Projektdokumentation

Entwicklung einer Mobile-App-Anwendung zum Annehmen von verschiedenen Aufgaben, die für die Auszubildenden im Unternehmen bereitgestellt werden

Auszubildender Fachinformatiker für Anwendungsentwicklung
Toni Kozarev
Projektzeitraum vom 11.10.2018 bis 08.11.2018

Auszubildender

Name:	Toni Kozarev
Prüflingsnummer:	40011
Geburtsdatum:	13.05.1996
Anschrift:	Düsternortstraße 110 27755 Delmenhorst

Ausbildungsbetrieb
Dataport Anstalt öffentlichen Rechts

Niederlassung:	Bremen
Anschrift:	Utbremer Str. 90 28217 Bremen

Inhaltsverzeichnis

Tabellenverzeichnis

Liste von Code-Anhängen

Abbildungsverzeichnis

1 Einführung

Diese Projektdokumentation beschreibt die Planung, Implementierung und das Testen einer Mobile-App zum Annehmen von verschiedenen Aufgaben, die für die Auszubildenden im Unternehmen bereitgestellt werden. Die Projektdokumentation berücksichtigt, dabei neben der Auswahl passender Frameworks, wichtige Aspekte, wie eine Kosten-Nutzen-Analyse und den Datenschutz.

1.1 Unternehmensprofil

Das Unternehmen Dataport stellt für die öffentliche Verwaltung IT bereit und bietet den Behörden ein umfassendes Angebot von Dienstleistungen wie IT-Beschaffung, Fortbildungen und Schulungen [1]. Neben den oben aufgelisteten Dienstleistungen werden Datensicherheitskonzepte, sowie E-Goverment-Lösungen und Anwendungen für Verwaltungsaufgaben angeboten. Dataport ist eine Anstalt des öffentlichen Rechts.
Der Betrieb wurde am 1. Januar 2004 durch den Zusammenschluss der Datenzentrale Schleswig-Holstein mit dem Landesamt für Informationstechnik und der Abteilung für Informations- und Kommunikationstechnik des Senatsamtes für Bezirksangelegenheiten gegründet und hat seinen Sitz in Altenholz. Neben den Sitz in Altenholz hat unser Betrieb Niederlassungen in Hamburg, Bremen, Rostock, Lüneburg, Halle und Magdeburg.
Die Aufgabe unseres Unternehmens ist die Versorgung von Kommunikations- und Informationstechnik für die Trägerländer Hamburg, Bremen, Niedersachsen, Mecklenburg-Vorpommern, Schleswig-Holstein und Sachsen-Anhalt als Full Service Provider. Dataport hat momentan über 2.700 Mitarbeiterinnen und Mitrarbeiter und erzielte 2017 einen Jahresumsatz von 547 Millionen Euro [2].

1.2 Themenschwerpunkte der Ausbildung

Die Schwerpunkte während der Ausbildung zum Fachinformatiker für Anwendungsentwicklung bei Dataport lagen in dem Testen von Software mit C# und in der Entwicklung von Programmen mit Java. Daneben waren weitere Ausbildungsinhalte wie Virtualisierung, Netzwerktechnik bzw. Netzwerkinfrastrukturen, Linux, Auftragssteuerung und Accounting, die am Standort in Bremen vermittelt wurden.

1.3 Projektbeschreibung

Es soll ein Prototyp einer Mobile-App für die Abstimmung einer betrieblichen Aufgabe von den Auszubildenden aller Standorte unserer Firma entwickelt werden. Diese soll auf dem Android-Betriebssystem [3] lauffähig sein.

Alle Benutzerdaten und wichtige Informationen für die Aufgaben sollen lokal gespeichert werden. Zu diesem Zweck sollte eine relationale SQL-Datenbank erstellt werden. Diese wurde auch für die Authentifizierung, Datenänderungen und Tests benutzt.

Im Betrieb werden die Auszubildenden in verschiedenen Teams verteilt und können sich ständig bei unterschiedlichen Tätigkeiten engagieren. Jedes Team ist für einen bestimmten Bereich verantwortlich, z.B. Technik-, Messe-, Software,- und Marketing-Team. Jede/r Auszubildende/r soll mindestens einem Team zugeordnet werden und regelmäßig sein/ihr Teil für dieses Team beitragen.

