

# **WEBANWENDUNG ZUR AUTOMATISIERTEN ERSTELLUNG VON WERTUNGSBLÄTTERN BEI TEAM-TURN WETTKÄMPFEN**

Ausgeführt im Schuljahr 2023/24 von:

Michael Laimer  
Laurin Losert-Nachbaur

5aWI  
5aWI

Betreuer/Betreuerin:

Diethard Kaufmann  
Bianca Franzoi

Dornbirn, am 02.04.2024

## Vorwort

Diese Diplomarbeit entstand aus der Problematik, dass die händische Eingabe von Wertungsblättern bei Team-Turn Wettkämpfen einen immensen Aufwand hervorbringt. Die Wertungsblätter müssen außerdem am Wettkampftag für jede Kampfrichterin und jeden Kampfrichter kopiert und nach Startreihenfolge sortiert werden, was oft ein zusätzlicher Stress für den Veranstalter ist. Die Lösung für dieses Problem ist eine Webanwendung, die eine effiziente digitale Eingabe von Wertungsblättern erlaubt.

Das Hauptziel der Diplomarbeit ist es, eine funktionierende Webanwendung zu entwickeln, welche eine Oberfläche für die Eingabe der Wertungsblätter bietet. Diese Wertungsblätter können dann am Wettkampftag als PDF-Dateien exportiert werden, und dadurch in der richtigen Reihenfolge und Menge ausgedruckt werden.

Zu der Erstellung der Applikation gehören ein funktionstüchtiges Frontend, das eine einfache und vor allem intuitive Eingabe der Wertungsblätter ermöglicht und ein Backend, welches mit der Datenbank kommuniziert, um die eingegebenen Daten zu speichern.

## Abstract

This diploma thesis arose from the problem that the manual entry of tariff forms at team gym competitions causes an immense amount of work. The tariff forms also have to be copied for each judge on the day of the competition and sorted by starting order, which can often act as an additional stress factor for the organizer. The solution to this problem is a web application that digitalizes the input of tariff forms.

The main goal of the diploma thesis is to develop a functioning web application that provides an interface for entering the tariff forms. These tariff forms can then be exported as PDF files on the day of the competition and printed out in the correct order and quantity.

The creation of the application includes a functional front end that enables simple and above all, intuitive entry of the tariff forms and a back end that communicates with the database to save the entered data.

## Danksagung

Diese Diplomarbeit hätte ohne die Hilfe mehrerer Personen nicht entstehen können. Besonders möchten wir uns hier bei zwei Personen bedanken, die es uns ermöglicht haben, diese Diplomarbeit umzusetzen.

Als erstes wollen wir uns bei unserem Projektbetreuer Diethard Kaufmann bedanken. Er ist uns während des ganzen Prozesses beiseite gestanden und hat uns bei Fragen und Problemen immer schnellstmöglich geholfen.

Des Weiteren wollen wir uns bei Bianca Franzoi bedanken. Sie hat uns auf der Seite des Turn-sport Austria betreut und hat uns bei Fragen über das Team Turnen immer aufgeklärt.

## **Eidesstattliche Erklärung**

# **WEBANWENDUNG ZUR AUTOMATISIERTEN ERSTELLUNG VON WERTUNGSBLÄTTERN BEI TEAM-TURN WETTKÄMPFEN**

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

---

Ort, Datum

---

Unterschrift, Michael Laimer

---

Unterschrift, Laurin Losert-Nachbaur

## Hinweise zur Textformatierung

In folgender Tabelle werden die in dem Dokument verwendeten Textformatierungen sowie die dazugehörigen Bedeutungen angeführt:

| Formatierung           | Erklärung   |
|------------------------|---|
| <b>Kursiv und Fett</b> | Begriffe, mit dieser Formatierung, sind im Glossar zu finden                                      |
| „Anführungszeichen“    | Technische und fachliche Begriffe sowie Bezeichnungen werden mit Ausführungszeichen hervorgehoben |

# Inhaltsverzeichnis

|       |  |    |
|-------|--|----|
| 1     | Impressum .....                                      | 9  |
| 1.1   | Projektteam .....                                    | 9  |
| 1.1.1 | Michael Laimer .....                                 | 9  |
| 1.1.2 | Laurin Losert-Nachbaur .....                         | 9  |
| 1.2   | Projektbetreuung .....                               | 9  |
| 1.2.1 | Diethard Kaufmann .....                              | 10 |
| 1.2.2 | Bianca Franzoi .....                                 | 10 |
| 1.3   | Turnsport Austria .....                              | 10 |
| 2     | Team-Turnen .....                                    | 11 |
| 2.1   | Geräte .....   | 11 |
| 2.1.1 | Boden .....  | 11 |
| 2.1.2 | Minitrampolin .....                                  | 12 |
| 2.1.3 | Tumbling .....                                       | 12 |
| 2.2   | Wertungsblatt .....                                  | 13 |
| 2.2.1 | Boden .....  | 13 |
| 2.2.2 | Minitrampolin .....                                  | 14 |
| 2.2.3 | Tumbling .....                                       | 14 |
| 3     | Projektmanagement .....                              | 15 |
| 3.1   | Projektauftrag .....                                 | 15 |
| 3.2   | Zieleplan .....                                      | 17 |
| 3.3   | Projektstrukturplan .....                            | 18 |
| 3.4   | Objektstrukturplan .....                             | 20 |
| 3.5   | Projektumweltanalyse .....                           | 21 |
| 4     | Planung und Evaluierung .....                        | 22 |
| 4.1   | Anforderungen .....                                  | 22 |
| 4.2   | Technologische Entscheidungen .....                  | 22 |
| 4.2.1 | Backend .....  | 22 |
| 4.2.2 | Frontend .....                                       | 24 |
| 4.2.3 | Datenbank .....                                      | 24 |
| 4.2.4 | Continuous Integration & Continuous Deployment ..... | 25 |
| 5     | Entwicklungsphase .....                              | 26 |
| 5.1   | Backend .....  | 27 |
| 5.1.1 | Dependency Injections .....                          | 27 |
| 5.1.2 | Generischer Konnektor zur Datenbank .....            | 28 |

---

|       |  |    |
|-------|--|----|
| 5.1.3 | Authentifizierung .....                      | 31 |
| 5.1.4 | Client Generierung mit NSwag .....           | 32 |
| 5.1.5 | PDF-Export .....                             | 33 |
| 5.2   | Frontend .....                               | 35 |
| 5.2.1 | Architektur des Frontends .....              | 35 |
| 5.2.2 | Administratoroberfläche .....                | 36 |
| 5.2.3 | Vereinsoberfläche .....                      | 41 |
| 5.2.4 | Login .....                                  | 52 |
| 5.3   | Deployment.....                              | 58 |
| 5.3.1 | Continuous Integration.....                  | 58 |
| 5.3.2 | Artefakte.....                               | 61 |
| 5.3.3 | Continuous Deployment.....                   | 62 |
| 5.3.4 | Hetzner Server.....                          | 63 |
| 5.3.5 | Docker.....                                  | 63 |
| 6     | Testphase.....                               | 67 |
| 6.1   | Testung durch Trainerinnen und Trainer ..... | 67 |
| 6.2   | Einsatz bei der Landesmeisterschaft.....     | 68 |
| 6.3   | Einsatz bei der Staatsmeisterschaft .....    | 68 |
| 7     | Fazit.....                                   | 69 |
| 8     | Technologien.....                            | 71 |
| 8.1   | C# .....                                     | 71 |
| 8.1.1 | Records.....                                 | 71 |
| 8.1.2 | Attribute.....                               | 71 |
| 8.2   | ASP .NET Core.....                           | 72 |
| 8.3   | MongoDB .....                                | 72 |
| 8.3.1 | GridFS.....                                  | 73 |
| 8.4   | TypeScript.....                              | 73 |
| 8.4.1 | Decorators .....                             | 73 |
| 8.5   | Angular .....                                | 74 |
| 8.5.1 | Komponenten.....                             | 74 |
| 8.5.2 | Direktiven.....                              | 74 |
| 8.5.3 | Services .....                               | 74 |
| 8.5.4 | Pipes .....                                  | 75 |
| 8.6   | Ionic .....                                  | 75 |
| 8.7   | SASS .....                                   | 75 |
| 8.8   | GitHub Actions.....                          | 76 |
| 8.9   | Nuke .....                                   | 76 |

---

|      |                             |    |
|------|-----------------------------|----|
| 8.10 | Docker .....                | 76 |
| 9    | Autorenverzeichnis.....     | 78 |
| 10   | Glossar .....               | 80 |
| 11   | Abkürzungsverzeichnis ..... | 83 |
| 12   | Abbildungsverzeichnis.....  | 86 |
| 13   | Literaturverzeichnis .....  | 89 |

# 1 Impressum

Folgende Personen waren an der Umsetzung des Projektes beteiligt waren oder haben das Projekt anderweitig unterstützt.

## 1.1 Projektteam

Das Projektteam hat sich direkt mit der Umsetzung des Projektes bzw. des Projektauftrages beschäftigt.

### 1.1.1 Michael Laimer

Michael Laimer übernahm die Rolle des Projektleiters und war für die Erstellung und Dokumentation des Backends zuständig sowie für die Planung, Einrichtung und Überwachung des Deployments.



Abbildung 1: Michael Laimer

### 1.1.2 Laurin Losert-Nachbaur

Laurin Losert-Nachbaur war für das Projektmanagement und somit für die Erstellung und Aktuellhaltung der Projektpläne zuständig. Außerdem übernahm er die Entwicklung und Dokumentierung des Frontends.



Abbildung 2: Laurin  
Losert-Nachbaur

## 1.2 Projektbetreuung

Die Projektbetreuung hat das Projektteam dazu aktiviert das Projekt bestmöglich umzusetzen.

### 1.2.1 Diethard Kaufmann

Für die Projektbetreuung war Herr Diethard Kaufmann zuständig. Er wirkte als Ansprechpartner auf der Seite der HTL Dornbirn.

Als Lehrer unterrichtet er Softwareentwicklung, Cloud Computing und Big Data, was ihn zu einer passenden Betreuungsperson für dieses Projekt machte.



Abbildung 3: Diethard Kaufmann

### 1.2.2 Bianca Franzoi

Auf der Vereinsseite war Bianca Franzoi für die Betreuung zuständig.

Bianca Franzoi ist Vizepräsidentin für das Team Turnen im technischen Komitee von European Gymnastics. Da Team Turnen aktuell nur in Europa ausgeübt wird, ist European Gymnastics das international maßgebende Gremium.<sup>1</sup>



Abbildung 4: Bianca Franzoi

<https://www.turnsport-austria.at/de/getpic/RMLnmMsB9AyY?.jpg>

## 1.3 Turnsport Austria

Turnsport Austria ist ein Dachverband für österreichische Turnvereine und wird momentan von rund 450 Vereinen mit insgesamt über 90.000 Mitgliedern vertreten. Von 1947 bis 2022 war der Verband unter dem Namen „Österreichischer Fachverband für Turnen“ bekannt.

Turnsport Austria ist der sechstgrößte Sportfachverband in Österreich und kann sich mit drei WM-Gold-Medaillen, einer Vielzahl von Weltcup-Medaillen und einem Gesamtweltcupsieg auszeichnen.

Vom Verband eingeschlossen werden die olympischen Sparten Kunstturnen, Rhythmische Gymnastik und Trampolinspringen, sowie die Sportarten Team Turnen, Parkour, Sportaerobic und Sportakrobatik.<sup>2</sup>



Abbildung 5: Logo Turnsport Austria  
<https://www.turnsport-austria.at/de/getpic/KSQMvkQ1wQG66?.jpg>

<sup>1</sup> vgl. *Bianca Franzoi jetzt Europa-Vizepräsidentin für Team-Turnen!*, o. J.

<sup>2</sup> vgl. *Turnsport Austria / Wir über uns*, o. J.

## 2 Team-Turnen

Team-Turnen ist eine Mannschaftssportart, bei der Teams von vier bis zwölf Turnerinnen und Turner jeden Alters zusammen in drei Disziplinen (auch Geräte genannt) antreten.

Team Turnen entspringt ursprünglich aus Skandinavien und gewinnt dort seit den 1970er Jahren an immer mehr Aufmerksamkeit. In den nordischen Ländern ist das Team Turnen noch beliebter als das olympische Kunstturnen und die Rhythmische Gymnastik.<sup>3</sup>

### 2.1 Geräte

Zu den drei Disziplinen beziehungsweise (bzw.) Geräten des Team Turnens gehören Boden, Minitrampolin und Tumbling. Im folgenden Kapitel wird auf diese genauer eingegangen.

#### 2.1.1 Boden

Bei der Disziplin Boden wird zu Instrumentalmusik eine gymnastische Übung geturnt, die zwischen einer Minute und zwei Minuten und 45 Sekunden dauert. Diese Übung wird auf einer 14m x 16m großen Fläche, die mit Matten ausgelegt ist, geturnt. Die Bodenübung beinhaltet dabei verschiedene „Elemente“. **Elemente** sind zum Beispiel eine Pirouette oder ein Spagat. Jedes **Element** hat eine bestimmte Schwierigkeit und umso schwerer das **Element** ist, umso mehr Punkte gibt es bei einer erfolgreichen Durchführung.<sup>4</sup>



Abbildung 6: Turnübung am Boden

[https://tszdornbirn.at/wp-content/gallery/team-gym-cup-2021-runde-2/TGC\\_r2\\_21\\_055.jpg](https://tszdornbirn.at/wp-content/gallery/team-gym-cup-2021-runde-2/TGC_r2_21_055.jpg)

---

<sup>3</sup> vgl. *Turnsport Austria—Team-Turnen*, o. J.

<sup>4</sup> vgl. *Turnsport Austria—Team-Turnen*, o. J.

## 2.1.2 Minitrampolin

Beim Minitrampolin bzw. Trampolin springen die Turnerinnen und Turner eines Teams nacheinander **Elemente**. Dazu wird auf einer Bahn Anlauf bis zum Trampolin genommen und dann in einem Sprung das **Element** durchgeführt. Hinter dem Trampolin liegt eine Matte für die Landung. Es gibt dabei drei Runden, in denen jeweils bis zu sechs Turnerinnen und Turner immer ein **Element** zeigen. In der ersten Runde (diese wird auch **Einheitsrunde** genannt) springen alle Turnerinnen und Turner dasselbe **Element**. In den darauffolgenden Runden können verschiedenste **Elemente** gesprungen werden, wobei leichtere **Elemente** zuerst und die schwierigeren am Schluss gesprungen werden.<sup>5</sup>



Abbildung 7: Turnübung am Trampolin  
[https://tszdornbirn.at/wp-content/TT-LM\\_r2\\_21\\_012.jpg](https://tszdornbirn.at/wp-content/TT-LM_r2_21_012.jpg)

## 2.1.3 Tumbling

Beim Tumbling bzw. Tumbling wird mit Anlauf auf einer Tumblingbahn beliebig viele **Elemente** in einer sogenannten „Serie“ präsentiert. Unabhängig davon, wie viele **Elemente** geturnt werden, werden nur die zwei schwierigsten **Elemente** gewertet. Die erste Runde ist eine **Einheitsrunde**, wie auch beim Trampolin. Darüber hinaus müssen in den anderen zwei Runden die **Serien** mit aufsteigender Schwierigkeit geturnt werden.



Abbildung 8: Turnübung am Tumbling  
[https://tszdornbirn.at/wp-content/gallery/team-gym-cup-2021-runde-2/IGC\\_r2\\_21\\_101.jpg](https://tszdornbirn.at/wp-content/gallery/team-gym-cup-2021-runde-2/IGC_r2_21_101.jpg)

---

<sup>5</sup> vgl. *Turnsport Austria—Team-Turnen*, o. J.

## 2.2 Wertungsblatt

Die turnerische Ausführung der **Elemente** wird von einem Wertungsgericht bewertet, welches aus sechs bzw. am Boden aus acht Wertungsrichterinnen und Wertungsrichtern besteht. Jede Wertungsrichterin und jeder Wertungsrichter bewertet hierbei einen anderen Aspekt der Übung. Dazu haben die Wertungsrichterinnen und Wertungsrichter jeweils ein Wertungsblatt zur Hand, auf dem definiert ist, welche **Elemente** geturnt werden und in welcher Reihenfolge dies passiert. Diese Wertungsblätter werden von den Trainerinnen und Trainern einer Mannschaft vor dem Wettkampf ausgefüllt.



Abbildung 9: Wertungsrichterinnen beim Bewerten einer Übung  
[https://tszdornbirn.at/wp-content/gallery/teamgym-international-competition-round-3/TGC\\_r3\\_23\\_006.jpg](https://tszdornbirn.at/wp-content/gallery/teamgym-international-competition-round-3/TGC_r3_23_006.jpg)

### 2.2.1 Boden

Am Boden wird das Wertungsblatt in unterschiedliche Formationen aufgeteilt. Dabei ist eine Formation eine Abbildung der Positionen der Turnerinnen und Turner zu einem bestimmten Zeitpunkt in der Übung. Jeder Formation können bestimmte **Elemente** oder Kompositionen zugeordnet werden, welche in bzw. aus dieser Formation heraus geturnt werden.<sup>6</sup> Die der Formation zugeordneten **Elemente** werden in der Spalte „Code“ durch den Elementencode und in der Spalte „Symbol“ durch das Elementsymbol definiert. In der Spalte „Komp.“ werden die Kompositionen, die dieser Formation zugeordnet sind, definiert.

| Tariff Form - Floor   |        |        |       |                  |
|---|--------|--------|-------|------------------|
| Team: TSZ Dornbirn Jun w  |        |        |       | Start no.: 29    |
| Formation   | Code   | Symbol | Komp. | Deductions/Notes |
| •   | HB604  | ↷      | LF    |                  |
| •   | HB1001 |        |       |                  |
| •   | F1003  | ↔-     |       |                  |
| •   | J1009A | ✗/     |       |                  |
| •   | A1013  |        |       |                  |
| •   | J1010  | ↔*     |       |                  |
| •   | JB17   |        |       |                  |
| •   | G1001  |        |       |                  |
| •   | SB806  | Y      | CF    |                  |
| •   | A601   | ⊗      |       |                  |
| D-Elements Check:   |        |        |       |                  |
| HB604<br>HB1001<br>F1003<br>J1009A<br>A1013<br>J1010<br>JB17<br>G1001<br>SB806<br>A601                                |        |        |       |                  |
| In correct Order  |        |        |       |                  |
| D-Elements F/H/N  |        |        |       |                  |
| 1. HB604<br>2. HB1001<br>3. F1003<br>4. J1009A<br>5. A1013<br>6. J1010<br>7. JB17<br>8. G1001<br>9. SB806<br>10. A601 |        |        |       |                  |
| D-Score:  |        |        |       |                  |
| E-Score:  |        |        |       |                  |
| C-Score:  |        |        |       |                  |
| Final Score:  |        |        |       |                  |

Abbildung 10: Wertungsblatt am Boden

<sup>6</sup> vgl. *TeamGym Code of Points V1.1, 2022*, S. 41.

## 2.2.2 Minitrampolin

Das Wertungsblatt am Trampolin wird in drei unterschiedliche Runden unterteilt. Dabei wird die erste Runde als ***Einheitsrunde*** bezeichnet, bei der alle Turnerinnen und Turner dasselbe ***Element*** turnen. In den anderen zwei Runden werden ***Elemente*** in aufsteigender Schwierigkeit geturnt. Die ***Elemente*** werden hierbei durch ihr gezeichnetes Symbol in der ersten Spalte dargestellt. Der Schwierigkeitswert des ***Elements*** wird in der zweiten Spalte mit zwei Nachkommastellen abgebildet. Am Ende jeder Runde wird die Summe der Schwierigkeitswerte eingefügt.

|  |                              |  |
|--|------------------------------|--|
|  | <b>Tariff Form - Trampet</b> |  |
| Team: TSZ Dornbirn Jun w   | Start no.: 29                |  |
| <b>Round 1</b>   |                              |  |
| 5:40   | D: 3.00                      |  |
|  |                              |  |
| <b>Round 2</b>   |                              |  |
| 1 ⚪ 770  | 0.60                         |  |
| 2 ⚪ 0  | 0.60                         |  |
| 3 ⚪ 0  | 0.60                         |  |
| 4 ⚪ 0  | 0.60                         |  |
| 5 VODV180  | 0.80                         |  |
| 6 VODV180  | 0.80                         |  |
|  | D: 4.00                      |  |
|  |                              |  |
| <b>Round 3</b>   |                              |  |
| 1 TSU  | 0.80                         |  |
| 2 TSU  | 0.80                         |  |
| 3 TSU  | 0.80                         |  |
| 4 TSU  | 0.80                         |  |
| 5 TSU V  | 0.90                         |  |
| 6 TSU V  | 0.90                         |  |
|  | D: 5.00                      |  |

*Abbildung 11: Wertungsblatt am Trampolin*

### 2.2.3 Tumbling

Der Aufbau des Wertungsblattes am Tumbling ähnelt demjenigen vom Trampolin. Der einzige Unterschied hierbei ist, dass für eine Turnerin bzw. einen Turner mehrere **Elemente** pro Runde definiert werden. Es werden also mehrere **Elemente** als Serie angegeben. In der zweiten Spalte jeder Runde ist der Schwierigkeitswert der beiden schwierigsten **Elemente** dargestellt. Die Summe der Schwierigkeitswerte aller Turnerinnen und Turner einer Runde wird am Ende der Rundentabelle eingefügt.

## Tariff Form - Tumble

Team: TSZ Dornbirn Jun w

Start no.: 29

### Round 1

D: 4.20

### Round 2

|   |  |     |
|---|--|-----|
| 1 |  | 360 |
| 2 |  | 360 |
| 3 |  | 360 |
| 4 |  | 360 |
| 5 |  | 360 |
| 6 |  | 360 |

D: 3.40

### Round 3

|   |  |     |
|---|--|-----|
| 1 |  | 360 |
| 2 |  | 360 |
| 3 |  | 360 |
| 4 |  | 720 |
| 5 |  | 720 |
| 6 |  | 720 |

D: 4.80

Created on 03/11/2024 at 20:50

e-Judge

*Abbildung 12: Wertungsblatt am Tumbling*

### **3 Projektmanagement**

Die im folgenden Abschnitt angeführten Projektpläne wurden am Anfang des Projektes erstellt und gegebenen Falls im Laufe des Projektes angepasst bzw. abgeändert.

Die Projektpläne sind essenziell bei der Planung des Projektes und helfen, die einzelnen Phasen der Diplomarbeit zeitlich einzuhalten.

