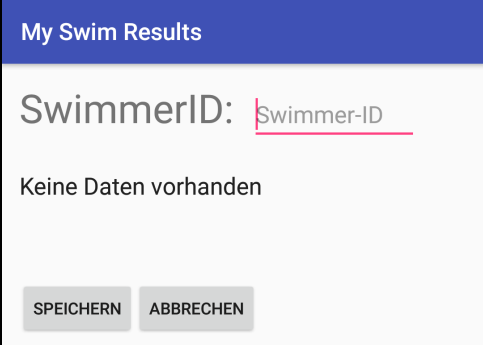


Die scrollbare RecyclerView zeigt alle in der lokalen Room-Datenbank enthaltenen *Swimmer* (Layout wie rechts abgebildet).

**Funktionalität:**

* Sortierung: Nachname, dann Vorname
* **Delete**: Beim Klicken auf den Papierkorb wird der *Swimmer* aus der Datenbank gelöscht und die Liste aktualisiert
* **Insert**: Beim Klicken auf das + im Options-Menü erscheint das *EditSwimmerFragment* (siehe unten), wo anhand der *SwimmerID* ein neuer *Swimmer* gesucht und erstellt wird.
* **Update**: Beim Klicken auf die gesamte *swimmer*\_*card* erscheint ebenfalls das *EditSwimmerFragment*, allerdings ist der selektierte *swimmer* bereits im Dialog vorausgefüllt. Trotzdem wird auch hier der *Swimmer* frisch von der Webseite geparsed, da sich z.B. die Vereinszugehörigkeit geändert haben könnte und deshalb der *Swimmer* in der DB aktualisiert werden muss.

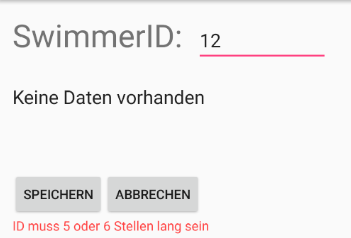
Zum Wechseln auf das *EditSwimmerFragment* ist eine **Navigation-Action** zu verwenden.



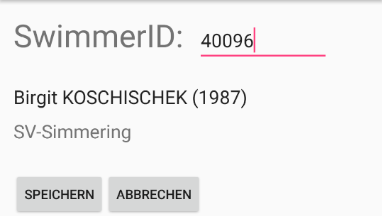
## GUI: EditSwimmerFragment – Insert

Mithilfe von bidirektionalem Databinding kann jede Änderung im numerischen Texteingabefeld Swimmer-ID an ein *LiveData*-Property im ViewModel des *EditSwimmerFragments* gebunden werden:

* Solange die *SwimmerID* nicht 5 oder 6 Zeichen enthält, gibt es keinen *swimmer* (noch ein LiveData) und daher wird im Textfeld für die Schwimmerdaten „**Keine Daten vorhanden**“ angezeigt.



* Sobald getippt wird, aber die ID weder 5 noch 6 Stellen enthält, wird eine rote Statusmeldung angezeigt: „**ID muss 5 oder 6 Stellen lang sein**“.
* Sobald die *SwimmerID* 5 oder 6 Stellen enthält, wird die Website aufgerufen und geparsed. Wenn es auf der Website eine/n SchwimmerIn mit dieser ID gibt, wird ein *Swimmer*-Objekt daraus erstellt. In der Folge werden die gefundenen Schwimmerdaten angezeigt – aber noch nicht in der Datenbank gespeichert!
* Erst mit Drücken auf „Speichern“ wird der erstellte *Swimmer* auch wirklich in die Datenbank gespeichert und mithilfe einer Navigation-Action wieder zurück in das *SwimmerListFragment* gewechselt.



* „Abbrechen“ speichert nichts und kehrt nur zurück in das *SwimmerListFragment*.

## GUI: EditSwimmerFragment – Update

Der einzige Unterschied zum Insert ist, dass beim Erzeugen des *EditSwimmerFragment* bereits eine existierende *SwimmerID* bekannt ist und daher die entsprechenden Daten aus der Datenbank bereits verfügbar sind.

Trotzdem wird auch hier die *SwimmerID* verwendet, um die Website zu parsen, und diese möglicherweise aktuelleren Daten angezeigt. Beim Speichern-Button erfolgt ein Update in der Datenbank nur dann, wenn es den *Swimmer* mit dieser *SwimmerID* in der Datenbank schon gibt und sich dessen Daten geändert haben sollten – z.B. wegen eines Vereinswechsels.

## Nicht-funktionale Anforderungen

* **Room** als Persistenz-Schicht
* Für asynchrone Abläufe sind **Coroutines** zu verwenden
* **Navigation** zwischen Fragmenten mit Navigation-Actions
* Für eine saubere Lebenszyklusresistenz sind Daten ausschließlich in **ViewModels** zu verwalten
* Nur im **EditSwimmerFragment** sind die ViewModel-Daten mithilfe von **Databinding** und **LiveData** an die Layout-Komponenten zu binden. Für RecyclerView/CardView im SwimmerListFragment ist Databinding möglich, aber nicht notwendig.

## Bonusaufgabe



Hinzufügen eines Buttons „Lade Ergebnisse“, welcher nur aktiv ist, wenn eine gültige SwimmerId gesetzt ist. Bei Auswahl dieses Buttons werden alle Bestzeiten (erste Zeilen in den Bestzeiten-Tabellen) für alle Disziplinen von der ÖSV-Homepage geladen (noch ohne Änderung in der Datenbank) und als einfacher Html-Textstring angezeigt.

In der links gezeigten Darstellung entspricht pro Disziplin der erste Wert der Kurzbahnzeit (25m-Becken) und der zweite der Langbahnzeit (50m-Becken).

**Viel Erfolg!**