üK-335 Mobile Applikation realisieren

Dokumentation-Pendenz-Verwaltung

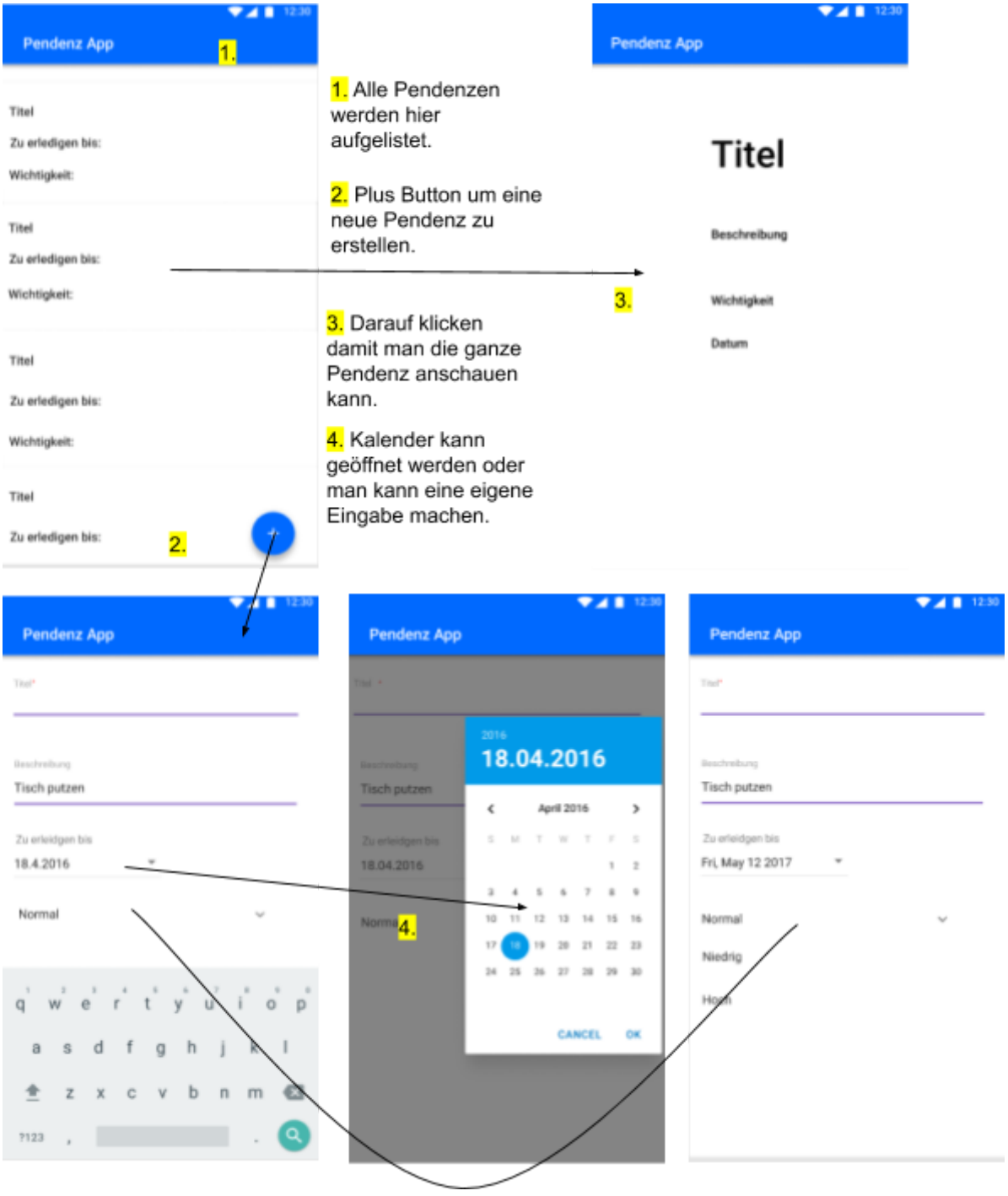
Version 1.0.0, 04.06.2020 | Alban Selimi und Samuel Sättler