#### **3.1 Projektauftrag**

Im Projektauftrag (siehe Abbildung 13) wird der genaue Umfang des Projektes festgehalten und die wichtigsten Aspekte zusammengefasst. Daraus abzulesen ist, dass das Projektstartereignis die Vorpräsentation des Themas der Diplomarbeit am 30.06.2023 in der HTL Dornbirn war. Das Projekt an sich hat aber erst mit dem 11.09.2023 gestartet, dies war zugleich der Starttermin für die fünfte Klasse. Der Abgabetermin für die Diplomarbeit fällt auf den 03.04.2024. Mit diesem Datum ist auch das Projekt beendet. Das inhaltliche Ende der Diplomarbeit ist mit der Abgabe der Dokumentation gekennzeichnet. Der formale Abschluss der Diplomarbeit findet am Tag der Präsentation und dem Verteidigungsgespräch statt.

| eJudge   |   | PROJEKT-AUFTAG        |  |                       |               |                  |          |   |             |  |
|--|---|-----------------------|--|-----------------------|---------------|------------------|----------|---|-------------|--|
| <b>Projektstartereignis:</b>   | <ul style="list-style-type: none"> <li>Vorpräsentation am 30.06.2023</li> </ul>   |                       |  |                       |               |                  |          |   |             |  |
| <b>Projektstarttermin:</b>   | <ul style="list-style-type: none"> <li>11.09.2023</li> </ul>  |                       |  |                       |               |                  |          |   |             |  |
| <b>Inhaltliches Projektendereignis:</b>  | <ul style="list-style-type: none"> <li>Projekt &amp; Dokumentation abgeschlossen</li> <li>Erste Verwendung bei einem Wettkampf</li> </ul>   |                       |  |                       |               |                  |          |   |             |  |
| <b>Formales Projektendereignis:</b>  | <ul style="list-style-type: none"> <li>Präsentation der Diplomarbeit</li> </ul>   |                       |  |                       |               |                  |          |   |             |  |
| <b>Projektziele:</b>   | <ul style="list-style-type: none"> <li>Erstellen von Wertungsblätter</li> <li>Eingabeportal für Vereine</li> <li>Admin-Oberfläche für Verwaltung</li> </ul>   |                       |  |                       |               |                  |          |   |             |  |
| <b>Nicht-Projektziele:</b>   | <ul style="list-style-type: none"> <li>Noteneingabe</li> <li>Notenauswertung</li> </ul>   |                       |  |                       |               |                  |          |   |             |  |
| <b>Hauptaufgaben (Projektphasen):</b>  | <ul style="list-style-type: none"> <li>Projektmanagement</li> <li>Vorbereitung</li> <li>Durchführung</li> <li>Fehlersuche- und Behebung</li> </ul>  |                       |  |                       |               |                  |          |   |             |  |
| <b>ProjektauftraggeberIn:</b>  | <b>ProjektleiterIn:</b> <ul style="list-style-type: none"> <li>Bianca Franzoi (TS Austria)</li> </ul>   |                       |  |                       |               |                  |          |   |             |  |
| <b>Projektressourcen und -kosten*:</b>   | <table border="1"> <thead> <tr> <th>Ressourcen-/Kostenart</th> <th>Mengeneinheit</th> <th>Kosten (in Euro)</th> </tr> </thead> <tbody> <tr> <td>V-Server</td> <td>1</td> <td>4 € / Monat</td> </tr> </tbody> </table> |                       |  | Ressourcen-/Kostenart | Mengeneinheit | Kosten (in Euro) | V-Server | 1 | 4 € / Monat |  |
| Ressourcen-/Kostenart  | Mengeneinheit   | Kosten (in Euro)      |  |                       |               |                  |          |   |             |  |
| V-Server   | 1   | 4 € / Monat           |  |                       |               |                  |          |   |             |  |
| <b>Projektteam:</b>  | <ul style="list-style-type: none"> <li>Laurin Losert-Nachbaur</li> </ul>  |                       |  |                       |               |                  |          |   |             |  |
| <i>Bianca Franzoi</i>  |   | <i>Michael Laimer</i> |  |                       |               |                  |          |   |             |  |
| Bianca Franzoi, (ProjektauftraggeberIn)      Michael Laimer, (ProjektleiterIn) |   |                       |  |                       |               |                  |          |   |             |  |

Abbildung 13: Projektauftrag

## 3.2 Zieleplan

Im Zieleplan ist genau festgehalten, was Teil der Diplomarbeit ist und was nicht. Dabei werden die Anforderungen des Projektes genau festgehalten und in erreichbare Ziele umgeformt. Diese halten dann den Rahmen des Projektes fest.

Hauptziele sind Ziele, die erreicht werden müssen, wenn das Projekt erfolgreich abgeschlossen werden soll. Zusatzziele müssen nicht unbedingt für einen erfolgreichen Projektabschluss erreicht werden, stellen jedoch mögliche Weiterentwicklungen für das Projekt dar. Nichtziele sind dazu da, das Projekt noch gründlicher einzugrenzen, bzw. Teile, die nicht Thema der Diplomarbeit sind, auszugrenzen.

Im Fall der vorliegenden Diplomarbeit wurden als Hauptziele die Bestandteile gewählt, welche die Applikation aufweisen muss, damit sie funktionieren kann und bei Wettkämpfen eingesetzt werden kann. Aus Abbildung 14 sind alle Hauptziele, Zusatzziele und Nicht-Ziele abzulesen.

| <b>Hauptziele</b>   | <b>Adaptierte Hauptziele</b> |
|---|------------------------------|
| Funktionierende Webapplikation mit Admin-panel zur Eingabe von Wertungsblättern |                              |
| Eingabe von Vereinen und deren Teams  |                              |
| Eingabe von Elementen für Trampolin, Tumbling und Boden                         |                              |
| Eingabe der Wertungsblätter durch Vereine am Boden, Tumbling und Trampolin      |                              |
| <b>Zusatzziele</b>  |                              |
| Testeinsatz bei einem Wettkampf   |                              |
| Zeitplanung von Wettkämpfen   |                              |
| <b>Nicht-Ziele</b>  |                              |
| Eingabe und Auswertung von Noten  |                              |
| Anmeldungen für Wettkämpfe  |                              |

Abbildung 14: Zieleplan

### 3.3 Projektstrukturplan

Der Projektstrukturplan gibt eine Übersicht über das Projekt. Das Projekt wird dabei in verschiedene Phasen geteilt, welche wiederum Arbeitspakete und Meilensteine enthalten. Der Projektstrukturplan erlaubt eine einfachere Zeitplanung bzw. Terminplanung und deren Umsetzung, in dem die einzelnen Arbeitspakete nacheinander abgearbeitet werden.

Die grundlegenden Phasen des Projektes wurden am Anfang der Diplomarbeit festgelegt. Die genauere Aufteilung der Arbeitspakete wurde erst nach der Planung des Projektes gemacht, als die zu verwendenden Technologien bereits feststanden. Die Meilensteine haben bei der Durchführung der Diplomarbeit einen guten Überblick über den Fortschritt verschafft und zur Einhaltung der Termine beigetragen.<sup>7</sup>

---

<sup>7</sup> vgl. BET-PM 08 PM Projektplanung FRB 2021.pdf, o. J.

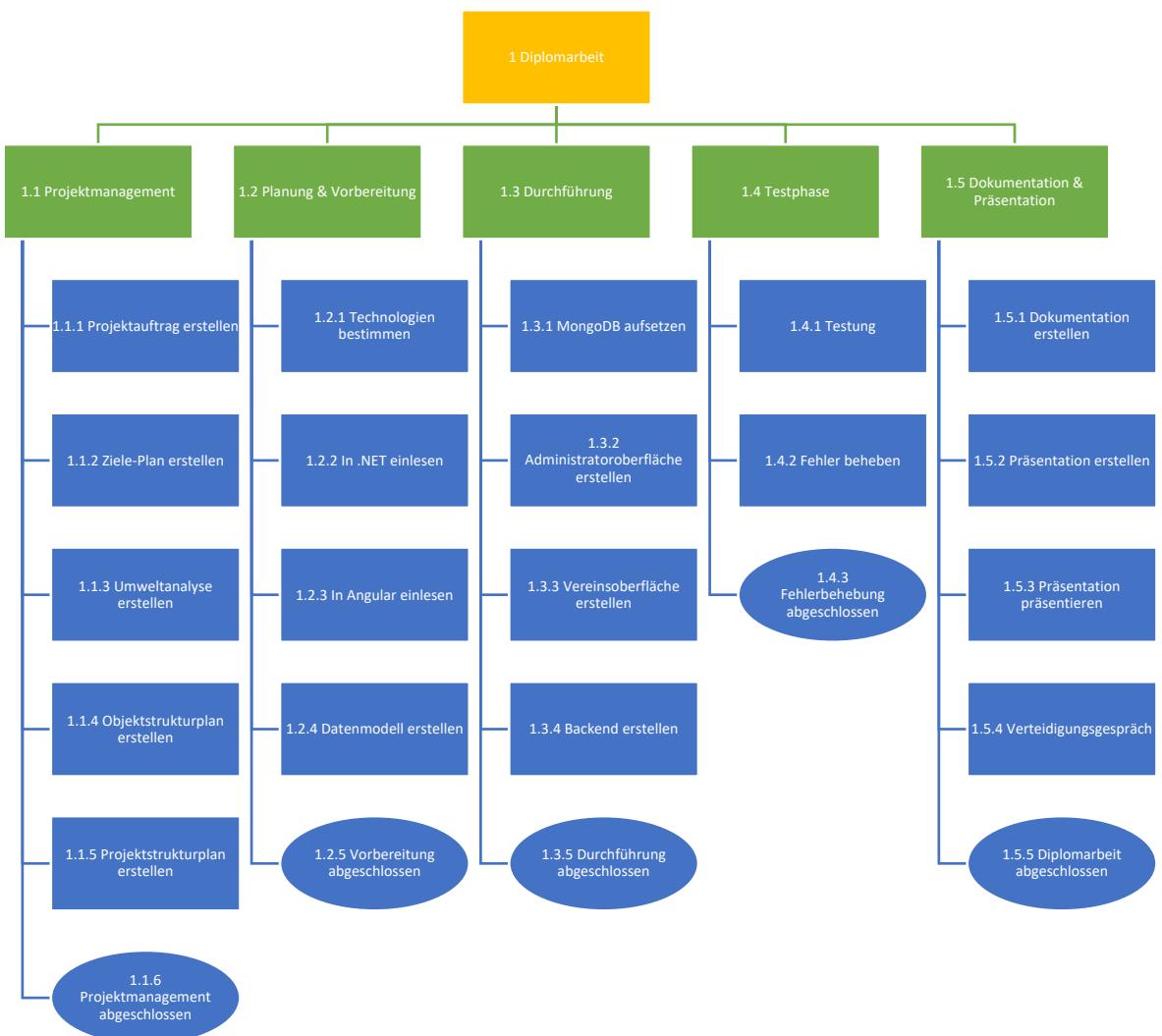


Abbildung 15: Projektstrukturplan

### Legende

Projekt

Projektphase

Arbeitspaket

Meilenstein

### 3.4 Objektstrukturplan

Der Objektstrukturplan ist dem Projektstrukturplan sehr ähnlich. Der Unterschied liegt darin, dass das Projekt vom Projektstrukturplan organisatorisch aufgeteilt wird, während der Objektstrukturplan das Projekt fachlich bzw. technisch aufteilt. Der Objektstrukturplan hilft dabei, das Projekt in greifbare, zu erstellende Objekte aufzuteilen.

Die Diplomarbeit wird dabei in die Administratoroberfläche und die Vereinsoberfläche geteilt, welche wiederum in die fachlich/technisch wichtigen Objekte geteilt werden.<sup>8</sup>

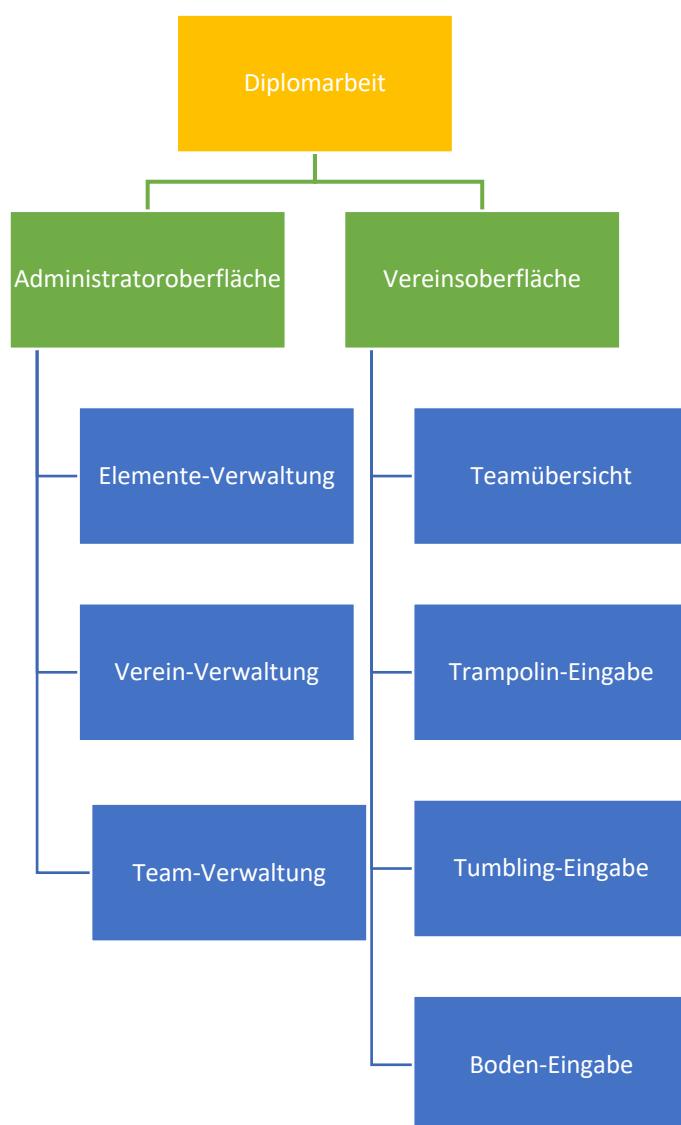


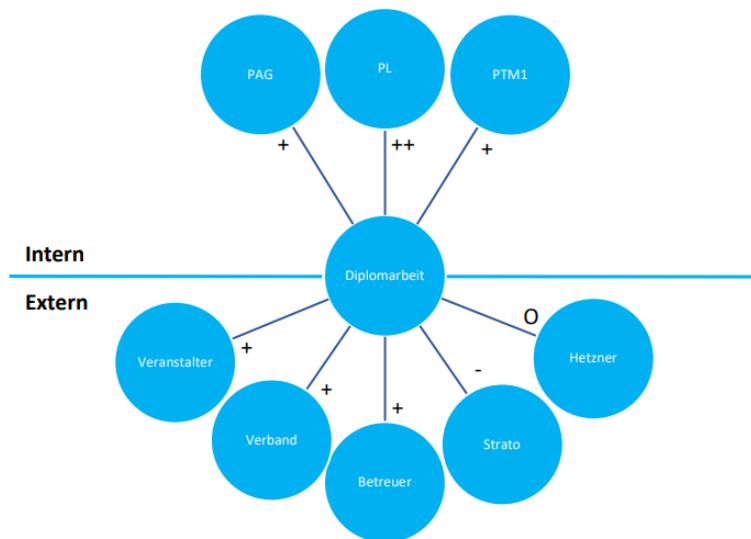
Abbildung 16: Objektstrukturplan

---

<sup>8</sup> vgl. BET-PM 08 PM Projektplanung FRB 2021.pdf, o. J.

### 3.5 Projektumweltanalyse

Die Projektumweltanalyse stellt die relevanten internen und externen Beziehungen für ein Projekt graphisch dar und bewertet diese. Mit der Umweltanalyse und den dazugehörigen Bewertungen können Probleme in Beziehungen festgestellt werden, wodurch dann notwendige Maßnahmen getroffen werden können, um diese zu lösen. Die Beziehungen mit den zugehörigen Maßnahmen werden für einen besseren Überblick tabellarisch zur Grafik beigelegt. Im Falle der vorliegenden Diplomarbeit gibt es keine relevanten Probleme im Zusammenhang mit den Beziehungen. Die Beziehung mit Strato wird als eher negativ dargestellt, weil der virtuelle Server anfangs auch über Strato gelaufen ist, dort Probleme aufgetreten sind. Deshalb wurde ist der aktuelle Server bei Hetzner gemietet, sodass nur noch die Domain über Strato läuft.<sup>9</sup>



| Bezeichnung  | Umwelt                       | Beziehung | Maßnahmen                     |
|--------------|------------------------------|-----------|-------------------------------|
| PAG          | Franzoi Bianca (TS Austria)  | +         | Vereinfachung von Wettkämpfen |
| Betreuer     | Kaufmann Diethard            | +         | Zufriedene Schüler            |
| PL           | Laimer Michael               | ++        | -                             |
| PTM1         | Losert-Nachbaur Laurin       | +         | Motivation durch gute Note    |
| -            | Hetzner (Server)             | O         | -                             |
| -            | Strato (Domain)              | -         | -                             |
| Veranstalter | Veranstalter des Wettkampfes | +         | Planung vereinfacht           |
| Verband      | Nationaler Turn Verband      | +         | Vereinfachung von Wettkämpfen |

Abbildung 17: Projektumweltanalyse

<sup>9</sup> vgl. Frankenhauser, 2021.

## 4 Planung und Evaluierung

In dieser Phase der Diplomarbeit wurden, die genauen Anforderungen des Auftraggebers festgehalten und analysiert. Anhand der Anforderungen wurde dann geplant, wie das Projekt umgesetzt werden soll. Bevor mit dem Projekt gestartet werden konnte, mussten Entscheidungen über die zu verwendenden Technologien getroffen werden.

### 4.1 Anforderungen

Die Anforderungen für die Applikation von Seiten des Turnsport Austria waren grundsätzlich nur fachlich bezogen. Die technische Umsetzung wurde gänzlich dem Projektteam überlassen. Die Anwendung sollte jedoch möglichst benutzerfreundlich sein und den Trainerinnen und Trainer der Vereine ein müheloses Eintragen der Wertungsblätter erlauben.

### 4.2 Technologische Entscheidungen

In den folgenden Kapiteln werden die technologischen Entscheidungen, die für die Ausarbeitung der Diplomarbeit genutzt wurden, beschrieben.

#### 4.2.1 Backend

Im Backend wurde ASP .NET Core als Framework verwendet. Die starke Typisierung der Variablen in C#, sowie die Möglichkeit zur genauen Definition des Datenmodells und der **End-points**, machten ASP .NET Core zu einer passenden Technologie zur Entwicklung des Backends. ASP .NET Core kommt außerdem mit einer Reihe an Zusatzbibliotheken, welche sich nahtlos in das Grundgerüst der API integrieren.

##### 4.2.1.1 Authentifizierung

Zur Authentifizierung wurde, die mit ASP .NET Core mitgelieferte Authentifizierungs-**Bibliothek** verwendet. Die einfache Implementierung, sowie die Sicherheitsanforderungen in diesem sensiblen Bereich der API sind Gründe für die Verwendung der integrierten Authentifizierungs-**Bibliothek**. Außerdem bietet diese **Bibliothek** eine Reihe an Anpassungsmöglichkeiten

und generischen Funktionen, was für die Architektur mit zwei Frontends von großem Vorteil war.

#### 4.2.1.2 PDF-Creator

Bei einem Wettkampf werden die Wertungsblätter in ausgedruckter Form benötigt. Um dies zu erreichen, müssen die Wertungsblätter in einem PDF-Format exportiert werden. Für dieses Vorhaben gibt es für ASP .NET Core mehrere **Bibliotheken**, welche zur Verfügung standen und im Folgenden näher erläutert werden.

##### 4.2.1.2.1 IronPDF

IronPDF ist eine umfangreiche ASP .NET Core-**Bibliothek**, welche es dem Entwickler erlaubt, HTML-Strings performant in PDF-Dateien umzuwandeln. Sie ermöglicht außerdem noch weitere Konvertierungen, wie ASPX zu PDF, XAML zu PDF, etc. Die Konvertierung ist sehr präzise, da jegliche **CSS**-Styles, unabhängig von ihrer Komplexität, übernommen werden. Allerdings ist die **Bibliothek** nicht kostenlos, sondern muss lizenziert werden. Aufgrund dessen und der erheblichen Vergrößerung des **Build**-Artefakts durch den Umfang der **Bibliothek** wurde von ihrer Verwendung abgesehen.<sup>10</sup>

##### 4.2.1.2.2 SelectPDF

Bei SelectPDF handelt es sich, ähnlich wie bei IronPDF, um einen HTML zu PDF-Konverter. Im Gegensatz zu IronPDF ist SelectPDF allerdings kostenlos.

Die **Bibliothek** setzt auf einen Aufruf einer API zur Generierung der PDF-Datei. Ein erheblicher Nachteil der **Bibliothek** ist, dass die **Bibliothek** nicht plattformübergreifend verfügbar ist, sondern nur auf Windows funktioniert.<sup>11</sup>

##### 4.2.1.2.3 MigraDocCore

MigraDocCore ist eine Weiterentwicklung der Open Source **Bibliothek** MigraDoc und unterscheidet sich hauptsächlich in ihrer plattformübergreifenden Verfügbarkeit von ihr. Bei MigraDoc wird der Inhalt eines PDF-Dokuments zuerst durch .NET-Code erstellt und anschließend in

---

<sup>10</sup> vgl. *C# PDF Library (All-in-One Solution) / IronPDF for .NET*, o. J.

<sup>11</sup> vgl. „*Pdf Library for .NET | Free Html To Pdf Converter*“, o. J.

eine Datei oder einen Memory-Stream gerendert. Die Erstellung von PDFs mag zuerst aufwendig erscheinen, allerdings lässt sich durch die statische Erstellung des PDFs durch Programm-codes ein verlässliches Layout erstellen, welches in Bezug auf die Unterstützung gewisser **CSS**-Eigenschaften keinerlei Nachteile mit sich bringt.<sup>12</sup>

### 4.2.2 Frontend

Für die Entwicklung des Frontends wurde Ionic mit **Angular** als Framework gewählt, wodurch die Auswahl der Programmiersprache automatisch auf TypeScript gefallen ist, da **Angular** nur mit TypeScript funktioniert.

Ionic ist ein hochwertiges Framework für die Entwicklung von Webapplikationen und weist eine große Auswahl an vorgefertigten **Komponenten** auf, was die Gestaltung der Applikation erleichtert.<sup>13</sup> Zudem hat Ionic eine schlichte Icon-**Bibliothek**, welche die wichtigsten Icons enthält.

Mit Ionic zusammen wurde außerdem **Angular** verwendet. **Angular** ist ein von Google gefertigtes Framework, welches eine übersichtliche Entwicklung mit TypeScript erlaubt. Die **Komponenten** werden dabei in Ordner unterteilt und der Code wird nach Funktionalität in verschiedene Dateien getrennt.

Zusätzlich wurde für eine überschaubare Ausarbeitung der Stylesheets **SASS** verwendet. **SASS** ist ein **Superset** von **CSS** und bietet verschiedene Syntax Verbesserungen und Erweiterungen um das Erstellen von Stylesheets zu erleichtern.

### 4.2.3 Datenbank

Als Datenbank wurde für die gesamte Applikation MongoDB ausgewählt. Diese Entscheidung wurde getroffen, da die Speicherung in Dokumenten eine angenehme Datenverwaltung und -aggregierung ermöglicht. Die redundante Speicherung von Daten in der MongoDB stellte hierbei kein großes Problem dar, da in diesem Anwendungsfall Daten fast nie gelöscht werden

---

<sup>12</sup> vgl. Ststeiger/PdfSharpCore: Port of the PdfSharp library to .NET Core—Largely removed GDI+ (only missing GetFontData—which can be replaced with freetype2), o. J.

<sup>13</sup> vgl. Ionic Framework—The Cross-Platform App Development Leader, o. J.

mussten. Der, mit MongoDB mitgelieferte, Konnektor erlaubt außerdem ein einfaches Abrufen und Abspeichern von Dokumenten.

#### 4.2.4 Continuous Integration & Continuous Delivery

Als Versionskontrollsystem wurde **Github** ausgewählt. Folglich wurde für die Continuous Integration **Github** Actions verwendet. Die nahtlose Integration von **Github** Actions in das Versionskontrollsoftware Git macht **Github** Actions zur passenden Wahl.

Als **Build** Tool wurde Nuke gewählt, da es, genauso wie das Backend, in C# programmiert werden kann und außerdem mehrere vorgefertigte Funktionen zur Ausführung des **Builds** mit **dotnet** und **NodeJS** implementiert.

## 5 Entwicklungsphase

Im folgenden Kapitel erfolgt eine detaillierte Beschreibung des Entwicklungsprozesses der Applikation, sowie eine genaue Darstellung dieser.