Die Idee dieser Anwendung basiert auf der Kanban-Methode. Die Aufgaben sollen in drei verschiedenen Zuständen organisiert werden - *Offen*, *In Bearbeitung*, *Erledigt*. Siehe Abbildung unten.

Es werden oft neue Aufgaben in der Datenbank von dem Administrator hinzugefügt, welche angenommen werden können. Manche Aufgaben brauchen mehrere Auszubildende, deswegen sollen sie mehrmals von dem Verteiler hinzugefügt werden, damit die notwendige Anzahl von Teilnehmern erreicht werden kann.

Alle Auszubildenden, die diese Android-App benutzen sollen, müssen zuerst registriert und erfolgreich angemeldet werden, sodass sie das Hauptmenü erreichen können. Dort sollen sie die Suchkriterien eingeben. Es soll eine Liste mit dem entsprechenden Team und Zustand erzeugt werden. Die ältesten Aufgaben werden am Anfang der Suchliste angezeigt, weil sie am dringendsten sind.

Der Benutzer kann jede Aufgabe von den gefundenen Ergebnissen öffnen und ihre Details durchlesen. Er hat die Möglichkeit den Titel, die Beschreibung, das Datum der Veröffentlichung, das Team und den Status anzuschauen, bevor er die Aufgabe annimmt. Die offenen Aufgaben sollen von allen registrierten Benutzern sichtbar sein. Nur die Person, die die Aufgabe zuerst akzeptiert hat, kann diese auch später als erledigt markieren, weil nur sie die Verantwortung für diese Tätigkeit trägt. Wenn eine Aufgabe angenommen wurde, soll sie ihr Status von „Offen“ auf „In Bearbeitung“ ändern. Solange eine Aufgabe fertig ist, kann ihren Status auf „Erledigt“ eingestellt werden und der Benutzer wird diese Aufgabe nachher unter „Erledigte“ Aufgaben finden können. Ihr Zustand sollte nicht mehr änderbar sein. Sie wird nur für Informationszwecke angezeigt werden.

Eine Sitzung wird nach der erfolgreichen Anmeldung eröffnet. Der Nutzer kann sich jederzeit sorglos die Mobile-App anschließen. Sobald der Benutzer angemeldet ist, wird seine Sitzung aktiv bleiben und er konnte die Mobile-App unbeschränkt von seinem Profil nutzen. Eine neue Anmeldung ist nur dann notwendig, wenn ein Nutzer sich von dem Hauptmenü explizit abmeldet und seine Sitzung beendet.

Aus dem Projekt soll hervorgehen, ob eine Fortführung des Projektes zu einem späteren Zeitpunkt sinnvoll und wirtschaftlich ist.

2 Projektvorbereitung

In diesem Kapitel wird der aktuelle Zustand des Projekts beschrieben, sowie der Soll-Zustand, der nach dem Projekt erreicht werden soll.

2.1 Ist-Analyse

Momentan verfügt Dataport über keine technische Umsetzung, wie bspw. eine Mobile-App für die Verteilung von Aufgaben durch die Auszubildenden. Es ist lediglich möglich über MS-Outlook am PC Nachrichten mit allen Tätigkeiten, die gemacht werden sollen, an eine bestimmte Gruppe von Personen zu versenden. Der Ersteller wird von allen Auszubildenden Zusage/Absage-Emails für jedes Angebot bekommen und sein Posteingang wird überflutet. Wenn es Unklarheiten über die Aufgaben gibt, dann soll der Verteiler die Fragen der Auszubildenden beantworten und diese Diskussionsrunde wird durch Emails durchgeführt, welche viel Zeit verbrauchen werden.

