

# **Programmierbeleg “Dota2Predict”**

Zhavarankau Illia

August 2021

Dozent	M. Sc. Dipl.-Inf. (FH) Knut Altroggen
Fach	Einführung in die Informatik II, Programmierbeleg
Hochschule	Hochschule Mittweida
Semester	Sommersemester 2021

## **Inhaltsverzeichnis**

<b>Inhaltsverzeichnis.....</b>	<b>2</b>
<b>1      Aufgabenstellung.....</b>	<b>3</b>
<b>2      Problemanalyse .....</b>	<b>3</b>
<b>3      Software-Architektur .....</b>	<b>4</b>
<b>4      Beschreibung der wesentlichen Implementierungsdetails.....</b>	<b>8</b>
<b>5      Benutzerleitpfaden.....</b>	<b>9</b>
<b>6      Installationsanleitung .....</b>	<b>11</b>
<b>7      Hilfsmittel und Entwicklungstools.....</b>	<b>12</b>
<b>8      Inhaltsverzeichnis Dota2Predict .....</b>	<b>12</b>
<b>9      Fazit .....</b>	<b>14</b>
<b>10     Reflexion .....</b>	<b>14</b>
<b>11     Eigenständigkeitserklärung .....</b>	<b>16</b>
<b>12     Literaturverzeichnis.....</b>	<b>17</b>

## 1. Aufgabenstellung

Eine Multiplayer-Vorhersage App für Dota2-Spiele sollte geschrieben werden. Die Anwendung sollte über eine grafische Oberfläche verfügen:

- Registrierung neuer Benutzer;
- Anmeldung in der App, woraufhin der Benutzername und die aktuelle Punktzahl angezeigt werden;
- Möglichkeit zur Änderung der Registrierungsdaten (Passwort und e-Mail);
- Informationen über kommende Dota2-Spiele;
- die Möglichkeit, eine Vorhersage für ein bevorstehendes Spiel zu treffen;
- Historie der Punktwertänderungen des Kontos;
- Gesamtwertung der Teilnehmer nach Punktzahl;
- die Möglichkeit, unbegrenzt viele Male die Punkte für das Spiel aufzuladen, wenn der Kontostand unter 1.000 fällt.

Spielmechanik. Die Nutzer können über die Punkte nach eigenem Gutdünken verfügen. Eine beliebige Anzahl von Punkten können eingesetzt werden. Wenn man gewinnt, werden die Punkte der unterlegenen Teilnehmer unter den Gewinnern im Verhältnis zu ihrem Anteil an der Gesamtpunktzahl aufgeteilt. Im Falle eines Verlustes werden die platzierten Punkte vom Konto abgezogen.

Die zu prognostizierenden Spiele stammen von der Liquipedia-Website [1]. Die Spiele sollten als Best of 1, 3 oder 5 (Bo1, Bo3, Bo5) gespielt werden. Die zu spielenden Teams sollten professionell sein, um die Auswertung für die Dotabuff-Website [2] automatisieren zu können. Die Annahme von Vorhersagen für das Spiel endet 15 Minuten nach Beginn des ersten Spiels der Serie.

## 2. Problemanalyse

Damit die Spieler miteinander interagieren können, wird ein Vermittler in Form eines Servers benötigt. Dementsprechend sollte die Anwendung in 2 logische Teile unterteilt werden: einen Client-Teil und einen Server-Teil.

Für die Registrierung, Autorisierung, Bewertung, Historie usw. wird eine Datenbank benötigt, um diese Informationen zu speichern und zu verarbeiten. Als Datenbank wird die beliebte Plattform MySQL gewählt. Für die Verbindung mit der Datenbank wird die MySQL Connector-Bibliothek verwendet. Für die Arbeit des Servers müssen neue Tabellen erstellen werden, Zellwerte hinzugefügt und geändert und nach Informationen gesucht werden.

Für die Arbeit mit der Datenbank wird 3 Haupttabellen benötigt. Die erste Tabelle enthält die Benutzerdaten (USER\_TABLE). Die zweite Tabelle enthält alle Matches (DOTA\_MATCH\_TABLE). Dritte Tabelle, die manuell zu füllen ist (DOTABUFF\_TABLE). Bei der Registrierung wird für jeden Benutzer eine neue Tabelle erstellt, in der sein gesamter Spielverlauf angezeigt wird. Außerdem sollte es bei der Vorhersage eines Spielergebnisses eine neue Tabelle geben, die alle Vorhersagen nur für dieses bestimmte Spiel enthält. Für die weitere Berechnung der Punkte für Gewinner und Verlierer. Die Wettdaten müssen gleichzeitig in 2 Tabellen eingegeben werden: in die Benutzertabelle und in die Tabelle dieses Spiels.

Falls erforderlich, MySQL wird lokal installiert oder ein Remote-Hosting wird verwendet(in meinem Fall ist der MySQL-Server auf <https://beget.com/>).

Um Spielereignisse zu kennen, muss sie von Websites mit einer bestimmten Zeitdauer entnommen werden. Weil neue Treffer nicht sehr häufig vorkommen, ist ein stündliches Intervall für diesen Zweck geeignet. Die jsoup-Bibliothek kann verwendet werden, um Daten von Websites abzurufen. Um die Spielergebnisse zu überwachen und die Vorhersagen der Nutzer zu berechnen, soll ein Intervall von etwa 1 Minute gewählt werden. Diese Überwachung wird vom Server anhand der bestehenden Tabelle der Spiele (DOTA\_MATCH\_TABLE) durchgeführt. Für die oben genannten Funktionen wird ein Server benötigt, der rund um die Uhr läuft. Derselbe Server kann für die Benutzerautorisierung und -registrierung verwendet werden. Der Server sollte Daten in der MySQL-Datenbank speichern und den Benutzern antworten.

Javafx-Bibliotheken werden für die Programmierung der grafischen Oberfläche der Client-Anwendung verwendet. Scene Builder eignet sich gut für die Arbeit mit fxml-Dateien. Zunächst muss die Anwendung das Hauptanmeldefenster starten. Vom Anmeldefenster sollte es möglich sein, zum Anmeldefenster und zurück zu gehen. Wenn die Anmeldung erfolgreich ist, sollte das Spielfenster erscheinen, wenn nicht, sollte ein Hinweis angezeigt werden. Es ist zweckmäßig, die Bewertungs-, Spiel- und Verlaufsfenster in Form von Tabellen anzuzeigen. Der Model-View-Controller wird für die Anzeige von Tabellendaten verwendet. Es ist praktisch, die Spielvorhersage in einem neuen Fenster statt im aktuellen zu öffnen, so dass es möglich ist, die Situation in mehreren Spielen zu verfolgen.

**Anmerkung.** Aus zeitlichen und fachlichen Gründen kann die Client-Anwendung unter Umgehung des Servers nach Autorisierung durch den Server direkt auf die Datenbank zugreifen.