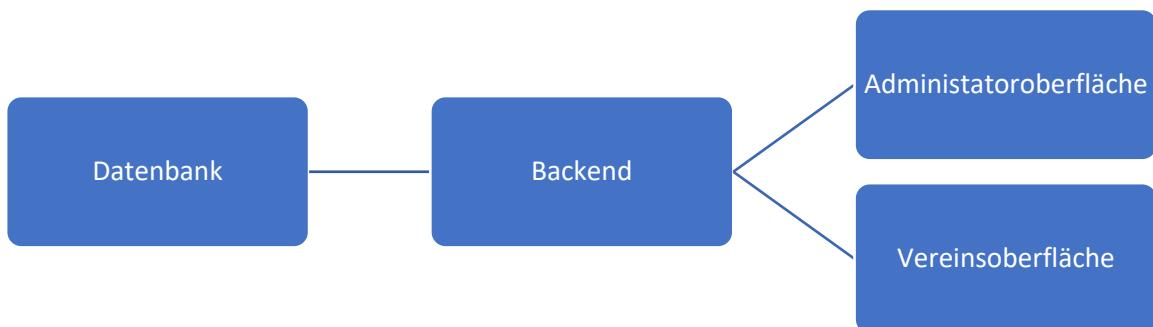


Abbildung 18: Übersicht Applikation

In Abbildung 18 ist die grundlegende Struktur der Applikation zu erkennen. Es gibt eine Datenbank, in welcher die Daten gespeichert werden. Das Backend kommuniziert mit der Datenbank, um die gespeicherten Daten aufzurufen, oder, um neue Daten abzuspeichern. Das Backend kommuniziert aber gleichzeitig auch mit den zwei Frontends, um auf die Eingaben von Benutzerinnen und Benutzern zu reagieren.

## 5.1 Backend

Im Folgenden wird die Entwicklung und Programmierung des Backends beschrieben. Die Projektmappe besteht hierbei aus vier verschiedenen Einzelprojekten:

| Name                      | Beschreibung  |
|---------------------------|---|
| eJ.Backend.Web            | Beinhaltet die API sowie die einzelnen <b>Endpoints</b> .   |
| eJ.Backend.Core           | Beinhaltet Datenmodelle, PDF-Erstellungs-Services sowie Services zum Aggregieren von Daten.                                       |
| eJ.Backend.Infrastructure | Beinhaltet den generischen Konnektor zur mongoDB Datenbank sowie Hilfsklassen zur generischen Auflösung von Referenzen (siehe 0). |
| eJ.Backend.SharedKernel   | Beinhaltet Interfaces, welche von allen 3 Projekten benötigt werden.  |

Das Projekt eJ.Backend.Web stellt hierbei das Startprojekt dar, in welchem auch die Konfiguration der API und Dependency Injections vorgenommen werden.

### 5.1.1 Dependency Injections

Um Abhängigkeiten zu injizieren, wird in der Program.cs Datei definiert, dass bestimmte Klassen mit bestimmten Interfaces in die **Konstruktoren** von anderen Klassen injiziert werden können. Dabei wird der „AddScoped“-Funktion das Interface und die zu injizierende Klasse übergeben. Das Interface muss dabei alle Funktionen enthalten, die in der fremden Klasse verwendet werden sollen.



```
1 builder.Services.AddScoped(typeof(IUserCredentialsVerification), typeof(UserCredentialsVerification));
```

Abbildung 19: Dependency Injection

In Abbildung 19 ist zu sehen wie eine Klasse namens „UserCredentialsVerification“ injiziert wird. Dabei wird der AddScoped-Funktion zuerst ein Interface, welches die Funktionen der Klasse definiert übergeben. Im zweiten Argument wird die zu injizierende Klasse der Funktion übergeben. Die übergebene Klasse erbt dabei vom im ersten Argument übergebenen Interface.

## 5.1.2 Generischer Konnektor zur Datenbank

Die Datenstruktur von Teams, Klassen und Wertungsblättern hat eine gewisse Komplexität. Insbesondere Referenzen zu anderen Typen sind aufgrund der fehlenden Relationalität von mongoDB nur durch zusätzliche Anfragen an den Server bzw. manuelles Auflösen der Referenzen im Backend möglich. Zur Behebung dieser ineffizienten Datenbeschaffung wurde ein generischer Konnektor zur Datenbank entwickelt, welcher zuvor in der Datenmodellstruktur definierte Referenzen auflöst (ähnlich zur Fremdschlüsselauflösung durch JOINS bei SQL-Datenbanken).

### 5.1.2.1 Referenztypen

Der generische Konnektor kann aktuell zwei Referenztypen behandeln. In Objekten verschachtelte sowie in Dictionaries verwendete Referenzen müssen nach wie vor manuell aufgelöst werden. Die referenzierten Objekte können alle durch ihre Vererbung vom Interface „IAggregateRoot“ vom Programmcode erkannt werden.

#### 5.1.2.1.1 Default

Beim Referenztyp **default** ist die referenzierte Property ein normales Objekt vom referenzierten Typ.

#### 5.1.2.1.2 Liste

Die referenzierte Property ist beim Referenztyp Liste ein **Array** (in C# eine Liste) aus einem oder mehreren Objekten vom referenzierten Typ. Der Typparameter der Liste erbt hierbei vom Interface IAggregateRoot.

#### 5.1.2.2 Initialisierung

Um den generischen Konnektor (im weiteren Text als MongoRepository referenziert) zu verwenden, werden im Vorhinein für jede Entität (z. B. Team) zwei Typdefinitionen (records) erstellt. Die erste definiert die Datenstruktur, die in der Datenbank abgebildet werden soll. In dieser Entität werden die Referenzen als IDs in Form von strings dargestellt. Die zweite Entität definiert das Dto, in welchem die Referenzen zu anderen Entitäten in Form deren Typs definiert werden. Beim Initialisieren des MongoRepositorys werden beide Typen durch Typparameter übergeben und vom **Konstruktor** die Referenzen in einer Liste abgespeichert. Hierbei wird die

in .NET integrierte **Bibliothek** Reflection verwendet, um Typen sowie deren Properties auf Runtimeebene zu untersuchen.

```
● ○ ●
1 var dtoProperties = typeof(TDto).GetProperties().Where(property => property.Name != "Events").ToList();
2 var foreignConstraints = new List<ForeignConstraint>();
3 for (int i = 0; i < dtoProperties.Count(); i++)
4 {
5     var proptype = dtoProperties[i].PropertyType;
6     var propinterface = proptype.GetInterface("IAggregateRoot");
7     if (propinterface is not null)
8     {
9         var constraint = new ForeignConstraint(dtoProperties[i], ConstraintType.Default, i, dtoProperties[i].PropertyType);
10        foreignConstraints.Add(constraint);
11    }
12    else if (proptype.IsGenericType && proptype.GetGenericTypeDefinition() == typeof(List<>))
13    {
14        var listType = proptype.GetGenericArguments()[0];
15        var listTypeInterface = listType.GetInterface("IAggregateRoot");
16        if (listTypeInterface is not null)
17        {
18            var constraint = new ForeignConstraint(dtoProperties[i], ConstraintType.List, i, listType);
19            foreignConstraints.Add(constraint);
20        }
21    }
22 }
```

Abbildung 20: Initialisierung des MongoRepositorys

Wie in Abbildung 20 ersichtlich ist, wird bei der Initialisierung des MongoRepositorys als erstes eine Liste aller Properties des Dtos erstellt. Anschließend wird bei jeder Property geprüft, ob sie vom Interface IAggregateRoot erbt, oder ob sie eine Liste von einem Typ, der von IAggregateRoot erbt, ist. Falls eine entsprechende Property gefunden wurde, wird ein neuer ForeignConstraint erstellt, welcher anschließend einer Liste mit weiteren ForeignConstraints hinzugefügt wird.

### 5.1.2.3 Abrufen von Daten

```

● ● ●

1 var result = await Collection
2   .FindAsyncBuilders<T>.Filter.Eq("_id", stringId, null, cancellationToken)
3   .ConfigureAwait(false);
4
5 var finalResult = result.FirstOrDefault(cancellationToken: cancellationToken);
6 if (finalResult == null)
7 {
8   return default;
9 }
10 var props = finalResult.GetType().GetProperties();
11 var propValues = props.Where(prop => prop.Name != "Events").ToList().Select(prop => prop.GetValue(finalResult, null)).ToList();
12 var idProp = propValues.Last();
13 propValues.RemoveAt(propValues.Count - 1);
14 propValues.Insert(0, idProp);
15 foreach (var foreignConstraint in foreignConstraints)
16 {
17   var property = finalResult.GetType().GetProperties().First(propertyIterator => propertyIterator.Name == foreignConstraint.property.Name);
18   var foreignKeys = property.GetValue(finalResult, null);
19   if (foreignKeys is not null)
20   {
21     var foreignRepoType = typeof(MongoRepository<,>).MakeGenericType(foreignConstraint.foreignType, foreignConstraint.foreignType);
22     dynamic? foreignRepo = Activator.CreateInstance(foreignRepoType, new object[] { _configuration });
23     if (foreignConstraint.constraintType == ConstraintType.Default)
24     {
25       var foreignValue = await foreignRepo?.GetByIdAsync(foreignKeys, cancellationToken);
26       propValues[foreignConstraint.index] = foreignValue;
27     }
28   else if (foreignConstraint.constraintType == ConstraintType.List)
29   {
30     var foreignType = foreignConstraint.property.PropertyType.GenericTypeArguments[0];
31     Type listType = typeof(List<>).MakeGenericType(foreignType);
32     dynamic? foreignValues = Activator.CreateInstance(listType);
33     var foreignKeysList = foreignKeys as List<string>;
34     if (foreignKeysList != null)
35     {
36       foreach (var foreignKey in foreignKeysList)
37     {
38       var foreignValue = await foreignRepo?.GetByIdAsync(foreignKey, cancellationToken);
39       foreignValues?.Add(foreignValue);
40     }
41     propValues[foreignConstraint.index] = foreignValues;
42   }
43 }
44 }
45 }
46 var returnDto = Activator.CreateInstance(typeof(TDto), propValues.ToArray());
47 if (returnDto != null)
48 {
49   return (TDto)returnDto;
50 }
51 else
52 {
53   return default;
54 }
55 }

```

Abbildung 21: Ablauf des Abrufens von Daten und Auflösung der Referenzen

In Abbildung 21 ist zu sehen, wie Daten aus der mongoDB abgerufen werden und anschließend die zuvor gespeicherten Referenzen (siehe 5.1.2.2) angewendet bzw. aufgelöst werden. Als erstes wird das in der mongoDB gespeicherte Objekt über den Standard mongoDB Konnektor geholt (Zeile 1-9). Die Werte der Properties werden in einem **Array** (im weiteren Text als Property **Array** referenziert) gespeichert (Zeile 10-14). Sie werden später für die Initialisierung des Dto Typs benötigt. Anschließend wird die zuvor erstellte Liste an Referenzen (siehe 5.1.2.2) schrittweise durchgegangen. Der Wert der Property von der zugehörigen aktuellen Iteration wird geholt (Zeile 17-18). Dieser Wert ist die ID des referenzierten Objekts. Danach wird in der Runtime ein neues MongoRepository initialisiert, welches als Typparameter den Typ der Michael Laimer, Laurin Losert-Nachbaur

referenzierten Entität hat (Zeile 21-22). Bei der Auflösung der Referenzen wird je nach Referenztyp unterschiedlich verfahren (siehe 5.1.2.3.1 bzw. 5.1.2.3.2). Nach der Auflösung aller Referenzen wird eine neue Instanz vom Typ des Dto erstellt. Das Property **Array** wird hierbei dem **Konstruktor** des Dto übergeben.

### 5.1.2.3.1 Auflösen von Default-Referenzen

Bei einer **Default** Referenz wird durch das neu initialisierte MongoRepository das Objekt anhand der vorher abgerufenen ID aus der Datenbank geholt (Zeile 25). Anschließend wird der Wert der Property im zuvor erstellten Property **Array** auf das neue Objekt geändert (Zeile 26).

### 5.1.2.3.2 Auflösen von Listen-Referenzen

Bei einer Referenz vom Typ „List“ besteht die referenzierende Property aus einer Liste von IDs. Die IDs werden schrittweise durchgegangen, die referenzierten Objekte geholt (siehe 5.1.2.3.1) und anschließend eine neue Liste mit den aufgelösten Referenzen erstellt (Zeile 30-40). Diese Liste wird anschließend an der entsprechenden Stelle in das zuvor erstellte Property **Array** eingefügt (Zeile 41).

## 5.1.3 Authentifizierung

Zur Authentifizierung wird die von .NET bereitgestellte Identity-**Bibliothek** verwendet. Die Authentifizierung erfolgt über einen **Bearer-Token**, welcher als HTTP-only Cookie an den „Client“ gesendet wird.

```
● ● ●
1 Response.Cookies.Append("X-eJudge-Token", jwtstring, new CookieOptions
2 {
3   HttpOnly = true,
4   SameSite = SameSiteMode.None,
5   Secure = true,
6   Expires = DateTime.UtcNow.AddDays(1)
7 });
```

Abbildung 22: Anfügen des Bearer Tokens als Cookie

### 5.1.3.1 Authentifizierungspolicies

Durch die Verwendung von zwei Frontends, welche beide auf dasselbe Backend zugreifen, werden zwei unterschiedliche Bearer-Signaturen benötigt. Somit muss auch zwischen den beiden **Bearer-Token** unterschieden werden, da gewisse **Endpoints** nur von der Administratoroberfläche und andere nur von der Vereinsoberfläche erreichbar sein dürfen.

| Policy       | Beschreibung   |
|--------------|--|
| OnlyFrontend | <b>Endpoints</b> mit dieser Policy können nur durch einen authentifizierten Benutzer des Admin Panels aufgerufen werden. Benutzer der Vereinsoberfläche bekommen den Fehlercode 401 (Unauthorized). Falls nicht anders definiert, wird diese Policy verwendet. |
| OnlyClubUnit | <b>Endpoints</b> mit dieser Policy können nur durch einen authentifizierten Benutzer der Vereinsoberfläche aufgerufen werden.  |
| AllPolicies  | <b>Endpoints</b> mit dieser Policy können von Benutzern der Administratoroberfläche als auch von der Vereinsoberfläche verwendet werden.   |

Die verwendete Policy wird im **Endpoint** mithilfe des Authorize-Attributs angegeben. Weiters werden **Endpoints**, welche keine Authentifizierung benötigen (z. B. der Login) mit dem „AllowAnonymous“ Attribut gekennzeichnet.



```

1  [HttpPost(LoginRequest.Route)]
2  [AllowAnonymous]
3  [SwaggerOperation(
4      Summary = "Endpoint for logging in",
5      Description = "Checks the User Credentials passed via POST",
6      OperationId = "User.Login",
7      Tags = new[] { Tag>LoginEndpoints })
8  ]

```

Abbildung 23: Verwendung des Attributs AllowAnonymous

### 5.1.4 Client Generierung mit NSwag

Mit NSwag kann aus den **Endpoints** und der daraus resultierenden swagger.json Definitionsdatei ein „Typescript-Client“ erstellt werden. Swagger erstellt bei jedem **Build** eine swagger.json Datei. In dieser sind alle Datenmodelle, sowie die **Endpoints** und deren Anfrageparameter definiert. Nach dem Kompilieren des Backends wird anhand der swagger.json-Datei eine client.ts Datei erstellt, welche alle in **Endpoints** vorkommenden Datenmodelle enthält und Funktionen für jeden **Endpoint** erstellt.

Diese Funktion muss im Frontend anschließend nur noch mit den benötigten Anfrageparametern aufgerufen werden, um die entsprechende HTTP-Anfrage zu senden.



Abbildung 24: Ablauf des Backend Build-Prozesses

## 5.1.5 PDF-Export

Vor dem Wettkampf müssen die Wertungsblätter als PDF exportiert werden. Dies geschieht in Klassen, welche für jedes Gerät angefertigt wurden.

### 5.1.5.1 Header

```

1  public void AddHeader()
2  {
3      var header = _documentSection.Headers.Primary;
4      var src = ImageSource.FromFile("./austria.png", 75);
5      var image = header.AddImage(src);
6      image.Height = 50;
7      image.WrapFormat.Style = WrapStyle.Through;
8      image.WrapFormat.DistanceTop = 0;
9      var heading = header.AddParagraph("Tariff Form - Floor");
10     heading.Format.SpaceBefore = 10;
11     heading.Format.Alignment = ParagraphAlignment.Center;
12     heading.Format.Font.Size = 14;
13     heading.Format.Font.Bold = true;
14     heading.Format.Font.Underline = Underline.Single;
15
16     var table = header.AddTable();
17     table.Format.SpaceBefore = 20;
18     table.TopPadding = 0;
19     table.BottomPadding = 0;
20     table.Format.Alignment = ParagraphAlignment.Left;
21
22     Column teamColumn = table.AddColumn();
23     teamColumn.Format.Alignment = ParagraphAlignment.Left;
24     teamColumn.Width = (_document.DefaultPageSetup.PageWidth - _document.DefaultPageSetup.LeftMargin - _document.DefaultPageSetup.RightMargin) / 2;
25
26     Column startnumberColumn = table.AddColumn();
27     startnumberColumn.Format.Alignment = ParagraphAlignment.Right;
28     startnumberColumn.Width = (_document.DefaultPageSetup.PageWidth - _document.DefaultPageSetup.LeftMargin - _document.DefaultPageSetup.RightMargin) / 2;
29
30     table.Rows.Height = 10;
31
32     Row row = table.AddRow();
33     row.VerticalAlignment = VerticalAlignment.Top;
34
35     var teamParagraph = row.Cells[0].AddParagraph("Team: " + _preValue.team.name);
36     teamParagraph.Format.Font.Bold = true;
37     teamParagraph.Format.Font.Size = 10;
38
39     var startNumberParagraph = row.Cells[1].AddParagraph("Start no.: " + _preValue.team.startnumber);
40     startNumberParagraph.Format.Font.Bold = true;
41     startNumberParagraph.Format.Font.Size = 10;
42
43
44     var BorderParagraph = header.AddParagraph("");
45     BorderParagraph.Format.Borders.Bottom.Visible = true;
46     BorderParagraph.Format.Borders.Bottom.Style = BorderStyle.Single;
47 }

```

Abbildung 25: Erstellung eines Headers auf dem Wertungsblatt

In Abbildung 25 ist ersichtlich, wie die Erstellung des Headers eines **Wertungsblattes** abläuft. Hierfür werden der Headersektion des Dokuments als erstes das Logo des Turnsport Austria und eine Überschrift hinzugefügt. Anschließend wird dem Dokument eine Tabelle mit zwei Michael Laimer, Laurin Losert-Nachbaur

Spalten angefügt, welche es erlaubt, mehrere Paragraphen auf derselben Höhe zu platzieren. In diese Tabelle werden auf der linken Seite der Teamname (Zeile 39-41) und auf der rechten Seite die Startnummer eingefügt. Die Objekte, welche durch die „AddParagraph“ Funktion zurückgegeben werden, beinhalten eine Reihe an **Properties**, die verwendet werden können, um verschiedene Eigenschaften der Schrift (z. B. Positionierung, Größe) anzupassen. Der generierte Header kann in Abbildung 26 betrachtet werden.



Abbildung 26: Header im exportierter PDF-Datei

### 5.1.5.2 Einfügen von Bildern

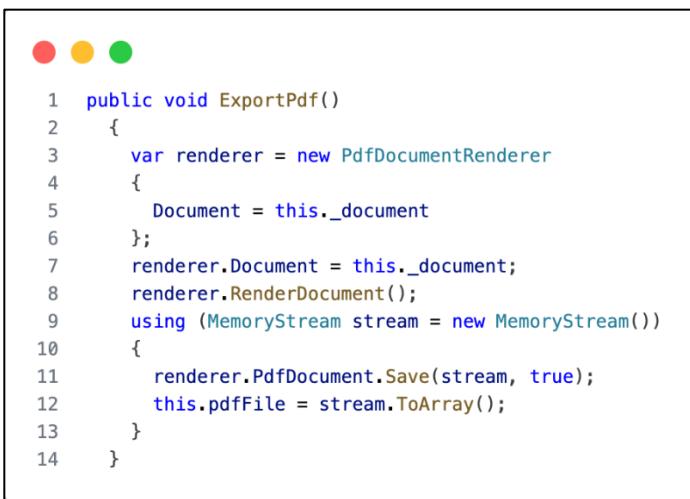
```
● ● ●
1 var image = await _trampetelementRepository.DownloadFileAsync(roundElements[0]?.fileId, cancellationToken);
2 var row = roundTable.AddRow();
3 row.TopPadding = 2.5;
4 row.BottomPadding = 2.5;
5 row.Height = 20;
6 using (MemoryStream stream = new MemoryStream(image))
7 {
8     var elementImage = row.Cells[0].AddImage(ImageSource.FromStream(roundElements[0].Id, () => stream));
9     elementImage.Height = 15;
10    elementImage.Left = 20;
11    elementImage.LockAspectRatio = true;
12 }
```

The code is a snippet of C# code within a code editor. It uses the `TrampetelementRepository` to download an image from MongoDB. This image is then converted into a `MemoryStream`. The code then adds this stream to the first cell of a row in a table, setting its height to 15 and locking its aspect ratio. The `LockAspectRatio` property is set to `true` to ensure the image maintains its original proportions when scaled.

Abbildung 27: Einfügen von Bildern in ein PDF

Wie in Abbildung 27 zu sehen ist, wird zur Einfügung von Bildern in die PDF-Datei zuerst das Bild als Byte-**Array** aus der MongoDB geholt. Anschließend wird das Byte-**Array** in einem Memory-Stream geöffnet und mit Hilfe einer statischen ImageSource-Funktion der AddImage Methode übergeben. Anschließend wird die Höhe festgelegt sowie das Seitenverhältnis des Bildes gesperrt.

### 5.1.5.3 Rendern des PDFs



```

1  public void ExportPdf()
2  {
3      var renderer = new PdfDocumentRenderer
4      {
5          Document = this._document
6      };
7      renderer.Document = this._document;
8      renderer.RenderDocument();
9      using (MemoryStream stream = new MemoryStream())
10     {
11         renderer.PdfDocument.Save(stream, true);
12         this.pdfFile = stream.ToArray();
13     }
14 }

```

Abbildung 28: Rendern einer PDF-Datei

Für den abschließenden Rendervorgang des PDFs wird das Dokument als erstes einem „**PdfDocumentRenderer**“ übergeben. Anschließend wird das Dokument gerendert. Das fertig gerenderte Dokument wird schlussendlich einem **Memorystream** übergeben, der es in ein **Byte-Array** umwandelt. Dieses **Byte-Array** kann dann als Antwort auf eine HTTP Anfrage zur Erstellung des PDFs verwendet werden.

## 5.2 Frontend

Das Frontend ist jener Teil der Applikation, mit welchem der User interagiert. Die zwei Frontends der Applikation wurden mithilfe von Ionic und **Angular** entwickelt.

### 5.2.1 Architektur des Frontends

Das Frontend besteht grundlegend aus zwei Teilen: Der erste Teil ist eine Oberfläche für die Verwaltung verschiedener Inhalte, wie zum Beispiel die verfügbaren Turnübungen, die aktiven Vereine und die jeweiligen Teams dieser Vereine. Der zweite Teil ist die Vereinsoberfläche, auf welcher das eigentliche Eingeben der Wertungsblätter stattfindet. Jeder teilnehmende Verein kann sich auf dieser Oberfläche anmelden und für die einzelnen Teams die Wertungsblätter über verschiedene Formulare eingeben.

Beide Frontends kommunizieren mit dem gleichen Backend und mit derselben Datenbank.

## 5.2.2 Administratoroberfläche

Die Administratoroberfläche dient der Verwaltung der gesamten Applikation. Das Design für diese Oberfläche wurde unkompliziert gehalten, da sie nur von wenigen Personen benutzt wird, welche zudem mit der Applikation vertraut sind.

Dennoch ist es wichtig, dass das Navigieren zwischen den Verwaltungsinterfaces und das Ändern von Daten einfach vorgenommen werden kann.