2.2 Soll-Konzept

Zweck dieses Projekts soll eine lauffähige Mobile-App sein, die auf dem Android-Betriebssystem verwendet werden kann. Dazu soll das Android SDK verwendet werden. Dieses Projekt soll darauf abzielen, eine Mobile-App, welche auf die Idee von Kanban-Brett basiert, darzustellen. Die Informationen für die Aufgaben und alle Benutzerdaten sollen in einem relationalen Datenbankmodell gespeichert werden. Diese Datenspeicherung soll authentisch, korrekt und sicher ablaufen.
Damit neue Benutzer sich anmelden können, wurde die Registrierung zuerst implementiert, welche nicht Teil des Projektes war, stellt sich aber sehr entscheidbar für die Weiterentwicklung der Mobile-App heraus, da die Passwörter der registrierten Benutzern geschützt in der Datenbank gespeichert werden sollen. Danach konnte die Anmeldung programmiert werden, wo das gespeicherte und das eingegebene Passwort, welches zuerst verschlüsselt werden soll, verglichen werden muss. Dazu sollen bestimmte Bibliotheken ausgewählt werden, um beiden Funktionen eine sichere Datenverschlüsselung bereitzustellen. Nachdem ein Benutzer erfolgreich angemeldet ist, kann er eine Liste mit allen verfügbaren Aufgabentiteln durch die Suchkriterien erstellen. Danach sollte es möglich sein, dass der Nutzer durch die angezeigten Titel eine Aufgabe aussuchen und eröffnen kann. Dementsprechend sollen alle Informationen für diese Tätigkeit angezeigt werden. Damit der Administrator neue Aufgaben in der Datenbank hinzufügen kann, wurde eine Admin-Plattform implementiert. Diese konnte nur mit der Email-Adresse *„admin@dataport.de“* erreicht werden. Das war auch kein Teil des Projektes, aber wird sehr hilfreich für den Verteiler der Aufgaben sein, damit er nicht stets die Datenbank korrigieren muss. Diese Funktionalität soll den Prozess erleichtern und beschleunigen.
Es sollen neben der Durchführung der eigentlichen Mobile-App auch automatisierte Tests entwickelt und implementiert werden. Darüber hinaus soll es möglich sein, dass jeder angemeldete Benutzer seine Sitzung bei Schließung der Applikation nicht verliert und angemeldet bleibt.

2.3 Kosten-Nutzen-Analyse

Für dieses Projekt werden insgesamt 70 Arbeitsstunden geplant. In diesem Zeitraum muss das ganze Projekt entworfen, implementiert und getestet werden. Am Ende soll es ausführlich dokumentiert werden. Wenn die App fertiggestellt ist, muss sie nur als .apk Datei verteilt und von dem Benutzer auf dem privaten Smartphone installiert werden. Dieser Ablauf sollte kaum mehr als 5 Minuten dauern.
Es werden durch die Entwicklung der Mobile-App weder Anschaffungskosten neben dem Arbeitslohn des Anwendungsentwicklers noch laufende Kosten anfallen. Die Rechnung entspricht

70 Arbeitsstunden $\cdot\ 70\frac{Euro}{Arbeitsstunden} = 4900$ Euro. Die Entwicklungskosten für dieses Projekt belaufen sich auf 4900 Euro.

Die Zeit, die durch die Nutzung der App gespart wird, ist nur ungenau zu berechnen, da jeder Nutzer unterschiedlich lang braucht, um die Details einer Aufgabe durchzulesen.
Zum Ersten soll der Verteiler der Aufgaben nicht mehr sorgen, wer von den Auszubildenden Zeit und Lust hat, eine Tätigkeit anzunehmen und die Verantwortung für sie zu tragen. Diese wurde komplett von der ganzen Mobile-App übernommen. Außerdem kann der Administrator jederzeit einen Blick in der lokalen Datenbank für Details werfen. Ein anderer Vorteil ist, dass es meistens mehrere Freiwillige gibt, die eine Aufgabe annehmen möchten. Solche Situationen, mit beschränkten Plätzen einer Aufgabe, führen zu Überschneidungen, wenn der Verteiler mit den Auszubildenden nur durch Emails kommunizieren kann.