### 3. Software-Architektur

Das Programm besteht aus 2 Teilen: dem Serverteil und dem Clientteil, die sich jeweils in den Server- und Anwendungspaketen befinden. Das UML-Klassendiagramm für den Serverteil ist in der Abbildung 1 dargestellt. Die Many\_Thread-Klasse startet unseren Spieleserver. Die Klasse Server bearbeitet Anfragen von Benutzern zur Registrierung und Anmeldung. Die Klassen Configs\_server und DB\_MySQL\_const enthalten Konstanten, mit denen der Serverteil bei Bedarf angepasst werden kann (Umbenennung von Tabellenspalten in geeignetere Spalten, Änderung des Ports usw.). Die Klasse DataBaseHandler enthält Funktionen für die Arbeit mit Datenbanken. Die Parser-Klasse enthält Funktionen zum Abrufen von Daten von Dota 2-Websites [1-2]. Die Klassen Check\_new\_game, Check\_live\_game, Check\_complited\_match füllen die Tabelle DOTA\_MATCH\_TABLE.

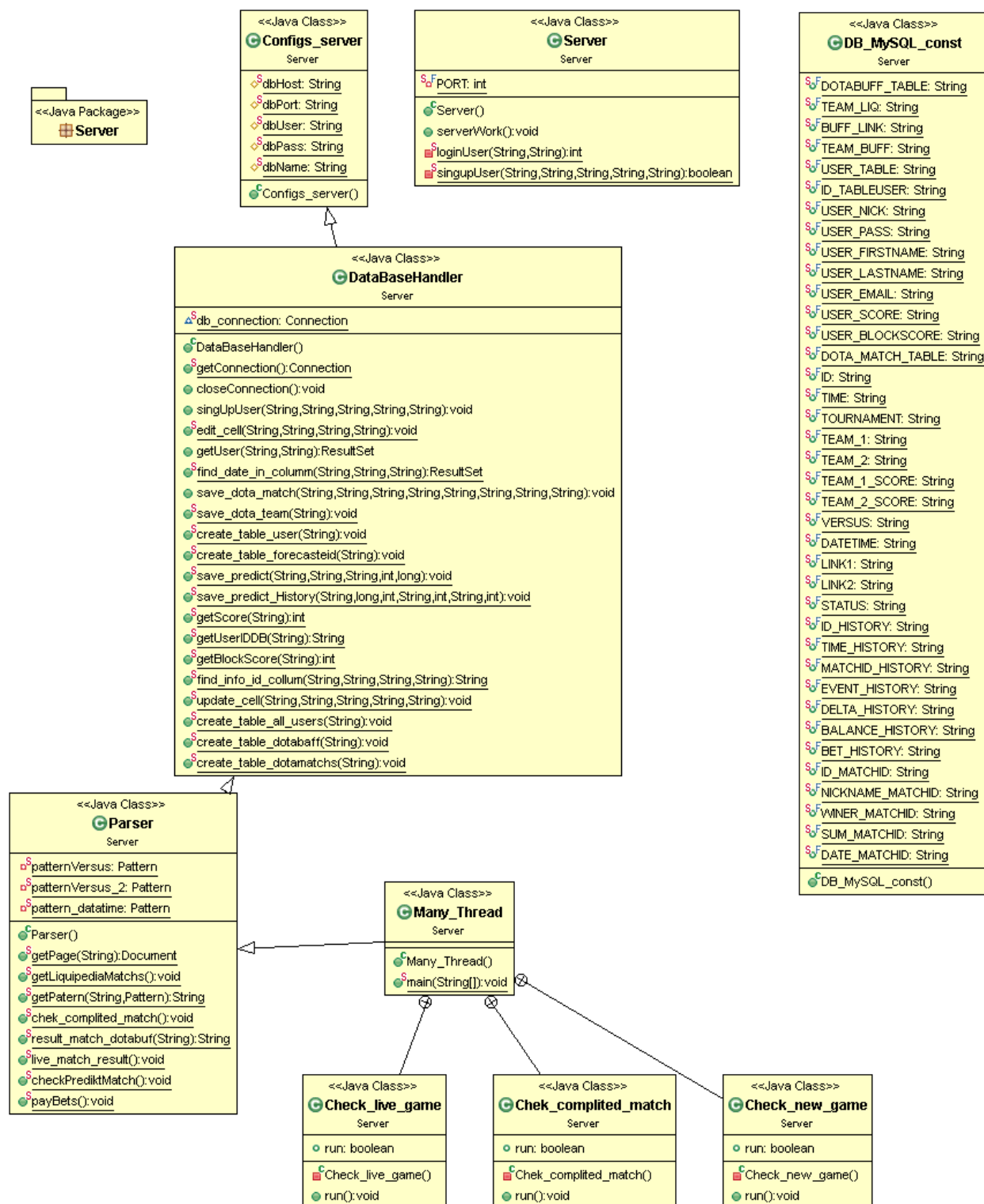
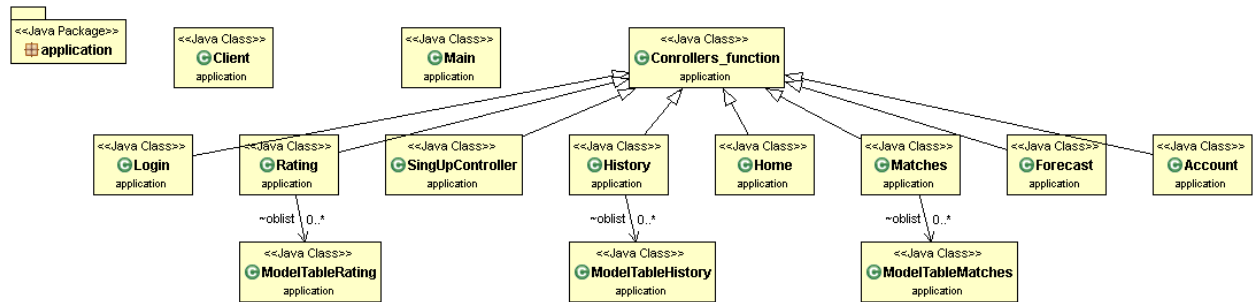


Abbildung 1. UML-Klassendiagramm: Package Server.

Ein verkürztes UML-Klassendiagramm Package Application ist in der Abbildung 2 dargestellt. Die Anwendung wird mit der Klasse Main gestartet. Die Klassen Rating, Matches und History werden zur Anzeige von Tabellen verwendet. Die Klassen Login, Rating, SingUpController, History, Home, Matches, Forecast, Account dienen als Controller für entsprechende fxml-Dateien. Und sie erben alle Funktionen von der Klasse Conrollers\_function. Die Klasse Forecast enthält alle Aktionen, um Informationen über

das aktuelle Spiel zu erhalten und auch die Möglichkeit, Vorhersagen über das ausgewählte Spiel zu treffen. Auf diese Klasse wird in Kapitel 4 näher eingegangen.