Deshalb gibt es im Menü der Administratoroberfläche fünf Seiten, auf die navigiert werden kann:

| Bezeichnung     | Beschreibung  |
|-----------------|---|
| <b>Elements</b> | Dient der Verwaltung der <b>Elemente</b> , die bei Übungen geturnt werden können. Diese Seite ist in drei weitere Seiten für die Geräte unterteilt.   |
| Teams           | Erlaubt die Verwaltung von einzelnen Teams, unabhängig vom Verein.  |
| Timetable       | Ermöglicht das Sortieren der Teams nach zeitlichem Ablauf des Wettkampfes. Die Sortierung wird benötigt, um die Wertungsblätter am Wettkampftag sofort in der richtigen Reihenfolge ausdrucken zu können. |
| Clubs           | Vereine können erstellt und bearbeitet werden. Es ist möglich, den Vereinen Teams zuzuordnen, sowie Benutzer für die Teams anzulegen.   |
| Prevalues       | Gestattet einen Überblick über die eingegebenen Wertungsblätter und bietet die Möglichkeit, diese auszudrucken.   |

### 5.2.2.1 Elemente-Verwaltung

In der **Elemente**-Verwaltung können die verfügbaren **Elemente** administriert werden. Dies ist notwendig, da **Elemente** nicht immer gleichbleiben, sondern sich über die Jahre ändern können. Jedes Gerät hat unterschiedliche **Elemente** mit verschiedenen Eigenschaften, die beim Erstellen bzw. Ändern festgelegt werden können. Darum gibt es für jedes Gerät eine eigene Verwaltungsoberfläche, um die entsprechenden Eigenschaften eintragen zu können. Das bedeutet, dass für jedes Gerät eine Übersichtsseite sowie **Komponenten** angelegt wurden, um jeweils **Elemente** erstellen und bearbeiten zu können.

Zudem gibt es Eigenschaften, welche alle **Elemente**, unabhängig vom Gerät, gemeinsam haben:

| Eigenschaft | Beschreibung  |
|-------------|---|
| Image       | Ein Bild, welches für das <b>Element</b> steht und auf den Wertungsblättern erscheinen, wird.                             |
| Code        | Der Code ist ein Kürzel für die jeweiligen <b>Elemente</b> . Dadurch ist ersichtlich, welches <b>Element</b> gemeint ist. |
| K-Code      | Ein Code, der das digitale Darstellen von <b>Elementen</b> erleichtern soll.  |
| Description | Eine kurze Beschreibung des <b>Elements</b>   |
| Value       | Der Wert für das jeweilige <b>Element</b> , welcher benötigt wird, um die Noten am Ende auszurechnen.                     |

### 5.2.2.1.1 Trampolinelemente-Verwaltung

Die Trampolinelemente-Verwaltung bietet einen Überblick über die derzeitigen **Elemente**. Des Weiteren gibt es Knöpfe zum Anlegen eines neuen **Elements**, sowie zum Löschen eines **Elements**. Bei einem Klick auf die Zeile eines **Elements** kann dieses auch bearbeitet werden.

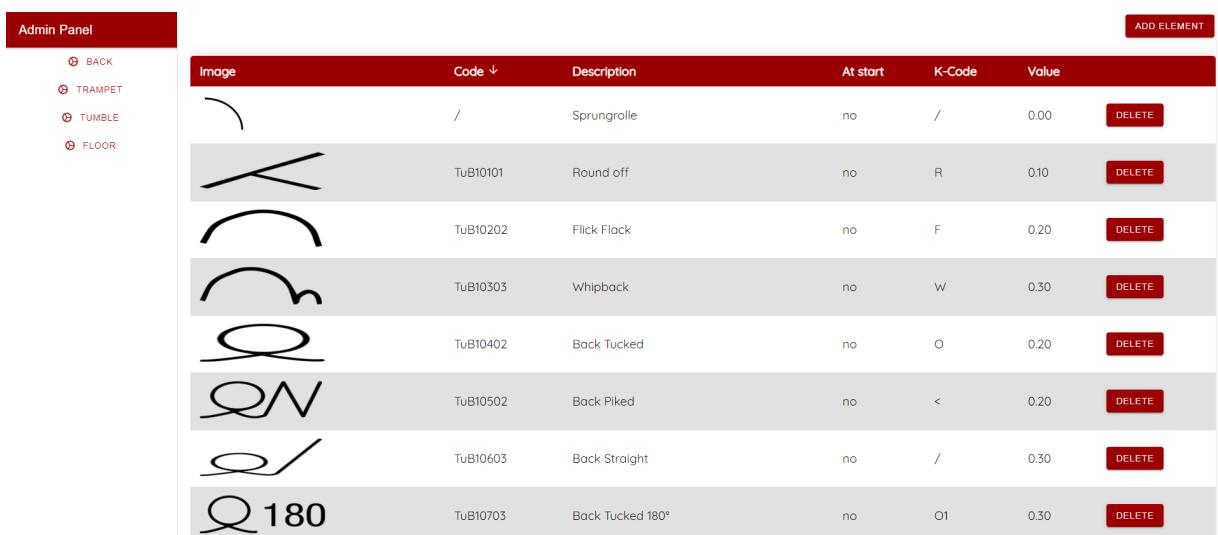
| Admin Panel                      |         | Trampolinelemente-Verwaltung |      |                                 |                |        |       |                         |
|----------------------------------|---------|------------------------------|------|---------------------------------|----------------|--------|-------|-------------------------|
|                                  |         | Image                        | Code | Description                     | Subapparatus   | K-Code | Value | Actions                 |
| <input checked="" type="radio"/> | BACK    | /                            | /    | Strecksprung                    | Trampet        | /      | 0.00  | <button>DELETE</button> |
| <input checked="" type="radio"/> | TRAMPET | V                            | /    | Bücksprung >45°                 | Trampet        | /      | 0.05  | <button>DELETE</button> |
| <input checked="" type="radio"/> | TUMBLE  | I=I                          | /    | Aufhocken - Strecksprung        | Vaulting Table | /      | 0.10  | <button>DELETE</button> |
| <input checked="" type="radio"/> | FLOOR   | Y                            | /    | Strecksprung mit ½ Drehung      | Trampet        | /      | 0.05  | <button>DELETE</button> |
|                                  |         | O                            | /    | Strecksprung mit ganzer Drehung | Trampet        | /      | 0.10  | <button>DELETE</button> |
|                                  |         | H                            | /    | Hocksprung                      | Trampet        | /      | 0.05  | <button>DELETE</button> |
|                                  |         | I=                           | /    | Durchhocken                     | Vaulting Table | /      | 0.20  | <button>DELETE</button> |
|                                  |         | A=                           | /    | Durchgrätschen                  | Vaulting Table | /      | 0.20  | <button>DELETE</button> |

Abbildung 29: Trampolinelemente-Verwaltung

Wie in Abbildung 29 zu erkennen ist, haben Trampolinelemente zu den normalen Eigenschaften zusätzlich noch eine Eigenschaft Namens „Subapparatus“. Diese Eigenschaft wird benötigt, da laut Regelwerk von den drei Trampolinrunden mindestens eine mit Sprungtisch und eine ohne Sprungtisch absolviert werden muss.<sup>14</sup> Mit Hilfe dieser Eigenschaft kann eine Differenzierung vorgenommen werden. Dadurch kann bei der Eingabe von Wertungsblättern auf diese Regel eingegangen werden.

### 5.2.2.1.2 Tumblingelemente-Verwaltung

Die Tumblingelemente-Verwaltung ist grundsätzlich sehr ähnlich zur Trampolinelemente-Verwaltung. Es gibt wieder die Möglichkeit **Elemente** zu erstellen, zu bearbeiten und zu löschen.



| Admin Panel |          |      |                  |          |        |       | <b>ADD ELEMENT</b> |
|-------------|----------|------|------------------|----------|--------|-------|--------------------|
|             | Image    | Code | Description      | At start | K-Code | Value |                    |
| ⊕ BACK      | /        |      | Sprungrolle      | no       | /      | 0.00  | <b>DELETE</b>      |
| ⊕ TRAMPET   | TuB10101 |      | Round off        | no       | R      | 0.10  | <b>DELETE</b>      |
| ⊕ TUMBLE    | TuB10202 |      | Flick Flack      | no       | F      | 0.20  | <b>DELETE</b>      |
| ⊕ FLOOR     | TuB10303 |      | Whipback         | no       | W      | 0.30  | <b>DELETE</b>      |
|             | TuB10402 |      | Back Tucked      | no       | O      | 0.20  | <b>DELETE</b>      |
|             | TuB10502 |      | Back Piked       | no       | <      | 0.20  | <b>DELETE</b>      |
|             | TuB10603 |      | Back Straight    | no       | /      | 0.30  | <b>DELETE</b>      |
|             | TuB10703 |      | Back Tucked 180° | no       | O1     | 0.30  | <b>DELETE</b>      |

Abbildung 30: Tumblingelemente-Verwaltung

Tumblingelemente benötigen die zusätzliche Eigenschaft „At start“. Dieses Attribut wird benötigt, da manche **Elemente** einen anderen Wert für die Berechnung der Note haben, wenn sie als erstes geturnt werden.<sup>15</sup> Dies muss auf dem Wertungsblatt bei der Angabe der Werte vermerkt werden, wodurch sich diese Unterscheidung bei der Eingabe von Tumblingelementen ergibt.

<sup>14</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 71.

<sup>15</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 63.

### 5.2.2.1.3 Bodenelemente-Verwaltung

Die Bodenelemente-Verwaltung gestaltet sich umfangreicher als die Verwaltungen für Trampolin- und Tumblingelemente.

| ELEMENTS |        | COMPOSITIONS            |                 |       |               |                         |
|----------|--------|-------------------------|-----------------|-------|---------------|-------------------------|
| Image    | Code   | Description             | Type            | Value | Allowed in DS |                         |
| →○       | DB201  | Forward pirouette 360°  | Dynamic Balance | 0.20  | X             | <button>DELETE</button> |
| →○       | DB601  | Forward pirouette 540°  | Dynamic Balance | 0.60  | X             | <button>DELETE</button> |
| →○       | DB801  | Forward pirouette 720°  | Dynamic Balance | 0.80  | X             | <button>DELETE</button> |
| →○       | DB1001 | Forward pirouette 900°  | Dynamic Balance | 1.00  | X             | <button>DELETE</button> |
| ←○       | DB202  | Backward pirouette 360° | Dynamic Balance | 0.20  | X             | <button>DELETE</button> |
| ←○       | DB602  | Backward pirouette 540° | Dynamic Balance | 0.60  | X             | <button>DELETE</button> |

Abbildung 31: Bodenelemente-Verwaltung

In der oben angeführten Abbildung ist zu sehen, dass die **Elemente** für den Boden kein K-Code haben, aber dafür zwei zusätzliche Eigenschaften: Die erste Eigenschaft ist der „Type“. Diese Eigenschaft ergibt sich daraus, dass die **Elemente** am Boden in verschiedene Typen unterteilt sind, wobei bei der Eingabe des Wertungsblatts bestimmte Regeln eingehalten werden müssen. Das bedeutet, dass gewisse **Elemente** in einer gewissen Anzahl vorkommen müssen, um ein gültiges Wertungsblatt zu bilden.<sup>16</sup> Die zweite Eigenschaft ist „Allowed in DS“. Diese beschreibt, ob ein **Element** während der Bewegungssequenz geturnt werden darf.<sup>17</sup>

Zusätzlich zu der **Elemente**-Verwaltung befindet sich hier auch die Verwaltung für die Kompositionen. Bei Wettkämpfen gibt es gewisse Kompositionsanforderungen, die eingehalten werden müssen, da sonst ein Punkteabzug erfolgt. Diese Kompositionen werden beim Bodenwertungsblatt zu den jeweiligen Formationen in einer eigenen Spalte eingetragen. Zwischen der Elemente- und Kompositionen-Verwaltung kann per Reiter hin und her gewechselt werden.

<sup>16</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 40.

<sup>17</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 41

### 5.2.2.2 Verein-Verwaltung

Die Verein-Verwaltung ist für die Verwaltung der einzelnen Vereine da. Es können neue Vereine angelegt werden und bestehende bearbeitet werden. Ein Verein wird grundsätzlich nur mit einem Namen und einer Nationalität angegeben.

| Name                      | Nationality |
|---------------------------|-------------|
| TSZ Dornbirn              | Austria     |
| TS Wolfurt                | Austria     |
| TS Lustenau               | Austria     |
| Tecnoplast TS Höchst      | Austria     |
| TS Hohenems               | Austria     |
| Sportunion Wien 3         | Austria     |
| MTV Hernals               | Austria     |
| Halleiner Turnverein 1866 | Austria     |

TSZ Dornbirn

- GENERAL**
- Name: TSZ Dornbirn
- Nationality: Austria

Abbildung 32: Verein-Verwaltung

Zu den Eigenschaften des Vereines können zusätzlich die Teams des Vereins hinzugefügt werden. Die Benutzerinnen und Benutzer des Vereins können ebenfalls über die Vereins-Verwaltung verwaltet werden.

### 5.2.2.3 Team-Verwaltung

Die Teams können über die Verein-Verwaltung grundlegend administriert werden. In der Team-Verwaltung können zusätzlich die zu dem Team gemachten Eingaben der Wertungsblätter angeschaut werden und theoretisch auch geändert werden. Eine Änderung der Wertungsblätter über die Administratoroberfläche ist aber generell nicht vorgesehen, sondern wird nur in speziellen Fällen verwendet. Es könnte zum Beispiel verwendet werden, um Wertungsblätter am Wettkampftag abzuändern, falls sich am Vortag ein Sportler oder eine Sportlerin beim Training verletzt haben sollte.

| Startnumber | Name                            | Class      | Club                      |
|-------------|---------------------------------|------------|---------------------------|
| 1           | Sportunion Wien 3 OK w          | Open Class | Sportunion Wien 3         |
| 2           | Sportunion Wien 3 OK w          | Open Class | Sportunion Wien 3         |
| 3           | MTV Hernals OK mx               | Open Class | MTV Hernals               |
| 4           | Halleiner Turnverein 1866 OK mx | Open Class | Halleiner Turnverein 1866 |
| 5           | TV Straßwalchen OK w            | Open Class | TV Straßwalchen           |
| 6           | TV Straßwalchen OK w            | Open Class | TV Straßwalchen           |
| 7           | TS Lustenau OK 1 w              | Open Class | TS Lustenau               |

Abbildung 33: Team-Verwaltung

## 5.2.3 Vereinsoberfläche

Die Vereinsoberfläche ist der Teil der Applikation, mit welchem die Vereine interagieren, um die Wertungsblätter einzugeben. Mit diesem Teil der Applikation interagieren die normalen Benutzerinnen und Benutzer, weshalb die Oberfläche einfach zu bedienen und übersichtlich sein sollte.

### 5.2.3.1 Aufbau und Design

Für die Vereinsoberfläche war es sehr wichtig, dass das Design ansprechend ist und die Benutzerinnen und Benutzer intuitiv wissen, wie die Seite zu navigieren ist. Die Oberfläche ist deshalb so aufgebaut, dass sich an der Seite eine Leiste befindet, auf welcher sich wichtige Informationen zum Wettkampf sowie Hinweise zu den Fristen für die Eingabe der Wertungsblätter befinden.

| Status | Team                 | Class      | Startnumber | Round |
|--------|----------------------|------------|-------------|-------|
| ✓      | TSZ Dornbirn OK w    | Open Class | 9           | 1     |
| ✓      | TSZ Dornbirn OK w    | Open Class | 10          | 1     |
| ✓      | TSZ Dornbirn J3 w    | Youth 3    | 14          | 2     |
| ✓      | TSZ Dornbirn J1 mx   | Youth 1    | 20          | 2     |
| ✓      | TSZ Dornbirn Jun w   | Juniors    | 29          | 3     |
| ✓      | TSZ Dornbirn Elite w | Elite      | 33          | 3     |

Abbildung 34: Vereinsoberfläche Teamübersicht

Ebenso ist auf der Seitenleiste ein Knopf für das Rückkehren auf die Hauptansicht und ein Knopf, mit dem man sich Ausloggen kann, zu finden (siehe Abbildung 34). Des Weiteren gibt es eine schmale Übersicht des Benutzers inklusive des Vereines des Benutzers.

Nach dem Login gelangt man auf die Übersichtsseite der Teams. Im Normalfall haben Vereine mehrere Teams, für die verschiedene Wertungsblätter eingegeben werden müssen. Die Teams des Vereins werden in einer tabellarischen Ansicht aufgelistet. Die erste Spalte der Tabelle ist mit „Status“ beschrieben und beinhaltet Icons, welche den Fortschritt der Wertungsblätter für ein Team anzeigen. Dadurch wird sichergestellt, dass nach dem Login sofort ersichtlich ist, welche Wertungsblätter schon fertig eingegeben wurden und welche noch einzutragen sind.

Nach dem Auswählen eines Teams wird eine Übersicht der Geräte angezeigt. Hier ist zu sehen, für welches Gerät welche Eingaben gemacht wurden und welche noch offen sind. Über diese Ansicht kann auf die jeweiligen Interfaces der Geräte navigiert werden.

The screenshot displays the software interface for managing competition forms. At the top, it shows the user's name 'Good evening laurin.losert!' and the logo of 'TURNSPORT AUSTRIA'. Below this, the competition details are listed: 'Staatsmeisterschaft Team Turnen Competition' and the date '25.11.2023'. The entry deadline for tariff forms is '18.11.2023 22:59' and the amendment deadline is '25.11.2023'. A link to 'What do the Deadlines mean?' is also present. On the left, there is a sidebar with a user profile picture, the name 'laurin.losert', and buttons for 'SEND FEEDBACK', 'HOME', and 'LOGOUT'. The main content area is titled 'TSZ Dornbirn OK w (9)'. It includes a navigation bar with 'Open Class Class', 'Round 1 Round', and 'national (Turnsport Austria) Ruleset'. The interface is divided into three sections: 'Floor', 'Tumble', and 'Trampet'. Each section has a 'Tariff Form on [Device]' sub-section with tables showing round numbers, entered participants, and status. Buttons for 'ENTER TARIFF FORM ON FLOOR', 'PREVIEW TARIFF FORM ON FLOOR', 'ENTER TARIFF FORM ON TUMBLE', 'PREVIEW TARIFF FORM ON TUMBLE', and 'ENTER TARIFF FORM ON TRAMPET' are located at the bottom of each sub-section.

Abbildung 35: Vereinsoberfläche Geräteübersicht

### 5.2.3.2 Trampolin-Eingabe

Die Trampolin-Eingabe ist vom Design an das Wertungsblatt vom Trampolin angelehnt, um den Benutzern eine möglichst einfache Handhabung zu ermöglichen. In der Kopfzeile der Seite wird noch einmal das Team angeführt, für welches das Wertungsblatt momentan ausgefüllt wird. Unter der Kopfzeile ist die Oberfläche geteilt. Auf der linken Seite kann mit dem Wertungsblatt direkt interagiert werden. Es können **Elemente** hinzugefügt, gelöscht und verschoben werden. Auf der rechten Seite befindet sich ein Überblick über die Regeln, die eingehalten werden müssen. Bei Nichtbefolgung dieser Regeln kann das Wertungsblatt nicht abgespeichert werden.

The screenshot shows a web-based application for entering trampoline scores. At the top, it displays the team name "TSZ Dornbirn OK w". Below this, there are three rows for "ROUND 2". Each row contains a diagram of a vault, the number "180", and the text "Straight salto 1/2". To the right of the table, there is a sidebar titled "Tariff Form Validation" with the following error messages:

- The Tariff Form is not valid
- There are at least 4 Elements required in Round 3.
- There must be at least one round jumped at vaulting table and one round jumped at trampet

| ROUND 1                         | ROUND 2  | ROUND 3 |
|---------------------------------|--|---------|
| 1.<br>180<br>Straight salto 1/2 | 0.30<br><br><input type="button" value="COPY"/><br><input type="button" value="PASTE"/><br><input type="button" value="REMOVE"/> |         |
| 2.<br>180<br>Straight salto 1/2 | 0.30<br><br><input type="button" value="COPY"/><br><input type="button" value="PASTE"/><br><input type="button" value="REMOVE"/> |         |
| 3.<br>180<br>Straight salto 1/2 | 0.30<br><br><input type="button" value="COPY"/><br><input type="button" value="PASTE"/><br><input type="button" value="REMOVE"/> |         |
| A                               |  |         |

Abbildung 36: Trampolin-Eingabe

Die Hälfte mit dem Wertungsblatt wird durch drei Segmente aufgeteilt, welche jeweils die einzelnen Durchläufe darstellen. Für jeden Durchlauf gibt es eine Tabelle, in welche die **Elemente** eingegeben werden können. Es kann dabei in jeder Zeile genau ein **Element** eingetragen werden. Der erste Durchlauf für das Trampolin ist eine sogenannte **Einheitsrunde**. In dieser Runde springen alle Turnerinnen und Turner das gleiche **Element**, weshalb diese Tabelle bei der Eingabe nur eine Reihe hat.<sup>18</sup>

<sup>18</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 71.

Die Auswahl der **Elemente** kann mit dem Plus-Knopf geöffnet werden und öffnet sich dann in Form eines **Modal-Dialogs**, welcher mit Segmenten in „Trampet“ und „Vaulting table“ unterteilt ist. Diese Unterteilung wurde vorgenommen, da mindestens ein Durchlauf ohne Sprungtisch und einer mit Sprungtisch gemacht werden muss. Die **Elemente** können auch mit Hilfe der Suchleiste aufgrund ihrer verschiedenen Attribute durchsucht werden.

Die andere Hälfte mit dem Überblick der Regeln ist eine Tabelle mit Warnungen, welche gegebenenfalls anzeigen, wenn eine wichtige Regel verletzt wird. Die Überprüfung der Regeln ist jedoch nicht vollständig, da es Situationen gibt, in denen Trainerinnen und Trainer diese Regeln absichtlich nicht beachten und den Abzug von Punkten in Kauf nehmen. Deshalb gibt es nicht nur Fehler, sondern auch Warnungen, die in der Tabelle angezeigt werden, um Trainerinnen und Trainer auf Fehler aufmerksam zu machen, die absichtlich gemacht werden.

Für die Überprüfung bzw. Validierung der eingegebenen **Elemente** gibt es eine Funktion, die immer aufgerufen wird, wenn ein **Element** eingefügt, kopiert oder gelöscht wird.

```
● ● ●
1 const elementsAscending = roundElements.every(
2   (element, i, { [i - 1]: prev }) => !prev || !element || prev?.value <= element.value
3 );
4 if (!elementsAscending) {
5   this.validationErrors.push(
6     new ValidationError(
7       ValidationErrorType.ERROR,
8       'The difficulties of the elements in round ' + (Number(roundIndex) + 1) + ' are not ascending'
9     )
10  );
11 }
```

Abbildung 37: Trampolin-Validierungsfunktion

Das obenstehenden Coding bildet einen Ausschnitt der Validierungsfunktion als Beispiel ab. Hierbei wird überprüft, ob die **Elemente** eines Durchlaufs in aufsteigender Schwierigkeit sind oder den gleichen Wert haben. Dafür wird eine Variable definiert, die eine „every“ Funktion aufruft. Diese überprüft, ob jedes **Element** der Runde in aufsteigender Reihenfolge (nach Wert) eingegeben wurde. Es wird zuerst überprüft, ob das **Element** und jenes davor nicht **null** sind. Falls eines von beiden, oder beide **null** sein sollten, wird ein **true** zurückgegeben und das nächste **Element** des Durchlaufs wird überprüft. Sind beide Variablen definiert, wird überprüft, ob der Wert des vorherigen **Elementes** kleiner oder gleich dem momentanen **Element** ist. Wenn dies für ein **Element** nicht stimmt, wird die Variable auf **false** gesetzt. In den Zeilen von

4-13 wird, falls die in der ersten Zeile definierte Variable nicht **true** sein sollte, ein neues Error-Objekt zu dem gesamten Error-**Array** angehängt. Beim Erstellen des neuen Error-Objektes wird außerdem der Typ des Errors in den **Konstruktor** mitgegeben. Der Typ kann dabei entweder ERROR sein für schwerwiegende Fehler, mit welchen das Wertungsblatt nicht gespeichert werden kann, oder WARNING für Fehler, die nur als Warnung angezeigt werden sollten, mit welchen das Wertungsblatt trotzdem gespeichert werden kann.

