



**Kantonsschule Büelrain**

---

# NOTENRECHNER

---

Ein Projekt in IDPA

KANTONSSCHULE BÜELRAIN WINTERTHUR  
RAFAEL GAHLER (3BI), SAMIRA STRAGIOTTI (3BI), NICO ZOLLINGER (3BI)  
Betreut von: Jaques Mock Schindler, Bernhard Marti  
Abgabe: 20.04.2023



## Inhalt

1. Abstract .....	3
2. Einleitung.....	4
3. Vertiefung.....	5
3.1. Use-Case .....	5
3.2. Erläuterung Funktionen .....	6
3.2.1. Berechnen.....	6
3.2.2. Import CSV .....	6
3.2.3. Export CSV .....	6
3.2.4. Speichern .....	6
3.2.5. Felder leeren.....	6
3.2.6. Prognose .....	6
3.2.7. Diagramm .....	7
3.2.8. Fächer hinzufügen .....	7
3.3. Aktivitätsdiagramme.....	7
3.3.1. Berechnen.....	7
3.3.2. Prognose .....	8
3.3.3. Speichern .....	8
3.3.4. Felder leeren.....	9
3.3.5. Import CSV .....	9
3.3.6. Export CSV .....	9
3.4. Erläuterung Code .....	10
3.4.1. save() .....	10
3.4.2. Diagrammfunktion.....	10
3.4.3. prognose() .....	11
3.4.4. delete() .....	12
3.4.5. import_csv().....	12
3.4.6. export_csv() und.....	13
3.4.7. berechnen() .....	14
4. Dependencies.....	15
5. Testfälle.....	17
6. Fazit .....	17
7. Bestätigung Eigenständigkeit .....	18
7.1. Ehrenwörtliche Erklärung .....	18
8. Quellenverzeichnis .....	19
8.1. Abbildungsverzeichnis .....	19



9.	Anhang .....	28
9.1.	Projektvereinbarung .....	28
9.2.	Arbeitsjournal .....	29
9.2.1.	Arbeitsjournal Rafael Gahler .....	29
9.2.2.	Arbeitsjournal Nico Zollinger .....	40
9.2.3.	Arbeitsjournal Samira Stragiotti .....	46
9.3.	Betreuungsgespräche .....	51
9.3.1.	Betreuungsgespräch 1 .....	51
9.3.2.	Betreuungsgespräch 2 .....	53
9.4.	QR-Code zu GitHub Repository (inkl. Programm ZIP Datei) .....	53



## 1. Abstract

Die Arbeit Notenrechner soll nicht nur als IDPA dienen, sondern soll auch wirklich als Tool genutzt werden können. Dazu muss das ganze Programm nutzerfreundlich und übersichtlich sein, und in seiner Funktion als Notenrechner fehlerfrei funktionieren. Um dies umzusetzen haben wir uns am bereits bestehenden Notenrechner der KBW orientiert, der als Excel-Mappe verfügbar ist. Der Vorteil von unserem Notenrechner ist, dass er als selbstständiges Programm funktioniert und nicht als Mappe in Excel. Ebenso wurden Funktionen eingebaut, die der bisherige Notenrechner so nicht bieten kann. Zum Beispiel gibt unser Notenrechner ein Diagramm über den Notenverlauf aus. Das Design des Programms hat sich leider ein wenig komplizierter gestaltet als erwartet, da wir mit JavaFX arbeiten. Mit Javascript wäre es uns einfacher gefallen, das Design vorzunehmen, jedoch wäre es dann komplizierter gewesen, ein Java-Backend zu benutzen, weshalb wir uns für JavaFX entschieden haben.



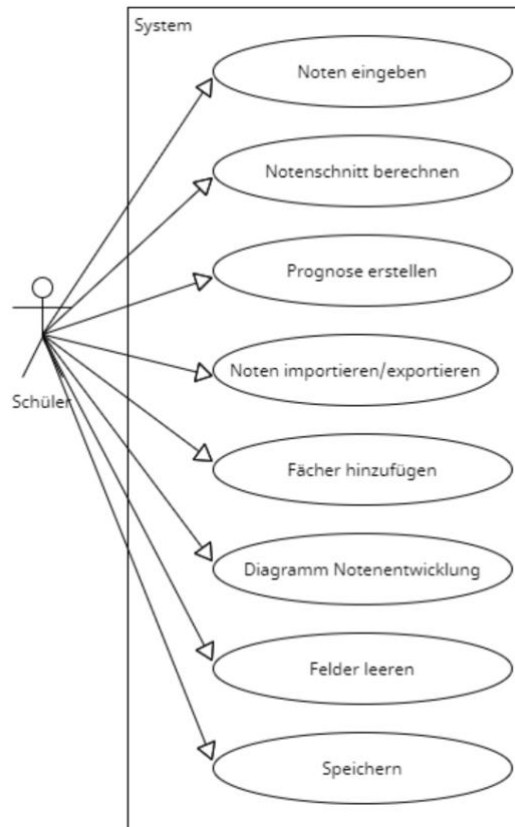
## 2. Einleitung

In unserer IDPA-Arbeit «Notenrechner» soll ein Notenrechner erstellt werden, anhand des bereits bestehenden Notenrechners der KBW. Der Notenrechner soll als selbstständiges Programm bzw. als Web-Applikation funktionieren, soll jedoch ein Java-Backend haben. Da wir nicht geübt sind in Web-Applikationen mit Java-Backend haben wir uns dazu entschieden, für das Frontend JavaFX zu verwenden. Der Notenrechner soll die Fächer, die im jetzigen Notenrechner vorhanden sind, beinhalten, jedoch soll man ebenfalls neue Fächer hinzufügen können. Aus den Semesternoten, die man in den Notenrechner eingibt, kann man berechnen, welche Note an der Abschlussprüfung notwendig ist, um den Abschluss zu bestehen. Ebenso soll es möglich sein, eine Prognose abzugeben, bevor man alle Semesternoten ausgefüllt hat. Zu dem Notenrechner soll ebenfalls eine technische Dokumentation erstellt sowie von jedem Mitglied der IDPA-Gruppe ein Arbeitsjournal geführt werden. Somit sollen unser Arbeitsablauf und die Vorgehensweise von Anfang bis zum Ende des Projekts nachvollziehbar gemacht werden. Zum Schluss folgt eine Reflexion unserer Arbeit bzw. dem Arbeitsprozess. Die Abgabe des Projekts ist am 20.04.2023, wobei wir bis zu diesem Datum 2 Zwischengespräche mit unseren Betreuern, Herrn Schindler und Herrn Marti haben, wofür wir zwischenzeitige Ziele festgelegt haben.

### 3. Vertiefung

#### 3.1. Use-Case

Für unser Projekt Notenrechner haben wir ein Use-Case Diagramm erstellt:



Der Notenrechner hat 8 Funktionalitäten, welche hier im Use-Case-Diagramm festgehalten sind.

1. **Noten eingeben:** Um den Notenschnitt berechnen zu können, muss man auch Seine Zeugnis- bzw. Abschlussprüfungsnoten eingeben können. Diese werden dann ausgelesen und gespeichert. Somit kann dann der Schnitt berechnet oder eine Prognose erstellt werden.
2. **Notenschnitt berechnen:** Aus den eingegebenen Noten soll dann der Schnitt berechnet werden können. Mathematisch ist das nicht schwer, jedoch müssen die Noten dazu zuerst ausgelesen und gespeichert werden.
3. **Prognose erstellen:** Aus seinen bisherigen Noten kann man sich eine Prognose für die benötigte Prüfungsnote berechnen lassen, um den Abschluss zu bestehen.
4. **Noten importieren/exportieren:** Das Programm soll auch Noten aus einer CSV-Datei auslesen können, die dann in die Felder eingefügt werden, bzw. Noten in eine CSV-Datei speichern können.
5. **Fächer hinzufügen:** Falls nebst unseren voreingestellten Fächern noch ein anderes Fach relevant für den Abschluss sein sollte, kann man dieses Fach hier ebenfalls erfassen und auch



dieses in die Berechnung einbeziehen. Somit ist das Programm für eine grössere Menge an Endbenutzern geeignet.

6. Diagramm Notenentwicklung: Aus den eingegebenen Noten wird ein Diagramm erstellt, welches den Notenverlauf über die Semester veranschaulicht.
7. Felder leeren: Um das Löschen von Noten aus der Tabelle zu vereinfachen, kann man dies mit einem Button tun. Die Funktion löscht dann alle Werte in der Tabelle auf einmal.
8. Speichern: Auch das Speichern von Noten soll möglich sein. Somit hat man die Werte auch noch, wenn man das Programm schliesst und ein anderes Mal wieder öffnet.

## 3.2. Erläuterung Funktionen

### 3.2.1. Berechnen

Die Funktion «Berechnen» ist dazu da, aus den eingegebenen Noten den Notenschnitt zu berechnen. Füllt man alle Semesternoten ein, gibt es einem den Notenschnitt, den man erreicht hat, aus. Füllt man nicht alle Semesternoten aus, so setzt es für die Leeren Felder den Wert Null und errechnet den Schnitt der Semester, die man ausgefüllt hat. Ebenso gibt es die eingegebenen Noten in einem Diagramm wieder, um den Notenverlauf zu veranschaulichen und die Entwicklung seiner Fachnoten besser einschätzen zu können.

### 3.2.2. Import CSV

Um den Umstieg vom bisherigen Excel-Notenrechner auf unseren Notenrechner zu vereinfachen, kann man den CSV-Export des Excel-Notenrechners hier einfügen. Alle im Excel ausgefüllten Felder werden dann auch in unserem Notenrechner ausgefüllt, wodurch man sich die Zeit spart, die Noten von Hand einzutippen.

### 3.2.3. Export CSV

Auch der Export einer CSV-Datei ist möglich. Mit dieser Funktion kann man die Werte im Notenrechner ganz einfach in eine CSV-Datei exportieren, um diese dann zum Beispiel an einen Klassenlehrer oder ähnliches zu verschicken.

### 3.2.4. Speichern

- Die Speichern Funktion speichert alle Werte, die in der Tabelle ausgefüllt sind, in das Programm. Somit sind die Werte auch noch vorhanden, wenn man das Programm schliesst und ein anderes Mal wieder öffnet.

### 3.2.5. Felder leeren

Das Löschen von Noten aus dem Notenrechner ist sehr einfach. Anstatt von Hand alle Noten einzeln Löschen zu müssen, gibt es einen Button, mit dem man alle Felder auf einmal leeren kann. Die Werte werden beim Betätigen des Buttons erst einmal nur für den Moment gelöscht. Will man sie endgültig aus dem Programm löschen, muss man, nachdem man die Felder geleert hat, auch wieder speichern.

### 3.2.6. Prognose

Die Funktion Prognose erstellt eine Prognose über die notwendigen Abschlussprüfungsnoten. Füllt man alle Felder aus, berechnet die Funktion den Schnitt der einzelnen Fächer und berechnet daraus die tiefst möglichen Noten für die Abschlussprüfung, mit denen man den

Abschluss besteht. Füllt man nicht alle Semesternoten aus, so wird für die leergelassenen Felder die Note 4.0 gesetzt und mit dem daraus berechneten Schnitt gerechnet. Die Funktion Prognose gibt ebenso eine Auswertung der Gesamtnote aus, in der angezeigt wird, ob man die Vorgaben bezüglich Gesamtnote, Anzahl ungenügender Noten und der Notenabweichung unter 4.0 erfüllt. Erfüllt man eine der Vorgaben nicht, so wird einem dies angezeigt. Nicht erfüllte Kriterien werden mit roter Farbe gekennzeichnet, erfüllte Kriterien mit der Farbe Grün.

### 3.2.7. Diagramm

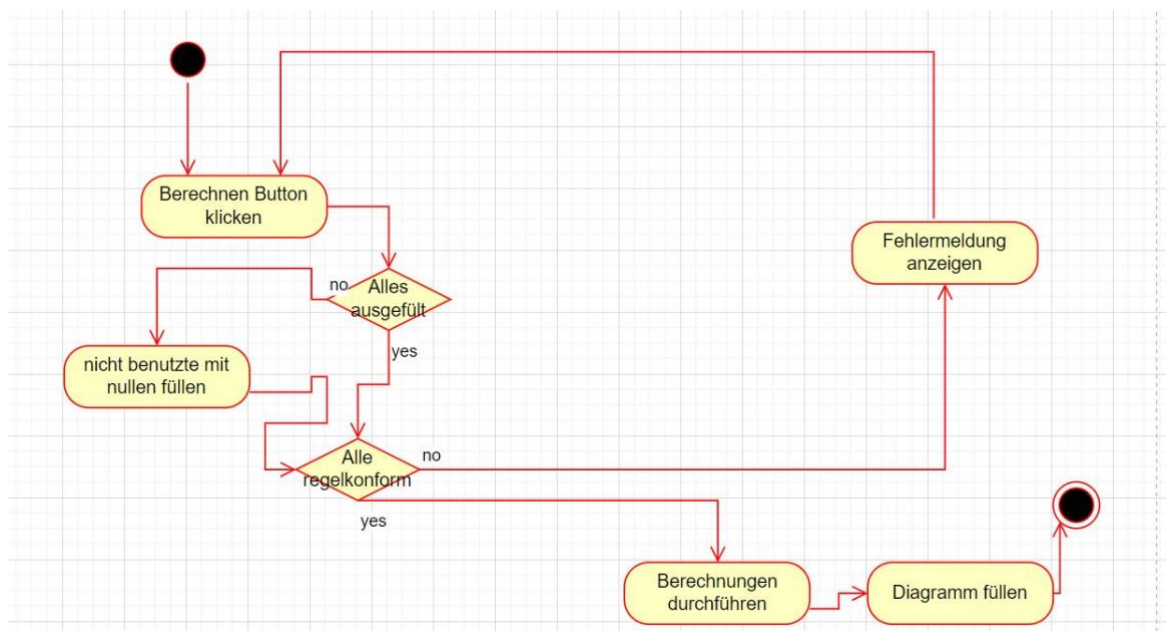
Die Diagramm Funktion füllt die Semesternoten in ein Diagramm ab und veranschaulicht so den Notenverlauf. Die einzelnen Schulfächer sind im Diagramm mit verschiedenfarbigen Linien gekennzeichnet, wobei sich unten am Diagramm eine Legende befindet, in welcher die für die Fächer benutzten Farben vermerkt sind.