Inhalt

1. Projektbeschreibung

Diese Applikation ist ein Teil der NOA Software der Noser Young und wurde erstellt, damit man die NOA Software auch per Mobile nutzen kann. NOA ermöglicht die Verwaltung von Personendaten von Lernenden und Mitarbeitern, die Verwaltung von Beobachtungen, Bildungsberichten, Arbeitsjournale, Schnupperlehren, Bewerbungen, oder Pendenzen. Pendenzen sind in dem Fall Tätigkeiten, welche eine bestimmte Person bis zu einem bestimmten Datum erledigen muss, beispielsweise ein Ämtli. Das Ziel unseres Projektauftrags ist es, eine Android Applikation zu erstellen, bei der ein Lernender seine Pendenzen selber erstellen und in einer Liste seine Pendenzen anzeigen kann. Jede Pendenz muss einen Titel, dazu eine Beschreibung, ein Abgabedatum und die Dringlichkeit des Auftrages haben. Die Pendenzen werden in eine Datenbank reingeschrieben, und beim Auflisten der Pendenzen in einer Liste herausgenommen und dann strukturiert aufgezeichnet. Bei der Liste der Pendenzen sieht man nur den Titel, das Abgabedatum und die Dringlichkeit. Wenn man dann auf eine Pendenz klickt, wird diese hervorgehoben und die Beschreibung wird zusätzlich eingeblendet.

2. Mockups



1. **MainActivity**
   In der MainActivity wird eine Liste aller Pendenzen von der Datenbank geholt und angezeigt. Jede Zeile hat einen Titel, eine Abgabedatum und die Dringlichkeit. Drückt man auf einen der Datensätze, wird dieser hervorgehoben und man hat eine grössere Ansicht der Daten der gewählten Pendenz kommen, dazu kommt noch die

Beschreibung der Pendenz. Ganz unten Rechts der Activity ist ein Plus Knopf, welcher auf eine nächste Activity, nämlich zur Erstellung einer neuen Pendenz führt.

2. **PendenceActivity**
   In der PendenceActivity kann der Benutzer eine neue Pendenz erstellen. Das Titel Feld und das Dringlichkeit Feld muss ausgefüllt werden, die Beschreibung und das Datum hingegen sind nur Optionale Felder. Durch das Drücken des “Pendenz Speichern” Buttons , werden die eingegebenen Daten validiert und bei korrekten Angaben wird die Pendenz erstellt und in die Datenbank reingeschrieben, danach wird die Activity geschlossen und man landet wieder auf der MainActivity. Man hat auch einen Zurück Knopf um direkt zur MainActivity kommt falls man keine Pendenz erstellen möchte, also zur Liste der Pendenzen.

3. **DetailedActivity**
   Diese Activity wird aufgerufen, sobald man eine der aufgelisteten Pendenzen klickt. Danach wird diese Pendenz geöffnet und man hat sie sie ist so gross wie der Screen selber. Zusätzlich sieht man noch die Beschreibung der Pendenz. Man hat auch einen schliess Button und kommt so auch wieder zur MainActivity Seite, wo alles aufgelistet ist.

3. Technische Realisierung

MainActivity
MainActivity ist die Launcher Activity von unserem Projekt. Das wird in der Manifest File festgehalten. In der MainActivity gibt es die Methode initialiseTheItemsList. Dort wird die Liste von Pendenzen mit Daten aus der DB gefüllt. In der OnCreate Method, die jedesmal aufgerufen wird, wenn die Activity gestartet wird, wird zuerst dem Applikation mitgeteilt, dass die activity_main, die View von der MainActivity ist. Danach wird dort die Methode createExampleList() aufgerufen. Die erstellt einen PendeceDao mit der wir eine Zeile später alle Pendenzen aus der DB holen. Danach initialisieren wir den RecylerView in der buildRecyclerView() Methode. Dort initalisieren wir unseren recyclerView Feld und gleichzeitig verbinden wir den RecyclerView aus dem Fragment mit unseren. Danach setzten wir unseren Pendenz Adapter als Adapter in RecycleView. Am Schluss hatten wir definiert was passieren würde, wenn jemand den Insertbutton drückt. Dann wird die openPendenceActivity() Methode ausgerufen. Die erstellt einen expliziten Intent der nach PendenceActivity geht. Ein Aktivität wird gestartet und der Intent wird mitgegeben. Am Schluss wird diese Activity durch die onStop() Methode zerstört. Ich habe das gemacht, weil ich wollte, dass wenn man eine neue Pendenz erstellt hat und man zurückgekehrt wird in die MainActivity, wollte ich, dass man den neuen Pendenz noch sieht und das wäre ohne die onStop() Methode nicht möglich.