### 5.2.3.3 Tumbling-Eingabe

Die Tumbling-Eingabe ist der Trampolin-Eingabe vom Design und der Funktion her ähnlich. Es gibt wieder die gleiche Aufteilung der Seite. Auch die Eingabe ist dem Wertungsblatt sehr ähnlich. Es gibt ebenfalls drei Durchläufe, die durch drei Segmente geteilt werden. Die erste Runde findet wieder als **Einheitsrunde** statt, was bedeutet, dass alle Turnerinnen und Turner dieselben **Elemente** turnen. Der große Unterschied zum Trampolin liegt in der Anzahl der **Elemente**: Beim Tumbling werden Abfolgen aus einer Kombination von mindestens drei **Elementen** geturnt, welche **Serien** genannt werden.<sup>19</sup> Es ist dabei möglich, **Elemente** einzufügen und zu löschen. Zusätzlich gibt es Funktionen für das Kopieren und Einfügen von ganzen **Serien**.

|         | ROUND 1 | ROUND 2 | ROUND 3  |
|---------|---------|---------|----------|
| Round 3 |         |         |          |
| 1       |         |         | 0.40<br> |
| 2       |         |         | 0.70<br> |
| 3       |         |         | 0.70<br> |
| 4       |         |         | 0.00<br> |
|         |         |         | 1.80     |

Abbildung 38: Tumbling-Eingabe

Die bereits eingetragenen **Elemente** können mit dem Edit-Knopf bearbeitet werden. Dabei öffnet sich ein **Modal-Dialog**, in welchem die **Elemente** verschoben und gelöscht werden können. Das Einfügen der **Elemente** findet, wie bei der Trampolin-Eingabe, über einen **Modal-Dialog** statt, der mit dem Plus-Knopf aufgerufen werden kann.

<sup>19</sup> vgl. TeamGym Code of Points V1.1, 2022, S. 58.

| Element | Code      | Description               | K-Code | Value |
|---------|-----------|---------------------------|--------|-------|
|         |           |                           |        |       |
|         | TuF10101  | Cartwheel                 | X      | 0.1   |
|         | TuF10202  | Handspring                | H      | 0.2   |
|         | TuF10302  | Flyspring                 | FS     | 0.2   |
|         | TuF10402S | Front Tucked (At start)   | O      | 0.2   |
|         | TuF10603S | Front Piked (At start)    | >      | 0.3   |
|         | TuF10803S | Front Straight (At start) | \      | 0.3   |
|         | TuF11003  | Front Tucked 180°         | O1     | 0.3   |
|         | TuF11103  | Arabian                   | 10     | 0.3   |
|         | TuF11204  | Front Piked 180°          | >1     | 0.4   |

Abbildung 39: Tumbling-Eingabe Elementauswahl

Wie in der obigen Abbildung zu sehen, werden die Tumblingelemente durch zwei Segmente geteilt. Beim Tumbling erfolgt die Einteilung nach vorwärts und rückwärts geturnten **Elementen**. Dabei muss mindestens ein Durchlauf nur aus Rückwärtselementen und ein Durchlauf nur aus Vorwärtselementen bestehen. Der übrige Durchlauf hat keine Anforderungen an die Richtung der **Elemente**.<sup>20</sup> Bei den Tumblingelementen gibt es außerdem unterschiedliche Schwierigkeiten für das gleiche **Element**, abhängig davon, ob dieses am Anfang einer **Serie** oder im Laufe der **Serie** geturnt wird.<sup>21</sup> Dadurch werden beim Einfügen des ersten **Elementes** nur **Elemente** zur Auswahl gegeben, die mit diesem Attribut gekennzeichnet sind.

Wie in Abbildung 38 zu erkennen, hat jede geturnte **Serie** einen Wert, der die Schwierigkeit der **Serie** bestimmt und für die Wertung notwendig ist. Dieser Wert wird mit folgendem Coding ausgerechnet:

<sup>20</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 58–59.

<sup>21</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 63.

```

● ● ●
1 import { Pipe, PipeTransform } from '@angular/core';
2 import { TumbleElement } from '../client/client';
3
4 @Pipe({
5   name: 'getElementsValue',
6   pure: false,
7 })
8 export class GetElementsValuePipe implements PipeTransform {
9   transform(elementArray: Array<TumbleElement>, participantCount: number | null): number {
10   if (elementArray && elementArray.length >= 1) {
11     let cleanedArray: Array<TumbleElement> = Array.from(new Set(elementArray.map((element) => element.id))).map(
12       (id) => {
13         return elementArray.find((element) => element.id === id);
14       }
15     );
16     if (cleanedArray.length === 1) {
17       return cleanedArray[0].value * (participantCount ?? 1);
18     }
19     cleanedArray.sort((a: TumbleElement, b: TumbleElement) => {
20       return b.value - a.value;
21     });
22     return (cleanedArray[0].value + cleanedArray[1].value) * (participantCount ?? 1);
23   }
24   return 0;
25 }
26 }

```

Abbildung 40: Serienschwierigkeit Pipe

In den ersten zwei Zeilen des Codes werden die Interfaces Pipe und PipeTransform und die Klasse TumbleElement importiert. Diese sind notwendig, um die Typensicherheit zu gewährleisten. In den Zeilen 4-7 wird die Pipe definiert und das Attribut pure auf **false** gesetzt, damit die Pipe bei jedem Aufruf ausgeführt wird.<sup>22</sup> In der Zeile 8 wird eine Klasse definiert, welche das PipeTransform interface implementiert und bis zu Zeile 26 reicht. In Zeile 9 wird in der Klasse eine Funktion definiert, die als Übergabeparameter ein **Array** von den **Elementen** in der **Serie** und die Anzahl der Teilnehmerinnen und Teilnehmer hat. Die Zahl der Teilnehmerinnen und Teilnehmer wird benötigt, um den Wert in der **Einheitsrunde** zu berechnen.

Der erste Schritt in der Funktion ist das Überprüfen, ob das übergebene **Array** vorhanden und nicht leer ist. Stimmt eines davon nicht, wird als Wert **null** zurückgegeben. Stimmt beides, wird in der nächsten Zeile ein neues **Array** deklariert, welches nur mit einzigartigen **Elementen** gefüllt wird. Falls das neue **Array** eine Länge von eins hat, also nur ein **Element** in der **Serie** ist, wird die Schwierigkeit, multipliziert mit der Teilnehmendenanzahl, zurückgegeben. Bei normalen Runden ist der participantCount auf **null** gesetzt, wodurch die Schwierigkeit mit Eins

---

<sup>22</sup> vgl. *Angular—Transforming Data Using Pipes*, o.J., Abschn. Detecting impure changes within composite objects.

multipliziert wird. In der **Einheitsrunde** hingegen ist der participantCount gesetzt und somit wird die Schwierigkeit mit der Teilnehmerzahl der **Einheitsrunde** multipliziert. Sollte in dem **Array** mehr als ein **Element** sein, wird das **Array** nach der Schwierigkeit absteigend sortiert (Zeile 19-21). In Zeile 22 werden die ersten zwei Einträge des **Arrays** addiert, welche in diesem Fall durch das Sortieren die zwei größten Schwierigkeiten sind. Die Schwierigkeit einer **Serie** wird nämlich durch das Addieren von den zwei schwersten **Elementen** ausgerechnet.<sup>23</sup> In der gleichen Zeile wird dann die Schwierigkeit wieder mit der Teilnehmerzahl multipliziert, um den Schwierigkeitswert der **Einheitsrunde** zu bilden. Dieser Wert wird dann von der Pipe zurückgegeben.

#### 5.2.3.4 Boden-Eingabe

Die Boden-Eingabe ist in Bezug auf die Eingabeoberfläche sehr unterschiedlich zu den anderen zwei Geräten. Anstatt einer einfachen Tabelle, in die **Elemente** eingefügt werden müssen, gibt es eine Tabelle mit mehreren Spalten pro Zeile, sowie **Elemente** (mit zugehörigen **Element-Codes**, Formationen, ...), die in die jeweiligen Spalten eingefügt werden müssen. Das grundlegende Layout ist immer noch einheitlich mit den Layouts der Eingaben für die anderen **Elemente**: In der Kopfzeile wird das Team, inklusive der Startnummer, angezeigt. Der interaktive Bereich wird in zwei Teile geteilt: Links die eigentliche Oberfläche für die Eingabe und rechts die Überprüfungen bzw. Validierungen.

---

<sup>23</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 62.

| Team: TSZ Dornbirn Elite w |                          |        |               |  | Start no.: 33 |
|----------------------------|--------------------------|--------|---------------|--|---------------|
| Formation                  | Code                     | Symbol | Comp.         |  |               |
|                            |                          |        | SF            | <input type="button" value="+ ADD"/><br><input checked="" type="button" value="EDIT"/> |               |
|                            | SB806<br>HB1001<br>F1003 |        |               | <input type="button" value="+ ADD"/><br><input checked="" type="button" value="EDIT"/> |               |
|                            | J1009A<br>DSJ823         |        | CF<br>↔<br>DS | <input type="button" value="+ ADD"/><br><input checked="" type="button" value="EDIT"/> |               |
|                            | A801                     |        | LF            | <input type="button" value="+ ADD"/><br><input checked="" type="button" value="EDIT"/> |               |
|                            | DD<br>G1001<br>A1013     |        |               | <input type="button" value="+ ADD"/><br><input checked="" type="button" value="EDIT"/> |               |
|                            |                          |        |               | <input type="button" value="+ ADD"/><br><input checked="" type="button" value="EDIT"/> |               |
|                            |                          |        |               | <input type="button" value="+ ADD"/>   |               |

**Requirements**

Compositions

- Small Formation
- Large Formation
- Curved Formation
- D. e. in moving sequence
- RS Start
- RS End
- Planes Backward
- Sideways Planes

Elements

- Handstand
- Difficulty Distribution
- 1 of 1 Standing Balance
- 2 of 3 Balance Elements
- 2 of 3 Jumps/Hops/Leaps
- 2 of 2 Acrobatic

Abbildung 41: Bodenwertungsblatt in Arbeit

Wie auf der rechten Seite in Abbildung 41 zu sehen, ist die Validierung der Eingabe in **Elemente**, Kompositionen und Sonstige (nicht in der Abbildung zu sehen) geteilt. Diese beziehen sich auf die vielen Anforderungen, die eingehalten werden müssen.

Für die Komposition müssen, je nach Klasse, verschieden viele Formationen geturnt werden. Diese werden in der Spalte ganz links in der Tabelle angezeigt. Bei einer neuen Eingabe hat die Tabelle so viele Zeilen, wie die Klasse des Teams Formationen haben muss (z. B. Acht bei Elite Teams, Vier bei der offenen Klasse, usw.). Mit einem Knopf unter der Tabelle können noch mehr Formationen hinzugefügt werden, sollte die/der eingebende Trainerin und Trainer dies wünschen. Eine neue Formation kann mit dem Plus-Knopf hinzugefügt werden. Es öffnet sich dann, als **Modal-Dialog**, ein Fenster auf dem Punkte mit der Maus platziert und nachträglich umhergeschoben werden können. Abbildung 42 zeigt dabei eine leere Formation (links) und eine ausgefüllte Formation (rechts).

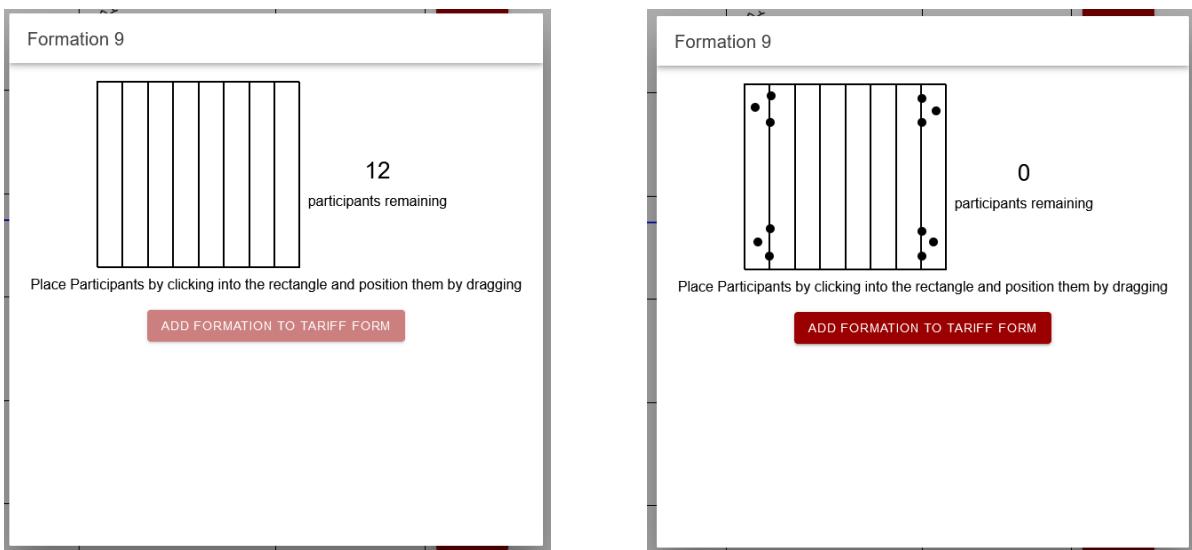


Abbildung 42: Leere und ausgefüllte Formation

Beim Öffnen werden dabei im Hintergrund als Parameter die Nummer der Formation, die Teilnehmerzahl und falls eine schon vorhandene Formation bearbeitet wird, die gesetzten Punkte, angezeigt. Die Linien, die in Abbildung 42 zu sehen sind, stellen Orientierungshilfen für die Trainerinnen und Trainer dar, um die Formationen genauer aufstellen zu können.

Die Punkte werden im Hintergrund mit x- und y-Koordinaten gespeichert, wobei es zwei unterschiedliche Variablen gibt: In einer Variable werden die eigentlichen Punkte festgehalten, die auch schlussendlich in der Datenbank gespeichert werden und in der anderen Variable werden die Punkte für das Anzeigen in der Grafik zwischengespeichert. Beim Klick auf „Add formation to tarrif form“ werden die Punkte als Parameter wieder zurück an die Boden-Eingabe gegeben und mit Hilfe einer eigenen **Komponente** in der Tabelle angezeigt.

In der Tabelle können nicht nur Formationen, sondern auch **Elemente** und Kompositionen eingefügt werden. Diese können am Rand jeder Zeile mit dem „Add“ Knopf hinzugefügt werden. Es öffnet sich dann, ähnlich wie bei den anderen zwei Geräten, ein **Modal-Dialog**, auf dem die **Elemente** aufgelistet sind und ausgewählt werden können.

The screenshot shows a modal dialog titled 'Add Element/Composition'. On the left, the 'Elements' section displays a table with four rows of data. The columns are labeled 'Image', 'Code', 'Description', 'Type', 'Value', and 'Allowed in DS'. The rows represent different dynamic balance elements:

| Image | Code   | Description            | Type            | Value | Allowed in DS |
|-------|--------|------------------------|-----------------|-------|---------------|
| →○    | DB201  | Forward pirouette 360° | Dynamic Balance | 0.20  | false         |
| →○○   | DB601  | Forward pirouette 540° | Dynamic Balance | 0.60  | false         |
| →○○○  | DB801  | Forward pirouette 720° | Dynamic Balance | 0.80  | false         |
| →○○○○ | DB1001 | Forward pirouette 900° | Dynamic Balance | 1.00  | false         |

On the right, the 'Compositions' section displays a table with six rows of data. The columns are labeled 'Code' and 'Description'.

| Code | Description      |
|------|------------------|
| SF   | Small Formation  |
| LF   | Large Formation  |
| CF   | Curved Formation |
| RS → | RS Start         |
| RS ← | RS End           |
| ↑    | Planes Backward  |

Abbildung 43: Elemente und Kompositionen Bodeneingabe

Es gibt die Möglichkeit, die **Elemente** mit der Suchleiste zu durchsuchen. Dafür gibt es viele Segmente, welche das Filtern von **Elementen** nach ihrem Typ möglich macht (siehe Abbildung 43). Auf der rechten Seite des **Modal-Dialogs** befinden sich die Kompositionen.

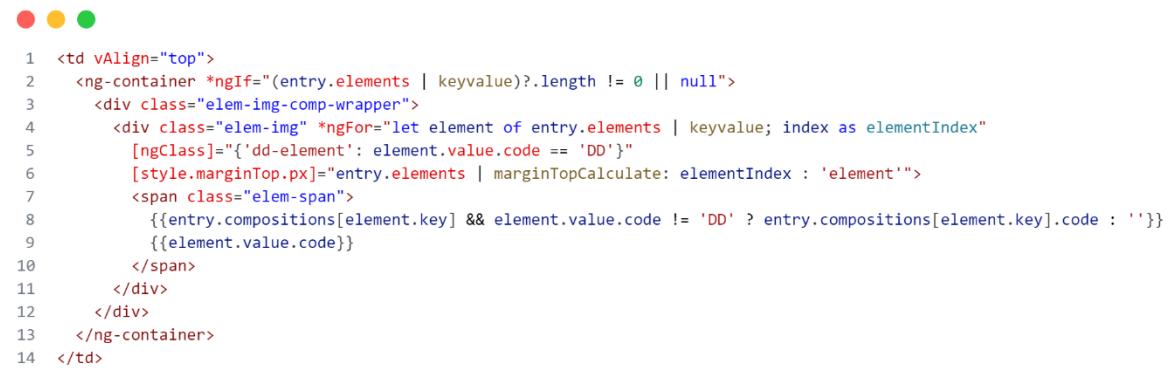
Für jede Formation, die in die Tabelle eingefüllt wird, können ein oder mehrere **Elemente** und Kompositionen in die Spalten daneben eingefügt werden. Diese müssen in jeder Reihe klar untereinander dargestellt werden. Um die Positionen der anzuzeigenden **Elemente** zu bestimmen, gibt es eine Pipe, die mit Hilfe der Anzahl an **Elementen** in der Reihe festlegt, welchen Abstand der Text oder die Bilder vom Rahmen der Zeile haben müssen.

|  |                  |        |               |
|--|------------------|--------|---------------|
|  | J1009A<br>DSJ823 | 2<br>Z | CF<br>↔<br>DS |
|--|------------------|--------|---------------|

Abbildung 44: Bodeneingabe Elementpositionen in Zeile

In Abbildung 44 ist zu sehen, dass in der ersten Spalte die Formation abgebildet ist und in der Spalte daneben die **Elemente**-Codes stehen. In der dritten Spalte werden die **Elemente** mit dem Bild angezeigt und in der letzten Spalte sind die Kompositionen eingetragen. Die **Elemente** und Kompositionen sind dabei alle untereinander aufgelistet. Die Reihenfolge kann mit dem Edit-Knopf neben der jeweiligen Zeile nachträglich noch bearbeitet werden.

Es gibt unter den **Elementen** die sogenannte „Difficulty distribution“ (abgekürzt DD). Dieses **Element** markiert den Zeitpunkt, an dem eine Minute und 30 Sekunden der Bodenübung vorbei sind. Sie wird mit einer durchgezogenen horizontalen Linie im Wertungsblatt gekennzeichnet (siehe Abbildung 41).<sup>24</sup>



```

1  <td vAlign="top">
2    <ng-container *ngIf="(entry.elements | keyvalue)?.length != 0 || null">
3      <div class="elem-img-comp-wrapper">
4        <div class="elem-img" *ngFor="let element of entry.elements | keyvalue; index as elementIndex"
5          [ngClass]={'dd-element': element.value.code == 'DD'}
6          [style.marginTop.px]="entry.elements | marginTopCalculate: elementIndex : 'element'"
7          <span class="elem-span">
8            {{entry.compositions[element.key] && element.value.code != 'DD' ? entry.compositions[element.key].code : ''}}
9            {{element.value.code}}
10         </span>
11       </div>
12     </div>
13   </ng-container>
14 </td>

```

Abbildung 45: Codeausschnitt - Difficulty Distribution

In der Oberfläche wird die DD als zwei Pixel dicke blaue Linie eingezeichnet. Dies wird mit Hilfe einer **Direktive** von **Angular** gemacht, die sich „NgClass“ nennt. Diese **Direktive** ermöglicht es, **CSS**-Klassen konditionell zu vergeben. Die Klasse wird daher nur vergeben, wenn es sich bei dem **Element** um die DD handelt.

Außerdem kann in dem abgebildeten Code erkannt werden, dass unter speziellen Konditionen vor den Code des **Elements** etwas geschrieben wird. Das kommt daher, dass eines der **Elemente** einer Bodenübung als „Difficulty Element in Moving Sequence“ (DS) geturnt werden muss und daher auch als solches gekennzeichnet werden muss (siehe Abbildung 44).<sup>25</sup>

## 5.2.4 Login

Der Login am Frontend findet mit Hilfe einer **Komponente** und einem Service statt. Die **Komponente** ist für die Interaktion mit dem User zuständig. Der Service handhabt die Kommunikation mit dem Backend sowie die Authentifizierung.

---

<sup>24</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 47.

<sup>25</sup> vgl. *TeamGym Code of Points V1.1*, 2022, S. 45.

### 5.2.4.1 Login-Komponente

Die Login-**Komponente** besteht, wie jede **Angular-Komponente**, aus einer HTML-Datei, einer **CSS**-Datei (bei diesem Projekt **SASS**) und zwei TypeScript-Dateien. Einer dieser TypeScript-Dateien beinhaltet die Definitionen für das Testen der **Komponente**. Die zweite TypeScript-Datei ist für die eigentliche Funktionalität zuständig. Die **SASS**-Datei ist für die Styles (Aussehen) der **Komponente** zuständig.

```

1 <ion-content>
2   <ion-card id="container">
3     <ion-card-header>
4       <ion-img src="../../assets/logo.png"></ion-img>
5     </ion-card-header>
6     <ion-card-content>
7       <form [formGroup]="ionicForm">
8         <ion-item>
9           <ion-label position="floating">Username</ion-label>
10          <ion-input formControlName="username" placeholder="Enter text"></ion-input>
11        </ion-item>
12        <ion-item>
13          <ion-label position="floating">Password</ion-label>
14          <ion-input formControlName="password" type="password" placeholder="Enter text"></ion-input>
15        </ion-item>
16        <ion-button type="submit" expand="block" (click)="performLogin()">Log in</ion-button>
17      </form>
18    </ion-card-content>
19  </ion-card>
20  <div *ngIf="this.authService.authState == this.authService.authStates.Initializing" class="spinner-overlay">
21    <ion-spinner class="spinner" name="crescent" color="primary"></ion-spinner>
22  </div>
23 </ion-content>
```

Abbildung 46: Login-Komponente HTML-Datei

Wie in Abbildung 46 zu erkennen, wird in der HTML-Datei der **Komponente** das Grundgerüst und die Struktur festgelegt. Es werden zum Beispiel Inputfelder für Username (Zeile 10) und Passwort (Zeile 14) eingefügt. Außerdem wird ein Knopf für den Login eingebaut. Dieser Knopf hat die „(click)“ **Direktive** (siehe Zeile 16). Diese **Direktive** bewirkt, dass die „performLogin“ Funktion bei einem Klick auf den Knopf aufgerufen wird.