Erfahrungswerte zeigen, dass es monatlich ungefähr 25-30 neue Aufgaben von verschiedenen Teams gibt, die gemacht werden müssen. Jede Aufgabe wurde vor der Implementierung der Mobile-App ca. 10 Minuten von dem Verteiler und dem/der Auszubildende/r diskutiert. Das sind 30 Auszubildenden, die 10 Minuten jeden Monat und 1 Verteiler, der 300 Minuten in solcher Diskussionen verbringen. Insgesamt sind das 600 Minuten, was auch ca. 10 Stunden monatlich sind oder ungefähr 120 Stunden pro Jahr. Mit der Applikation konnte der Benutzer in ca. 3-5 Minuten die Beschreibung einer Aufgabe lesen und entscheiden, ob er die Aufgabe annehmen möchte oder nicht. Das Hinzufügen von Aufgaben monatlich wird auch nicht mehr als 1 Stunde dauern.

Die **pessimistische Berechnung** soll entsprechen:
30 Auszubildenden $\cdot\ 5$ Minuten $= 150$ Minuten
150 Minuten $+\ 60$ Minuten $= 210$ Minuten (addiert die verbringte Zeit des Verteilers)
210 Minuten $\cdot\ 12$ Monate $= 2520$ Minuten pro Jahr
Insgesamt $\frac{2520 Minuten}{60 Minuten} = 42$ Stunden/jährlich

Die **optimistische Berechnung** soll so aussehen:
30 Auszubildenden $\cdot\ 3$ Minuten $= 90$ Minuten
90 Minuten $+\ 60$ Minuten $= 150$ Minuten (addiert die verbringte Zeit des Verteilers)
150 Minuten $\cdot\ 12$ Monate $= 1800$ Minuten pro Jahr
Insgesamt $\frac{1800 Minuten}{60 Minuten} = 30$ Stunden/jährlich

Also mit dieser Applikation sollten alle Nutzer jährlich zwischen $120 - 42 = 78$ Stunden und $120 - 30 = 90$ Stunden ersparen.

Eine objektive Einschätzung der Ersparnis wäre eine Verbesserung der Arbeitszeit um ca. 65-75%. Bei geschätzten Lohnkosten von 70 Euro pro Stunde belaufen sich die Entwicklungskosten so auf 4900 Euro und die jährlichen Einsparungen auf 5460 Euro bis 6300 Euro. Mit der Amortisierung der App ist so bereits nach weniger als ein Jahr nach Einführung zu rechnen.

2.4 Testplanung

Die Testfälle für diese Android-App sind so erstellt, dass es für einige Java Klassen eine Testsuite gibt. Alle Tests sind automatisiert und werden mithilfe des Frameworks Espresso gemacht. Es gibt Tests, die zum Beispiel die Aktionen: ein Benutzer registrieren oder anmelden, eine Aufgabe mit bestimmten Suchkriterien suchen oder Aufgaben erstellen (nur als Administrator möglich), testen. Darüber hinaus soll überprüft werden, ob eine Aufgabe von dem angemeldeten Benutzer korrekt

angenommen werden kann. Dabei werden ebenfalls auch triviale Tests durchgeführt, die Tests sollen überprüfen, ob jede Aktion, die an einen Button gebunden ist, erfolgreich ausgeführt wird. Es werden noch verschiedene UI Elemente getestet, u.a.:

- Layout
- Button
- Edit Text
- List View

Weitere Informationen bzgl. der Tests, siehe Unterabschnitt 3.5.

#	Testszenarien	einzelne Testfälle
1	Start der App	Sind alle UI Elemente korrekt geladen und angezeigt?
2.1	Anmelden als Administrator	Kann der Administrator anmelden?
2.2	Aufgaben hinzufügen	Kann der Administrator neue Aufgaben in der Datenbank hinzufügen?
3	Anmelden als Benutzer	Ist eine Anmeldung erfolgreich oder nicht? Ist die Email/Passwort gültig?
4	Registrieren	Ist die Registrierung erfolgreich oder wurde die Email schonmal benutzt?
5	Menü	Funktionieren den Abmeldungsbutton und das Menü fehlerfrei?
6	Listenergebnisse	Zeigt die Liste die richtige Ergebnisse an?
7	Aufgabe anzeigen	Kann ein/e Auszubildende/r eine Aufgabe annehmen/erledigen?
8	Hinzugefügte Aufgaben	Können die neuen Aufgaben auf die Listenergebnisse gefunden werden?