### 3.2.8. Fächer hinzufügen

Die Funktion Fächer hinzufügen war ursprünglich gedacht, um das Programm flexibler zu gestalten. Somit könnte es auch immer noch eingesetzt werden, wenn es zum Beispiel eine Veränderung im Lehrplan gibt. Leider konnten wir die Funktion nicht fehlerfrei umsetzen, weshalb wir sie schliesslich gänzlich aus dem Programm genommen haben.

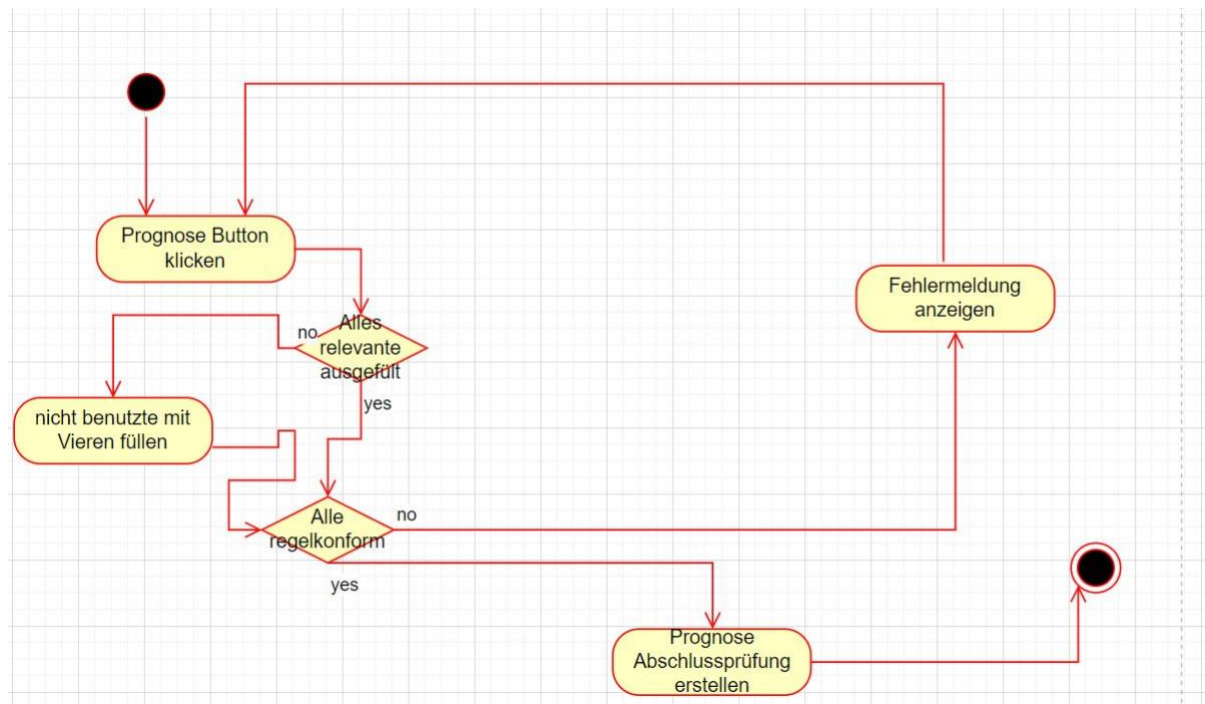
## 3.3. Aktivitätsdiagramme

### 3.3.1. Berechnen

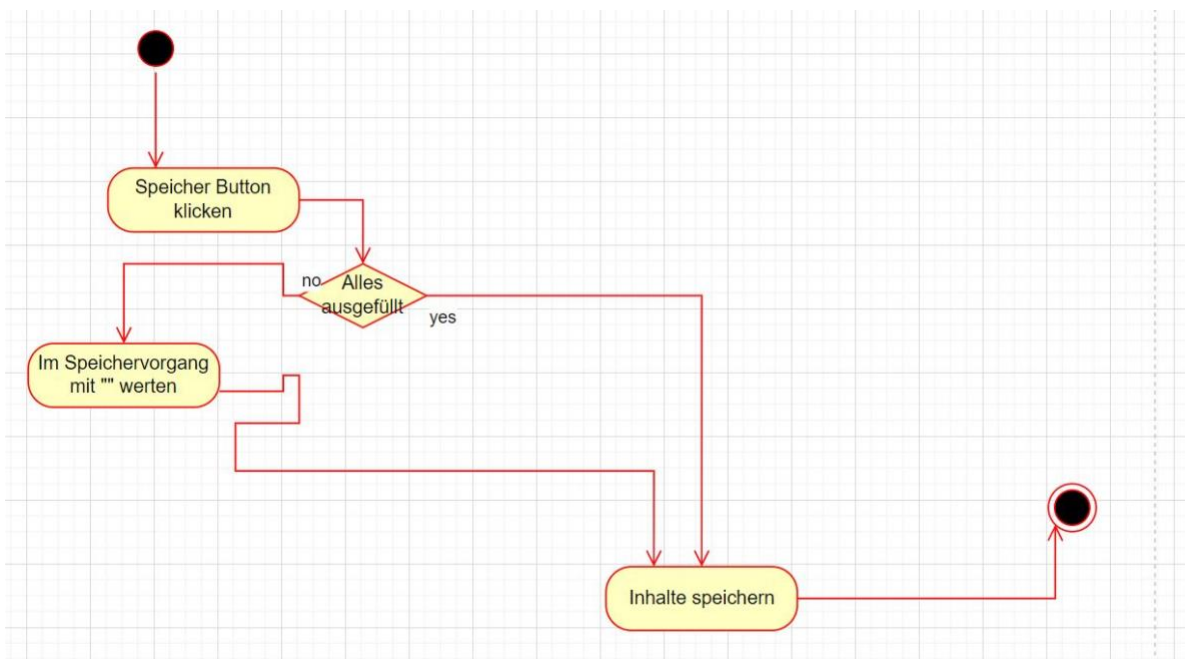




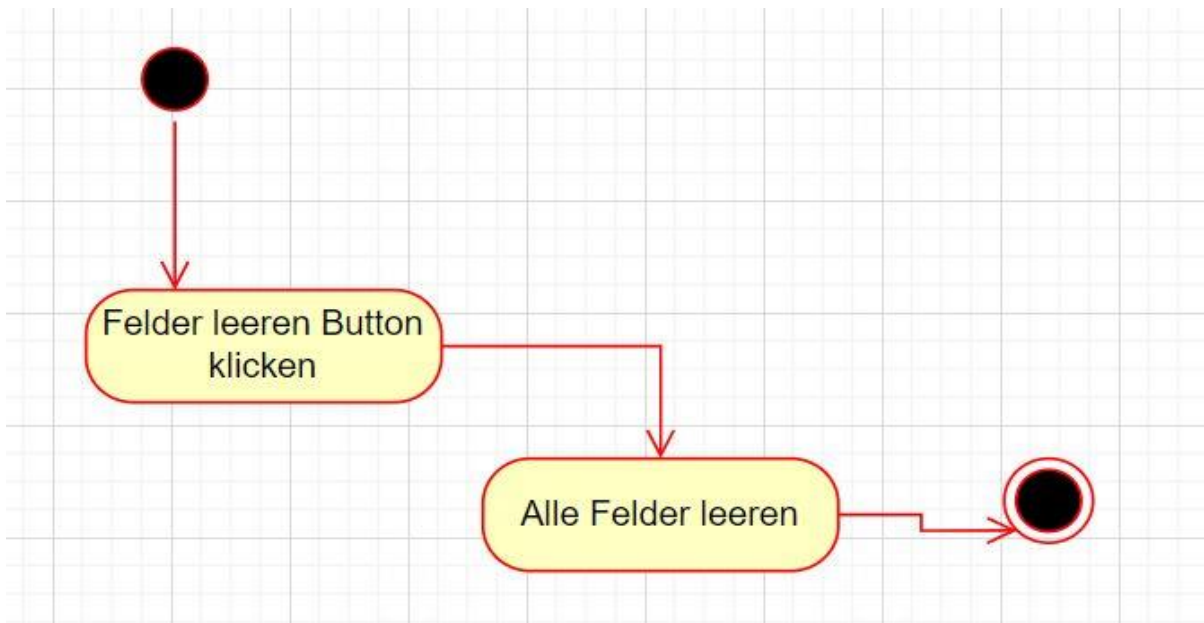
### 3.3.2. Prognose



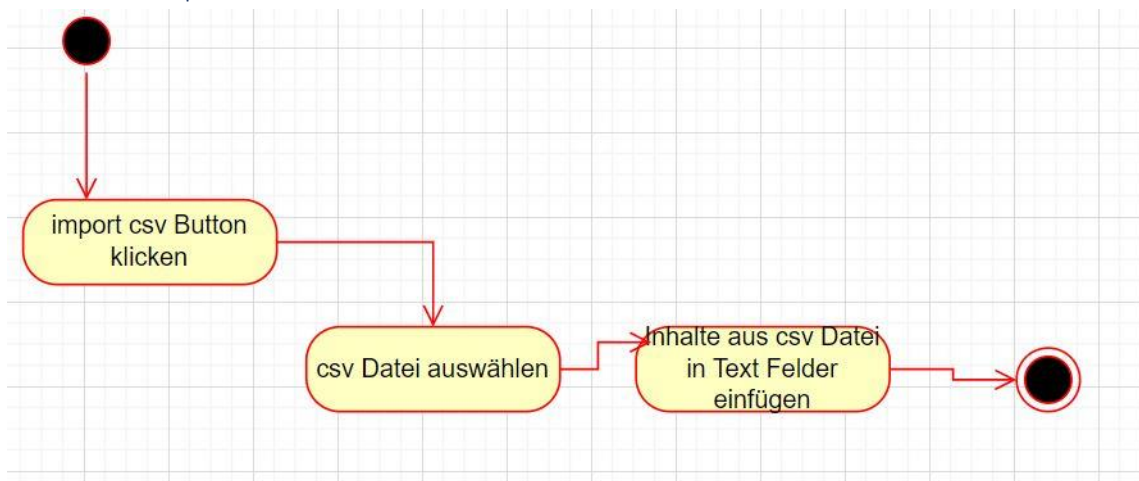
### 3.3.3. Speichern



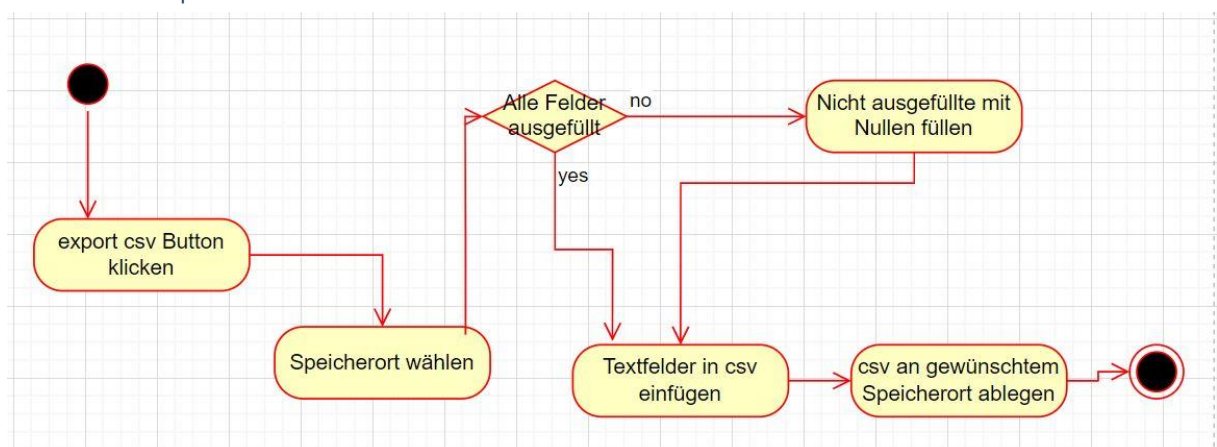
### 3.3.4. Felder leeren



### 3.3.5. Import CSV



### 3.3.6. Export CSV



### 3.4. Erläuterung Code

#### 3.4.1. save()

```
306 for (int i = 0; i < fieldNames.length; i++) {
307     String value = fields[i].getText();
308     if (value != null && !value.isEmpty()) {
309         jsonObject.put(fieldNames[i], value);
310     } else {
311         jsonObject.put(fieldNames[i], "");
312     }
313 }
314
315
316 try (FileWriter fileWriter = new FileWriter(filePath)) {
317     fileWriter.write(JSONValue.toJSONString(jsonObject));
318 } catch (IOException e) {
319     e.printStackTrace();
320 }
```

Um die Textfelder zu speichern, wird deren Inhalt mithilfe obiger Methode in ein .json File geschrieben, woraus dieser beim nächsten aufstarten des Programmes wieder eingelesen wird.