Die Funktion ist in der funktionellen TypeScript-Datei der **Komponente** definiert, wie Abbildung 47 zeigt.

```

● ● ●

1 import { Component, OnInit } from '@angular/core';
2 import { FormBuilder, FormControl, FormGroup } from '@angular/forms';
3 import { AlertController } from '@ionic/angular';
4 import { ApiException } from '../shared/client/client';
5 import { AuthService } from '../shared/services/auth.service';
6
7 @Component({
8   selector: 'app-login',
9   templateUrl: './login.component.html',
10  styleUrls: ['./login.component.scss'],
11 })

```

Abbildung 47: Login-Komponente TS-Datei 1/3

Hierfür werden als erstes in der **Komponente** die verwendeten Module importiert (Zeile 1-5). Dann wird mit Hilfe eines **Decorators** die **Komponente** genau definiert (Zeile 7-11). Anschließend wird der Name der **Komponente** (Zeile 8), der Pfad zu der HTML-Datei (Zeile 9) und der Pfad zu der **CSS-** bzw. **SASS**-Datei angegeben.

```

● ● ●

1 export class LoginComponent {
2   public ionicForm: FormGroup;
3   constructor(
4     public authService: AuthService,
5     private alertController: AlertController
6   ) {
7     this.ionicForm = new FormGroup({
8       username: new FormControl(),
9       password: new FormControl(),
10    });
11 }

```

Abbildung 48: Login-Komponente TS-Datei 2/3

Nach dem **Decorator** folgt die eigentliche Klasse. In **Angular** werden **Komponenten** immer als Klassen definiert und exportiert. In der ersten Zeile wird also die Klasse der **Komponente** definiert und auch exportiert. Danach folgt die Definition einer öffentlichen Variable namens „**ionicForm**“ vom **Datentyp** „**FormGroup**“. Diese Variable wird beim Login benötigt, um den eingegebenen Usernamen und das Passwort auszulesen. In den Zeilen 3-11 wird der **Konstruktor** der Klasse definiert. Mithilfe der Parameter des **Konstruktors** werden zwei Klassen namens „**AuthService**“ und „**AlertController**“ injiziert. Der „**AuthService**“ ist der Service, der für die Authentifizierung von Benutzern zuständig ist. Der „**AlterController**“ ist eine von Ionic mitgelieferte **Komponente**, die möglich macht, eine Warnung anzuzeigen. Im Körper des **Konstruktors** (Zeile 7-9) wird die vorher definierte Variable namens „**ionicForm**“ mit einer neuen Instanz eines „**FormGroup-Objektes**“ gefüllt. Dieses „**FormGroup-Objekt**“ beinhaltet die zwei

Eigenschaften „username“ und „password“, welche mit einem FormControl-Objekt instanziert werden. Die Eigenschaften werden beim Login gebraucht, um den Benutzernamen und das Passwort aus den einzelnen Inputfeldern zu lesen. Diese Eigenschaften werden in der HTML-Datei (Abbildung 46) mit der „formControlName“ **Direktive** referenziert.



```

1  public performLogin() {
2    let username = this.ionicForm.value.username;
3    let password = this.ionicForm.value.password;
4    this.ionicForm.reset();
5    this.authService.tryLogin(username, password).subscribe({
6      next: (response) => {
7        this.authService.authState = this.authService.authStates.LoggedIn;
8        this.authService.user = response;
9        this.authService.getUser();
10     },
11      error: async (error: ApiException) => {
12        this.authService.authState = this.authService.authStates.LoggedOut;
13        if (error.status == 400) {
14          this.authService.authState = this.authService.authStates.Unauthorized;
15          const alert = await this.alertController.create({
16            header: 'Wrong credentials',
17            message: 'The given username and/or password are wrong',
18            buttons: ['OK'],
19            cssClass: 'unauthorized-alert',
20          });
21          await alert.present();
22        }
23      },
24    });
25  }

```

Abbildung 49: Login-Komponente TS-Datei 3/3

In Abbildung 49 ist die „perfomLogin“ Funktion zu sehen, welche bei einem Klick auf den Login-Knopf aufgerufen wird. In den ersten Zeilen der Funktion (Zeile 2-4) werden die eingegebenen Daten aus den Inputfeldern extrahiert und in Variablen gespeichert. Anschließend werden die beiden Eingabefelder wieder geleert. Als nächstes wird die „tryLogin“ Funktion des Authentifizierung-Services (siehe 5.2.4.2) aufgerufen. Mit der angehängten „subscribe“ Funktion können die verschiedenen Antworten der „tryLogin“ Funktion abgehandelt werden: Kommt eine positive Antwort ohne Fehler zurück, springt das Programm in den „next“ Teil. In diesem wird der Login-Status auf eingeloggt geändert (Zeile 7), die Benutzervariable mit dem eingeloggten Benutzer gefüllt (Zeile 8) und die „getUser“ Funktion des Authentifizierungs-Service aufgerufen (Zeile 9). Kommt bei der Login-Anfrage ein Fehler zurück, springt das Programm in den „error“ Teil. Dort wird zuerst der Login-Status auf nicht eingeloggt gesetzt. Danach wird überprüft, ob es sich beim zurückgegebenen Fehler um einen Fehler mit dem Code 400 handelt. Ist

dies der Fall, wird zuerst der Login-Status auf unautorisiert gesetzt und eine Meldung auf der **UI** angezeigt, welche besagt, dass es sich um die falschen Anmeldedaten handelt (Zeile 13 bis 21).

#### 5.2.4.2 Authentifizierungsservice

Die wichtigsten Methoden und Variablen für den Status des Benutzers werden in einem Service abstrahiert. Diese Variablen und Methoden werden in unterschiedlichen Teilen der Anwendung gebraucht, wodurch sich ein Service perfekt für diese Aufgabe eignet. Der Service wird innerhalb der Applikation mit „authService“ bezeichnet. In Abbildung 50 kann ein Teil der „authService“ Datei gesehen werden.

```

● ● ●
1 import { Injectable } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { AlertController } from '@ionic/angular';
4 import { Observable } from 'rxjs';
5 import { AuthenticationState } from 'src/app/app.component';
6 import {
7   ApiException,
8   eJudgeClient,
9   LoginRequest,
10  UserDto,
11 } from '../client/client';
12
13 @Injectable({
14   providedIn: 'any',
15 })
16 export class AuthService {
17   public authStates = AuthenticationState;
18   public authState = AuthenticationState.LoggedOut;
19   public user: UserDto | null = null;
20   constructor(
21     private client: eJudgeClient,
22     private router: Router,
23     private alertController: AlertController
24   ) {}
25
26   public tryLogin(username: string, password: string): Observable<UserDto> {
27     this.authState = this.authStates.Initializing;
28     return this.client.user_Login(
29       new LoginRequest({ username: username, password: password })
30     );
31   }

```

Abbildung 50: AuthService 1/2

Dabei werden zuerst die wichtigsten Module importiert (Zeile 1-11). Anschließend wird mit dem „**Injectable**“ **Decorator** der Service in jeder anderen Datei verfügbar gemacht (Zeile 13-15). Als nächstes wird die Klasse des Services deklariert. Die Klasse hat dabei drei öffentliche Variablen, die zum Beispiel in der **Login-Komponente** benötigt und verändert werden. Die Variable „authStates“ (Zeile 17) beinhaltet alle Login-Status. „authState“ (Zeile 18) ist die Variable, welche denn aktuellen Login-Status beinhaltet. Diese wird deshalb am Anfang auf nicht

eingeloggt gesetzt. In der „user“ Variable (Zeile 19) wird der momentan angemeldete User abgespeichert, insofern ein Benutzer angemeldet ist, ansonsten ist die Variable **null**. Von Zeile 20-24 spannt sich der **Konstruktor** des Services. Dabei werden drei Klassen injiziert: Die Klasse „client“ (Zeile 21), welche die Definitionen der **Endpoints** des Backends enthält, die Klasse „router“ (Zeile 22), die für die programmatische Navigation in der Anwendung gebraucht wird und die Klasse „alertController“, welche für das Anzeigen von Warnungen auf der Oberfläche gebraucht wird.

In den Zeilen 26-31 befindet sich die „tryLogin“ Funktion, die von der Login-**Komponente** aufgerufen wird. Es wird dabei zuerst der Login-Status auf initialisierend gesetzt (Zeile 27) und danach der **Endpoint** „user\_Login“ mit den eingegeben Logindaten aufgerufen. Das vom **Endpoint** zurückgegebene Objekt wird auch als Rückgabewert der Funktion zurückgegeben.

```

● ● ●

1  public getUser() {
2    this.authState = AuthenticationState.Initializing;
3    this.client.user_CheckAuth().subscribe({
4      next: (user) => {
5        if (this.router.url == '/') {
6          this.router.navigate(['user-ui']);
7        }
8        this.user = user;
9        this.authState = AuthenticationState.LoggedIn;
10   },
11   error: async (error: ApiException) => {
12     this.authState = AuthenticationState.Unauthorized;
13     if (error.status == 401) {
14       const alert = await this.alertController.create({
15         header: 'Session Expired',
16         message: 'You have been logged out. Please log in again',
17         buttons: ['OK'],
18         cssClass: 'unauthorized-alert',
19       });
20       await alert.present();
21       this.router.navigate(['']);
22       return true;
23     } else if (error.status == 0) {
24       this.authState = AuthenticationState.ConnectionFailed;
25     }
26   },
27 });
}

```

Abbildung 51: AuthService 2/2

Die „getUser“ Funktion (siehe Abbildung 51) hat die Aufgabe, den aktuellen **Bearer-Token** zu überprüfen und gegebenenfalls den Login-Status zu ändern. Dazu wird der „user\_CheckAuth“ **Endpoint** aufgerufen (Zeile 3). Kommt kein Fehler vom **Endpoint** zurück, wird die aktuelle Route des Users angeschaut: Sollte die Route „/“ sein, wird sie auf „/user-ui“ geändert (Zeile 5-7). Anschließend wird die „user“ Variable mit dem aktuellen Benutzer gefüllt und der Login-Status auf eingeloggt gesetzt. Gibt der **Endpoint** einen Fehler zurück, wird zuerst der

Login-Status auf nicht eingeloggt gesetzt (Zeile 12). Bei einem Fehlercode von 401 wird eine Warnung angezeigt, welche besagt, dass die derzeitige Session nicht mehr gültig ist. Zudem wird auf die Route „/“ navigiert (Zeile 13-22). Bei einem Fehlercode von **null** wird der Login-Status auf „Verbindung Fehlgeschlagen“ gesetzt (Zeile 23-25).

## 5.3 Deployment

Im folgenden Teil wird das Deployment von der Continuous Integration per **Github Actions** bis zur Bereitstellung der Applikation als Docker Container beschrieben.

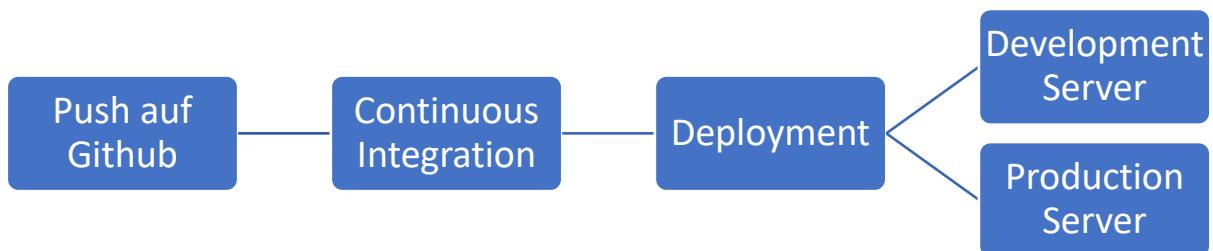


Abbildung 52: Ablauf Deployment

### 5.3.1 Continuous Integration

Die Continuous Integration wird mit dem **Build** Tool Nuke sowie mit dem **GitHub**-Service **GitHub Actions** realisiert. Hierbei wird bei jedem „commit“, welcher auf das Repository übertragen wird, ein Workflow ausgelöst. Jeder Workflow Ablauf besteht wiederum aus mehreren Schritten.

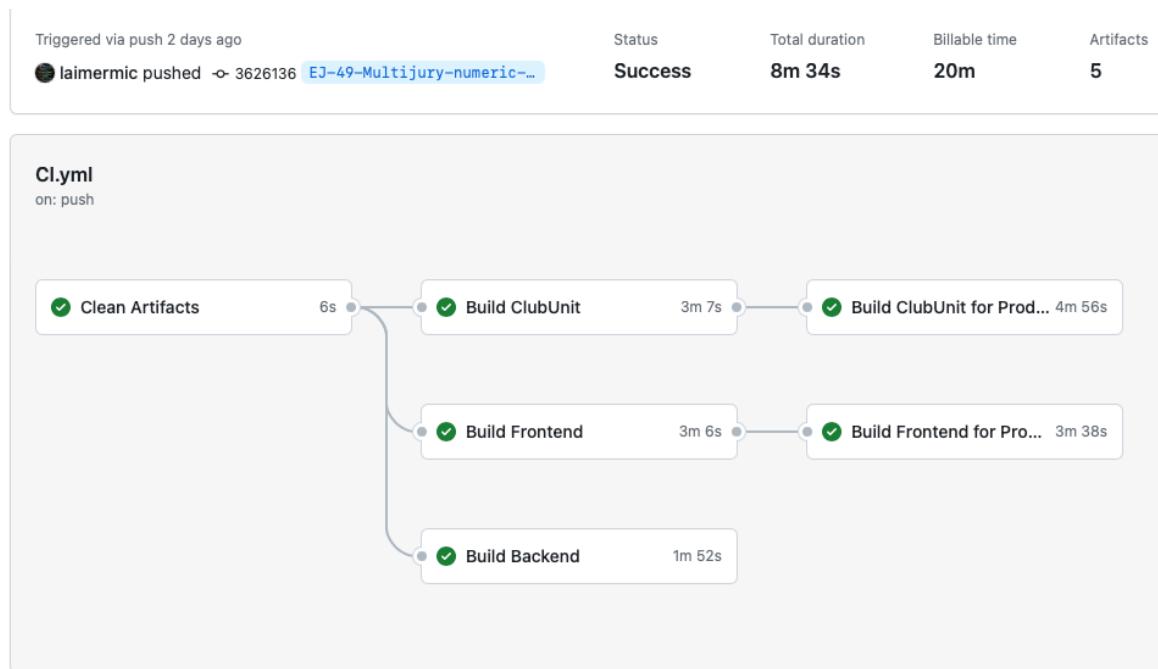


Abbildung 53: Ablauf Continuous Integration

### 5.3.1.1 Alte Artefakte löschen

Da **GitHub** nur begrenzten Speicher zur Speicherung von Artefakten bereitstellt (für Schüler 2 GB)<sup>26</sup>, müssen alte Artefakte gelöscht werden, um das Hochladen von neuen **Build**-Artefakten nicht zu blockieren. Hierzu wird ein Open Source Workflow verwendet, welcher eine einfache http-Anfrage an die **GitHub** API sendet, mit dem Befehl, alle Artefakte zu löschen.

### 5.3.1.2 Backend Build

Beim Backend **Build** über das **Build**-Tool Nuke werden alle externen Abhängigkeiten aufgelöst und anschließend der Code in Maschinencode kompiliert. Im Anschluss werden die kompilierten Dateien in ein spezielles Verzeichnis kopiert, von welchem aus sie dann als Artefakte zu **GitHub** hochgeladen werden.

<sup>26</sup> vgl. *Pricing · Plans for every developer · GitHub*, o. J.

```
1 ► Run ./build.cmd BackendBuild
4 GNU bash, version 5.1.16(1)-release (x86_64-pc-linux-gnu)
5 Microsoft (R) .NET SDK version 8.0.202
6
7 [REDACTED]
8 [REDACTED]
9 [REDACTED]
10 [REDACTED]
11 [REDACTED]
12 [REDACTED]
13
14 NUKE Execution Engine version 6.0.1 (Linux,.NETCoreApp,Version=v6.0)
15
16 22:58:39 [INF] > /usr/bin/dotnet /home/runner/.nuget/packages/gitversion.tool/5.10.3/tools/net5.0/any/gitversion.dll /nocache
17 ► Info
25 ► Clean
36 ► RestoreBackend
44 ► CompileBackend
455 ► PublishBackend
465 ► Warnings & Errors
664
665
666 

| Target         | Status    | Duration |
|----------------|-----------|----------|
| Info           | Succeeded | < 1sec   |
| Clean          | Succeeded | < 1sec   |
| RestoreBackend | Succeeded | 0:18     |
| CompileBackend | Succeeded | 0:20     |
| PublishBackend | Succeeded | 0:07     |


673
674 Total 0:47
675
676
677 Build succeeded on 03/29/2024 22:59:27. \ (^.^) /
```

Abbildung 54: Ablauf Backend Build

### 5.3.1.3 Build der Administrationsoberfläche

Die Transpilierung der Administrationsoberfläche ist in zwei Teile aufgeteilt (siehe Abbildung 53): Im ersten Teil wird das Frontend für den Development-Server transpiliert. Dabei werden Variablennamen nicht gekürzt, und auf Performanceoptimierungen verzichtet, welche das Debuggen im Browser erschweren können. Im zweiten Teil wird das Frontend für ein Deployment auf den Produktionsserver transpiliert. Dabei wird der Code optimiert, um eine möglichst kleine Bundle-Size, sowie eine möglichst hohe Performance für die Nutzerinnen und Nutzer zu ermöglichen. Wie aus Abbildung 53 ersichtlich, ist die Transpilierung für die Produktion abhängig vom Ergebnis der Transpilierung des Development-Servers. Das heißt, schlägt die Transpilierung für den Development-Server fehl, wird die Transpilierung für die Produktion nicht ausgeführt. Bei beiden Vorgängen werden zuerst die Abhängigkeiten an npm-**Bibliotheken** aufgelöst, Frontend von TypeScript in JavaScript transpiliert und anschließend als Artefakt auf **GitHub** hochgeladen.

```
1 ► Run ./build.cmd FrontendBuild
4 GNU bash, version 5.1.16(1)-release (x86_64-pc-linux-gnu)
5 Microsoft (R) .NET SDK version 8.0.202
6
7 [REDACTED]
8 [REDACTED]
9 [REDACTED]
10 [REDACTED]
11 [REDACTED]
12 [REDACTED]
13
14 NUKE Execution Engine version 6.0.1 (Linux,.NETCoreApp,Version=v6.0)
15
16 22:58:47 [INF] > /usr/bin/dotnet /home/runner/.nuget/packages/gitversion.tool/5.10.3/tools/net5.0/any/gitversion.dll /nocache
17 ► Info
25 ► Clean
36 ► RestoreFrontend
56 ► CompileFrontend
147 ► PublishFrontend
148 ► Warnings & Errors
157
158 =====
159 Target      Status      Duration
160
161 Info          Succeeded    < 1sec
162 Clean        Succeeded    < 1sec
163 RestoreFrontend    Succeeded    0:30
164 CompileFrontend    Succeeded    0:27
165 PublishFrontend    Succeeded    < 1sec
166
167 Total                    0:58
168
169
170 Build succeeded on 03/29/2024 22:59:46. \ (^o^) /
```

Abbildung 55: Ablauf Frontend Build

#### **5.3.1.4 Build der Vereinsoberfläche**

Beim **Build** der Vereinsoberfläche ist der Ablauf im Grunde genommen derselbe, wie beim **Build** der Verwaltungsoberfläche (vgl. 5.3.1.3). Auch hier wird in zwei Schritten die Vereinsoberfläche zuerst für einen Development-Server und anschließend für einen Produktionsserver transpiliert.

### **5.3.2 Artefakte**

Nach dem komplett abgeschlossenen Workflow werden von **GitHub** die Artefakte bereitgestellt. Diese Artefakte beinhalten die fertig kompilierten/transpilierten **Komponenten** der Anwendung. Die Artefakte für die Administratoroberfläche und die Vereinsoberfläche bestehen aus einer index.html und mehreren JS-Dateien und lassen sich direkt von jedem Webserver (z. B. Apache, nginx, etc.) bereitstellen<sup>27</sup>. Das Artefakt für das Backend muss durch einen dotnet Server ausgeführt werden.

<sup>27</sup> vgl. *Angular—Deployment*, o. J.

| Artifacts               |         |
|-------------------------|---------|
| Produced during runtime |         |
| Name                    | Size    |
| Backend                 | 116 MB  |
| Frontend                | 24.6 MB |
| FrontendProd            | 4.18 MB |
| clubUnit                | 23.7 MB |
| clubUnitProd            | 4.2 MB  |

Abbildung 56: Artefakte nach einem Workflow

### 5.3.3 Continuous Delivery

Das Continuous Delivery besteht aus zwei Workflows mit den Namen „deploy-dev“ und „deploy-prod“. Der einzige Unterschied zwischen den beiden Workflows ist, dass „deploy-dev“ auf einen Development-Container bereitgestellt, während „deploy-prod“ auf einen Produktions-Container bereitstellt. Ansonsten sind beide Workflows identisch.

Nach dem manuellen Starten des Workflows werden als erstes die Docker-Container, auf welche bereitgestellt wird, gestoppt. Die Kommunikation für diesen Befehl, sowie die spätere Übertragung der Artefakte wird über eine SSH-Verbindung abgewickelt, welche über einen private Key gesichert wird. Dieser private Key ist in den Umgebungsvariablen von **GitHub** gespeichert. Hierbei wird der SSH-Key aus den **GitHub**-Umgebungsvariablen zuerst auf in eine Datei geschrieben und in der SSH-Instanz registriert. Anschließend wird eine vorgefertigte SSH-Konfigurationsdatei vom Repository in das SSH-Verzeichnis kopiert. Weiters wird der Server im Vorhinein zu den vertrauenswürdigen Hosts hinzugefügt. Dadurch kann anschließend eine SSH-Verbindung aufgebaut und Befehle abgesetzt werden.

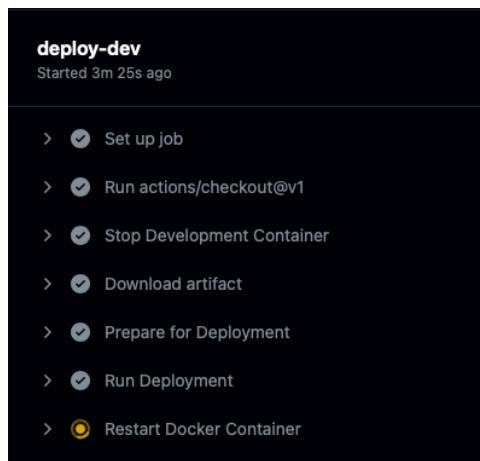


Abbildung 57: Ablauf eines Deployments

```
● ● ●
1 eval `ssh-agent`
2 mkdir ~/.ssh
3 echo "Adding SSH Private Key for Production Server..."
4 echo "$PRIVATE_KEY" > ~/.ssh/id_rsa
5 chmod 0600 ~/.ssh/id_rsa
6 echo $PRIVATE_PASS | SSH_ASKPASS=/bin/cat setsid -w ssh-add ~/.ssh/id_rsa
7 echo "Authorizing Servers for Connection..."
8 cp ./deploy/config ~/.ssh/config
9 echo "$SSH_KNOWN_HOSTS" > ~/.ssh/known_hosts
10 echo "Authorization complete!"
11 ssh prod 'cd ejjudgedev; docker-compose down --rmi local; rm -R -f Frontend > /dev/null; rm -R -f Backend > /dev/null;'
```

Abbildung 58: SSH-Skript zum Stoppen des Docker Containers

Anschließend wird das neueste Artefakt (siehe 5.3.2) von **GitHub** heruntergeladen, sowie die Konfigurationsdateien (welche die **URL** des Backends enthalten) je nach Deployment Umgebung (Development oder Production), angepasst. Danach werden die Artefakte über rsync auf den Server übertragen. Zum Schluss werden die Docker Container neu gestartet.

### 5.3.4 Hetzner Server

Der virtuelle Server, welcher bei Hetzner gemietet wurde, läuft mit der Linux-Distribution Ubuntu und stellt 40 GB Festplattenspeicher, sowie 4 GB Arbeitsspeicher und eine statische IPv4 bzw. IPv6 Adresse zur Verfügung. Der Zugriff erfolgt über eine SSH-Verbindung, wobei beim Deployment die Authentifizierung über einen Private Key erfolgt, welcher in den Environment Secrets des **GitHub**-Repositorys abgespeichert ist.