Tabelle 1: Tesplanung der Integrationstests

3 Projektdurchführung

Im folgenden Kapitel werden alle Schritte bei der Implementierung der Android-App beschrieben und welche unerwarteten Hindernisse bzw. Probleme bei der Realisierung der geforderten Anforderungen auftraten.

3.1 Auswahl der Software

Die Auswahl der richtigen Software und Frameworks ist ein wichtiger Teil des Projektes. Die Entwicklungsumgebung **Android Studio** [4] ist eine sehr beliebte Umgebung für die Erzeugung von Android Applikationen und wird meist empfohlen. Außerdem musste eine lokale Datenbank mit der Software **Sqliteman** [5] erstellt werden. Die Dokumentation [6] in **Sqlite** wurde sehr verständlich und ausführlich geschildert, sowie öffentlich zugänglich und kommt mit vielen Beispielen.

3.1.1 Auswahl der Entwickungsumgebung

Für die Implementierung der Mobile-App wurde sich für die Entwicklungsumgebung Android Studio entschieden. Diese Umgebung wird empfohlen für den Einsatz der Android SDK und bringt bereits Gradle [7] mit, das für die Anwendung der Mobile-App benötigt wird.

3.1.2 Auswahl der Frameworks

Am Anfang des Projektes wurde sich für bestimmte Frameworks und Bibliotheken entschieden, die für die Durchführung des Projektes fundamental sind.
Dazu zählt die Bibliothek **BCrypt**, die für die Verschlüsselung des Benutzer-Passworts benutzt werden sollte, bevor dieses in der lokalen Datenbank gespeichert wird.
Damit die ausgewählten Frameworks/Bibliotheken funktionieren, müssen sie in der Gradle-Konfiguration eingerichtet werden. Nach der Implementierung einer Dependency haben alle Klassen und Methoden Zugriff auf die enthaltenen Funktionalitäten in den Frameworks.

```
1  implementation 'dependency'
```

Hierbei muss **dependency** durch den Namen ersetzt werden, dessen Dependency hinzugefügt werden soll. Diese ist üblicherweise beim Anbieter des Frameworks oder der Bibliotheken zu finden. In diesem Fall soll die **build.gradle** Konfigurationsdatei mit der unten stehenden Zeile erweitert werden.

```
1  implementation 'org.mindrot:jbcrypt:0.4'
```

3.1.3 Auswahl der Tests Frameworks

Außerdem wird für das Testen ein sehr populäres Framework benutzt: **Espresso** [8]. Dieses Framework wurde für die Erstellung der automatisierten Tests ins Spiel gebracht. Diese UI-Tests liefen auf einem Smartphone oder Emulator und überprüfen, ob bei einer Änderung am User Interface das gleiche Ergebnis vorkommt. Diese Tests werden auch Instrumented Tests genannt. Mit den nächsten Zeilen können sie in der Gradle-Konfiguration eingerichtet werden:

```
1  androidTestImplementation 'com.android.support.test:runner:1.0.2'
2  androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
3  androidTestImplementation 'com.android.support.test:rules:1.0.2'
```

Für die erstellte JUnit Tests musste noch eine Zeile hinzugefügt werden:

```
1  testImplementation 'junit:junit:4.12'
```

Die verwendeten Frameworks sind Open Source Software und es fallen keine Kosten hierfür an.

3.2 Entwicklung des UI Designs der Mobile-App

Als Erstes wird ein sogenanntes Mockup der App erstellt. Ein Mock-Up bedeutet einen Wegwerfprototyp der Benutzerschnittstelle einer zu erstellenden Software. Die Abbildung 6 stellt ein Beispiel für ein Mockup dar, das für das Projekt angefertigt wurde. Dieser Prototyp wurde in **Adobe Spark** [9] erzeugt. Auf dem Screenshot kann man sehen, wie die Positionierung der einzelnen Elemente geplant war und berücksichtigt wird, dass die Elemente sich nicht überschneiden. Die Schriftgröße der Texte und die Größe der Menü Buttons müssen auch entsprechend gewählt werden, um das Lesen und das Navigieren in der Mobile-App zu erleichtern. Die Implementierung dieses Mockups wird in der Abbildung 6 repräsentiert.