**Abbildung 2. Verkürztes UML-Klassendiagramm Package Application.**

Ein detailliertes UML-Klassendiagramm für Package Application ist in der Abbildung 3 dargestellt.

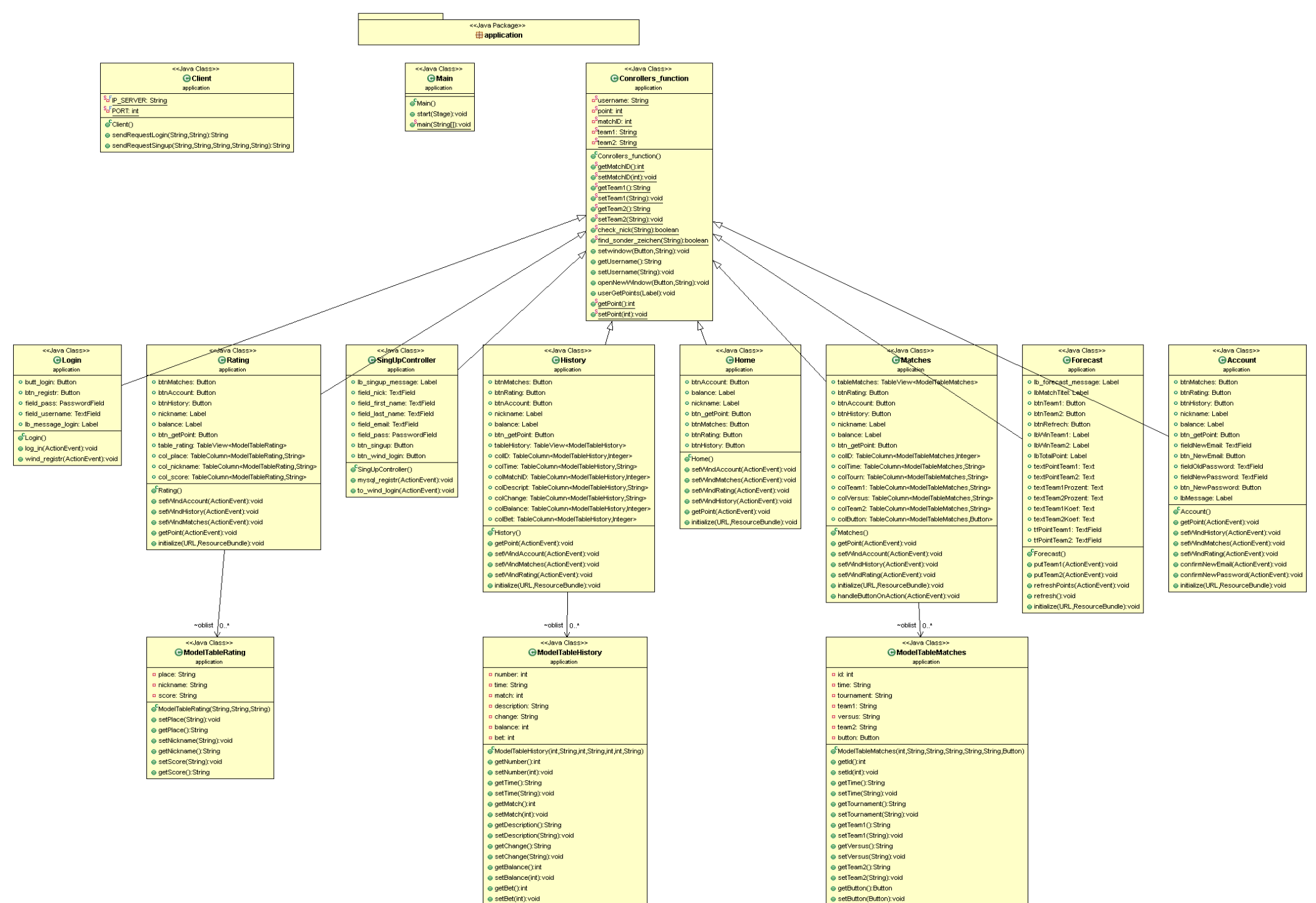


Abbildung 3. UML-Klassendiagramm Package Application.

## 4. Beschreibung der wesentlichen Implementierungsdetails

Es wird die Implementierung der Klasse Forecast betrachtet, deren UML-Klassendiagramm in der Abbildung 4 dargestellt ist.

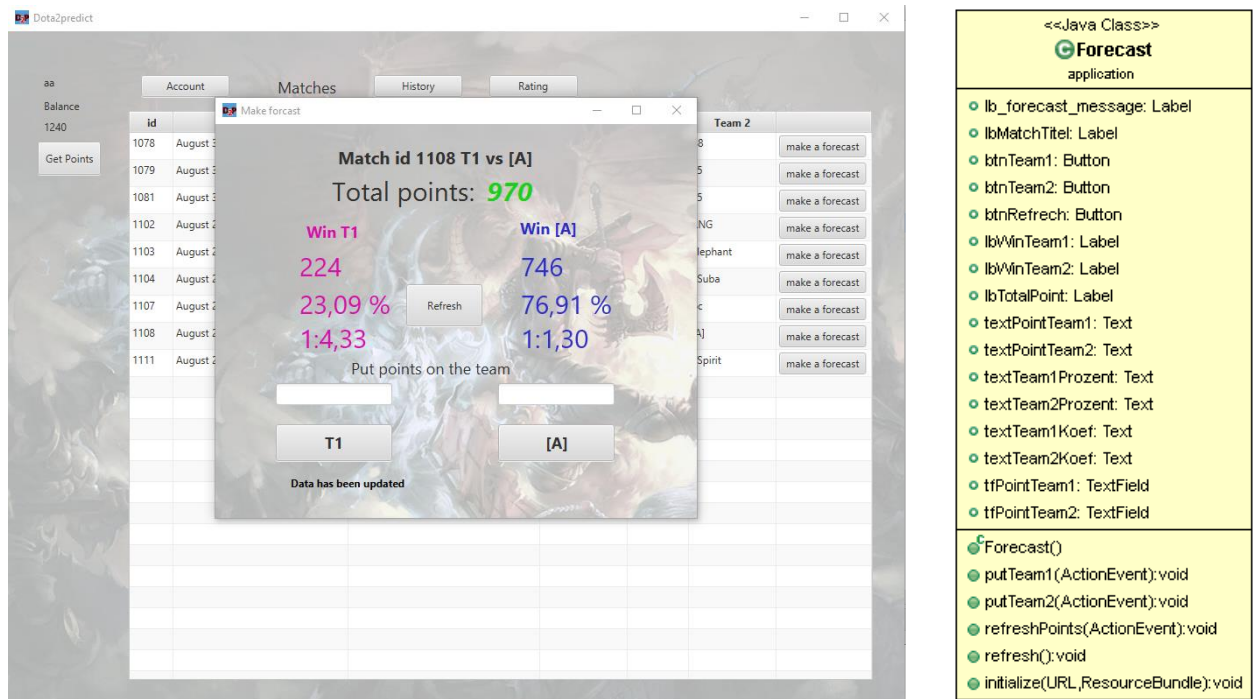


Abbildung 4. Fensterdarstellung und UML-Klassendiagramm Forecast.