Man geht auf Zeile 306 einmal durch den gesamten Array fieldNames durch, um alle Textfelder zu lesen und sie dann in die ArrayList jsonObject zu schreiben. Im Array fieldNames sind alle Namen der benötigten Variablen. Im Falle eines leeren Textfeldes wird ein leeres Anführungszeichen der Liste hinzugefügt auf Zeile 311. Falls dies nicht der Fall ist, wird die Variable value an dieser Stelle in die Liste eingefügt. In diese speichert man auf Zeile 307 den Wert der Variable im Array fields, indem die Variablen gespeichert sind. Wenn man so durch alle Felder gegangen ist, schreibt man auf Zeile 316-320 die gesamte Liste im .json Format mithilfe von json.simple Funktionen in das .json File dessen Pfad den «filePath» darstellt.

#### 3.4.2. Diagrammfunktion

```
911 // Create a series for each subject
912 for (int i = 0; i < subjects.length; i++) {
913     XYChart.Series<Number, Number> series = new XYChart.Series<>();
914     series.setName(subjects[i]);
915     series.getData().add(new XYChart.Data<>(< 0, y: 0));
916
917     TextField[] actuallist = new TextField[textFields[i].length];
918     for (int j = 0; j < textFields[i].length; j++) {
919         if (!textFields[i][j].getText().isEmpty()) {
920             actuallist[j] = textFields[i][j];
921         }
922     }
923     for (int j = 0; j < actuallist.length; j++) {
924         if (actuallist[j] != null && !actuallist[j].getText().isEmpty() && !Objects.equals(actuallist[j].getText(), "0")) {
925             if (actuallist[j] != null && !actuallist[j].getText().isEmpty() && !Objects.equals(actuallist[j].getText(), "0.0")) {
926                 series.getData().add(new XYChart.Data<>(< j + 1, Double.parseDouble(actuallist[j].getText())));
927             }
928         }
929     }
930
931 // Add the series to the chart
932 lineChart.getData().add(series);
933 }
```

In diesem zentralen Textausschnitt geht man durch eine Liste der Fächer als String.

Man erstellt eine neue Series für jedes Fach und setzt den Namen den String in der Liste. Die erste stelle wird auf null gesetzt. Danach wird eine Liste erstellt auf Zeile 917 in der man folgend im Falle, dass die Variable aus dem Array «textFields» nicht leer ist dessen Wert einspeichert. Dies macht man für jedes Objekt des Arrays. Danach geht man durch den neu gefüllten Array und überprüft diesen, ob er leer 0 oder 0.0 Werte hat. Falls nicht werden sie der series hinzugefügt. Falls doch werden sie ignoriert. Zum Schluss fügt man diese series dem Diagramm hinzu um sie anzuzeigen.

### 3.4.3. prognose()

```
546:   for (int i = 0; i < textFields.length; i++) {  
547:       double sum = 0;  
548:       int count = 0;  
549:       for (int j = 0; j < textFields[i].length; j++) {  
550:           sum += Double.parseDouble(textFields[i][j].getText());  
551:           count++;  
552:       }  
553:       double average = sum / count;  
554:       double rounded = Math.round(average * 2) / 2.0;  
555:       average = rounded;  
556:       double prognoseSchnitt = 7.5 - average;  
557:       double prognoseEinzel = prognoseSchnitt - 0.25;  
558:       double prognoseS = prognoseEinzel + 0.25;  
559:       double prognoseM = prognoseEinzel - 0.25;  
560:       for (int k = 0; k < 1; k++) {  
561:           switch (ziele[i].length) {  
562:               case 1:  
563:                   ziele[i][k].setText(Double.toString(prognoseS));  
564:                   break;  
565:               case 2:  
566:                   ziele[i][k].setText(Double.toString(prognoseS));  
567:                   ziele[i][k+1].setText(Double.toString(prognoseM));  
568:                   break;  
569:           }  
570:       }
```

Dieser Code geht zuerst durch den Array textFields. Es werden zwei Variablen für die Summe und den Zähler erstellt. Danach geht man durch das i. te Objekt des Arrays und rechnet dessen Wert an Stelle j der Summe hinzu. Den Zähler zählt man hoch. Danach um den Schnitt zu rechnen, dividiert man die Summe durch den Zähler und rundet diese. Danach rechnet man 7.5-die bekommene Zahl, da man insgesamt aus Abschlussprüfung und Semestern 7.5 erhalten muss, um die 4 zu erreichen. Danach werden die zwei Variablen prognoseS und M gesetzt. In einem weiteren loop geht man einmal durch und überprüft wie viele Prüfungen, dessen Variablen im Array ziele gespeichert sind, das Fach hat.

Im Falle 1 wird die Prüfung zu prognoseS gesetzt. Im Falle 2 setzt man die 1. Variable prognoseS und die zweite prognoseM.

## 3.4.4. delete()

```
483 public void delete() {
484     faeher.clear();
485     for (Node node : grid1.getChildren()) {
486         if (node instanceof TextField) {
487             TextField textField = (TextField) node;
488             textField.clear();
489         }
490     }
491 }
```

In diesem Code Abschnitt wird die Methode delete dargestellt. Zuerst werden die Fächer in der Klasse Fach geleert, sodass da die Werte auf null gesetzt werden. Danach geht man durch alle Textfelder im grid1 und leert sie mit «textField.clear()».

## 3.4.5. import\_csv()

```
327 public void import_csv() {
328     //choose file
329     FileChooser fileChooser = new FileChooser();
330     fileChooser.setTitle("Open CSV file");
331     fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("CSV files", ...strings: "*.csv"));
332     File selectedFile = fileChooser.showOpenDialog(window: null);
333     if (selectedFile != null) {
334         // read data
335         String line = "";
336         String splitBy = ",";
337         try {
338             // parsing a CSV file into BufferedReader class constructor
339             BufferedReader br = new BufferedReader(new FileReader(selectedFile.getPath()));
340             for(int i = 0; (line = br.readLine()) != null; i++) {
341                 String[] faeherCSV = line.split(splitBy);
342                 // add read data as "Fach" to faeher list
343                 if(i >= 1) {
344                     faeher.add(new Fach(faeherCSV[0]));
345                     ArrayList<Double> noten = new ArrayList<>(List.of(
346                         Double.parseDouble(faeherCSV[1]),
347                         Double.parseDouble(faeherCSV[2]),
348                         Double.parseDouble(faeherCSV[3]),
349                         Double.parseDouble(faeherCSV[4]),
350                         Double.parseDouble(faeherCSV[5]),
351                         Double.parseDouble(faeherCSV[6])));
352                     faeher.get(i-1).setZeugnisnoten(noten);
353                     faeher.get(i-1).setSchriftlich(Double.parseDouble(faeherCSV[7]));
354                     faeher.get(i-1).setMündlich(Double.parseDouble(faeherCSV[8]));
355                     faeher.get(i-1).setErfahrungsnotePos1(Double.parseDouble(faeherCSV[9]));
356                     faeher.get(i-1).setPrüfungsnotePos2(Double.parseDouble(faeherCSV[10]));
357                 }
358             }
359
360             // set TextFields from faeher list
361             d1.setText(faeher.get(0).getZeugnisnotenBySemester(1)+"");
362             d2.setText(faeher.get(0).getZeugnisnotenBySemester(2)+"");
```

Bei der Methode import\_csv() wird zuerst mithilfe von FileChooser über den Benutzer eine .csv Datei ausgewählt, die dann eingelesen wird mittels BufferedReader. Es wird durch abgesehen von der ersten (weil das nur der Header ist) durch alle Zeilen iteriert und dabei für jedes Fach ein Objekt Fach erstellt und der facher Liste hinzugefügt. Anschliessend werden die Felder der FXML-Datei damit befüllt.



## 3.4.6. export\_csv() und

```
457 public void export_csv() throws IOException {
458     berechnen();
459     // get data read for csv file
460     String csv = "Fach,1. Semester,2. Semester,3. Semester,4. Semester,5. Semester,6. Semester,schriftlich,muendlich
461     for(Fach fach : faecher) {
462         csv += fach.toCSV();
463     }
464
465     FileChooser fileChooser = new FileChooser();
466     fileChooser.setTitle("Save CSV file");
467     fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter( s: "CSV files", ...strings: "*.csv"));
468     File selectedFile = fileChooser.showSaveDialog( window: null);
469     if (selectedFile != null) {
470         if (selectedFile != null) {
471             try (PrintWriter writer = new PrintWriter(selectedFile)) {
472                 // write file
473                 writer.write(csv);
474             } catch (FileNotFoundException e) {
475                 e.printStackTrace();
476             }
477         }
478     }
479 }
```

```
101 public String toCSV() {
102     String csv = name + ",";
103     // zeugnisnoten
104     if(zeugnisnoten.size() == 6) {
105         for (Double note : zeugnisnoten) {
106             csv += note + ",";
107         }
108     } else if(zeugnisnoten.size() == 4) {
109         for (Double note : zeugnisnoten) {
110             csv += note + ",";
111         }
112         csv += "0,0,";
113     } else if(zeugnisnoten.size() == 2) {
114         csv += "0,0,";
115         for (Double note : zeugnisnoten) {
116             csv += note + ",";
117         }
118         csv += "0,0,";
119     } else if(zeugnisnoten.size() == 1) {
120         csv += "0,0,0,0,0,";
121         for (Double note : zeugnisnoten) {
122             csv += note + ",";
123         }
124     } else {
125         csv += "0,0,0,0,0,0,";
126     }
127
128     // schriftlich - falls vorhanden
129     if(schriftlich != 0.0) {
130         csv += schriftlich + ",";
131     } else {
132         csv += "0,";
133     }
134
135     // mündlich - falls vorhanden
136     if(muendlich != 0.0) {
137         csv += muendlich + ",";
138     } else {
139         csv += "0,";
140     }
141
142     // prüfungsnotePos2 - falls vorhanden
143     if(prüfungsnotePos2 != 0.0) {
144         csv += prüfungsnotePos2 + ",";
145     } else {
146         csv += "0,";
147     }
148
149     // erfahrungsnotePos1 - falls vorhanden
150     if(erfahrungsnotePos1 != 0.0) {
151         csv += erfahrungsnotePos1 + ",";
152     } else {
153         csv += "0,";
154     }
155
156     // fachnote - falls vorhanden
157     if(fachnote != 0.0) {
158         csv += fachnote;
159     } else {
160         csv += "0,";
161     }
162
163     return csv += "\n";
164 }
165 }
```

Die Methode `export_csv()` exportiert ähnlich, wie auch bei `import_csv()` dorthin, wo der Benutzer den Pfad auswählt. Mithilfe der `toCSV()` Methode aus der `Fach` Klasse wird für jedes Fach aus der `faecher` Liste ein passender String für die `.csv` Datei generiert und anschliessend als solche erstellt.