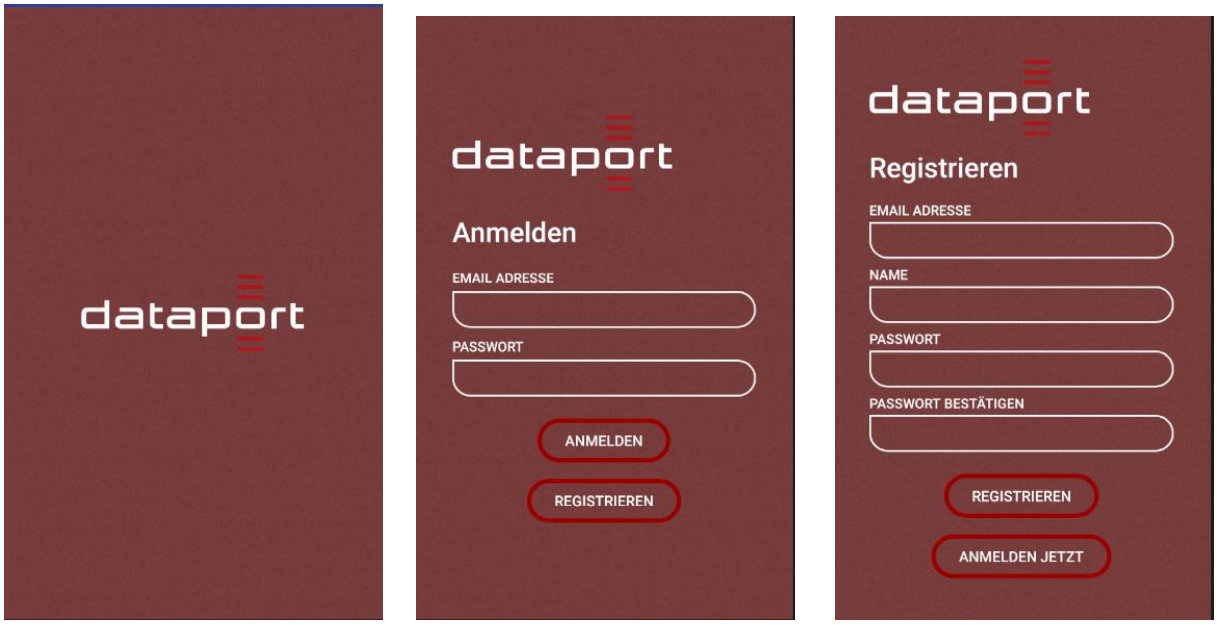
Das Hintergrundbild wurde mithilfe **Background Image Generator** [10] erstellt. Dies wurde dann mit dem **Batch Import Plugin** im Android Studio zu dem Projekt hinzugefügt, damit es für alle Größe des Bildschirms eines Smartphones das entsprechende Bild angepasst wurde.
Das runde Icon 7 wurde mithilfe **Launcher Icon Generetor** [11] gemacht. Außerdem wurden verschiedene Layouts mit allen notwendigen UI Elemente gemacht. Alle Elemente werden hauptsächlich durch die XML-Dateien generiert, aber sie können auch in dem Java-Code geändert werden.

Um die App universell und speziell für unser Unternehmen zu machen und keine populären Layouts zu benutzen, wurde das ganze Projekt in einem Vollbildansicht-Thema gemacht. Dazu wurde ein Splashscreen mit einer Animation hinzugefügt. Die Navigation durch die verschiedenen Aktivitäten wurde leicht und intuitiv gemacht, um die App nutzerfreundlicher zu machen. Es werden meistens Farben benutzt, die im Unternehmens Logo, auf der Webseite und in den Dokumenten von Dataport verwendet werden.