Diese Klasse erbt Hilfsfunktionen von Conrollers\_function. Um Daten anzeigen zu können, verfügt diese Klasse über ein Interface Initializable. Die Felder Label und Text erhalten darin die Basiswerte aus den DOTA\_MATCH\_TABLE - und matchID-Tabellen des jeweiligen Spiels.

```
public class Forecast extends Conrollers_function implements Initializable{
```

Das Hauptmerkmal der Eingabefelder ist, dass nur Ziffern von 0-9 erlaubt sind. die Lösung sieht folgendermaßen aus:

```
UnaryOperator<javaafx.scene.control.TextFormatter.Change>
integerFilter = change -> {
    String input = change.getText();
    if (input.matches("[0-9]*")) {
        return change;
    }
    return null;
};
tfPointTeam1.setTextFormatter(new TextFormatter<String>(integerFilter));
tfPointTeam2.setTextFormatter(new TextFormatter<String>(integerFilter));
```

Um das Spiel zu spielen, muss es überprüft werden, ob man auf das Spiel wetten darf. Wenn es die Möglichkeit gibt, ein paar Minuten vor Ende des Spiels zu wetten, dann ist dieses Spiel nicht sinnvoll. Zuerst soll eine Verbindung zu dieser Datenbank hergestellt werden, dann es wird in der Tabelle nach der ID unseres Matches nachgeschaut, ob der Status 0 ist (noch nicht gestartet), dann wird der Code weiter durchgeführt.

```
DataBaseHandler handler =new DataBaseHandler();
```



```

ResultSet rs =null;
Connection con = null;
con = DataBaseHandler.getConnection();
rs = con.createStatement().executeQuery("SELECT * FROM "+
    DB_MySQL_const.DOTA_MATCH_TABLE+ " WHERE "+
    DB_MySQL_const.ID+"="+getMatchID());
    if (rs.next()) {
        if (rs.getInt(DB_MySQL_const.STATUS) == 0 ) {

```

Damit ein Spieler eine Vorhersage treffen kann, sollte es geprüft werden, ob seine Punkte höher sind als sein Einsatz. Wenn der Spieler bereits Vorhersagen getroffen hat, die Ergebnisse aber noch nicht bekannt sind, werden die Punkte nicht abgezogen. Deshalb hat jeder Spieler auch gesperrte Punkte, die derzeit in anderen Prognosen verwendet werden. Der Spieler kann die Punkte nutzen, die sich aus der Differenz zwischen den Punkten und den gesperrten Punkten ergeben.

```

    int block=handler.getBlockScore(getUsername());
    int score=handler.getScore(getUsername());

    if (team1<score -block && team1>0 ){

```

Die Daten werden dann in die matchID-Tabelle dieser Übereinstimmung eingefügt, und die Punkte werden in der Tabelle USER\_TABLE bearbeitet.

```

handler.save_predict("match"+getMatchID(),getUsername(), getTeam2(),
    team2,dateNow);
handler.save_predict_History(getUsername(), dateNow, getMatchID(),
    getTeam1()+ " vs "+getTeam2()+": Win "+getTeam2(), team2,"expect",
    score);
handler.edit_cell(DB_MySQL_const.USER_TABLE,
    handler.getUserIDDB(getUsername()), DB_MySQL_const.USER_BLOCKSCORE,
    Integer.toString (block+team2));

```

Diese Klasse enthält auch den Refresh-Knopf mit der entsprechenden Funktion public void refresh(). Der Benutzer kann die Daten selbst aktualisieren, und diese Funktion wird auch automatisch bei der Vorhersage aufgerufen. Zum Schluss wird die Verbindung zur Datenbank geschlossen.

## 5. Benutzerleitpfaden

### 5.1. Für den Benutzer.

Um das Spiel nutzen zu können, muss sich der Nutzer registrieren. Die Verwendung des Leerzeichens \, \*, ' , " beim Ausfüllen der Felder ist nicht zulässig. Nach der Registrierung wird der Benutzer zum Home-Fenster geleitet, in dem die grundlegenden Funktionen des Programms kurz erläutert werden. Zu Beginn hat der Benutzer 0 Punkte. Um das Kontostand aufzufüllen, muss der Nutzer auf den Knopf "Punkte erhalten" klicken. Der Kontostand wird mit 1000 Punkten erhöht. Der aktuelle Kontostand wird in der linken oberen Ecke angezeigt. Um das Passwort oder die E-Mail zu ändern, kann ein Benutzer auf die Seite "Account" gehen. Um die für die Vorhersage verfügbaren Spiele zu sehen, gehen Sie auf die Seite "Matches".

Hier werden kurze Informationen über das bevorstehende Spiel angezeigt. Um eine Prognose für das ausgewählte Spiel abzugeben, müssen Sie auf den Knopf "Prognose abgeben" klicken. Nach dem

Drücken diesem Knopf öffnet sich ein neues Fenster, in dem Sie Informationen über die gespielten Punkte, die Anzahl der auf jede Mannschaft gesetzten Punkte und die Gewinnquote sehen können.

Man kann die Lieblingsmannschaft auswählen und die Anzahl der Punkte bis zum Kontostand des Benutzers festlegen. Die Vorhersage wird erfolgreich angenommen, woraufhin eine Benachrichtigung erscheint. Auf der Seite "History" können Sie die Entwicklung der Punkte auf Ihrem Konto verfolgen. Es wird der Zeitpunkt angezeigt, zu dem das Konto finanziert wurde, sowie eine Zusammenfassung der getroffenen Vorhersagen. Solange das Spiel nicht beendet ist, wird in der Spalte "Change" den Wert "expected" angezeigt. Nach Beendigung des Spiels wird eine Berechnung vorgenommen. Auf der Seite "Rating" werden alle Spieler mit ihrem aktuellen Punktestand angezeigt.

## 5.2. Für den Server-Administrator.

Für die Unterstützung der Serverseite ist es wichtig, Folgendes zu wissen. Leider war es nicht möglich, den Prozess der Erfassung von Übereinstimmungsergebnissen vollständig zu automatisieren. Daher muss die DOTABUFF\_TABLE-Tabelle manuell ausgefüllt werden. Die dort auszufüllenden Mannschaften erscheinen automatisch aus der DOTA\_MATCH\_TABLE-tabelle, danach müssen die Felder "leer" Abbildung 5 ausgefüllt werden, um korrekt zu funktionieren. Es handelt sich um 2 Felder: den abgekürzten Mannschaftsnamen auf dem Dotabuff[2] und den Link zum Mannschaftsprofil.