### 3.4.7. berechnen()

```
593 public void berechnen() throws IOException {
594     for (Node node : grid1.getChildren()) {
595         if (node instanceof TextField) {
596             TextField textField = (TextField) node;
597             if (textField.getText() != null && !textField.getText().isEmpty()) {
598                 try {
599                     double v = Double.parseDouble(textField.getText());
600                     if (!textField.getText().matches(regex: "^[1-5](\\.5)?$|^([0-6])$|^([0-6](\\.0)?)$")) {
601                         allFieldsFilled = false;
602                         break;
603                     } else {
604                         allFieldsFilled = true;
605                     }
606                 } catch (NumberFormatException e) {
607                     allFieldsFilled = false;
608                     break;
609                 }
610             }
611             // check if TextField is not set and set it to 0
612             else {
613                 if (textField.getText().isEmpty()) {
614                     textField.setText("0");
615                 }
616             }
617         }
618     }
619 }
```

In der `berechnen()` Methode wird zuerst geprüft, ob die Werte der `TextFields` zulässig sind mittels `RegEx`. Ist ein Feld leer, dann wird der Wert 0 gesetzt.

```
620 if (allFieldsFilled) {
621     faecher.clear();
622     // daten als "Fach" objekt speichern und berechnen
623     Fach d = new Fach( name: "Deutsch");
624     d.setZeugnisnoten(new ArrayList<>(Arrays.asList(
625         Double.parseDouble(d1.getText()),
626         Double.parseDouble(d2.getText()),
627         Double.parseDouble(d3.getText()),
628         Double.parseDouble(d4.getText()),
629         Double.parseDouble(d5.getText()),
630         Double.parseDouble(d6.getText()))));
631     d.setSchriftlich(Double.parseDouble(dS.getText()));
632     d.setMündlich(Double.parseDouble(dM.getText()));
633     d.setErfahrungsnotePos1(d.getPos1());
634     d.setPrüfungsnotePos2(d.getPos2());
635
636     ChangeListener<String> fnDListener = (observable, oldValue, newValue) -> {
637         if (d.getFachnote() < 4) {
638             FnD.setStyle("-fx-background-color: #FFA07A !important;");
639         } else {
640             FnD.setStyle("-fx-background-color: #90EE90 !important;");
641         }
642     };
643     FnD.textProperty().addListener(fnDListener);
644
645     zD.setText(d.getErfahrungsnotePos1()+"");
646     pD.setText(d.getPrüfungsnotePos2()+"");
647     FnD.setText(d.getFachnote()+"");
648     faecher.add(d);
649 }
```

Sind alle Werte gültig, so werden für alle Fächer wie beim `import_csv()` je ein Objekt Fach erstellt. Dannach werden entsprechend mit Hilfsmethoden die Erfahrungsnote (Position 1), Prüfungsnote (Position 2) und somit die Fachnote kalkuliert. Wenn einer dieser drei Werte ungenügend ist, dann wird dieses Feld rot angezeigt.

```
863 // gesamtnote, anzahl tiefnoten und tiefpunkte berechnen
864
865ChangeListener<String> gesamtnoteListener = (observable, oldValue, newValue) -> {
866     if (!(((double) Math.round(getGesamtnote()*10)/10) >= 4)) {
867         gesamtnote.setStyle("-fx-background-color: #FFA07A !important;");
868     } else {
869         gesamtnote.setStyle("-fx-background-color: #90EE90 !important;");
870     }
871 };
872
873 gesamtnote.textProperty().addListener(gesamtnoteListener);
874
875ChangeListener<String> anzahlTiefnotenListener = (observable, oldValue, newValue) -> {
876     if (getAnzahlTiefnoten() > 2) {
877         anzahl_tiefnoten.setStyle("-fx-background-color: #FFA07A !important;");
878     } else {
879         anzahl_tiefnoten.setStyle("-fx-background-color: #90EE90 !important;");
880     }
881 };
882
883 anzahl_tiefnoten.textProperty().addListener(anzahlTiefnotenListener);
884
885ChangeListener<String> tiefpunkteListener = (observable, oldValue, newValue) -> {
886     if (getTiefpunkte() > 2) {
887         tiefpunkte.setStyle("-fx-background-color: #FFA07A !important;");
888     } else {
889         tiefpunkte.setStyle("-fx-background-color: #90EE90 !important;");
890     }
891 };
892
893 tiefpunkte.textProperty().addListener(tiefpunkteListener);
894
895 gesamtnote.setText((((double) Math.round(getGesamtnote()*10)/10)+"" );
896 anzahl_tiefnoten.setText(getAnzahlTiefnoten()+"" );
897 tiefpunkte.setText(getTiefpunkte()+"" );
```

Schlussendlich werden die Gesamtnote, Anzahl Tiefnoten und Tiefpunkte berechnet und gesetzt. Gelten diese als erfüllt, so werden sie grün und sonst rot eingefärbt.

## 4. Dependencies

Für Java FXML haben wir die Dependency openjfx benutzt.

```
<dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-controls</artifactId>
    <version>19</version>
</dependency>
<dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-fxml</artifactId>
    <version>19</version>
</dependency>
```





Für JUnit Tests haben wir auch eine Dependency dafür ausgewählt.

```
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>${junit.version}</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-engine</artifactId>
  <version>${junit.version}</version>
  <scope>test</scope>
</dependency>
```

Um die Daten in einer .json Datei zu speichern und auszulesen benötigen wir diese zwei json Dependencies.

```
<dependency>
  <groupId>com.vaadin.external.google</groupId>
  <artifactId>android-json</artifactId>
  <version>0.0.20131108.vaadin1</version>
</dependency>
<dependency>
  <groupId>com.googlecode.json-simple</groupId>
  <artifactId>json-simple</artifactId>
  <version>1.1.1</version>
</dependency>
```

Gleich wie bei den .json Dateien benötigt auch der Import sowie auch Export von .csv Dateien eine .csv Dependency.

```
<dependency>
  <groupId>com.opencsv</groupId>
  <artifactId>opencsv</artifactId>
  <version>4.1</version>
</dependency>
```



## 5. Testfälle

Nr	Input	Output	erfüllt
1	Es werden mehr als 2 neg. Noten eingetragen, weniger als einen 4er Schnitt und mehr als zwei Tiefpunkte erreicht.	Promotionsstand negativ (sonst positiv)	✓
2	Noten werden eingetragen	Notenschnitt sowie Promotionsstand wird angegeben.	✓
3	Noten werden mittels .csv Datei importiert	Noten werden automatisch eingetragen. (vgl. Test Nr 1)	✓
4	Noten werden exportiert	Der Anwender erhält ein PDF/PNG mit den Noten, -schnitte und Promotionsentscheide.	✓
5	Fach wird hinzugefügt	Es können Noten für das neu hinzugefügte Fach eingetragen werden und sie werden im Promotionsstand berücksichtigt.	✗
6	Entwicklung (Notenschnitt/Zeit) anzeigen	Ein Diagramm wird gezeigt, das die Entwicklung des Schnitts über die Zeit anzeigt.	✓
7	Bei Aufrufen des Programs wird eine Authentifizierung verlangt	Man kann sich nur mit gültigen Anmeldedaten authentifizieren oder evtl. neu registrieren.	✗

Vermerkung zu Nummer 5 und 7: Aufgrund von Zeitdruck haben wir unsere Prioritäten auf die anderen (Haupt-)Funktionen gelegt.

## 6. Fazit

Die Arbeit am Notenrechner ist anspruchsvoll, da das Design mit JavaFX komplizierter ist als gedacht. Trotzdem sind keine grossen Probleme aufgetaucht. Wir konnten den Notenrechner mit allen seinen Hauptfunktionen umsetzen und haben sogar mehr Funktionen als der Excel-Notenrechner der Schule. Leider konnten wir die Funktion Fächer hinzufügen nicht umsetzen, die anderen Funktionen funktionieren aber einwandfrei.

Die Gruppenarbeit hat grundsätzlich gut funktioniert; wir hatten einen Plan, wer was zu tun hat und haben diesen auch so umsetzen können. In unserem GitHub Repository haben wir in der Form eines Issues eine [ToDo Liste](#) erstellt, welche uns während des Projekts als gute Übersicht bot. Problematisch war, dass das Verbinden mit GitHub für Rafael zuerst nicht funktioniert hat und dass das Projekt plötzlich nicht mehr lief und wir es neu aufsetzen mussten. Dennoch haben wir fast alles rechtzeitig fertig geschafft und sind mit unserem Ergebnis zufrieden.



## 7. Bestätigung Eigenständigkeit

### 7.1. Ehrenwörtliche Erklärung

Wir erklären, dass wir die vorliegende Arbeit selbstständig, ohne fremde Hilfe und in eigener Sprache verfasst haben und keine anderen als die angegebenen Hilfsmittel benutzt worden sind. Insbesondere versichern wir, dass wörtliche und sinngemässe Übernahmen aus anderen Werken (Internet, Bücher, Zeitschriften u.ä.) mit Fussnoten und im Quellenverzeichnis erwähnt werden. Wir nehmen zur Kenntnis, dass eine unentschuld bare, verspätete Ablieferung der Arbeit zu einem Notenabzug führt.

Ort: Winterthur Datum: 19.04.2023

Unterschriften Autoren/Autorinnen:

Rafael Gahler:

Rafael Gahler

Samira Stragiotti:

Samira Stragiotti

Nico Zollinger:

N. Zollinger

Die Arbeit darf einem interessierten Publikum gezeigt werden.

## 8. Quellenverzeichnis

1. Für das Schreiben des Codes wurden folgende Websites zur Hilfe genommen:
  - a. Oracle: <https://docs.oracle.com/en/> (Abrufdatum 19.04.2023)
  - b. Stackoverflow: <https://stackoverflow.com/> (Abrufdatum 19.04.2023)
2. Für das Erstellen der Prototypen wurde die Applikation «Figma» verwendet.
3. Für das Erstellen der Diagramme wurde folgende Website benutzt:
  - a. Diagrams: <https://app.diagrams.net/>
4. Das Icon des Programms haben wir von folgender Website genommen:
  - a. Flaticon: [https://www.flaticon.com/free-icon/colors\\_10235999?term=grades&page=1&position=28&origin=search&related\\_id=10235999](https://www.flaticon.com/free-icon/colors_10235999?term=grades&page=1&position=28&origin=search&related_id=10235999) (Abrufdatum 19.04.2023)
5. Das Icon der Fehlermeldungen ist von folgender Website:
  - a. Flaticon: [https://www.flaticon.com/free-icon/prohibition\\_929457?term=x&page=2&position=26&origin=search&related\\_id=929457](https://www.flaticon.com/free-icon/prohibition_929457?term=x&page=2&position=26&origin=search&related_id=929457) (Abrufdatum 19.04.2023)
6. Während dem Projekt haben wir ab und zu im anschließenden Buch nachgeschaut:
  - a. «Markt + Technik Verlag (2017), Das Grosse Computerlexikon»

### 8.1. Abbildungsverzeichnis

1.

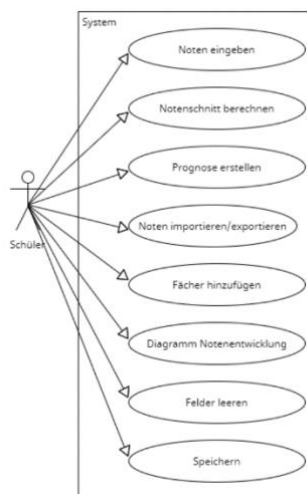


Abbildung 1: Use-Case Diagramm

2.

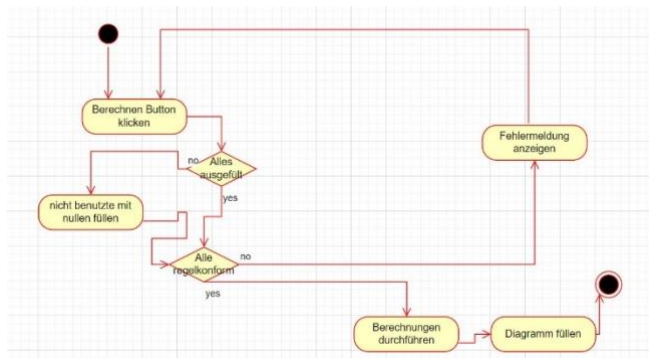


Abbildung 2: Aktivitätsdiagramm Berechnen-Funktion

3.

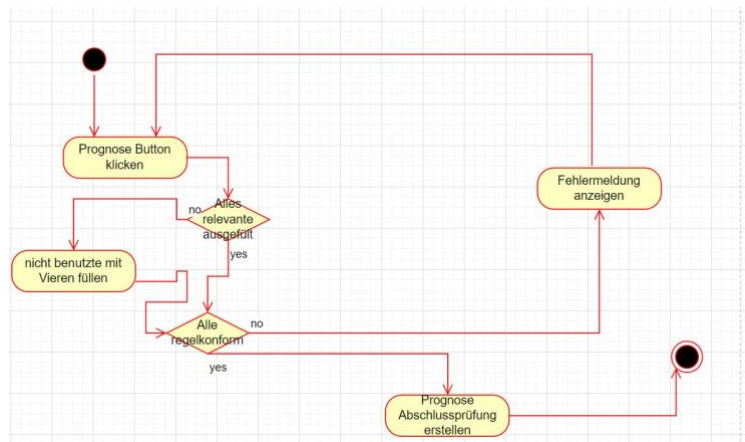


Abbildung 3: Aktivitätsdiagramm Prognosefunktion

4.

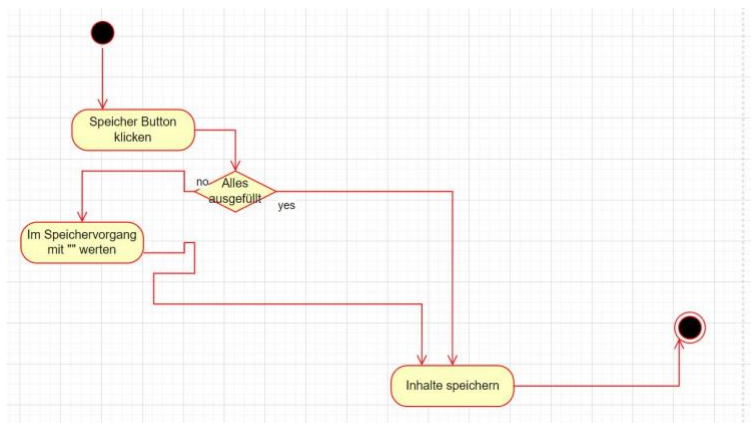


Abbildung 4: Aktivitätsdiagramm Speicherfunktion

5.