PendenceActivity
Man kommt in der PendenceActivity nur wenn man den insertButton in der MainActivity gedrückt hat. Zuerst wird in die onCreate() Methode, die activity_pendece.xml als unsere View gesetzt. Danach werden durch die Methode initialiseTheViews(), unsere UI Komponente, die wir oben deklariert haben, mit dem UI Komponenten von der View verbunden. In diesen Vorgang werden sie gleichzeitig auch initialisiert. Danach wird der OnClickListener von unserer Save-Button initialisiert. Mehr dazu später. Danach wird geschaut, dass der Datum TextView namens eText initialisiert wird und gleichzeitig mit dem createDate TextView aus dem View verbunden wird. Dann wird der OnClickListener von diesen Button auch erstellt. In der OnClick Methode von eText wird ein Kalender instanziiert und der Datum, den wir ausgewählt haben wird in dd.mm.yyyy Format eingegeben. In der OnClick() Methode von unserem Save-Button werden zuerst alle Werte, die in den UI Feldern eingegeben waren, in Strings gespeichert. Danach wird bei der Beschreibung und dem Titel geschaut, dass sie richtig validiert sind. Der Titel darf nicht leer sein und er darf nicht über 50 Zeichen haben. Die Beschreibung darf nicht über 500 Zeichen haben. Danach schaut man, ob man die Datum richtig eingegeben wurde. Die Ergebnisse von diesen Tests werden in Booleans gespeichert. Falls diese Daten nicht richtig validiert wurden, dann gibt es einen false zurück. Danach schaut man, ob diese Booleans true sind. Falls sie es sind, dann wird eine Pendenz Instanz erstellt. Sie bekommt nachher, die Werte vom Formular und wird anschliessend in der DB gespeichert. Am Schluss wird man in die MainActivity Klasse gebracht.

DetailedActivity
In diesen Activity wird in der onCreate() Methode als allererstes die activity_detailled als unsere View gesetzt. Danach werden unsere TextView Felder initialisiert, indem sie mit dem den UI Komponenten der View verbunden werden. Danach wird die getData() Methode aufgerufen, die alle Daten aus dem Intent holt. Man kommt in der DetailedActivity nur wenn man in der MainActivity einen CardView antippt. Beim Antippen einer CardView wird ein Intent erstellt, das alle Daten aus der CardView speichert. Dieser Intent geht nachher in die DetailedActivity. Die Daten werden in Strings gespeichert. Nachdem die Strings initialisiert werden, dann wird die setData() Methode aufgerufen. Sie setzt als Value in den TextView Felder die entsprechenden Strings.