Beispiel für das Ausfüllen des Teamnamens durch Abkürzung auf der Liquidapedia [1] VG. (Wenn die Abkürzungen der Mannschaften nicht bekannt sind, können die vollständigen Mannschaftsnamen in der Liquidapedia oder Google gesucht werden). Rausfinden VG – Vici Gaming. Dann es wird gesucht auf Dotabuff[2]. Es gibt die folgende Seite: <https://www.dotabuff.com/esports/teams/726228-vici-gaming>. Als Link wird das Ende der Seite benötigt, und als Abkürzung wird den TEAM-TAG in der Teambeschreibung auf dieser Seite **VG** genommen.

←T→	id	team	link	team_dotabuff
<input type="checkbox"/>	1	VG	leer	leer
<input type="checkbox"/>	2	MagMa	leer	leer
<input type="checkbox"/>	3	iG	leer	leer
<input type="checkbox"/>	4	Elephant	leer	leer
<input type="checkbox"/>	5	4Z	leer	leer
<input type="checkbox"/>	6	NoPing	leer	leer
<input type="checkbox"/>	7	Aries	leer	leer
<input type="checkbox"/>	8	Aster	leer	leer

←T→	id	team	link	team_dotabuff
<input type="checkbox"/>	1	VG	726228-vici-gaming	VG
<input type="checkbox"/>	2	MagMa	leer	leer
<input type="checkbox"/>	3	iG	leer	leer
<input type="checkbox"/>	4	Elephant	leer	leer
<input type="checkbox"/>	5	4Z	leer	leer
<input type="checkbox"/>	6	NoPing	leer	leer
<input type="checkbox"/>	7	Aries	leer	leer
<input type="checkbox"/>	8	Aster	leer	leer

a. Leere Tabelle

b. Ausfüllen der ersten Zeile

**Abbildung 5. Manuelles Ausfüllen der Tabelle**

Der abgekürzte Namen können in der Dotabuff anders lauten als in der Liquidapedia. Deshalb sollten Abkürzungen von Dotabuff ausfüllen. Denn daraus werden die endgültigen Spielergebnisse entnommen.

Die Projektdateien enthalten eine fertige Tabelle (August 2021), die importiert werden kann. Es werden jedoch ständig neue Mannschaften gebildet oder neu zusammengestellt, die dann hinzugefügt oder alte Mannschaftsdaten geändert werden müssen.

Wenn es im automatischen Modus keine Ergebnisse gibt, gibt es die Möglichkeit, manuell in die DOTA\_MATCH\_TABLE -tabelle einzugeben. Es muss den Status des Spiels geändert werden. Ein Spiel kann einen der folgenden Status haben:

0 – das Spiel wird erwartet. Zu diesem Zeitpunkt sind Vorhersagen erlaubt. Sie sind in der Spieltabelle aufgeführt.

1 – das Spiel hat begonnen. Vorhersagen sind nicht verfügbar, das Spiel ist in der Liste der Spiele ausgeblendet.

2 – das Spiel ist vorbei. Es ist zu prüfen, ob es Prognosen dazu gibt.

4 – die Vorhersagen werden berechnet, wenn sie gemacht wurden.

5 – das Spiel ist beendet und es gibt keine weiteren Operationen erforderlich.

Es ist notwendig, die Tabellenzellen mit den Ergebnissen für jede Mannschaft zu ändern und den Spielstatus auf 2 zu setzen. Wenn ein Spiel (für das nicht vorhergesagt wurde) abgesagt wird, kann den Status auf 5 geändert werden. Wenn es die Vorhersage für das Spiel gibt, muss den Status auf 2 geändert werden und das Ergebnis geprüft werden (es muss ein 0:0 – Unentschieden sein), um die Punkte zurückzugeben und die blockierten Punkte zu entfernen.

## 6. Installationsanleitung

**6.1.** Für die Nutzung der Client-App ist keine Installation erforderlich, es genügt, die Datei D2P.exe auszuführen. Wenn Java 16-Version und JavaFX 16-Version installiert sind und Pfade zu Umgebungsvariablen im Betriebssystem geschrieben werden. In einem anderen Fall kann das Spiel über eine launch.bat-Datei in einem Ordner (D2Pfull) gestartet werden, das Java und JawaFX enthält.

**6.2.** Der Server ist bereits standardmäßig in Betrieb. Erforderlichenfalls wird er jedoch wie folgt eingestellt.

1. Bevor den Server startee wird, muss die Verbindung zur MySQL-Datenbank in der Datei Configs\_server konfiguriert werden. Um eine Verbindung zu einer entfernten Datenbank herzustellen, sind benötigt:

dbHost–die DNS-Datenbank oder die IP;

dbPort – der Verbindungsport;

dbUser – der Benutzername der Datenbank;

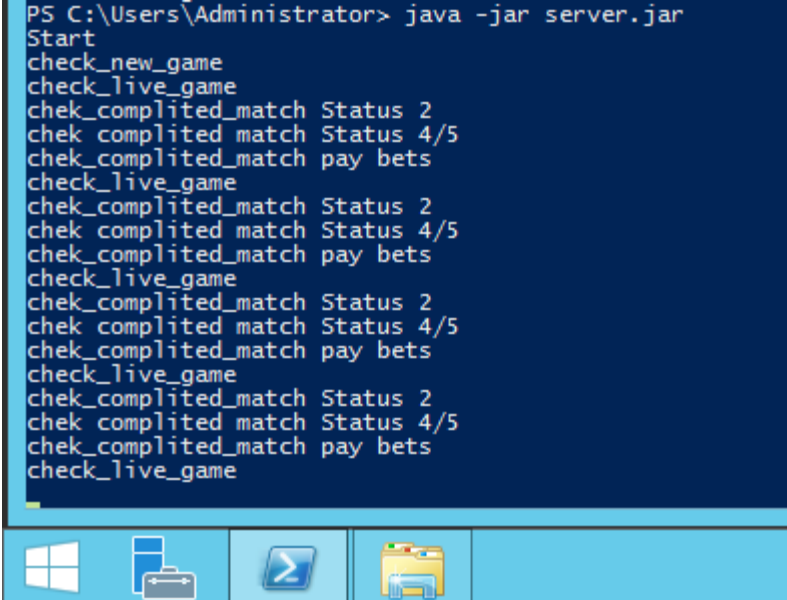
dbName – der Name der Datenbank;

dbPass – das Kennwort.

Diese Daten sollten mit dem Anbieter abgeklärt werden.