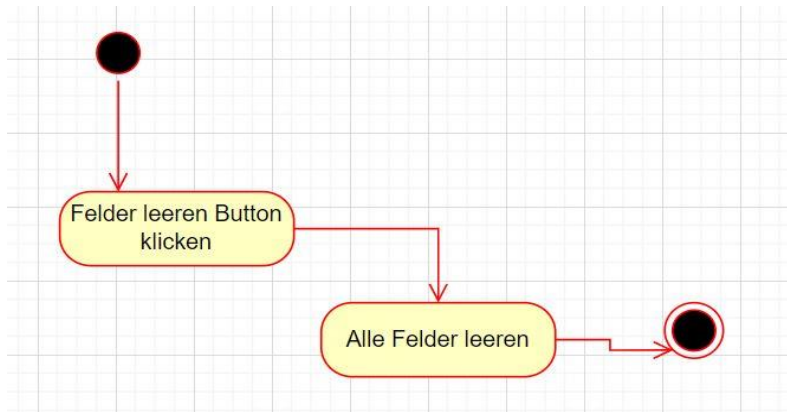


Abbildung 5: Aktivitätsdiagramm Felder leeren Funktion

6.

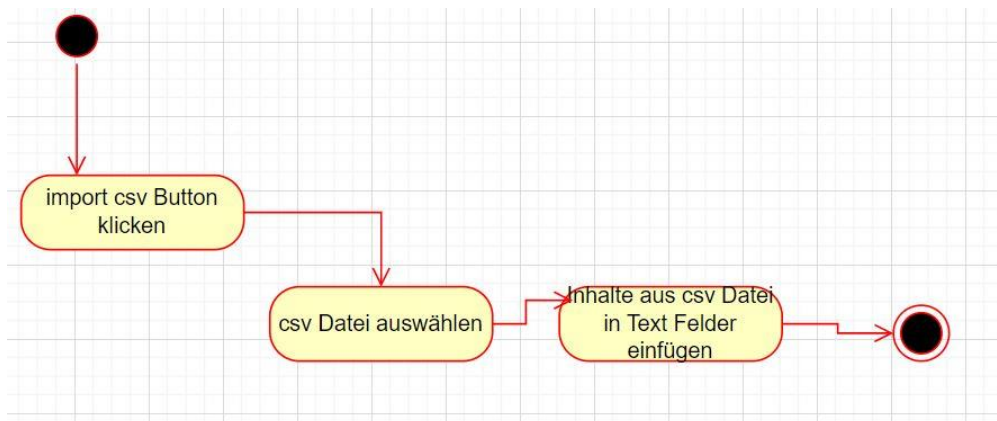


Abbildung 6: Import CSV Funktion

7.

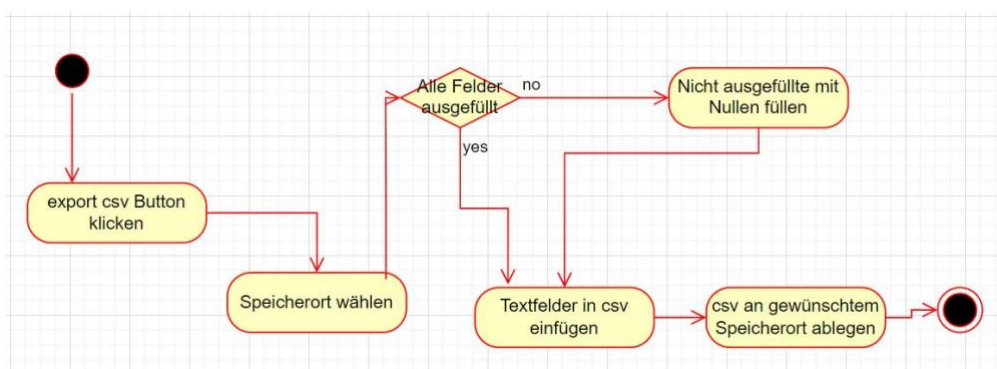


Abbildung 7: Export CSV Funktion

8.

```
306 for (int i = 0; i < fieldNames.length; i++) {
307     String value = fields[i].getText();
308     if (value != null && !value.isEmpty()) {
309         jsonObject.put(fieldNames[i], value);
310     } else {
311         jsonObject.put(fieldNames[i], "");
312     }
313 }
314
315
316 try (FileWriter fileWriter = new FileWriter(filePath)) {
317     fileWriter.write(JSONValue.toJSONString(jsonObject));
318 } catch (IOException e) {
319     e.printStackTrace();
320 }
```

Abbildung 8: Codeausschnitt Speicherfunktion

9.

```
911 // Create a series for each subject
912 for (int i = 0; i < subjects.length; i++) {
913     XYChart.Series<Number, Number> series = new XYChart.Series<>();
914     series.setName(subjects[i]);
915     series.getData().add(new XYChart.Data<>(< 0, y: 0));
916
917     TextField[] actuellList = new TextField[textFields[i].length];
918     for (int j = 0; j < textFields[i].length; j++) {
919         if (!textFields[i][j].getText().isEmpty()) {
920             actuellList[j] = textFields[i][j];
921         }
922     }
923     for (int j = 0; j < actuellList.length; j++) {
924         if (actuellList[j] != null && !actuellList[j].getText().isEmpty() && !Objects.equals(actuellList[j].getText(), "0.0")) {
925             if (actuellList[j] != null && !actuellList[j].getText().isEmpty() && !Objects.equals(actuellList[j].getText(), "0.0")) {
926                 series.getData().add(new XYChart.Data<>(< j + 1, Double.parseDouble(actuellList[j].getText())));
927             }
928         }
929     }
930
931     // Add the series to the chart
932     lineChart.getData().add(series);
933 }
```

Abbildung 9: Codeausschnitt Diagrammfunktion

10.

```
546. for (int i = 0; i < textFields.length; i++) {  
547.     double sum = 0;  
548.     int count = 0;  
549.     for (int j = 0; j < textFields[i].length; j++) {  
550.         sum += Double.parseDouble(textFields[i][j].getText());  
551.         count++;  
552.     }  
553.     double average = sum / count;  
554.     double rounded = Math.round(average * 2) / 2.0;  
555.     average = rounded;  
556.     double prognoseSchnitt = 7.5 - average;  
557.     double prognoseEinzelN = prognoseSchnitt - 0.25;  
558.     double prognoseS = prognoseEinzelN + 0.25;  
559.     double prognoseM = prognoseEinzelN - 0.25;  
560.     for (int k = 0; k < 1; k++) {  
561.         switch (ziele[i].length) {  
562.             case 1:  
563.                 ziele[i][k].setText(Double.toString(prognoseS));  
564.                 break;  
565.             case 2:  
566.                 ziele[i][k].setText(Double.toString(prognoseS));  
567.                 ziele[i][k+1].setText(Double.toString(prognoseM));  
568.                 break;  
569.         }  
570.     }
```

Abbildung 10: Codeausschnitt Prognosefunktion



11.

```
327 public void import_csv() {
328     //choose file
329     FileChooser fileChooser = new FileChooser();
330     fileChooser.setTitle("Open CSV file");
331     fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("CSV files", ...strings: "*.csv"));
332     File selectedFile = fileChooser.showOpenDialog( window: null);
333     if (selectedFile != null) {
334         // read data
335         String line = " ";
336         String splitBy = ",";
337         try {
338             // parsing a CSV file into BufferedReader class constructor
339             BufferedReader br = new BufferedReader(new FileReader(selectedFile.getPath()));
340             for(int i = 0; (line = br.readLine()) != null; i++) {
341                 String[] faeherCSV = line.split(splitBy);
342                 // add read data as "Fach" to faeher list
343                 if(i >= 1) {
344                     faeher.add(new Fach(faeherCSV[0]));
345                     ArrayList<Double> noten = new ArrayList<>(List.of(
346                         Double.parseDouble(faeherCSV[1]),
347                         Double.parseDouble(faeherCSV[2]),
348                         Double.parseDouble(faeherCSV[3]),
349                         Double.parseDouble(faeherCSV[4]),
350                         Double.parseDouble(faeherCSV[5]),
351                         Double.parseDouble(faeherCSV[6]));
352                     faeher.get(i-1).setZeugnisnoten(noten);
353                     faeher.get(i-1).setSchriftlich(Double.parseDouble(faeherCSV[7]));
354                     faeher.get(i-1).setMündlich(Double.parseDouble(faeherCSV[8]));
355                     faeher.get(i-1).setErfahrungsnotePos1(Double.parseDouble(faeherCSV[9]));
356                     faeher.get(i-1).setPrüfungsnotePos2(Double.parseDouble(faeherCSV[10]));
357                 }
358             }
359
360             // set TextFields from faeher list
361             d1.setText(faeher.get(0).getZeugnisnotenBySemester(1)+"");
362             d2.setText(faeher.get(0).getZeugnisnotenBySemester(2)+"");
```

Abbildung 11: Code Ausschnitt import\_csv()

12.

```
457 public void export_csv() throws IOException {
458     berechnen();
459     // get data read for csv file
460     String csv = "Fach,1. Semester,2. Semester,3. Semester,4. Semester,5. Semester,6. Semester,schriftlich,muendlich
461     for(Fach fach : faeher) {
462         csv += fach.toCSV();
463     }
464
465     FileChooser fileChooser = new FileChooser();
466     fileChooser.setTitle("Save CSV file");
467     fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("CSV files", ...strings: "*.csv"));
468     File selectedFile = fileChooser.showSaveDialog( window: null);
469     if (selectedFile != null) {
470         if (selectedFile != null) {
471             try (PrintWriter writer = new PrintWriter(selectedFile)) {
472                 // write file
473                 writer.write(csv);
474             } catch (FileNotFoundException e) {
475                 e.printStackTrace();
476             }
477         }
478     }
479 }
```

Abbildung 12: Code Ausschnitt export\_csv()

13.

```
101 public String toCSV() {
102     String csv = name + ",";
103     // zeugnisnoten
104     if(zeugnisnoten.size() == 6) {
105         for (Double note : zeugnisnoten) {
106             csv += note + ",";
107         }
108     } else if(zeugnisnoten.size() == 4) {
109         for (Double note : zeugnisnoten) {
110             csv += note + ",";
111         }
112         csv += "0,0,";
113     } else if(zeugnisnoten.size() == 2) {
114         csv += "0,0,";
115         for (Double note : zeugnisnoten) {
116             csv += note + ",";
117         }
118         csv += "0,0,";
119     } else if(zeugnisnoten.size() == 1) {
120         csv += "0,0,0,0,0,";
121         for (Double note : zeugnisnoten) {
122             csv += note + ",";
123         }
124     } else {
125         csv += "0,0,0,0,0,0,";
126     }
127
128     // schriftlich - falls vorhanden
129     if(schriftlich != 0.0) {
130         csv += schriftlich + ",";
131     } else {
132         csv += "0,";
133     }
134
135     // mündlich - falls vorhanden
136     if(mündlich != 0.0) {
137         csv += mündlich + ",";
138     } else {
139         csv += "0,";
140     }
```

Abbildung 13: Code Ausschnitt toCSV() Teil 1

14.

```
142 // prüfungsnotePos2 - falls vorhanden
143 if(prüfungsnotePos2 != 0.0) {
144     CSV += prüfungsnotePos2 + ",";
145 } else {
146     CSV += "0,";
147 }
148
149 // erfahrungsnotePos1 - falls vorhanden
150 if(erfahrungsnotePos1 != 0.0) {
151     CSV += erfahrungsnotePos1 + ",";
152 } else {
153     CSV += "0,";
154 }
155
156 // fachnote - falls vorhanden
157 if(fachnote != 0.0) {
158     CSV += fachnote;
159 } else {
160     CSV += "0,";
161 }
162
163 return CSV += "\n";
164 }
165 }
```

Abbildung 14: Code Ausschnitt toCSV() Teil 2

15.

```
593 public void berechnen() throws IOException {
594     for (Node node : grid1.getChildren()) {
595         if (node instanceof TextField) {
596             TextField textField = (TextField) node;
597             if (textField.getText() != null && !textField.getText().isEmpty()) {
598                 try {
599                     double v = Double.parseDouble(textField.getText());
600                     if (!textField.getText().matches( regex: "^([1-5](\\.5)?)|^([0-6])|^([0-6](\\.0)?)$" )) {
601                         allFieldsFilled = false;
602                         break;
603                     } else {
604                         allFieldsFilled = true;
605                     }
606                 } catch (NumberFormatException e) {
607                     allFieldsFilled = false;
608                     break;
609                 }
610             }
611             // check if TextField is not set and set it to 0
612             else {
613                 if (textField.getText().isEmpty()) {
614                     textField.setText("0");
615                 }
616             }
617         }
618     }
619 }
```

Abbildung 15: Code Ausschnitt berechnen() Teil 1

16.

```

620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648

if (allFieldsFilled) {
    faecher.clear();
    // daten als "Fach" objekt speichern und berechnen
    Fach d = new Fach( name: "Deutsch");
    d.setZeugnisnoten(new ArrayList<>(Arrays.asList(
        Double.parseDouble(d1.getText()),
        Double.parseDouble(d2.getText()),
        Double.parseDouble(d3.getText()),
        Double.parseDouble(d4.getText()),
        Double.parseDouble(d5.getText()),
        Double.parseDouble(d6.getText()))));
    d.setSchriftlich(Double.parseDouble(dS.getText()));
    d.setMündlich(Double.parseDouble(dM.getText()));
    d.setErfahrungsnotePos1(d.getPos1());
    d.setPrüfungsnotePos2(d.getPos2());

    ChangeListener<String> fnDListener = (observable, oldValue, newValue) -> {
        if (d.getFachnote() < 4) {
            FnD.setStyle("-fx-background-color: #FFA07A !important;");
        } else {
            FnD.setStyle("-fx-background-color: #90EE90 !important;");
        }
    };
    FnD.textProperty().addListener(fnDListener);

    zD.setText(d.getErfahrungsnotePos1()+"");
    pD.setText(d.getPrüfungsnotePos2()+"");
    FnD.setText(d.getFachnote()+"");
    faecher.add(d);