### 5.3.5 Docker

Auf dem Hetzner Server wurden vier Docker-Container eingerichtet, welche im folgenden Abschnitt beschrieben werden. Die Verwaltung läuft dabei über eine docker-compose.yml Datei.

#### 5.3.5.1 Reverse Proxy

Alle Anfragen an den Server werden durch einen Reverse Proxy beantwortet und je nach angefragter Domain orchestriert. Hierfür wird das Docker Image *jwilder/nginx-proxy* verwendet. Über die Umgebungsvariable *VIRTUAL\_HOST* kann jedem Container eine Domain zugewiesen werden (siehe Abbildung 59). Anfragen an diese Domain werden dann an diesen Container weitergeleitet.

The diagram illustrates the structure of a Docker Compose file. The file defines three services: ejudgeDev, ejudgeFrontend, and clubUnit. The ejudgeDev service is highlighted with a red border and labeled 'Backend'. The ejudgeFrontend service is highlighted with a green border and labeled 'Verwaltungsoberfläche'. The clubUnit service is highlighted with a blue border and labeled 'Vereinsoberfläche'.

```
version: '3.7'
services:
  ejudgeDev:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: ejudge_Backend
    ports:
      - "49250:49250"
    restart: unless-stopped
    environment:
      - VIRTUAL_HOST=api.ejudgedev.terrex.at
      - MongoDBConnectionString=${MongoDBConnectionString}
      - BearerToken=${BearerToken}
      - ClubUnitBearerToken=${ClubUnitBearerToken}
      - MongoDBDatabase=${MongoDBDatabase}
      - MailUser=${MailUser}
      - MailPass=${MailPass}
    networks:
      - proxy-network
  ejudge_Frontend:
    image: nginx:latest
    container_name: ejudge_Frontend
    ports:
      - "8080:80"
    restart: unless-stopped
    volumes:
      - ./Frontend:/usr/share/nginx/html
      - ./conf/nginx.conf:/etc/nginx/conf.d/default.conf
    environment:
      - VIRTUAL_HOST=ejudgedev.terrex.at
    networks:
      - proxy-network
  clubUnit:
    image: nginx:latest
    container_name: clubUnit_Frontend
    ports:
      - "8081:80"
    restart: unless-stopped
    volumes:
      - ./clubUnit:/usr/share/nginx/html
      - ./conf/clubUnit.conf:/etc/nginx/conf.d/default.conf
    environment:
      - VIRTUAL_HOST=clubunitdev.terrex.at
    networks:
      - proxy-network
      name: proxy-network
      external: true
```

Abbildung 59: Docker compose Datei



```
version: "3"
services:
  nginx-proxy:
    container_name: terrex_proxy
    image: jwilder/nginx-proxy
    restart: unless-stopped
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - "/etc/nginx/vhost.d"
      - "/usr/share/nginx/html"
      - "/var/run/docker.sock:/tmp/docker.sock:ro"
      - "./certs:/etc/nginx/certs:rw"
      - "./max_upload_size.conf:/etc/nginx/conf.d/client_max_body_size.conf:rw"
    networks:
      - proxy-network
  networks:
    proxy-network:
      driver: bridge
      name: proxy-network
```

Abbildung 60: Docker compose Datei des Reverse Proxys

In der oben abgebildeten „docker compose“ Datei wird die „docker.sock“ Datei des Servers an den Container weitergegeben. Dies erlaubt dem Reverse Proxy laufende Docker Container zu beobachten. Aus diesen Informationen wird anschließend eine Konfigurationsdatei mit den entsprechenden virtuellen Hosts (Domains) erstellt. Über die „docker compose“ Datei werden außerdem SSL-Zertifikate, sowie erweiterte Konfigurationen über Docker-Volumes übergeben.

### 5.3.5.2 Backend

Das Backend wird durch ein eigenes Docker Image bereitgestellt, welches bei jedem Neustart über docker-compose neu erstellt wird. Beim Erstellen des Images wird das ASP .NET Core Image von Microsoft als Basis verwendet. Anschließend werden verschiedene benötigte Abhängigkeiten bzw. **Bibliotheken** installiert, welche unter anderem für die PDF-Erstellung wichtig sind. Außerdem wird das Backend Artefakt (welches vorher von **GitHub** übertragen wurde) in das Docker Image kopiert und das Environment bzw. die exposed Ports gesetzt.

```
● ● ●
1 FROM mcr.microsoft.com/dotnet/aspnet:6.0
2 COPY ./Backend /dist
3 WORKDIR /dist
4 LABEL author="Michael Laimer"
5 RUN echo "deb http://deb.debian.org/debian stable main contrib non-free" > /etc/apt/sources.list \
6     && echo "ttf-mscorefonts-installer msttcorefonts/accepted-mscorefonts-eula select true" | debconf-set-selections \
7     && apt-get update \
8     && apt-get install -y \
9         ttf-mscorefonts-installer \
10    && apt-get clean \
11    && apt-get autoremove -y \
12    && rm -rf /var/lib/apt/lists/*
13 RUN apt-get update && apt-get install -y libgdplus
14 RUN ln -sf /usr/share/zoneinfo/posix/Europe/Vienna /etc/localtime
15 ENV ASPNETCORE_URLS=http://api.ejudgedev.terrex.at:49250
16 ENV ASPNETCORE_ENVIRONMENT=Development
17
18 EXPOSE 49250
19 ENTRYPOINT dotnet /dist/eJ.Backend.Web.dll
```

Abbildung 61: Dockerfile des Backends

Dieses Image wird bei jedem Deployment neu erstellt. Dadurch ist das Backend auf keinerlei Dateien außerhalb des Images angewiesen. In der „docker-compose.yml“ Datei werden anschließend einige Umgebungsvariablen (Secrets) des Hosts an den Container übergeben (siehe 5.3.5). Die Umgebungsvariable *VIRTUAL\_HOST* signalisiert dem Reverse Proxy, Anfragen an die URL „api.ejudgedev.terrex.at“ an diesen Container weiterzuleiten.

### 5.3.5.3 Administrations- und Vereinsoberfläche

Die Administrator- und Vereinsoberfläche besteht jeweils aus einem einfachen nginx-Server, welcher das Artefakt der Administratoroberfläche als Root-Verzeichnis verwendet. Hierbei wird kein eigenes Image erstellt, sondern das, in der Docker Registry verfügbare, nginx-Image verwendet. Ein Docker Volume lässt das HTML-Rootverzeichnis auf das Artefakt zeigen.

## 6 Testphase

In der Testphase wurde die entwickelte Applikation gründlich auf Fehler durchsucht und auf Verbesserungsvorschläge eingegangen. Vor dem ersten richtigen Einsatz bei einem Wettbewerb wurde die Anwendung durch Trainerinnen und Trainer des TSZ Dornbirn getestet. Anschließend wurde auf das Feedback der Trainer bei der Landesmeisterschaft in Lustenau eingegangen. Als letzter Test wurde die Staatsmeisterschaft in Wien genutzt.

### 6.1 Testung durch Trainerinnen und Trainer

Vor dem ersten Einsatz bei einem richtigen Wettkampf, wurde das Programm von mehreren Trainerinnen und Trainern des TSZ Dornbirn getestet. Diese führten die Testung durch, indem sie die Wertungsblätter ihrer Teams für die Landesmeisterschaft eingaben. Ein Verbesserungsvorschlag, der dabei aufkam, war ein Auswahlraster für die Bodeneingabe, mit welchem man die Liste der **Elemente** nach Kategorie filtern kann. Dieser Vorschlag wurde noch vor der Landesmeisterschaft implementiert (siehe Abbildung 62).

| Elements                    |        |                         |                 |       |               |   |
|-----------------------------|--------|-------------------------|-----------------|-------|---------------|---|
|                             | ALL    | DB                      | SB              | HB    | J             | A |
| <input type="text"/> Search |        |                         |                 |       |               |   |
| Image                       | Code   | Description             | Type            | Value | Allowed in DS |   |
| →○                          | DB201  | Forward pirouette 360°  | Dynamic Balance | 0.20  | false         |   |
| →○                          | DB601  | Forward pirouette 540°  | Dynamic Balance | 0.60  | false         |   |
| →○                          | DB801  | Forward pirouette 720°  | Dynamic Balance | 0.80  | false         |   |
| →○                          | DB1001 | Forward pirouette 900°  | Dynamic Balance | 1.00  | false         |   |
| ←○                          | DB202  | Backward pirouette 360° | Dynamic Balance | 0.20  | false         |   |

Abbildung 62: Möglichkeit zur Filterung der Elemente nach Kategorie

Des Weiteren gab es den Vorschlag, für die Trampolin- und Tumbling-Eingabe, Knöpfe für das Kopieren und Einfügen von **Elementen** bzw. **Serien** hinzuzufügen. Dies wurde ebenfalls schnellstmöglich realisiert, um damit vor dem Einsatz bei der Landesmeisterschaft fertig zu werden.

## 6.2 Einsatz bei der Landesmeisterschaft

Am 21. Oktober 2023 fanden in Lustenau die Landesmeisterschaften im Team-Turnen statt. Zur Eingabe der Wertungsblätter wurde die entwickelte Anwendung verwendet. Zur Sicherheit wurden die Vereine gebeten, die Wertungsblätter zusätzlich in analoger Form zum Wettkampf mitzubringen.



Abbildung 63: Vorarlberger Landesmeisterschaften im Team-Turnen 2023

[https://tszdornbirn.at/wp-content/gallery/team-turnen-landesmeisterschaft-2023-siegerehrung/TT\\_LM\\_win\\_23\\_001.jpg](https://tszdornbirn.at/wp-content/gallery/team-turnen-landesmeisterschaft-2023-siegerehrung/TT_LM_win_23_001.jpg)

Bei der Landesmeisterschaft wurde angemerkt, dass es bei der Tumbling-Eingabe mühsam sei, eine **Serie** zu bearbeiten, falls ein Fehler gemacht wurde, bzw. falls sich die **Serie** geändert haben sollte. Zur Zeit der Landesmeisterschaft gab es nämlich noch keine Möglichkeit, die **Elemente** in einer **Serie** zu bearbeiten, wodurch zum Ändern einer **Serie** die ganze **Serie** neu gemacht werden musste. Darum wurde im Anschluss an die Landesmeisterschaft genau diese Bearbeitungsmöglichkeit in die Applikation aufgenommen.

## 6.3 Einsatz bei der Staatsmeisterschaft

Die Staatsmeisterschaft am 25. November 2024 in Wien diente als nächste und letzte Testung für die Applikation. Die Verbesserungsvorschläge wurden in der Zwischenzeit teilweise in die Applikation implementiert. Der Ablauf war ähnlich zur Landesmeisterschaft (siehe 6.2). Die Rückmeldung aller Beteiligten war äußerst positiv. Als Feedback war bei der Staatsmeisterschaft vor allem die Möglichkeiten zur Erweiterung der digitalen Eingabe auf andere Bereiche (z. B. Musik) aufgekommen.

## 7 Fazit

Anhand der vorliegenden Diplomarbeit wurde gezeigt, dass eine automatisierte Erstellung von Wertungsblättern bei Team Turn Wettkämpfen möglich ist. Diese Applikation wurde mit Hilfe von Angular und Ionic für das Frontend sowie ASP .NET Core für das Backend realisiert. Für die Speicherung der Daten wurde eine MongoDB verwendet.

Bei der Planung der Anwendung kam das Problem auf, dass nicht nur eine Eingabeoberfläche benötigt wird, sondern auch eine Möglichkeit verschiedene Informationen zu Verwalten. Deshalb wurde das Frontend in zwei geteilt.

Die Datenstruktur der verschiedenen Entitäten waren oft komplex und haben sich gegenseitig referenziert, was im Angesicht der Schemafreiheit von MongoDB ein Problem darstellte. Diese Schwierigkeit konnte durch die Entwicklung eines generischen Konnektors zur Datenbank, welcher die Referenzen automatisch auflöste, bewältigt werden.

Im Entwicklungsprozess stellte sich heraus, dass die Bibliothek, welche zur PDF-Erstellung verwendet wurde, nur auf Windows funktionsfähig ist. Konkret bedeutete dies, dass eine andere Bibliothek zum Erstellen und Rendern der PDF-Dateien verwendet werden musste.

Als der Vorschlag aufkam, die Anwendung bei der Landesmeisterschaft zu testen, war noch nicht klar, ob diese bis zu dem Termin weit genug entwickelt werden kann. Die Eingabe des Trampolins und des Tumblings waren kein Problem. Bei der Eingabe für das Bodenwertungsblatt hingegen war nicht sicher, ob dies bis zu dem vereinbarten Termin umgesetzt werden kann. Wie sich aber herausstellte konnte die Bodeneingabe aber schneller umgesetzt werden als gedacht. Die Applikation konnte deshalb bei der Landesmeisterschaft getestet werden.

Nach dem Einsatz der Anwendung bei der Landesmeisterschaft ist viel positives Feedback zurückgekommen. Vom veranstaltenden Verein kam die Rückmeldung, dass der gesparte Aufwand vom Kopieren und Sortieren der Wertungsblätter groß sei. Von den Trainerinnen und

Trainer der Vereine kam als Feedback, dass die Oberfläche für die Eingabe sehr einfach zu bedienen sei und es gab bei der Landesmeisterschaft keine großen Probleme

Weiters wurde angefragt, ob das Entwicklungsteam die Anwendung weiter betreuen und entwickeln möchte, um die Organisation von Wettkämpfen weiter zu erleichtern. Angedacht wäre hierbei auch eine Noteneingabe für die Wertungsrichter, sowie eine automatisierte Erstellung von Ergebnislisten. Dieser Vorschlag wurde angenommen und die ersten Pläne für die Implementierung dieser Anregungen nach diesem Projekt wurden erstellt.

Die Problemstellung konnte somit erfolgreich bearbeitet werden. Die dabei entwickelte Applikation konnte außerdem bei den Vorarlberger Landesmeisterschaften, sowie bei den österreichischen Staatsmeisterschaften im Team Turnen ausreichend getestet werden.

## 8 Technologien

Im folgenden Kapitel wird genauer auf die Technologien eingegangen, die für die Umsetzung des Projektes verwendet wurden.

### 8.1 C#

C# ist eine typsichere, plattformübergreifende Programmiersprache, welche 2001 von Microsoft veröffentlicht wurde. C# basiert auf dem Konzept der objektorientierten Programmierung. Durch die hohe Typsicherheit werden Kompatibilitätsprobleme oft bereits beim Kompilieren und nicht erst in der Runtime bemerkt. Beim Kompilieren wird der Code nicht sofort in Maschinensprache, sondern zuerst in eine Zwischensprache übersetzt.<sup>28</sup>

#### 8.1.1 Records

Records in C# sind Klassen oder Strukturen, welche eine spezielle Syntax zur Definition von Datenmodellen bereitstellen. Wenn für einen Record ein **Konstruktor** definiert wird, erstellt der Compiler automatisch passende „public **Properties**“ zu den **Parametern** im **Konstruktor**. Diese Properties haben keinen „Setter“ und können daher, außer während der Initialisierung, nicht geändert werden.<sup>29</sup>

#### 8.1.2 Attribute

Durch Attribute können Informationen über eine Property oder Klasse in deklarativer Form mit dem Code verknüpft werden. Das Attribut wird hierbei in eckigen Klammern über der Definition der Klasse/Methode/Property eingefügt. Dabei können dem **Konstruktor** auch Argumente übergeben werden.

Beispielsweise gibt es das „Obsolete-Attribut“. Mit ihm können Klassen, Methoden und Properties als veraltet markiert werden. Bei Verwendung des „Obsolete-Attributs“ wird dann beim

---

<sup>28</sup> vgl. *C-Sharp*, 2024.

<sup>29</sup> vgl. *Records in C#—C# | Microsoft Learn*, o. J.

Komplizieren eine Warnung ausgegeben, welche besagt, dass das Ziel veraltet ist und nicht mehr verwendet werden sollte.<sup>30</sup>

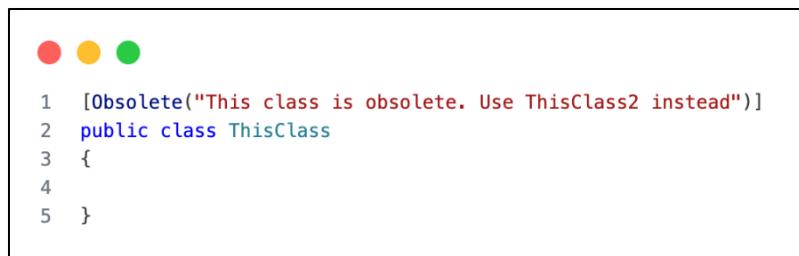


Abbildung 64: Beispielanwendung des Attributs Obsolete

## 8.2 ASP .NET Core

Bei ASP .NET Core handelt es sich um ein plattformübergreifendes Framework, mit welchem Webentwickler moderne, cloudfähige Apps entwickeln können. ASP .NET Core wurde im Juni 2016 veröffentlicht und ist als Open Source verfügbar. Weiters kann .NET auf so gut wie jeder Plattform ausgeführt werden, was vorteilhaft bei der Entwicklung von Anwendungen ist.

Die Blazor **Komponente** ist ein auf „WebAssembly“ aufbauender Teil von ASP .NET Core. Im November 2023 ist Version 8 von ASP .NET Core veröffentlicht.<sup>31</sup>

## 8.3 MongoDB

MongoDB ist eine NoSQL Datenbank, welche 2007 ins Leben gerufen wurde und im Februar 2009 zum ersten Mal veröffentlicht wurde. MongoDB ist eine dokumentenorientierte Datenbank, die das Abspeichern, Abrufen und Aggregieren von **JSON**-Dokumenten ermöglicht.<sup>32</sup> MongoDB verfügt außerdem über eine dynamische Schemastruktur und ist aufgrund der Speicherung als **JSON**- bzw. **BSON** nicht an ein zuvor definiertes Datenbankschema gebunden.<sup>33</sup>

---

<sup>30</sup> vgl. *Tutorial: Definieren und Lesen von benutzerdefinierten Attributen—C# | Microsoft Learn*, o. J.

<sup>31</sup> vgl. *ASP.NET Core*, 2024.

<sup>32</sup> vgl. *4\_NoSQL\_Document\_storage\_v1.3.pdf*, o. J., S. 13.

<sup>33</sup> vgl. *Was ist MongoDB und wie funktioniert es? | Pure Storage*, o. J.

### 8.3.1 GridFS

MongoDB GridFS ist eine Funktion von MongoDB, die Dateien in der Datenbank als **Array** aus Bytes abbildet. MongoDB teilt hierbei die Datei in „Chunks“ auf, die jeweils höchstens 255 kB groß sind. Zur Speicherung der Dateien werden zwei „Collections“ verwendet: In der ersten „Collection“ wird jede Datei registriert und mit einer eindeutigen ID identifiziert. In der zweiten „Collection“ werden die einzelnen „Chunks“ der Dateien gespeichert. Dabei wird jeder „Chunk“ seiner zugehörigen Datei aus dem Dateiregister zugeordnet.<sup>34</sup>

## 8.4 TypeScript

„TypeScript“ ist eine Programmiersprache, die auf JavaScript aufbaut und diese erweitert. Der Vorteil von „TypeScript“ gegenüber „JavaScript“ ist das überarbeitete Typen-System, welches in „JavaScript“ zu Fehlern führen kann. Das geänderte Typen-System bietet zusätzlichen Syntax, mit dem Variablen Typen zugeschrieben bekommen können.

„TypeScript“ wird vor dem Kompilieren zu „JavaScript“ konvertiert. Dadurch kann „TypeScript“ überall ausgeführt werden, wo auch „JavaScript“ ausgeführt werden kann.<sup>35</sup>

### 8.4.1 Decorators

**Decorators** sind ein Feature von „TypeScript“, welches es ermöglicht, Klassen oder Klassenmitgliedern zusätzliche Annotationen und Metadaten zu geben. Diese werden dabei mit einem ‚@‘ und einem Ausdruck beschrieben (z. B. @Pipe). Der Ausdruck muss dabei mit der Bezeichnung einer Funktion übereinstimmen. Diese wird dann während der Laufzeit mit den Informationen des **Decorators** aufgerufen.<sup>36</sup>

---

<sup>34</sup> vgl. *GridFS — MongoDB Manual*, o. J.

<sup>35</sup> vgl. *TypeScript—JavaScript With Syntax For Types.*, o. J.

<sup>36</sup> vgl. *TypeScript: Documentation—Decorators*, o. J.

## 8.5 Angular

**Angular** ist ein Framework, das auf TypeScript aufbaut und mit Hilfe von **Komponenten** die Entwicklung von Webapplikationen in alle Größen ermöglicht. **Angular** stellt außerdem eine Reihe von **Bibliotheken** zur Verfügung, welche eine große Variation von Funktionalitäten beinhalten, wie zum Beispiel Routing, Client-Server-Kommunikation, Animationen und Vieles mehr.<sup>37</sup>

### 8.5.1 Komponenten

**Komponenten** sind die grundlegenden Bausteine beim Erstellen einer **Angular**-Applikation. **Komponenten** helfen dabei, die Struktur der Anwendung zu teilen und somit eine leichtere und überschaubarere Organisation zu schaffen. Eine **Komponente** besteht dabei aus einem Template, in welchem mit Hilfe von HTML definiert wird, was im **DOM** gerendert werden soll.<sup>38</sup>

### 8.5.2 Direktiven

**Direktiven** bieten die Möglichkeit das Verhalten von HTML-Elementen und **Angular-Komponenten** zu erweitern. Es gibt viele verschiedene **Direktiven**, die vielfältige Funktionen bieten. Es gibt beispielsweise **Direktiven** für das konditionelle Rendern, Rendern von Listen oder die dynamische Vergabe von **CSS**-Styles. Es gibt Fälle, in denen die von **Angular** mitgelieferten **Direktiven** nicht ausreichen. Daher können auch selbst **Direktiven** für spezielle Situation erstellt werden.<sup>39</sup>

### 8.5.3 Services

Ein Service ist eine in **Angular** integrierte Möglichkeit, Logik zwischen verschiedenen **Komponenten** zu teilen. Services helfen dabei, die Modularität und Wiederverwendbarkeit zu erhöhen und Funktionalitäten, die nicht einer **Komponente** direkt zugeschrieben werden können,

---

<sup>37</sup> vgl. *Angular—What is Angular?*, o. J.

<sup>38</sup> vgl. *Angular—What is Angular?*, o. J., Abschn. Components.

<sup>39</sup> vgl. *Angular—What is Angular?*, o. J., Abschn. Directives.

abzuschotten. Sie werden zum Beispiel für Aufgaben wie das Abrufen von Daten vom Server, oder das Überprüfen von Benutzereingaben verwendet.<sup>40</sup>

### 8.5.4 Pipes

Pipes sind Funktionen, welche dabei helfen, Daten in die richtige Form zu transformieren. Es handelt sich dabei um Funktionen, die Eingabewerte entgegennehmen und einen Ausgabewert liefern. Pipes sind nützlich, da diese nur einmal für die ganze Applikation deklariert werden müssen und überall verwendet werden können. Es gibt vorgefertigte Pipes, die bei **Angular** mitgeliefert werden, wie zum Beispiel „UpperCasePipe“, welche den eingehenden „String“ nimmt, alles großschreibt und den Wert dann wieder zurückgibt. Ebenso ist es möglich, selbst Pipes für spezifische Datentransformationen zu erstellen.<sup>41</sup>

## 8.6 Ionic

Ionic ist ein Open Source Toolkit für die Erstellung von modernen und qualitativ hochwertigen **UIs**. Eine Codebasis, die mit Ionic entwickelt wird, kann als Webapplikation, aber auch als native App für Android und iOS kompiliert werden. Für die Entwicklung mit Ionic können verschiedene JavaScript-Frameworks als Grundlage verwendet werden. Standardmäßig werden die Frameworks **Angular**, **React** und **Vue** unterstützt. Mit Ionic wird eine umfangreiche **Bibliothek** von mehr als 100 **UI-Komponenten** mitgeliefert, was die Entwicklung von benutzerfreundlichen Oberflächen vereinfacht.<sup>42</sup>

## 8.7 SASS

**SASS** steht für „Syntactically awesome style sheets“ und ist eine Stylesheet-Sprache. **SASS** bietet dabei zusätzliche Funktionen und überarbeitet vorhandene Features mit neuer Syntax, die bei normalem **CSS** nicht vorhanden sind.