Für die Verbindung mit einer lokalen Datenbank muss diese installiert und konfiguriert werden. Laden Sie die Installationsdatei vom Link [3] herunter. Konfigurieren Sie dann die Einstellungen gemäß den Anweisungen [4]. Ändern Sie danach in der Datei Configs\_server die Serverdaten (dbHost, dbPort, dbUser, dbPass, dbName), die bei der Installation eingestellt wurden.

2. Ausführen des Serverteils für Windows. Um den Server zu starten, öffnen Sie die Befehlszeile und führen Sie die ausführbare jar-Datei aus. Drücken Sie dazu **Win+R**. Geben Sie in diesem Fenster **cmd** ein. Als nächstes müssen Sie in den Ordner mit der Datei server.jar wechseln. Geben Sie in der Befehlszeile den folgenden Befehl ein: **cd C:\Users** (geben Sie den Pfad an, in dem sich der Server befindet). Nachdem Sie die Serverdatei selbst mit folgendem Befehl ausgeführt haben: **java -jar server.jar**. Wenn alles korrekt ausgeführt wurde, erscheint die folgende Meldung im Terminal: start, wie in Abbildung 6 dargestellt. Der Serverteil ist erfolgreich gestartet.



```

PS C:\Users\Administrator> java -jar server.jar
Start
check_new_game
check_live_game
chek complited match Status 2
chek complited match Status 4/5
chek complited match pay bets
check_live_game
chek complited match Status 2
chek complited match Status 4/5
chek complited match pay bets
check_live_game
chek complited match Status 2
chek complited match Status 4/5
chek complited match pay bets
check_live_game
chek complited match Status 2
chek complited match Status 4/5
chek complited match pay bets
check_live_game

```

Abbildung 6. Terminalfenster beim Starten des Serverteils.

## 7. Hilfsmittel und Entwicklungstools

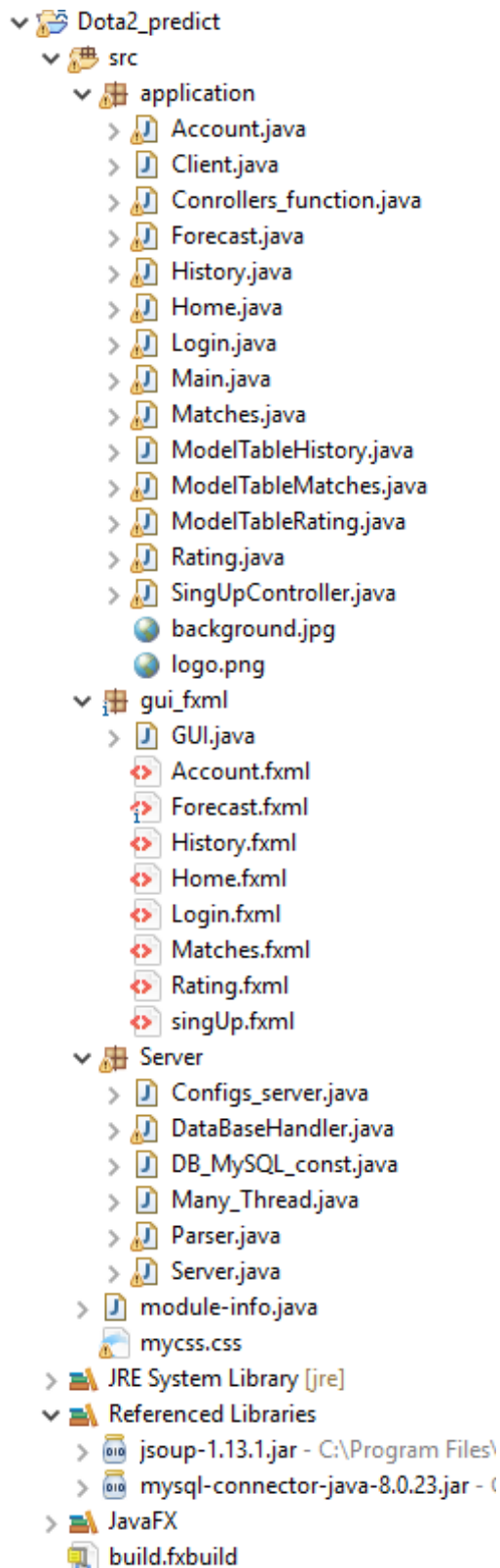
Als Begleitmaterial wurden die Vorlesungen "Informatik I-Programmierung" und "Weiterführende Programmierung" des Programmierkurses und deren Zusatzmaterialien verwendet. Für die Entwicklung wurden Eclipse IDE und Scene Builder verwendet. Die w3schools[5] wurde als Hilfsquelle für die Erarbeitung der SQL-Abfragesprache verwendet. Das ObjectAid-Plugin [6] für Eclipse wurde zur Erstellung von UML-Klassendiagrammen verwendet. Reguläre Ausdrücke wurden verwendet, um die Ergebnisse von Übereinstimmungen oder andere Muster auf der Website zu finden. Die Informationen zur Verwendung regulärer Ausdrücke stammen von javarush.ru[7].

Für eine visuelle Präsentation wurden YouTube-Videos als zusätzliche Ressourcen verwendet. Das Video "Single Event handle for Multiple buttons"[8] war hilfreich für das Hinzufügen von Aktionen mit den in der Spalte befindlichen Knöpfen. Model-View-Controller Konzept [9] wurde verwendet, um die Daten in tabellarischer Form anzuzeigen. Einige Beispiele für die Verwendung der jsoup-Bibliothek [10-11]. Mit Launch4j [12] wurde eine Datei mit der Erweiterung exe erstellt. Für Codekommentare und Dokumentation wurde javadoc [13] verwendet.

## 8. Inhaltsverzeichnis Dota2Predict

Die Anwendung ist in 3 Pakete unterteilt. Erstes Paket ist Client-Teil, zweite ist für die GUI (für fxm-Dateien), dritte ist der Server-Teil. Auch unterteilt in 2 Ordnern verwendete Bibliotheken: javafx und

referenzierte Bibliothek. Die Zusammensetzung der Ordnerdateien Dota2Predict ist in Abbildung 7 dargestellt.



**Abbildung 7. Inhaltsverzeichnis Dota2Predict**

### **package application**

Account – Controller für Account. Benutzereinstellungen: Name und Passwort.  
 Client – Client verbindet sich mit Server.  
 Conrollers\_function – gemeinsame Funktionen für Klassencontroller.  
 Forecast – Controller für das Prognosefenster.  
 History – Controller für das Verlaufsfenster. Ereignisverlaufs-tabelle im Konto.  
 Home – Controller für das Hauptfenster. Eine kurze Beschreibung der Spielfunktionen.  
 Login – Controller für Login. Das Anmeldefenster.  
 Main – Die Hauptklasse, in der die Anwendung gestartet wird.  
 Matches – Controller für Matches. Spieltabelle.  
 ModelTableHistory – Parameter zum Füllen der Verlaufstabelle.  
 ModelTableMatches – Parameter zum Füllen der Spieltabelle.  
 ModelTableRating – Parameter zum Füllen der Ratingstabelle.  
 Rating – Controller für Rating. die Ratingstabelle.  
 SingUpController – Controller für Registrierung.  
 background.jpg – der App-Hintergrund.  
 logo.png – das App-Verknüpfungslogo.