```

Abbildung 16: Code Ausschnitt berechnen() Teil 2

17.

```

863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893

// gesamtnote, anzahl tiefnoten und tiefpunkte berechnen
ChangeListener<String> gesamtnoteListener = (observable, oldValue, newValue) -> {
    if (!(((double) Math.round(getGesamtnote()*10)/10) >= 4)) {
        gesamtnote.setStyle("-fx-background-color: #FFA07A !important;");
    } else {
        gesamtnote.setStyle("-fx-background-color: #90EE90 !important;");
    }
};
gesamtnote.textProperty().addListener(gesamtnoteListener);

ChangeListener<String> anzahlTiefnotenListener = (observable, oldValue, newValue) -> {
    if (getAnzahlTiefnoten() > 2) {
        anzahl_tiefnoten.setStyle("-fx-background-color: #FFA07A !important;");
    } else {
        anzahl_tiefnoten.setStyle("-fx-background-color: #90EE90 !important;");
    }
};
anzahl_tiefnoten.textProperty().addListener(anzahlTiefnotenListener);

ChangeListener<String> tiefpunkteListener = (observable, oldValue, newValue) -> {
    if (getTiefpunkte() > 2) {
        tiefpunkte.setStyle("-fx-background-color: #FFA07A !important;");
    } else {
        tiefpunkte.setStyle("-fx-background-color: #90EE90 !important;");
    }
};
tiefpunkte.textProperty().addListener(tiefpunkteListener);

gesamtnote.setText(((double) Math.round(getGesamtnote()*10)/10)+"");
anzahl_tiefnoten.setText(getAnzahlTiefnoten()+"");
tiefpunkte.setText(getTiefpunkte()+"");

```

Abbildung 17: Code Ausschnitt berechnen() Teil 3



## 9. Anhang

### 9.1. Projektvereinbarung



## Projektauftrag IDPA (IMS 2022/2023)

**Thema/Arbeitstitel:**  
Notenrechner

**Personen (Klasse):**

Schülerteam: Stragiotti Samira (3bl)  
Gahler Rafael (3bl)  
Zollinger Nico (3bl)

Betreut von Bernhard Marti (BBW) und Jacques Mock Schindler (KBW)

**Projektidee und Eingrenzung des Themas:**

Übertragung Sie den Excel Notenrechner in ein Java-Programm mit grafischer Oberfläche. Erwartet wird, dass der aktuelle Stand abgespeichert und wieder eingelesen werden kann. Ausserdem soll das Programm einem dabei unterstützen, Prognoserechnungen anzustellen.

**Termine**

- **Kick-off-Gespräch/Erteilung Auftrag:** Mittwoch, 18. Januar 23, 16:00 Uhr
- **Planung** Abgabe Grobdisposition und Prozessbeschreibung\* (gemäss IDPA-Vorgaben 4.1) bis Di, 21.2.2023
- **Erstes Beratungsgespräch:** 1. März 23  
Angepasste Grobdispo/Prozessbeschreibung, Journal, offene Fragen bis 28.3.2022
- **Zweites Beratungsgespräch:** 28. März 23, 16:00  
Angepasste Grobdispo/Prozessbeschreibung, Journal, offene Fragen bis 6.4.2023
- **Abgabe der Arbeit:** 20.4.2023 (vgl. Vorgaben)

**Organisatorisches**

Die Termine sind verbindlich. Bei Fragen oder Problemen gilt das Prinzip der Holschuld. Dokumente sind über GitHub abzulegen.

\*Der Arbeitsprozess ist hauptsächlich über GitHub ersichtlich zu machen

**Bestätigung / Unterschriften**

Winterthur, 18. Januar 2023



## 9.2. Arbeitsjournal

### 9.2.1. Arbeitsjournal Rafael Gahler

Datum der Lerneinheit und aufgewendete Zeit:	18.01.23, 0.5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Informationen über den Ablauf der Arbeit und das Thema besprochen mit den Aufsichtspersonen.
Erreichte Kompetenzen und notwendiges Wissen:	Die Arbeit ermöglicht und allfällige Themen zur Arbeit geklärt und beantwortet. Termine für die Gespräche festgelegt.
Gemachte Erfahrungen:	Erfahrung in der Arbeits Essenz erlangt und so den Start der Arbeit ermöglicht.
Standortbestimmung und Selbsteinschätzung:	Am Gespräch beteiligt und alles Nötige abgeklärt.
Planung, nächste Schritte:	Das 1. Gespräch liegt in Sichtweite. Vorbereitungen darauf treffen.

Datum der Lerneinheit und aufgewendete Zeit:	23.01.23, 0.5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Geforderte Anforderungen von Herr Marti durchgelesen und begonnen mit denen. Technologie festgelegt und mit Teamkollegen besprochen.
Erreichte Kompetenzen und notwendiges Wissen:	Festgelegt welche Technologie wir verwenden und wie wir diese einsetzen.
Gemachte Erfahrungen:	-
Standortbestimmung und Selbsteinschätzung:	Gut und in Kommunikation mit dem Team gearbeitet.
Planung, nächste Schritte:	Eine Anforderungsliste für den Notenrechner erstellen.



Datum der Lerneinheit und aufgewendete Zeit:	12.02.23, 1.5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Anforderungen für unseren Notenrechner erarbeiten und festhalten. Dem Team übermitteln und mit ihnen überarbeiten.
Erreichte Kompetenzen und notwendiges Wissen:	Die genauen Anforderungen für den Notenrechner ermittelt und die Vorlage des Excelrechners genau analysiert.
Gemachte Erfahrungen:	Anforderungen in Erfahrung gebracht.
Standortbestimmung und Selbsteinschätzung:	Gut und in Kommunikation mit dem Team gearbeitet.
Planung, nächste Schritte:	1. Gespräch mit Herr Marti durchführen und abklären ob Teamkollegen noch Hilfe Benötigen um fürs Gespräch bereit zu sein.

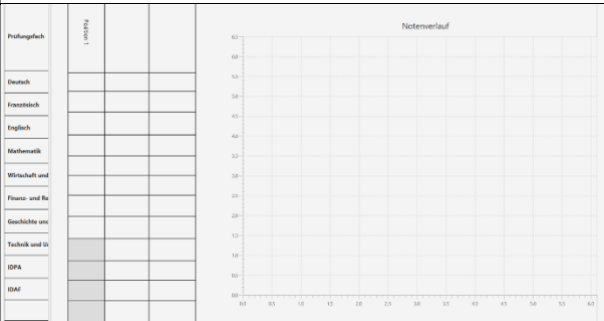
Datum der Lerneinheit und aufgewendete Zeit:	01.03.23, 1h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Erarbeitete Dinge mir Begleitperson besprechen und erweitern.
Erreichte Kompetenzen und notwendiges Wissen:	Die genauen Anforderungen für den Notenrechner herausarbeiten und erweitern.
Gemachte Erfahrungen:	Jetzt ist es uns möglich mit dem Projekt zu beginnen.
Standortbestimmung und Selbsteinschätzung:	Gut und in Kommunikation mit dem Team gearbeitet.
Planung, nächste Schritte:	Mit dem Frontend des Notenrechners beginnen.



Datum der Lerneinheit und aufgewendete Zeit:	06.03.23, 2h																																																																																	
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	An Frontend und Projekterstellung gearbeitet.																																																																																	
Erreichte Kompetenzen und notwendiges Wissen:	Javafx verwendet und das Frontend erarbeitet.																																																																																	
Gemachte Erfahrungen:	Den Umgang mit Javafx wiedererlernt und mich mit der Arbeit vertraut gemacht.																																																																																	
Standortbestimmung und Selbsteinschätzung:	Effizient und energisch gearbeitet.																																																																																	
Planung, nächste Schritte:	Weiter am Frontend arbeiten und dieses genau ausarbeiten.																																																																																	
Bild zum Fortschritt:	<div> <table border="1"> <caption>IMS Notenrechner 2023</caption> <thead> <tr> <th>Prüfungsfach</th> <th>Deutsch</th> <th>Englisch</th> <th>Mathematik</th> <th>Wirtschaft und Recht</th> <th>Finanz- und Rechnungswesen</th> <th>Geschichte und Politik</th> <th>Natur und Umwelt</th> <th>IDPA</th> </tr> </thead> <tbody> <tr> <td>Deutsch</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Englisch</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Mathematik</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Wirtschaft und Recht</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Finanz- und Rechnungswesen</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Geschichte und Politik</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Natur und Umwelt</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>IDPA</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table> </div>	Prüfungsfach	Deutsch	Englisch	Mathematik	Wirtschaft und Recht	Finanz- und Rechnungswesen	Geschichte und Politik	Natur und Umwelt	IDPA	Deutsch									Englisch									Mathematik									Wirtschaft und Recht									Finanz- und Rechnungswesen									Geschichte und Politik									Natur und Umwelt									IDPA								
Prüfungsfach	Deutsch	Englisch	Mathematik	Wirtschaft und Recht	Finanz- und Rechnungswesen	Geschichte und Politik	Natur und Umwelt	IDPA																																																																										
Deutsch																																																																																		
Englisch																																																																																		
Mathematik																																																																																		
Wirtschaft und Recht																																																																																		
Finanz- und Rechnungswesen																																																																																		
Geschichte und Politik																																																																																		
Natur und Umwelt																																																																																		
IDPA																																																																																		





Datum der Lerneinheit und aufgewendete Zeit:	13.03.23, 3h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	An Frontend und Projekterstellung gearbeitet. (ScrollPane Notenergebnis und Diagramm eingefügt und daran gearbeitet)
Erreichte Kompetenzen und notwendiges Wissen:	Anwendung von Javafx vertiefen und damit effizient arbeiten.
Gemachte Erfahrungen:	Erfahrungen in Javafx erarbeitet.
Standortbestimmung und Selbsteinschätzung:	Effizient und energisch gearbeitet.
Planung, nächste Schritte:	Frontend überarbeiten und ID's verteilen.
Bild zum Fortschritt:	

Datum der Lerneinheit und aufgewendete Zeit:	20.03.23, 0.5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Alle Id's zu den bisherig eingefügten Feldern hinzufügen
Erreichte Kompetenzen und notwendiges Wissen:	Id's zuweisen und in javafx navigieren.
Gemachte Erfahrungen:	Arbeit mit id's und dem Frontend in der Funktionalität
Standortbestimmung und Selbsteinschätzung:	Effizient gearbeitet, Probleme selbst behoben.
Planung, nächste Schritte:	Mit dem Speicher feature, dem Diagramm Feature oder dem row add Feature beginnen.



Datum der Lerneinheit und aufgewendete Zeit:	25.03.23, 4h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Mit row add Feature beginnen und in kleinem Projekt ausprobieren.
Erreichte Kompetenzen und notwendiges Wissen:	Erweitern eines GridPanes mit javafx und diese auch wieder Entfernen.
Gemachte Erfahrungen:	Dieses Feature nicht umsetzbar, da ich es nicht geschafft habe die Eingefügten rows wieder zu Entfernen.
Standortbestimmung und Selbsteinschätzung:	Ziel nicht erreicht aber an den Fehlern gelernt.
Planung, nächste Schritte:	Das Speicher Feature starten und das Diagramm Feature beginnen.

Datum der Lerneinheit und aufgewendete Zeit:	26.03.23, 6h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Mit speicher Feature beginnen und in kleinem Projekt ausprobieren.
Erreichte Kompetenzen und notwendiges Wissen:	Speichern des Inhaltes eines GridPanes in eine JSON Datei um sie beim nächsten Aufruf wieder in die Tabelle zu schreiben und so die Werte zu speichern.
Gemachte Erfahrungen:	Speicher feature erfolgreich in kleinem Projekt umgesetzt nach langen Schwierigkeiten mit dem Format und der Library.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und gutes Ergebnis bekommen.
Planung, nächste Schritte:	Das Speicher Feature implementieren und Diagramm feature einbauen.



Datum der Lerneinheit und aufgewendete Zeit:	27.03.23, 2h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Versuchen das Speicher feature ins Projekt einzubinden.
Erreichte Kompetenzen und notwendiges Wissen:	Speichern eines grossen mit TextFields gefüllten GridPanes mithilfe von JSON.
Gemachte Erfahrungen:	Ich habe es nicht ganz geschafft, da konstant neue Fehlermeldungen auftauchten. Aber wertvolle Erfahrung gesammelt.
Standortbestimmung und Selbsteinschätzung:	Ziel nicht erreicht.
Planung, nächste Schritte:	Das Speicher Feature implementieren und Diagramm feature einbauen.