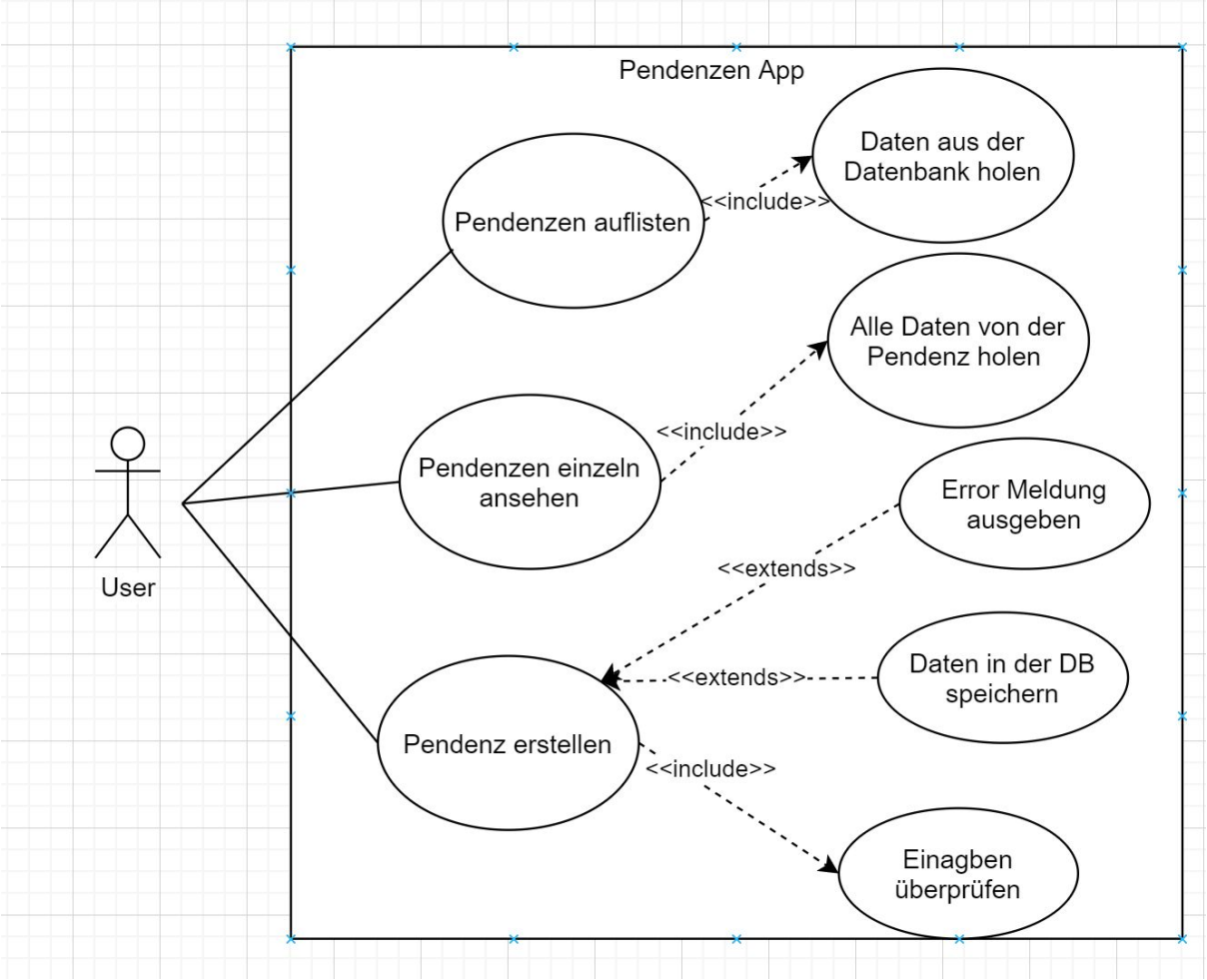
Pendence
Diese Klasse wird als Entity in unseren Room DB gebraucht. Deswegen hat es auch für jeden Feld eine Annotation. Der dateToFinish Feld braucht ausserdem einen DateConverter, deswegen hat dieses Feld noch eine zusätzliche TypeConverters Annotation. Diese Klasse besteht aus zwei Konstruktoren. Einem der fast alle Felder, ausser dem id Feld, als Parameter braucht und aus einem Default Konstruktor, der keine Parameter braucht. Ausserdem besitzt diese Klasse für jeden Feld einen Feld einen Getter und Setter.

PendenceAdapter
PendenceAdapter ist dafür zuständig, die Daten aus der Datenbank erfolgreich in die UI zu präsentieren. Der Adapter wird nur in der MainActivity initialisiert. Danach wird er als Adapter vom RecyclerView gesetzt. Es ist sehr wichtig, dass der Adapter von der Klasse RecyclerView.Adapter erbt. Dadurch können wir wichtige Methoden für diese Klasse

initialisieren. Diese drei Methoden heißen onCreateViewHolder(), onBindViewHolder() und getItemCount(). Die onCreateViewHolder() Methode gibt an, welcher Fragment als ViewHolder gebraucht wird. Diese Method gibt einen ViewHolder zurück. Dieser ViewHolder erbt von der RecyclerView.ViewHolder Klasse. Danach werden in dem ViewHolder Konstruktor die TextView Felder initialisiert, indem man sie mit der TextViewFeldern aus der UI verbunden hat. Nachher wird die onBindViewHolder() aufgerufen. Hier bekommen die TextView Felder aus der UI, die Daten aus der Datenbank. Eine List aus Pendenzen wird dem Adapter als Parameter eingegeben. Aus dieser Liste werden die Daten geholt und in Strings gespeichert. Nachher bekommen, die TextViews die Daten aus den Strings. Später wird ein OnClickListener für jeden CardView erstellt. Falls ein CardView berührt wird, dann erstellt der Adapter einen Intent und holt alle Daten aus dieser CardView und gibt Sie an Intent weiter. Der Intent schickt den User nachher in die DetailedActivity