### **package gui\_fxml**

GUI - eine leere Klasse, die zum Exportieren der Anwendung erforderlich ist (Zugriff auf fxml-Dateien).  
 Account.fxml – das Kontofenster.  
 Forecast.fxml – das Prognosefenster.  
 History.fxml – der Kontoverlauf.  
 Home.fxml- die erste Seite der Anwendung mit kurzen Erläuterungen, ist bei jedem Start nur einmal sichtbar.  
 Login.fxml – das Anmeldefenster.  
 Matches.fxml – die Spieltabelle.  
 Rating.fxml – die Ratingstabelle  
 SingUpController.fxml – das Registrierungs-fenster.

### **package Server**

Configs\_server – Einstellungen für die Verbindung zum Server mit der Datenbank.  
 DataBaseHandler – Methode zum Arbeiten mit der Datenbank.  
 DB\_MySQL\_const – Konstanten für Tabellen- und Spaltennamen.  
 Many\_Thread – Gemeinsamer Server zum Ausführen von Client-Anfragen und Empfangen von Daten von der Webseiten.

Parser – Zusatzfunktionen für die Arbeit mit der Website und das Speichern von Daten in Tabellen.

Server – Server mit Anmelde- und Registrierungsanfragen.  
mycss.css – das Anwendungsdesign.

#### Verzeichnis **Referenced Library**:

jsoup-1.13.1.jar – die Bibliothek für die Arbeit mit Webseiten.

mysql\_connector\_java-8.0.23.jar – die Bibliothek für die Arbeit mit der Datenbank.

#### Verzeichnis **JavaFX**:

javafx.base.jar ,javafx.controls.jar, javafx.fxml.jar, javafx.graphics.jar, javafx.media.jar, javafx.swing.jar, javafx.web.jar, javafx.swt.jar – die Bibliotheken zum Erstellen von GUI-Anwendungen.

## 9. Fazit

Im zweiten Semester wurde eine Client-Server-Anwendung entwickelt, die mit einer MySQL-Datenbank arbeitet. Die Entwicklung wurde unabhängig und in der vorgegebenen Zeit durchgeführt. Die Anwendung führt die in der Aufgabe beschriebenen Funktionen in vollem Umfang aus. Durch die eigenständige Arbeit wurden Erfahrungen bei der Umsetzung der Anwendung von der Idee bis zum Endprodukt gesammelt.

Das Spiel sieht aus wie ein fertiges Produkt mit einer minimalen Anzahl von Funktionen. Ein stabiler Betrieb erfordert jedoch eine ständige Überwachung der Serverseite, da die Spielergebnisse nicht immer auf der Website angezeigt werden oder die Website vorübergehend nicht verfügbar ist.

## 10. Reflexion

### Warum habe ich dieses Thema ausgewählt?

In meiner Schulzeit, vor etwa 15 Jahren, war ich gut im Dota-Spiel. Viele meiner Freunde haben auch gespielt, einige tun es immer noch. Wir schauen uns oft gemeinsam große Turniere an und prognostizieren den Sieger. Um bequem mit Freunden spielen zu können, müssen wir eine Statistik über unsere Vorhersagen führen. Daraufhin hatte ich die Idee, meine neuen Fähigkeiten als Programmierer einzusetzen, um den Prozess der Prognosen zu automatisieren. Aus technischer Sicht ist dieses Programm ein Totalisator für eine Sportveranstaltung. Viele Wettbüros bieten an, dass man gegen sie spielen kann, und zwar zu ihren Quoten. Deshalb wollte ich ein Programm schreiben, in dem die Leute den Ausgang von Spiele der gegeneinander erraten.

### Was soll dieser Beleg zeigen?

Dieser Programmierbeleg hat gezeigt, dass ich in der Lage bin, selbstständig zu arbeiten. Durchführung des geplanten Projekts von Anfang bis Ende. Dies ist eine großartige Erfahrung für die Organisation der nächsten Startup-Projekte als Leiter oder als Teil eines Teams. Während des Studiums habe ich einige Methoden oder Klassen erstellt, ist es nicht immer offensichtlich, wie man sie für den Benutzer ausführt oder wie man sie in ein größeres Projekt implementiert. Dabei habe ich gelernt, das Projekt als Ganzes zu sehen. Außerdem bin ich auf eine Reihe von Teilaufgaben gestoßen, bei denen ich mich selbstständig mit dem Thema befassen und neue Kenntnisse erwerben musste.

Ein weiterer wichtiger Aspekt dieser Tätigkeit ist die Erstellung eines Lebenslaufs für eine künftige Stelle. In der Regel zeigen junge Programmierer ohne Berufserfahrung ihre Hausaufgaben, die sie für eigene Zwecke oder im Auftrag angefertigt haben, damit der Arbeitgeber das Niveau beurteilen kann. Dieses

Projekt ist also nicht nur für den Lernprozess und die Leistungspunkte wichtig, sondern auch für den Lebenslauf bei einer künftigen Beschäftigung.

### **Was habe ich bei der Anfertigung gelernt?**

Viele der in den beiden Semestern erworbenen Kenntnisse flossen in die Erstellung dieses Programms ein. Allerdings musste ich mich zusätzlich mit der Abfragesprache MySQL und der jdbc-Bibliothek vertraut machen. Die jsoup-Bibliothek wird auch für die Webseiten verwendet, die die Matches und ihre Ergebnisse enthalten. Zusammenfassend kann man sagen, dass die folgenden Fähigkeiten während der Entwicklung verbessert wurden:

1. Analyse des Problems, Erstellung einer Idee und Einschätzung der Möglichkeit der Umsetzung
2. Entwicklung eines Layout-Diagramms der Anwendung
3. Programmierung einer Client-Anwendung mit grafischer Oberfläche und Übergang zwischen Fenstern (javafx, Scene Builder)
4. Programmierung eines kleinen Serverteils (nur Login und Registrierung) und eines Teils zum Abrufen von Daten für Spiele (jsoup, MySQL)
5. Dokumentation und Javadoc-Erstellung
6. Export der fertigen Anwendung, Erstellen einer jar- oder exe-Datei
7. Verantwortungsbewusstsein und Unabhängigkeit

### **Was würde ich anders machen?**

Da ich bereits über wenig Entwicklungserfahrung verfüge, hätte ich etwas mehr Zeit darauf verwendet, das Konzept im Detail zu planen und zu entwickeln. Als erstes habe ich versucht, das zu programmieren, was ich kannte, und das war nicht optimal oder führte zu nichts, da es problematisch war, es in anderen Teilen des Codes zu verwenden. Aus diesem Grund mussten einige Funktionen umgeschrieben oder entfernt und neue geschrieben werden, was zu einem zusätzlichen Zeitverlust führte. Der größte Nachteil der aktuellen Version ist, dass meine Verbindung zur Datenbank chaotisch ist. Ich stelle eine Verbindung her, schließe die Verbindung oder nicht, usw. Es gibt keine klare Struktur. Als mir das klar wurde, hatte ich schon zu viel geschrieben, um es noch aufzuräumen, also beschloss ich, es so zu belassen, wie es ist, weil es funktioniert und ich es schon lange so getestet habe. Ich wollte nicht riskieren, in den letzten 2 Wochen alles noch einmal zu machen.