Datum der Lerneinheit und aufgewendete Zeit:	28.03.23, 0.5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	1. Gespräch mit Herr Mock
Erreichte Kompetenzen und notwendiges Wissen:	Besprechung des gemachten und des weiteren Vorgehens.
Gemachte Erfahrungen:	Kommunikationsfähigkeiten gestärkt.
Standortbestimmung und Selbsteinschätzung:	Alle Fragen geklärt und auf gutem Kurs.
Planung, nächste Schritte:	Das Speicher Feature implementieren und Diagramm feature einbauen.



Datum der Lerneinheit und aufgewendete Zeit:	28.03.23, 2h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Versuchen das Speicher feature ins Projekt einzubinden.
Erreichte Kompetenzen und notwendiges Wissen:	Speichern eines grossen mit TextFields gefüllten GridPanes mithilfe von JSON.
Gemachte Erfahrungen:	Ich habe einen neuen Ansatz gewählt, mit dem es viel einfacher funktioniert hatte.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Das Speicher Feature optimieren und Diagramm feature einbauen.

Datum der Lerneinheit und aufgewendete Zeit:	30.03.23, 5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Das Diagramm feature ins Projekt einbauen
Erreichte Kompetenzen und notwendiges Wissen:	Werte aus dem Diagramm in ein Diagramm eingeben und da aufzeigen.
Gemachte Erfahrungen:	Ich habe gelernt wie ich mit javaFX Diagrammen arbeiten kann.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Das Speicher Feature optimieren und Fehlermeldung erstellen.



Datum der Lerneinheit und aufgewendete Zeit:	01.04.23, 2h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Optimieren des Codes der Speicherfunktion.
Erreichte Kompetenzen und notwendiges Wissen:	Umgang mit dem feature und Arrays und for-loops.
Gemachte Erfahrungen:	Ich habe meine Erfahrungen im Bereich der Arrays massiv weiterentwickelt.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingebaut.
Planung, nächste Schritte:	Fehlermeldung einbauen und möglicherweise Diagramm faecture code optimieren.

Datum der Lerneinheit und aufgewendete Zeit:	02.04.23, 5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Die Speicherfunktion erschaffen.
Erreichte Kompetenzen und notwendiges Wissen:	Öffnen einer neuen Scene und fehler melden.
Gemachte Erfahrungen:	Mit Scene sehr umständlich.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Fehlermeldung umbauen und möglicherweise Diagramm faecture code optimieren.



Datum der Lerneinheit und aufgewendete Zeit:	03.04.23, 2h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Die Speicherfunktion umbauen.
Erreichte Kompetenzen und notwendiges Wissen:	Arbeit mit Alert feature von JavaFX.
Gemachte Erfahrungen:	Deutlicher besserer Ansatz als zuvor.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Diagramm faecture code optimieren.

Datum der Lerneinheit und aufgewendete Zeit:	08.04.23, 1h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Diagramm code optimieren.
Erreichte Kompetenzen und notwendiges Wissen:	Cleaneren Code erschaffen.
Gemachte Erfahrungen:	Geht schon besser als beim Speichern.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Prognose feature einbauen, Felder leeren Feature einbauen und Arbeitsjournal zusammenfügen.



Datum der Lerneinheit und aufgewendete Zeit:	09.04.23, 1h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Felder leeren feature einbauen.
Erreichte Kompetenzen und notwendiges Wissen:	Die Felder in einem Textfeld alle auf Knopfdruck leeren.
Gemachte Erfahrungen:	Ohne Probleme verlaufen.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Prognose feature einbauen und Arbeitsjournal zusammenfügen.

Datum der Lerneinheit und aufgewendete Zeit:	09.04.23, 5h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Prognose feature einbauen.
Erreichte Kompetenzen und notwendiges Wissen:	Eine Prognose zur Mindestnote der Abschlussprüfungen erstellen. Und im Falle von leeren Feldern diese mit 4 ern füllen.
Gemachte Erfahrungen:	Ohne Probleme verlaufen.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und eingefügt.
Planung, nächste Schritte:	Alles überarbeiten und Arbeitsjournal zusammenfügen.

Datum der Lerneinheit und aufgewendete Zeit:	10.04.23, 8h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Alles am Code überarbeiten, wo möglich zu Cleancode ändern, inkludierung des neu gemachten codes im Projekt, Versuch Pdf exporter einzufügen.
Erreichte Kompetenzen und notwendiges Wissen:	Den ganzen code, seine Funktionalität behaltend, überarbeiten und den Aufwand optimieren.
Gemachte Erfahrungen:	Am pdf Export bin ich leider gescheitert, da ich nur Wege mithilfe neuer Librarys gefunden hätte, welche ich nicht verwenden wollte.
Standortbestimmung und Selbsteinschätzung	Bis auf den Pdf Export ist die Überarbeitung reibungslos verlaufen.



Datum der Lerneinheit und aufgewendete Zeit:	15.04.23, 2h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Arbeitsjournal zusammenfügen und überarbeiten.
Erreichte Kompetenzen und notwendiges Wissen:	Das Arbeitsjournal so weit gut erschaffen.
Gemachte Erfahrungen:	Ohne Probleme verlaufen.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und gemacht.
Planung, nächste Schritte:	Projekt in eine exe Datei kopieren und Dokumentation fertigstellen.

Datum der Lerneinheit und aufgewendete Zeit:	17/18.04.23, 10h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Verfeinerungen am Code vornehmen und geschriebene Funktionen für die Doku beschreiben.
Erreichte Kompetenzen und notwendiges Wissen:	Umgang mit JavaFx und Umgang mit word und Strukturen.
Gemachte Erfahrungen:	Mit gewissen Komplikationen und Zeitdruck nur beschränkt effizient aber schlussendlich erfolgreich.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und gemacht.
Planung, nächste Schritte:	Doku Drucken und Projekt zum exe machen.





Datum der Lerneinheit und aufgewendete Zeit:	19.04.23, 6h
Behandelte Fachthemen: (Kompetenz, Hinweis Portfolio)	Konvertieren eines JavaFX Projekts in .jar Datei. Danach diese in einem .exe aufrufen und für die benutzende Person zugänglich machen. + Einige Bug fixes im Programm.
Erreichte Kompetenzen und notwendiges Wissen:	Umgang mit JavaFx und Umgang mit Python und Java Versionen. Analyse von Fehlermeldungen.
Gemachte Erfahrungen:	Das Verwenden einer veralteten java Version führt zu massiven Problemen in der Umsetzung des .jar Files. Man sollte immer die neuste benutzen.
Standortbestimmung und Selbsteinschätzung:	Ziel erreicht und gemacht.
Planung, nächste Schritte:	Doku Drucken und fertigstellen. Abgabe.

Zeitaufwand Rafael Gahler: Etwa 71 Stunden

#### 9.2.2. Arbeitsjournal Nico Zollinger

18.01.2023	0.5h
Ausgeführte Arbeiten	Erstgespräch mit Aufsichtspersonen
Erreichte Ziele	Thema besprochen sowie Daten für Zwischengespräche festgelegt
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	Keine Hilfeleistung beansprucht, lediglich im Team mit den Aufsichtspersonen alles abgeklärt.
Vergleich mit dem Zeitplan	Zeitlich gut im Plan, Arbeit am Projekt kann jetzt beginnen.
Reflexion	Gespräch hat gut funktioniert, alle Fragen wurden geklärt und passende Daten für die Zwischengespräche konnten gefunden werden.



21.01.2023	0.5h
Ausgeführte Arbeiten	Gespräch im Team, erste Arbeitsschritte auf Gruppenmitglieder aufgeteilt
Erreichte Ziele	Arbeitsschritte aufgeteilt
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	Abklären, welche Aufgaben etwa wie viel Zeit in Anspruch nehmen werden.
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Nach Gespräch gut im Zeitplan, Daten für die Vervollständigung dieser Arbeiten wurden entsprechend gesetzt
Reflexion	Das Gespräch war hilfreich, so wissen wir nun genau, wer was machen muss und bis wann dies fertig sein sollte.

23.01.2023	0.5h
Ausgeführte Arbeiten	Im Team Mit Anforderungen an den Notenrechner auseinandergesetzt und begonnen die Umsetzung davon zu planen.
Erreichte Ziele	Wichtiger Schritt in der Planung des Projekts abgeschlossen
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	Bei gewissen Anforderungen mussten die Details mit den Verantwortlichen besprochen werden bzw. Fragen geklärt werden.
Vergleich mit dem Zeitplan	Gut im Zeitplan, mit dem Wissen über die Anforderungen können wir mit der Umsetzung des Rechners beginnen.
Reflexion	Die Arbeit war keine grosse Sache und wir haben uns schnell ein Bild über die Anforderungen machen können.



27.02.2023	1h
Ausgeführte Arbeiten	In Zusammenarbeit mit Rafael Gahler eine Liste von Meilensteinen unserer Arbeit erstellt.
Erreichte Ziele	Grober Zeitplan mit Meilensteinen erstellt, wichtiger Schritt in der Planung des Projektes erledigt.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Immernoch gut im Zeitplan, das nächste Gespräch mit den Aufsichtspersonen steht bald an, dazu haben wir alle nötigen Schritte schon gemacht und sind vorbereitet.
Reflexion	Arbeit konnte effizient erledigt werden und dem nächsten Zwischengespräch steht nichts mehr im Weg.

26.03.2023	3.5h
Ausgeführte Arbeiten	Dokumentation bis zum jetzigen Stand des Projekts nachgeführt.
Erreichte Ziele	Grundgerüst der Dokumentation steht, sowie erste Arbeiten in der Dokumentation festgehalten.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	Absprache mit Teamkollegen bezüglich Einzelheiten der Dokumentation
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Gut im Zeitplan, mit dem Erstellen der Dokumentation können ab jetzt neue Elemente einfach eingefügt werden
Reflexion	Die Dokumentation zu erstellen war nicht schwer und hat keine Probleme hervorgerufen. Für einige Einzelheiten musste ich mich mit Gruppenmitgliedern absprechen, jedoch konnte ich alles ausfüllen.



28.03.2023	0.5h
Ausgeführte Arbeiten	Zweites Zwischengespräch mit den Aufsichtspersonen
Erreichte Ziele	Zwischengespräch mit Herrn Mock, jetzigen Stand und nächste Schritte besprochen, sowie Fragen geklärt
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Gut im Zeitplan, Fragen wurden geklärt, somit können wir jetzt an unserem Projekt weiterarbeiten
Reflexion	Gespräch war sehr konstruktiv und hilfreich für das weitere Vorgehen.

14.04.2023	1h
Ausgeführte Arbeiten	Erstellen einer Prognosefunktion
Erreichte Ziele	Die Prognose-Funktion funktioniert noch nicht ganz, jedoch ist schonmal ein gutes Grundgerüst erstellt.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Gut im Zeitplan, die Prognosefunktion muss nur noch fertiggestellt werden, dann sind alle Funktionen einsatzbereit.
Reflexion	Arbeit an der Prognosefunktion ist gut gegangen, einige kleinere Probleme haben mich davon abgehalten, die Funktion fertigzustellen, jedoch werde ich das noch beheben.



14.04.2023	1h
Ausgeführte Arbeiten	Fertigstellen der Prognose-Funktion
Erreichte Ziele	Die Prognose-Funktion konnte fertiggestellt werden und funktioniert jetzt.
Aufgetretene Probleme	-
Durchgeführte Tests	Testen, ob die Prognose-Funktion mit allen gültigen Eingaben das richtige Resultat ausgibt → Ja
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	Funktion in Zusammenarbeit mit Rafael Gahler fertiggestellt, Notendurschnitt wird aus der von Samira Stragiotti erstellten Funktion ausgelesen.
Vergleich mit dem Zeitplan	Mit der Prognosefunktion haben wir alle Funktionen nur erstellt und implementiert, somit steht der Rechner schon einmal.
Reflexion	Zusammenarbeit mit Rafael Gahler hat gut funktioniert, Arbeit konnte effizient erledigt werden.

17.04.2023	2h
Ausgeführte Arbeiten	Arbeit an der Dokumentation
Erreichte Ziele	Weitere Elemente der Dokumentation wurden vervollständigt, finales Produkt kommt immer näher.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	Abklärung einzelner Fragen mit Teamkollegen.
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Gut im Zeitplan, Dokumentation und Arbeitsjournal muss noch fertig gemacht werden, ansonsten fertig
Reflexion	Gutes Vorankommen an der Dokumentation, wenn dieses Arbeitstempo beibehalten werden kann, müsste die Doku bald fertig sein.



18.04.2023	2.5h
Ausgeführte Arbeiten	Arbeit an der Dokumentation
Erreichte Ziele	Weitergekommen an der Dokumentation
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	Absprache mit Teamkollegen bezüglich der Doku
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Etwas in Verzug, einige Elemente beanspruchen einen höheren Zeitaufwand als gedacht
Reflexion	Arbeit ging gut voran, keine Probleme aufgetaucht.