4. Diagramme

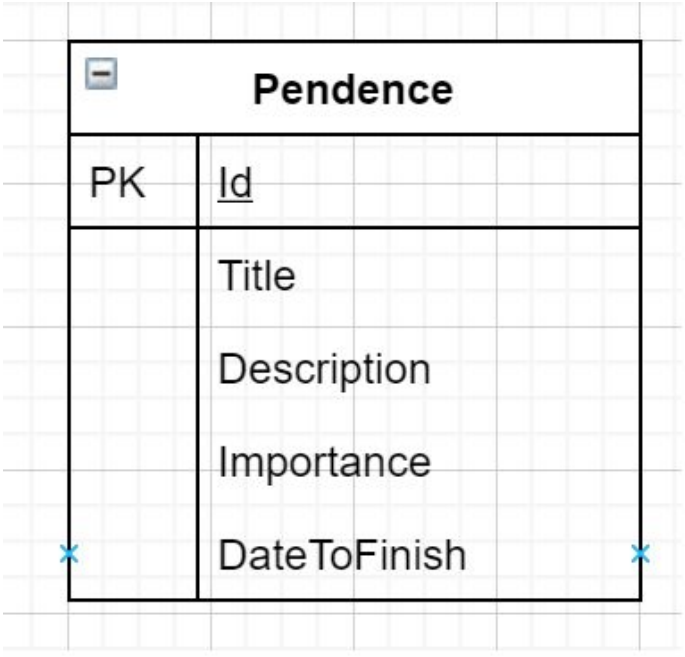
4.1 Use Case Diagramm



Ein User sollte drei Funktionen mit der Applikation ausführen können. Wenn man Pendenzen auflisten will, dann included die “Pendenzen auflisten” Use Case die “Daten aus

der Datenbank holen” Use Case. Wenn der User eine Pendenz einzeln ansehen will, dann included der “Pendenz einzeln ansehen” Use Case vom “Alle Daten von der Pendenz holen” Use Case. Da falls man nicht alle Daten von der Pendenz holt, dann kann man eine Pendenz nicht einzeln ansehen. Falls man eine Pendenz erstellen will, dann included die “Pendenz erstellen” Use Case von der “Eingaben überprüfen” Use Case. Die Usereingaben werden immer überprüft, aber nicht immer werden die Daten in der DB gespeichert und auch nicht immer wird ein Error Meldung ausgegeben.

4.2 Entity-Relationship-Diagramm



So sieht bei uns die einzige Entität aus. Die Id ist der PK von unserer Entität. Der Title wird gebraucht um den Titel der Pendenz zu speichern. Im Description Feld werden die Beschreibungen gespeichert. Bei Importance speichert man die Dringlichkeit der Pendenz und bei DateToFinish wird das Enddatum gespeichert, wann es fertig sein sollte.

5. Testing

Testfall-Nr.	1
Testfall-Bezeichnung	Pendenzenliste / Pendenz hinzufügen
Testumgebung	Google Pixel 2 API 27 Pendence Application
Zu testende Funktionalität	Testet, wenn man auf der Liste ist, eine neue Pendenz erstellen kann.

Datum der Testdurchführung		05.06.20			
Tester		Samuel Sättler			
Testauswertung:					
Nr.	**Aktion**	**Erwartetes Ergebnis**	**Effektives Ergebnis**	**Erfolgreich**	**Bemerkung**
1.	Starten Sie die Applikation	Die Pendenzenliste wird angezeigt	Wie erwartet	Ja	
2.	Drücken Sie auf den Plus Button	Die “Pendenz erstellen” Seite öffnet sich	Wie erwartet	Ja	