3.3 Implementierung der Klassen

In diesem Kapitel wird die ganze Implementierung des Projektes beschrieben. Beim Starten der App wird ein Splashscreen aktiviert. Nach dem Laden der Applikation sollte eine Anmeldungsmaske erzeugt werden. Um ein/e Auszubildende/r anzumelden, sollte er/sie vorher registriert werden sein. Dazu wird es einen Button geben, der den Benutzer zu einer anderen Aktivität weiterleitet, wo die Registrierung möglich ist. Wenn ein Konto erstellt wird, sollte die Person zunächst versuchen, sich mit ihren Benutzerdaten anzumelden. Wenn die Anmeldung erfolgreich war, wird eine Sitzung gestartet und das Menü angezeigt.
Dort kann jeder Benutzer eine Suche mit den Spinner durchführen. Als Ergebnis wird eine neue Seite mit den Listenergebnisse geöffnet. In dieser Liste werden alle gefundene Aufgaben mit den ausgewählten Suchkriterien aufgezählt. Beim Klicken auf einen Aufgabentitel werden die Details für sie angezeigt. Danach kann diese Tätigkeit angenommen werden. Am Ende sollte einen Abmeldungsbutton in dem Hauptmenü implementiert werden, der die Sitzung des Benutzers beendet.
Screenshots: 3

Abbildung 1: Screenshots: Splashscreen, Anmeldung, Registrierung



(a) Splashscreen (b) Anmeldung (c) Registrierung

3.3.1 Implementierung der Register

Beim Starten der Registrierungsmaske wird ein Splashscreen aktiviert. Die Registrierung von neuen Benutzern soll in dieser Klasse durchgeführt werden. Damit die Registrierung erfolgreich ist, sollen einige Bedingungen erfüllt werden:

1) Die Email-Adresse sollte nicht registriert werden. (Die *checkIfExists*-Methode von der *DatabaseHelper.java*, welche eine SQL-Anfrage enthält, überprüft diese Bedingung.)
2) Die Email sollte gültig sein. (siehe unten gezeigte *isEmailValid*-Methode)
3) Die Passwörter sollen übereinstimmen.
4) Alle Felder müssen ausgefüllt sein.
5) Das Passwort soll mindestens 6 Zeichen enthalten.

Wenn alles erfüllt ist, wird das Passwort mit der **BCrypt**-Verschlüsselung [12] geschützt und in der Datenbank gespeichert.

```
1  String generatedPasswordHash = BCrypt.hashpw(passwordStr, BCrypt.gensalt(12));
```

Die **hashpw** ist eine Methode von der BCrypt-Bibliothek, welche für die Verschlüsselung einer Zeichenkette benutzt wird. Diese wurde mit der nachfolgende Zeile importiert:

```
1  import org.mindrot.jbcrypt.BCrypt;
```

Danach wird ein Objekt vom Typ *User* erzeugt. Dort werden mit allen *setter*-Funktionen, die in der Klasse User.java vorhanden sind, die Informationen von den Felder - den Name, das Passwort und die Email-Adresse in diesem Objekt gespeichert. Das ganze Objekt wird dann in der Funktion *insertUser* hinzugefügt, welche aus der Klasse **DatabaseHelper.java** vorkommt. Die SQL-Anfrage in dieser Methode wird sich darum kümmern, die Informationen dieses Benutzers korrekt und sicher in

der lokalen Datenbank zu speichern.

Die *isEmailValid*-Funktion [13] validiert, ob eine Email-Adresse gültig für eine Registrierung ist:

```
1    public static boolean isEmailValid(String email) {
2          String emailPattern = "^[a-zA-Z0-9_*-]+(?:\\." +
3                "[a-zA-Z0-9_+&*-]+)*@" +
4                "(?:[a-zA-Z0-9-]+\\.)+[a-z" +
5                "A-Z]{2,7}$";
6          Pattern pat = Pattern.compile(emailPattern);
7          return email != null && pat.matcher(email).matches();
8      }
```

Diese Funktion benutzte dieses Muster [14] und sollte mit der folgenden Zeilen in der Klasse importiert werden, bevor der eigentlichen Implementierung der Methode:

```
1    import java.util.regex.Pattern;
```