19.04.2023	7h
Ausgeführte Arbeiten	Arbeit an der Dokumentation, sowie Vervollständigung des Arbeitsjournals.
Erreichte Ziele	Dokumentation und Arbeitsjournal vervollständigt und bereit zur Abgabe.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Wissensbeschaffung	Viel Arbeiten unter Absprache mit Teamkollegen, bezüglich Fragen zu Elementen der Dokumentation.
Beanspruchte Hilfeleistung	-
Vergleich mit dem Zeitplan	Dokumentation und Arbeitsjournal rechtzeitig fertig geworden, somit steht der Abgabe nichts mehr im Weg.
Reflexion	Arbeit war anstrengend, vor allem das Arbeitsjournal, da das nachträgliche Vervollständigen viel Zeitaufwand benötigt hat. Trotzdem bin ich fertig geworden und das Projekt ist somit fertig.

Zeitaufwand Nico Zollinger: Etwa 20 Stunden



### 9.2.3. Arbeitsjournal Samira Stragiotti

20.02.2023	Arbeitszeit (gerundet auf h): 1
Ausgeführte Arbeiten	Vorbereitung erstes Gespräch: UseCase Diagramm erstellen und allfällige Fragen klären.
Erreichte Ziele	Start mit technischer Dokumentation und Arbeitsjournal.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Reflexion	Mit dem Projekt gestartet nun folgt die effektive Umsetzung und darauffolgende technische Dokumentation.

13.03.2023	Arbeitszeit (gerundet auf h): 2
Ausgeführte Arbeiten	Mit technischer Dokumentation und Arbeitsjournal angefangen (Layout und Vorlage).
Erreichte Ziele	Start mit technischer Dokumentation und Arbeitsjournal.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Reflexion	Nach Gespräch mit Herr Marti wussten wir, dass unser Projekt auf Java basieren soll und als selbstständiges Programm funktionieren soll. Nun können wir das Projekt aufsetzen und unser GitHub Repository einrichten.

20.03.2023	Arbeitszeit (gerundet auf h): 4
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- GitHub aufgesetzt</li><li>- README.md erstellt</li><li>- Maven Projekt erstellt</li><li>- Grundgerüst des Projekts erstellt</li><li>- .fxml File von Rafael eingebunden</li><li>- Eine To-Do-Liste auf GitHub erstellt</li></ul>
Erreichte Ziele	Grundgerüst des Projekts erstellt.
Aufgetretene Probleme	Da Rafael mit einer anderen JDK Version gearbeitet musste ich das bei mir im .fxml File anpassen.
Durchgeführte Tests	-
Reflexion	Projekt ist aufgesetzt und es können nun die benötigten Klassen erstellt werden.



23.03.2023	Arbeitszeit (gerundet auf h): 2
Ausgeführte Arbeiten	Import und Export von .csv Files testen.
Erreichte Ziele	Grundgerüst des Projekts erweitern.
Aufgetretene Probleme	Leere Felder konnten nicht eingelesen werden.
Durchgeführte Tests	.csv Dateien einlesen sowie ausgeben.
Reflexion	Der Import sowie Export von .csv Files funktioniert grundsätzlich und bereit, um in den Controller eingebunden zu werden.

27.03.2023	Arbeitszeit (gerundet auf h): 4
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Klassen; Fach und Faechermanager*, erstellen</li><li>- Dazugehörige Methoden definieren</li></ul> <p>*Die Klasse Faechermanager habe ich später entfernt, da sie sich im Verlauf des Projekts als überflüssig herausstellte.</p>
Erreichte Ziele	Grundgerüst des Projekts erweitern.
Aufgetretene Probleme	-
Durchgeführte Tests	-
Reflexion	Die Klassen sind soweit definiert und es kann als nächstes mit dem Controller weitergefahren werden.

03.04.2023	Arbeitszeit (gerundet auf h): 2
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Controller Methoden von Rafael eingebunden<ul style="list-style-type: none"><li>o initial()</li><li>o save()</li><li>o delete()</li><li>o berechnen()</li></ul></li><li>- berechnen() Methode ergänzt</li></ul>
Erreichte Ziele	Grundgerüst des Projekts erstellt.
Aufgetretene Probleme	Da Rafael mit einer anderen JDK Version gearbeitet hat musste ich das bei mir im .fxml File anpassen.
Durchgeführte Tests	-
Reflexion	Das Projekt läuft nun mit Frontend und Backend. Es müssen jetzt noch die benötigten Funktionen implementiert werden.





10.04.2023	Arbeitszeit (gerundet auf h): 4
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Das ganze Projekt scheint aus dem nichts nicht mehr zu funktionieren – neu aufgesetzt</li></ul>
Erreichte Ziele	Projekt nochmals neu aufsetzen.
Aufgetretene Probleme	Das Programm läuft nicht mehr.
Durchgeführte Tests	-
Reflexion	Leider konnte ich die Ursache nicht ausfindig machen und habe mich dazu entschieden das Projekt neu aufzusetzen, was im Endeffekt die schnellste Lösung war. GitHub muss noch angepasst werden.

13.04.2023	Arbeitszeit (gerundet auf h): 3
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- GitHub anpassen (neues Projekt)</li><li>- Ausbau von berechnen() Methode</li></ul>
Erreichte Ziele	GitHub neu eingerichtet; neuer Main Branch. Bei der berechnen() Methode die Fächer als Fächer Objekte gespeichert, sowie die .csv Datei angepasst (effizienter gemacht).
Aufgetretene Probleme	-
Durchgeführte Tests	-
Reflexion	Es können nun alle über GitHub weiter am Projekt feilen. Die

14.04.2023	Arbeitszeit (gerundet auf h): 4
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Fächer Objekte erstellen</li><li>- Anpassung .csv Datei</li></ul>
Erreichte Ziele	-
Aufgetretene Probleme	-
Durchgeführte Tests	-
Reflexion	Dank der Anpassung der .csv Datei kann nun auch die import und export Methode erstellt werden.



15.04.2023	Arbeitszeit (gerundet auf h): 8
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Berechnen() Korrekturen</li><li>- Import_csv() Grundfunktionalität fertiggestellt</li><li>- TextFields updaten</li><li>- Einbau Null Handling der Methoden</li><li>- Fachnote berechnen sowie deren Implementierung in den anderen Methoden</li></ul>
Erreichte Ziele	-
Aufgetretene Probleme	-
Durchgeführte Tests	-
Reflexion	Error- beziehungsweise das Null Handling der Methoden verbessert und das Programm somit verbessert. Die Fachnoten der einzelnen Fächer werden berechnet und so kann nun die Prognose sowie auch das Diagramm erstellt werden.

16.04.2023	Arbeitszeit (gerundet auf h): 8
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Fxml Anpassungen<ul style="list-style-type: none"><li>o IDPA und IDAF waren vertauscht</li><li>o TextFields (nur Output gebende Felder) zu Labels geändert</li></ul></li><li>- Berechnung sowie Ausgabe von Gesamtnote, Anzahl Tiefnoten und Tiefpunkte</li><li>- export_csv() erweitert</li><li>- Bug fixes</li></ul>
Erreichte Ziele	-
Aufgetretene Probleme	<ul style="list-style-type: none"><li>- .csv Export funktioniert noch nicht; Datei kann ausgewählt werden – wird jedoch nicht richtig gelesen</li></ul>
Durchgeführte Tests	-
Reflexion	Kleine Verbesserungen sowie auch die letzten Berechnungen wurden getätigt und letztlich liegt der Fokus noch vor allem auf der Export Methode der .csv Datei, Error Handling und clean code.



17.04.2023	Arbeitszeit (gerundet auf h): 8
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- prognose() von Rafael implementiert</li><li>- bei Berechnen der Fachnote auf 0.5 und bei Gesamtnote auf eine Kommastelle runden</li><li>- Error Handling</li><li>- File Selector eingebaut bei Import sowie Export von .csv Datei</li></ul>
Erreichte Ziele	-
Aufgetretene Probleme	-
Durchgeführte Tests	<ul style="list-style-type: none"><li>- Ob Datei bzw. Speicherpfad selektiert werden kann beim Import bzw. Export einer .csv Datei</li></ul>
Reflexion	Das Error Handling sollte nun gelöst sein, sowie das Runden der Gesamtnote und das Selektieren der .csv Datei beim Export und Import.

18.04.2023	Arbeitszeit (gerundet auf h): 6
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Clean Code</li><li>- Fehlerbehebung import_csv()</li><li>- JavaDoc Comments ausprobieren</li></ul>
Erreichte Ziele	<ul style="list-style-type: none"><li>- Code verschönert</li></ul>
Aufgetretene Probleme	<ul style="list-style-type: none"><li>- import_csv() verursacht immer noch Probleme</li></ul>
Durchgeführte Tests	<ul style="list-style-type: none"><li>- Mithilfe Test Dateien Daten importieren/einlesen</li></ul>
Reflexion	Alles funktioniert bis auf das Importieren einer .csv Datei, was mein letztes Ziel für dieses Projekt ist. Das Kommentieren bzw. Dokumentieren mittels JavaDoc empfand ich in unserem Fall bei diesem Projekt als nicht viel besser als das herkömmliche Kommentieren des Codes.

19.04.2023	Arbeitszeit (gerundet auf h): 3
Ausgeführte Arbeiten	<ul style="list-style-type: none"><li>- Erfolgreiche Fehlerbehebung import_csv()</li></ul>
Erreichte Ziele	<ul style="list-style-type: none"><li>- export_csv() fix</li><li>- letzte Verbesserungen</li><li>- Testing</li></ul>
Aufgetretene Probleme	-
Durchgeführte Tests	In der <a href="#">Spezifikation</a> erfasst.
Reflexion	Die letzten Verbesserungen wurden getätigt und das Programm muss jetzt nur noch als ausführbare Datei exportiert werden.



Zeitaufwand Samira Stragiotti: Etwa 59h

Wissensbeschaffung/ Beanspruchte Hilfeleistung	Ich stütze mich bezüglich der Wissensbeschaffung und Hilfeleistung hauptsächlich auf Internetrecherchen und vergangene Projekte (Schule und privat) sowie das Vorgaben IDPA-Dokument der KBW.
Vergleich mit dem Zeitplan	Über das ganze Projekt hinweg gesehen fand ich, dass ich gut in der Zeit war und lediglich am Schluss (v. a. letzte Woche) ein bisschen unter Druck kam.

### 9.3. Betreuungsgespräche

#### 9.3.1. Betreuungsgespräch 1

Betreuungsgespräch 1 (mit Bernhard Marti)	
Kennen wir die genauen Anforderungen? Haben wir die nötigen Informationen?	In diesem Gespräch haben wir uns vor allem über genau diese Anforderungen unterhalten und unser bisherig gemachtes besprochen. Wir haben Anforderungen an das Ergebnis geschaffen und debattiert. Wir haben alle nötigen Informationen, um mit der Arbeit anzufangen, sobald abgeklärt ist, ob wir mit Java arbeiten müssen.
Was haben wir bereits erledigt? Ergeben sich Abweichungen zur Planung oder zum Thema?	Wir haben bereits die von Herr Marti gestellten Anforderungen zur Vorbereitung auf das Gespräch erledigt. Darin haben sich keine Abweichungen zu unserem Thema ergeben, jedoch müssen wir diese nochmals überarbeiten und nochmals besprechen, um sicher zu gehen, dass wir die Grundbausteine des Projektes gut legen.



Was bleibt zu tun, wie gehen wir weiter vor?	Als nächstes werden wir uns in der Gruppe zusammensetzen und erörtern, wer was macht. Wenn dies erledigt ist, werden wir mit dem Arbeiten und der Dokumentation beginnen. Es sollten Prototypen und Aktivitätsdiagramme erstellt werden, um einen Plan zu haben, wie wir das ganze umsetzen wollen und welche genauen Ziele wir in der Programmierungsphase verfolgen müssen. Bis zum nächsten Gespräch sollten die Grundbausteine des Systems stehen und nur noch Verfeinerungen zu erledigen sein.
--	--



### 9.3.2. Betreuungsgespräch 2

Betreuungsgespräch 2 (mit Jaques Mock Schindler)	
Welche Konsequenzen haben allfällige Abweichungen von der Planung?	Da wir eigentlich keine Abweichung von der Planung haben können, diese auch keine Folgen mit sich tragen.
Entspricht die Arbeit den Anforderungen?	Die Arbeit entspricht soweit noch nicht den Anforderungen, da wir noch an allem etwas mehr machen müssen, jedoch stehen die Grundlegenden Bausteine und wir können ohne Komplikationen darauf aufbauen. Das Feature neue Fächer hinzuzufügen, streichen wir aus unseren Plänen, da es uns nach langem probieren nicht als möglich erscheint. Die restlichen Features werden jedoch implementiert und verfeinert.
Was lief bisher gut / schlecht? Welche Verbesserungen sind nötig / möglich?	Bisher lief die Zusammenarbeit in der Gruppe eher etwas holprig, die Kommunikation hat nicht gestimmt und bei Rafael Gahler hat GitHub nicht funktioniert. Gut lief die individuelle Arbeit aller Teammitglieder, jeder hat bereits signifikante Teile seines Aufgabenbereichs abgeschlossen und kann weiter darin abtauchen. Auch die Schaffung des Produkts verläuft bis jetzt trotz kleinen Missverständnissen nach Plan. Wir müssen unsere Zusammenarbeit im Team stark verbessern und mehr kommunizieren.

### 9.4. QR-Code zu GitHub Repository (inkl. Programm ZIP Datei)