---

<sup>40</sup> vgl. *Angular—What is Angular?*, o. J., Abschn. Services.

<sup>41</sup> vgl. *Angular—Transforming Data Using Pipes*, o. J.

<sup>42</sup> vgl. *Ionic Framework—The Cross-Platform App Development Leader*, o. J.

Die Syntax von **SASS** existiert in zwei Formen: Die ältere Syntax verwendet Einrückungen für das Teilen von Codeblöcken und die neuere bedient sich der gleichen Teilung wie **CSS** (durch Klammern).<sup>43</sup>

## 8.8 GitHub Actions

Bei **GitHub** Actions handelt es sich um eine Plattform für **Continuous Integration** bzw. **Continuous Delivery**, mit welcher sich Workflows für **GitHub** Repositories automatisieren lassen. Ein **GitHub** Actions Workflow wird immer durch ein bestimmtes Ereignis ausgelöst (z. B. Erstellung einer neuen Pull Request) und kann mehrere Schritte enthalten, die voneinander abhängig sein können. Jeder Schritt des Workflows wird dabei in einer eigenen VM, welche entweder unter Linux, Windows oder macOS läuft, ausgeführt. Vom Ergebnis eines Workflow-Durchlaufs kann anschließend abhängig gemacht werden, ob eine Pull Request mit dem master Branch zusammengeführt werden kann.<sup>44</sup>

## 8.9 Nuke

Nuke ist ein **Build** Automatisierungstool, geschrieben in C#. Durch Nuke können sehr einfach dotnet Projekte kompiliert und **Angular** Projekte transpiliert werden. Die einzelnen **Build** Schritte können außerdem voneinander abhängig gemacht werden. Nuke bietet weiters die Möglichkeit, Konfigurationsdateien für CI/CD Pipelines von **GitHub**, Gitlab, Bitbucket und viele weitere Versionsverwaltungssoftwares zu erstellen.

## 8.10 Docker

Bei Docker handelt es sich um eine „Containerisierungssoftware“ von Docker Inc., mit welcher schlanke, virtuelle Maschinen erstellt werden können. Dabei verwendet Docker die, in Linux integrierten, Funktionen „Cgroups“ und „Namespaces“, um die einzelnen Container zu isolieren und unabhängig voneinander zu betreiben. Alle Container laufen hierbei mit

---

<sup>43</sup> vgl. *Sass (style sheet language)*, 2023.

<sup>44</sup> vgl. *Grundlegendes zu GitHub Actions—GitHub-Dokumentation*, o. J.

demselben Kernel. Docker verwendet ein imagebasiertes Deployment-Modell. Das heißt, dass Docker Container aus Images erstellt werden, welche zuvor mit einer speziellen Definitionsdatei („Dockerfile“) gebaut wurden.<sup>45</sup>

---

<sup>45</sup> vgl. *Was ist Docker? Welche Vorteile bieten Container?*, o. J.

## 9 Autorenverzeichnis

| Kapitel/Abschnitt                                  | Autor(en)               |
|--|-------------------------|
| Vorwort  | Laimer, Losert-Nachbaur |
| Abstract   | Laimer, Losert-Nachbaur |
| Danksagung   | Laimer, Losert-Nachbaur |
| 1 Impressum  | Losert-Nachbaur         |
| 1.1 Projektteam                                    | Losert-Nachbaur         |
| 1.2 Projektbetreuung                               | Losert-Nachbaur         |
| 1.2.1 Diethard Kaufmann                            | Losert-Nachbaur         |
| 1.2.2 Bianca Franzoi                               | Laimer                  |
| 1.3 Turnsport Austria                              | Laimer                  |
| 2 Team Turnen                                      | Laimer                  |
| 2.1 Geräte   | Laimer                  |
| 2.2 Wertungsblatt                                  | Laimer, Losert-Nachbaur |
| 3 Projektmanagement                                | Losert-Nachbaur         |
| 4 Planung und Evaluierung                          | Losert-Nachbaur         |
| 4.1 Anforderungen                                  | Losert-Nachbaur         |
| 4.2 Technologische Entscheidungen                  | Losert-Nachbaur         |
| 4.2.1 Backend                                      | Laimer                  |
| 4.2.2 Frontend                                     | Losert-Nachbaur         |
| 4.2.3 Datenbank                                    | Laimer                  |
| 4.2.4 Continuous Integration & Continuous Delivery | Laimer                  |
| 5 Entwicklungsphase                                | Losert-Nachbaur         |
| 5.1 Backend  | Laimer                  |
| 5.2 Frontend                                       | Losert-Nachbaur         |
| 5.3 Deployment                                     | Laimer                  |
| 6 Testphase  | Laimer, Losert-Nachbaur |
| 7 Fazit  | Laimer, Losert-Nachbaur |
| 8 Technologien                                     | Losert-Nachbaur         |

|                    |                         |
|--------------------|-------------------------|
| 8.1 C#             | Laimer                  |
| 8.2 ASP .NET Core  | Laimer                  |
| 8.3 MongoDB        | Laimer                  |
| 8.4 TypeScript     | Losert-Nachbaur         |
| 8.5 Angular        | Losert-Nachbaur         |
| 8.6 Ionic          | Losert-Nachbaur         |
| 8.7 SASS           | Losert-Nachbaur         |
| 8.8 GitHub Actions | Laimer                  |
| 8.9 Nuke           | Laimer                  |
| 8.10 Docker        | Laimer                  |
| 10 Glossar         | Laimer, Losert-Nachbaur |

## 10 Glossar

| Begriff              | Beschreibung   |
|----------------------|--|
| Angular              | Ein Framework für die Entwicklung von Benutzeroberflächen, welches von Google entwickelt wird.                                       |
| Array                | Ein Array ist eine Datenstruktur für das Speichern von mehreren Daten mit dem gleichen Datentyp. <sup>46</sup>                       |
| Bearer-Token         | Signierter Token, welcher zur Authentifizierung genutzt wird.  |
| Bibliothek           | Eine Sammlung von Code-Teilen, die bei der Entwicklung von anderen Programmen verwendet werden können.                               |
| Boolean              | Datentyp, der zwei Werte annehmen kann: <b>true</b> und <b>false</b> .   |
| BSON                 | Binäre Form zur Darstellung von Datenstrukturen.   |
| Build                | Kompilierung/Transpilierung von Anwendungscode.  |
| CSS                  | CSS ist eine Stylesheet-Sprache, mit der das Aussehen von Websites manipuliert werden kann. <sup>47</sup>                            |
| Data Transfer Object | Ein Objekt, welches Daten aus verschiedenen Quellen aggregiert. Dies erleichtert die Übertragung von Daten vom Backend zum Frontend. |
| Datentyp             | Bestimmt, welche Werte eine Variable annehmen kann und welche Operationen auf diese Variable möglich sind.                           |
| Decorator            | Notationsmöglichkeit in TypeScript, um Klassen und Funktionen Metadaten anzuhängen.  |
| Default              | Englisch für „Standard“.   |
| Direktive            | Möglichkeit, HTML-Elemente zu steuern.   |
| DOM                  | Bezeichnet eine Darstellungsweise für HTML-Dokumente als Baumstruktur. <sup>48</sup>   |

---

<sup>46</sup> vgl. *Array (data structure)*, 2024.

<sup>47</sup> vgl. „Cascading Style Sheets“, 2024.

<sup>48</sup> vgl. *Document Object Model*, 2023.

|               |  |
|---------------|--|
| dotnet        | Quelloffene Software-Platform zur Entwicklung und Ausführung von Anwendungsprogrammen  |
| Einheitsrunde | Die Bezeichnung für die erste Runde bei den Disziplinen Trampolin und Tumbling, wobei alle Turnerinnen und Turner die gleichen Elemente bzw. das gleiche Element turnen. |
| Element       | Kleinste Turneinheit (z. B. Salto oder Pirouette).   |
| Endpoint      | Aufrufbarer Endpunkt einer API-Schnittstelle.  |
| false         | Wert für „nicht wahr“ des Datentyps Boolean.   |
| GitHub        | Eine Plattform für Entwickler, die das Erstellen, Speichern und Managen von Codebasen ermöglicht. <sup>49</sup>  |
| JSON          | Kompaktes Format zur Datennotation.  |
| Komponente    | Wiederverwendbarer Code, der ein Stück der Website definiert und darstellt (z. B. ein Knopf oder ein Eingabefeld).   |
| Konstruktor   | Eine Funktion in der objektorientierten Programmierung, welche beim Instanziieren einer Klasse aufgerufen wird.  |
| Modal-Dialog  | Ein Fenster, welches sich über dem Hauptfenster der Anwendung öffnet und die Interaktion mit diesem deaktiviert, es aber sichtbar lässt. <sup>50</sup>                   |
| NodeJS        | Plattformübergreifende Laufzeitumgebung für JavaScript   |
| null          | Wert der Variable fehlt; Variable hat keinen Wert.   |
| React         | Ein von Meta entwickeltes Frontend-Framework, welches das Erstellen von Webapplikationen mittels Komponenten ermöglicht.   |
| SASS          | SASS ist eine Erweiterung für CSS, die verschiedene zusätzliche Funktionen einbringt. <sup>51</sup>  |
| Serie         | Eine Reihe von Elementen, die beim Tumbling geturnt werden.  |

---

<sup>49</sup> vgl. GitHub, 2024.

<sup>50</sup> vgl. *Modal window*, 2024.

<sup>51</sup> vgl. *Sass (style sheet language)*, 2023.

|          |  |
|----------|--|
| Superset | Supersets erweitern bestehende Programmiersprachen mit neuer Syntax und/oder neuen Funktionen. |
| true     | Wert für „wahr“ des Datentyps Boolean.   |
| UI       | Oberfläche, mit der ein Benutzer interagiert.  |
| URL      | Adresse einer Website.   |
| Vue      | Ein Open Source Framework für das Bauen von Frontend-Anwendungen.                              |

## 11 Abkürzungsverzeichnis

---

### A

*API*

Application Programming Interface ..... 22, 23, 27, 59, 81

---

### B

*bzw.*

beziehungsweise ..... 9, 13, 14, 30, 31, 54, 63, 65, 72, 76, 81

---

### C

*CD*

Continuous Delivery ..... 76

*CI*

Continuous Integration ..... 76

*CSS*

Cascading Style Sheets ..... 23, 24, 52, 53, 54, 74, 75, 76

---

### D

*DD*

Difficulty distribution ..... 52

*DOM*

Document Object Model ..... 74, 80

*DS*

Difficulty Element in Moving Sequence ..... 39, 52

*Dto*

Data Transfer Object ..... 28, 30, 31

---

### H

*HTL*

Höhere technische Lehranstalt ..... 10, 15

**HTML**

Hypertext Markup Language ..... 23, 53, 54, 55, 66, 74, 80

**HTTP**

Hypertext Transfer Protocol ..... 31, 33, 35

---

**I**

**ID**

Identifier ..... 30, 31, 73

---

**J**

**JS**

JavaScript ..... 61

**JSON**

JavaScript Object Notation ..... 72, 81

---

**K**

**kB**

Kilobyte ..... 73

---

**O**

**o. J.**

ohne Jahr ..... 10, 11, 12, 18, 20, 23, 24, 47, 59, 61, 71, 72, 73, 74, 75, 76, 77, 89, 90, 91

---

**S**

**SASS**

Syntactically Awesome Style Sheets ..... 24, 53, 54, 75, 76, 79

**SSH**

Secure Shell ..... 62, 63

---

**T**

**TSZ**

Turnsportzentrum ..... 67

---

## **U**

*UI*

User Interface ..... 56, 75, 81

*URL*

*Uniform Resource Locator* ..... 63, 66, 81

*usw.*

und so weiter ..... 49

---

## **V**

*vgl.*

vergleiche ..... 10, 11, 12, 13, 18, 20, 21, 23, 24, 38, 39, 43, 45, 46, 47, 48, 52, 59, 61, 71, 72, 73, 74, 75, 76, 77, 80, 81

*VM*

Virtuelle Maschine ..... 76

---

## **W**

*WM*

Weltmeisterschaft ..... 10

---

## **Y**

*yml*

yet another markup language ..... 63, 66

---

## **Z**

*z. B.*

zum Beispiel ..... 28, 32, 34, 49, 61, 68, 73, 76, 81

## 12 Abbildungsverzeichnis

|  |           |
|--|-----------|
| Abbildung 1: Michael Laimer .....  | 9         |
| Abbildung 2: Laurin Losert-Nachbaur .....                                      | 9         |
| Abbildung 3: Diethard Kaufmann .....   | 10        |
| Abbildung 4: Bianca Franzoi .....  | 10        |
| Abbildung 5: Logo Turnsport Austria.....                                       | 10        |
| Abbildung 6: Turnübung am Boden .....  | 11        |
| Abbildung 7: Turnübung am Trampolin.....                                       | 12        |
| Abbildung 8: Turnübung am Tumbling .....                                       | 12        |
| Abbildung 9: Wertungsrichterinnen beim Bewerten einer Übung .....              | 13        |
| Abbildung 10: Wertungsblatt am Boden.....                                      | 13        |
| Abbildung 11: Wertungsblatt am Trampolin .....                                 | 14        |
| Abbildung 12: Wertungsblatt am Tumbling .....                                  | 14        |
| Abbildung 13: Projektauftrag .....   | 16        |
| Abbildung 14: Zieleplan .....  | 17        |
| Abbildung 15: Projektstrukturplan.....   | 19        |
| Abbildung 16: Objektstrukturplan.....  | 20        |
| Abbildung 17: Projektumweltanalyse .....                                       | 21        |
| Abbildung 18: Übersicht Applikation.....                                       | 26        |
| Abbildung 19: Dependency Injection .....                                       | 27        |
| Abbildung 20: Initialisierung des MongoRepositorys.....                        | 29        |
| Abbildung 21: Ablauf des Abrufens von Daten und Auflösung der Referenzen ..... | 30        |
| Abbildung 22: Anfügen des Bearer Tokens als Cookie .....                       | 31        |
| Abbildung 23: Verwendung des Attributs AllowAnonymus .....                     | 32        |
| Abbildung 24: Ablauf des Backend Build-Prozesses .....                         | 33        |
| Abbildung 25: Erstellung eines Headers auf dem Wertungsblatt.....              | 33        |
| Abbildung 26: Header im exportierter PDF-Datei .....                           | 34        |
| Abbildung 27: Einfügen von Bildern in ein PDF.....                             | 34        |
| Abbildung 28: Rendern einer PDF-Datei .....                                    | 35        |
| Abbildung 29: Trampolinelemente-Verwaltung .....                               | 37        |
| Michael Laimer, Laurin Losert-Nachbaur   | 86 von 91 |

|   |    |
|---|----|
| Abbildung 30: Tumblingelemente-Verwaltung .....                 | 38 |
| Abbildung 31: Bodenenelemente-Verwaltung.....                   | 39 |
| Abbildung 32: Verein-Verwaltung .....                           | 40 |
| Abbildung 33: Team-Verwaltung.....                              | 41 |
| Abbildung 34: Vereinsoberfläche Teamübersicht.....              | 41 |
| Abbildung 35:Vereinsoberfläche Geräteübersicht .....            | 42 |
| Abbildung 36: Trampolin-Eingabe .....                           | 43 |
| Abbildung 37: Trampolin-Validierungsfunktion.....               | 44 |
| Abbildung 38: Tumbling-Eingabe .....                            | 45 |
| Abbildung 39: Tumbling-Eingabe Elementauswahl .....             | 46 |
| Abbildung 40: Serienschwierigkeit Pipe .....                    | 47 |
| Abbildung 41: Bodenwertungsblatt in Arbeit.....                 | 49 |
| Abbildung 42: Leere und ausgefüllte Formation .....             | 50 |
| Abbildung 43: Elemente und Kompositionen Bodeneingabe.....      | 51 |
| Abbildung 44: Bodeneingabe Elementpositionen in Zeile .....     | 51 |
| Abbildung 45: Codeausschnitt - Difficulty Distribution .....    | 52 |
| Abbildung 46: Login-Komponente HTML-Datei .....                 | 53 |
| Abbildung 47: Login-Komponente TS-Datei 1/3.....                | 54 |
| Abbildung 48: Login-Komponente TS-Datei 2/3.....                | 54 |
| Abbildung 49: Login-Komponente TS-Datei 3/3.....                | 55 |
| Abbildung 50: AuthService 1/2 .....                             | 56 |
| Abbildung 51: AuthService 2/2 .....                             | 57 |
| Abbildung 52: Ablauf Deployment .....                           | 58 |
| Abbildung 53: Ablauf Continuous Integration.....                | 59 |
| Abbildung 54: Ablauf Backend Build .....                        | 60 |
| Abbildung 55: Ablauf Frontend Build .....                       | 61 |
| Abbildung 56: Artefakte nach einem Workflow .....               | 62 |
| Abbildung 57: Ablauf eines Deployments .....                    | 62 |
| Abbildung 58: SSH-Skript zum Stoppen des Docker Containers..... | 63 |
| Abbildung 59: Docker compose Datei .....                        | 64 |
| Abbildung 60: Docker compose Datei des Reverse Proxys.....      | 65 |
| Abbildung 61: Dockerfile des Backends .....                     | 66 |

|  |    |
|--|----|
| Abbildung 62: Möglichkeit zur Filterung der Elemente nach Kategorie .....  | 67 |
| Abbildung 63: Vorarlberger Landesmeisterschaften im Team-Turnen 2023 ..... | 68 |
| Abbildung 64: Beispielanwendung des Attributs Obsolete .....               | 72 |

## 13 Literaturverzeichnis

*4\_SQL\_Document\_storage\_v1.3.pdf.* (o. J.).

*Angular—Deployment.* (o. J.). Abgerufen 15. Februar 2024, von <https://angular.io/guide/deployment>

*Angular—Transforming Data Using Pipes.* (o. J.). Abgerufen 15. Februar 2024, von <https://angular.io/guide/pipes#creating-pipes-for-custom-data-transformations>

*Angular—What is Angular?* (o. J.). Abgerufen 17. Dezember 2023, von <https://angular.io/guide/what-is-angular>

*Array (data structure).* (2024, Februar 7). Wikipedia. [https://en.wikipedia.org/w/index.php?title=Array\\_\(data\\_structure\)&oldid=1204418952](https://en.wikipedia.org/w/index.php?title=Array_(data_structure)&oldid=1204418952)

*ASP.NET Core.* (2024, Februar 19). Wikipedia. [https://en.wikipedia.org/w/index.php?title=ASP.NET\\_Core&oldid=1208938934](https://en.wikipedia.org/w/index.php?title=ASP.NET_Core&oldid=1208938934)

*BET-PM 08 PM Projektplanung FRB 2021.pdf.* (o. J.).

*Bianca Franzoi jetzt Europa-Vizepräsidentin für Team-Turnen!* (o. J.). Abgerufen 18. Januar 2024, von <https://www.turnsport-austria.at/de/sport/team-turnen/boxnewsar-chivshow14-bianca-franzoi-jetzt-europa-vizepraesidentin-fuer-team-turnen>

*C# PDF Library (All-in-One Solution) | IronPDF for .NET.* (o. J.). Abgerufen 30. Januar 2024, von <https://ironpdf.com/>

*Cascading Style Sheets.* (2024). In Wikipedia. [https://de.wikipedia.org/w/index.php?title=Cascading\\_Style\\_Sheets&oldid=242828989](https://de.wikipedia.org/w/index.php?title=Cascading_Style_Sheets&oldid=242828989)

*C-Sharp.* (2024, Februar 19). Wikipedia. <https://de.wikipedia.org/w/index.php?title=C-Sharp&oldid=242345003>

*Document Object Model.* (2023, Juli 10). Wikipedia. [https://de.wikipedia.org/w/index.php?title=Document\\_Object\\_Model&oldid=235351737](https://de.wikipedia.org/w/index.php?title=Document_Object_Model&oldid=235351737)

Frankenhauser, B. (2021). *Projektmanagement BET.*

*GitHub.* (2024, März 4). Wikipedia. <https://en.wikipedia.org/w/index.php?title=GitHub&oldid=1211842327>

*GridFS — MongoDB Manual.* (o. J.). Abgerufen 6. März 2024, von <https://www.mongodb.com/docs/manual/core/gridfs/>

*Grundlegendes zu GitHub Actions—GitHub-Dokumentation.* (o. J.). Abgerufen 30. Januar 2024, von <https://docs.github.com/de/actions/learn-github-actions/understanding-github-actions>

*Ionic Framework—The Cross-Platform App Development Leader.* (o. J.). Ionic Framework. Abgerufen 2. Januar 2024, von <https://ionicframework.com/>

*Modal window.* (2024, Januar 23). Wikipedia. [https://en.wikipedia.org/w/index.php?title=Modal\\_window&oldid=1198138897](https://en.wikipedia.org/w/index.php?title=Modal_window&oldid=1198138897)

*Pdf Library for .NET | Free Html To Pdf Converter.* (o. J.). SelectPdf.Com. Abgerufen 23. Januar 2024, von <https://selectpdf.com/pdf-library-for-net/>

*Pricing · Plans for every developer · GitHub.* (o. J.). Abgerufen 4. Februar 2024, von <https://github.com/pricing>

*Records in C#—C# / Microsoft Learn.* (o. J.). Abgerufen 7. März 2024, von <https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/types/records>

*Sass (style sheet language).* (2023, September 27). Wikipedia. [https://en.wikipedia.org/w/index.php?title=Sass\\_\(style\\_sheet\\_language\)&oldid=1177288725](https://en.wikipedia.org/w/index.php?title=Sass_(style_sheet_language)&oldid=1177288725)

*Ststeiger/PdfSharpCore: Port of the PdfSharp library to .NET Core—Largely removed GDI+ (only missing GetFontData—which can be replaced with freetype2).* (o. J.). Abgerufen 30. Januar 2024, von <https://github.com/ststeiger/PdfSharpCore>

*TeamGym Code of Points V1.1.* (2022). <https://backend.europeangymnastics.com/sites/default/files/paragraph/document/2022%20Team-Gym%20Code%20of%20Points%20V1.1.pdf>

*Turnsport Austria | Wir über uns.* (o. J.). Abgerufen 23. Januar 2024, von <https://www.turnsport-austria.at/de/verband/wir-ueber-uns>

*Turnsport Austria—Team-Turnen.* (o. J.). Abgerufen 3. Januar 2024, von <https://www.turnsport-austria.at/de/sport/team-turnen>

*Tutorial: Definieren und Lesen von benutzerdefinierten Attributen—C# / Microsoft Learn.* (o. J.). Abgerufen 8. März 2024, von <https://learn.microsoft.com/de-de/dotnet/csharp/advanced-topics/reflection-and-attributes/attribute-tutorial>

*TypeScript: Documentation—Decorators.* (o. J.). Abgerufen 15. Februar 2024, von <https://www.typescriptlang.org/docs/handbook/decorators.html>

*TypeScript—JavaScript With Syntax For Types.* (o. J.). Abgerufen 17. Dezember 2023, von <https://www.typescriptlang.org/>

*Was ist Docker? Welche Vorteile bieten Container?* (o. J.). Abgerufen 8. März 2024, von <https://www.redhat.com/de/topics/containers/what-is-docker>

*Was ist MongoDB und wie funktioniert es? | Pure Storage.* (o. J.). Abgerufen 6. März 2024, von <https://www.purestorage.com/de/knowledge/what-is-mongodb.html>