Testfall-Nr.		2			
Bezeichnung		Pendenzenliste / Pendenz anzeigen			
Testumgebung		Google Pixel 2 API 27 Pendence Application			
Zu testende Funktionalität		Testet, ob man auf der Liste der Pendenzen auf eine darauf drücken kann und sie sich öffnet			
Datum der Testdurchführung		05.06.20			
Tester		Samuel Sättler			
Testauswertung:					
Nr.	**Aktion**	**Erwartetes Ergebnis**	**Effektives Ergebnis**	**Erfolgreich**	**Bemerkung**
1.	Starten Sie die Applikation	Pendenzenliste wird angezeigt	Wie erwartet	Ja	
2.	Klicken Sie auf eine Pendenz drauf	Alle erforderlichen Pendenz Daten werden angezeigt	Wie erwartet	Ja	

Testfall-Nr.		3			
Bezeichnung		Pendenz erstellen - Eingaben für Pendenz erstellen			
Testumgebung		Google Pixel 2 API 27 Pendence Application			
Zu testende Funktionalität		Testet, ob alle Eingaben akzeptiert werden und ein neuer Eintrag in der Datenbank entstanden ist.			
Datum der Testdurchführung		05.06.20			
Tester		Samuel Sättler			
Testauswertung:					
Nr.	**Aktion**	**Erwartetes Ergebnis**	**Effektives Ergebnis**	**Erfolgreich**	**Bemerkung**

1.	Eingabe in Feld “Titel”: “Tisch putzen”	Eingabe wird akzeptiert	Wie erwartet	Ja	
2.	Eingabe in Feld “Beschreibung”: “Bitte alle Tische putzen.”	Eingabe wird akzeptiert	Wie erwartet	Ja	
3.	Eingabe in Feld “Datum eingeben”: Datum per klick wählen oder “01.09.2020” eingeben.	Eingabe wird akzeptiert	Wie erwartet	Ja	
4.	Eingabe in Feld “Wichtigkeit”: Auf Dropdown klicken und Hoch auswählen	Eingabe wird akzeptiert	Wie erwartet	Ja	
5.	Auf Pendenz speichern drücken	Alle Pendenz Daten werden in die Datenbank geschrieben und man landet auf der Pendenz auflistung Seite und sieht den neuen Eintrag.	Wie erwartet	Ja	

Testfall-Nr.		4			
Bezeichnung		Negativer Test Pendenz erstellen - Eingaben für Pendenz erstellen			
Testumgebung		Google Pixel 2 API 27 Pendence Application			
Zu testende Funktionalität		Testet, ob man auf der Liste der Pendenzen auf eine darauf drücken kann und sie sich öffnet			
Datum der Testdurchführung		05.06.20			
Tester		Samuel Sättler			
Testauswertung:					
Nr.	**Aktion**	**Erwartetes Ergebnis**	**Effektives Ergebnis**	**Erfüllt**	**Kommentar**
1.	Eingabe in Feld “Titel” mit mehr als 50 Zeichen	Die Seite bleibt bestehen mit einer Fehlermeldung, dass das Feld nicht mehr als 50 Zeichen haben darf.	Wie erwartet	Ja	
2.	Eingabe in Feld “Beschreibung”, mit mehr als 500 Zeichen”	Die Seite bleibt bestehen mit einer Fehlermeldung, dass das Feld nicht mehr als 500 Zeichen haben darf.	Wie erwartet	Ja	
3.	Datum mit falschen Format schreiben	Die Seite bleibt bestehen mit einer Fehlermeldung, dass Datum im DD.MM.YYYY Format sein muss.	Wie erwartet		

6. Fazit

Bei dem ÜK 335 (Mobile-Applikation realisieren) ging es darum, uns allgemein über Mobile Applikationen zu unterrichten und das erstellen einer App näher zubringen. Was unserer Meinung nach sehr gelungen ist, da wir sehr viel über Mobile-Apps gelernt haben und als Highlight sogar selber eine Android App programmieren konnten. Wie uns aber auch von anfang an gesagt wurde, ist das programmieren mit Android Studios manchmale sehr kompliziert ist, auch in den diversen XML Files gab es ab und zu Probleme mit dem fehleranfälligen und unsauberen Code. Als abschließendes Fazit für unser Programm und allgemein den Kurs, können wir sagen, dass wir sehr viele spannende neue Informationen bezüglich Mobile-Applikationen bekommen haben und dass wir mit unserem Projekt schlussendlich sehr zufrieden sind und das uns Spaß gemacht hat.