Was die Arbeit im Allgemeinen anbelangt, so sind 100 Stunden für eine perfekt funktionierende Anwendung auf meinem Programmierniveau nicht ausreichend. Es gibt wahrscheinlich vieles, was an dieser Applikation verbessert werden könnte, aber ich kenne zumindest die folgenden Punkte:

- die Anmeldebestätigung per e-Mail hinzufügen, Passwörter per Hash-Funktion in der Datenbank speichern (jetzt Speicherung im Original, um testen zu können);
- die Verbesserung der Server- und Client-Seite (Verlagerung der Datenbankzugriffsfunktionen auf den Server);
- die Optimierung der Datenbankhandhabung, Aufteilung der MySQL-Tabellen in mehrere Datenbanken je nach Zweck (Konten, Spielprognosen);
- die Echtzeit-Aktualisierung der Tabellen in der Anwendung (nur beim Wechsel zwischen den Seiten aktualisiert wird);
- ein Design entwickeln und hinzufügen (css-Datei);
- Detaillierte Durcharbeitung des Spieldesigns (Überlegen, wo die Knöpfe fehlen, wo sie nützlich wären), z. B. der Logout-Knopf;

– Verbesserung des Gameplays, Hinzufügen von täglichen Aufgaben, um das Interesse am Spiel zu erhöhen.

Alle diese Bereiche können in Zukunft weiter ausgebaut werden. Im Moment ist nur eine minimale Funktionalität implementiert, die es erlaubt, das Spiel zu spielen, aber ohne viel Komfort.

### Kritisches Fazit

Trotz meiner Bemühungen ist das Programm noch nicht in perfektem Zustand. Natürlich gibt es keine Grenzen der Perfektion, aber es lohnt sich, danach zu streben. Ich bin jedoch sehr froh, dass ich das Projekt in der vorgesehenen Zeit abschließen konnte. Diese Arbeit ermöglichte es mir, die Softwareentwicklung in ihrer Gesamtheit zu sehen, und gab mir die Möglichkeit, in einem größeren Rahmen zu denken als nur als jemand, der Code schreiben kann. Es hat mir auch einige der Schwierigkeiten vor Augen geführt, die Entwickler auf diesem Weg erwarten.

Abschließend möchte ich mich bei den Dozenten bedanken, die diese Aufgabe im 2. Semester übernommen haben. Je früher man Erfahrungen mit der Entwicklung eines Gesamtprojekts sammelt, desto eher beginnt man meiner Meinung nach, eine globale und strategische Planung zu entwickeln, die eine wichtige Rolle für die integrierte Entwicklung in verschiedenen Bereichen der Programmplanung spielt.

## 11 Eigenständigkeitserklärung

Ich erkläre, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.



Waldheim am 25. August 2021

Illia Zhavarankau



## 12. Literaturverzeichnis

1. Liquipedia Dota 2 Wiki. [Online]. Verfügbar unter: [https://liquipedia.net/dota2/Main\\_Page](https://liquipedia.net/dota2/Main_Page). Zugriff am: 30. April 2021.
2. Dotabuff. [Online]. Verfügbar unter: <https://www.dotabuff.com/>. Zugriff am: 30. April 2021.
3. Oracle Corporation and/or its affiliates, MySQL Community Downloads. Verfügbar unter: <https://dev.mysql.com/downloads/mysql/8.0.html>. Zugriff am: 16. Mai 2021.
4. Oracle Corporation and/or its affiliates, MySQL Server Configuration with MySQL Installer. [Online]. Verfügbar unter: <https://dev.mysql.com/doc/mysql-installation-excerpt/8.0/en/mysql-installer-workflow-server.html>. Zugriff am: 16. Mai 2021.
5. W3Schools, SQL Tutorial. [Online]. Verfügbar unter: <https://www.w3schools.com/sql/default.asp>. Zugriff am: 16. Mai 2021.
6. ObjectAid LLC, The ObjectAid UML Explorer. [Online]. Verfügbar unter: <https://www.objectaid.com/home>. Zugriff am: 16. Mai 2021.
7. JavaRush, Регулярные выражения в Java [Reguläre Ausdrücke in Java]. [Online]. Verfügbar unter: <https://javarush.ru/groups/posts/regulyarnye-vyrazheniya-v-java>. Zugriff am: 22. April 2021.
8. Youtube, JavaFX | How to handle Button click Event | Single Event handle for Multiple buttons. [Online]. Verfügbar unter: [https://www.youtube.com/watch?v=BwL\\_jL-Dmmg&ab\\_channel=CoolITHelp](https://www.youtube.com/watch?v=BwL_jL-Dmmg&ab_channel=CoolITHelp). Zugriff am: 27. Juni 2021.
9. Wikipedia, Model View Controller. [Online]. Verfügbar unter: [https://de.wikipedia.org/wiki/Model\\_View\\_Controller](https://de.wikipedia.org/wiki/Model_View_Controller). Zugriff am: 21. Juni 2021.
10. Jsoup HTML parser, Download and install jsoup. [Online]. Verfügbar unter: <https://jsoup.org/download>. Zugriff am: 13. April 2021.
11. Youtube, Приложение "Курс Валют" Часть 2/Парсинг сайта/Уроки по Android Studio Java [Anwendung "Währungskurs" Teil 2 / Parsing der Website / Tutorials auf Android Studio Java]. [Online]. Verfügbar unter: [https://www.youtube.com/watch?v=7GeDoh7k6l4&ab\\_channel=N.E.C.ORU](https://www.youtube.com/watch?v=7GeDoh7k6l4&ab_channel=N.E.C.ORU). Zugriff am: 13. April 2021.
12. SourceForge, Launch4j Executable Wrapper. [Online]. Verfügbar unter: <https://sourceforge.net/projects/launch4j/>. Zugriff am: 3. Juli 2021.
13. Javadoc, javadoc - The Java API Documentation Generator . [Online]. Verfügbar unter: <https://docs.oracle.com/javase/1.5.0/docs/tooldocs/windows/javadoc.html#javadoctags>. Zugriff am: 3. August 2021.