3.3.2 Implementierung der DatabaseHelper

Damit die Klasse **DatabaseHelper** erstellt wurde, sollten einige Importe gemacht werden:

```
1  import android.content.ContentValues;
2  import android.content.Context;
3  import android.database.Cursor;
4  import android.database.sqlite.SQLiteDatabase;
5  import android.database.sqlite.SQLiteOpenHelper;
```

In dieser Klasse werden alle Datenbankänderungen implementiert. Hier sind alle SQL-Anfragen beschrieben, die das Programm benutzt. In der Datenbank sollen zwei Tabellen vorhanden sein - **Tasks** 8, **Users** 8

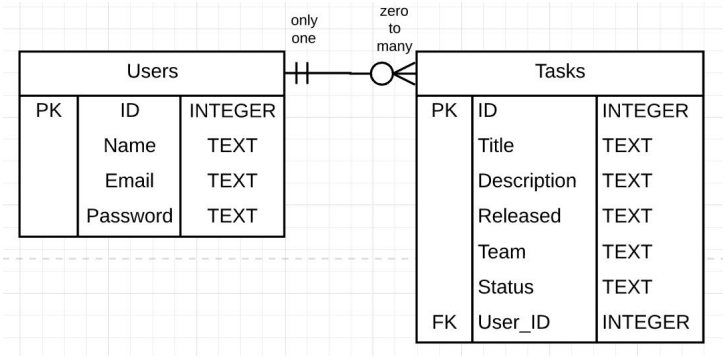
Das Objekt vom Typ **ContentValues** sollte für das Hinzufügen neuer Werte in der Datenbank benutzt werden. Am Anfang wird auf das **SQLiteDatabase db**-Objekt die Methode *getWritableDatabase* aufgerufen. Der Cursor wird mithilfe der SQL-Anfragen gefundene Werte in der Datenbank hinzugefügt. Danach wird auf das **db**-Objekt die Methode *insert* aufgerufen und der Cursor bzw. die Datenbank anschließen. Als Beispiel dient eine Methode von dieser Klasse, die in der Datenbank Informationen hinzufügen sollte:

```
1  public void insertUser(User user) {
2        db = this.getWritableDatabase();
3        ContentValues cv = new ContentValues();
4        String query = "SELECT * FROM " + TABLE_NAME;
5        Cursor crs = db.rawQuery(query, null);
6        int count = crs.getCount() + 1;
7        cv.put(COLUMB_ID, count);
8        cv.put(COLUMB_EMAIL, user.getEmail());
9        ...
10       db.insert(TABLE_NAME, null, cv);
11       crs.close();
12       db.close();
13   }
```

Das Lesen von der Datenbank sollte auch nicht kompliziert sein. Zuerst wird auf das **SQLiteDatabase db**-Objekt die Methode *getReadableDatabase* aufgerufen. Dann wird der Cursor alle Werte, die die entsprechende SQL-Anfrage findet, durchgehen. In dem folgenden Beispiel wird den Cursor, der die Ergebnisse hat, zurückgegeben. Dieser kann später in einer anderen Java-Datei benutzt und mit einer WHILE-Schleife durchgelaufen werden:

```
1  public Cursor queryList() {
2        db = this.getReadableDatabase();
3        String query = "SELECT * FROM " + TABLE_NAME2;
4        return db.rawQuery(query, null);
5    }
```

Die Datenbank hat die Struktur von dem Bild unter.



3.3.3 Implementierung der Login

Beim Starten der Anmeldungsmaske wird ein Splashscreen aktiviert. Danach sollte diese Klasse ermöglichen, dass sich ein Benutzer in der Anwendung anmeldet. Damit die Anmeldung erfolgreich ist, sollen einige Bedingungen erfüllt werden:

1) Die Email-Adresse muss gültig sein.
2) Die Email-Adresse oder das Passwort dürfen nicht leer sein.
3) Die Passwörter müssen übereinstimmen.

Wenn es alles erfüllt ist, wird das eingegebene Passwort mit der **BCrypt** verschlüsselt und mit dem gespeicherten Passwort dieses Benutzers verglichen. Diese wird mit der Methode **checkpw** durchzuführen.

```
1  boolean ifMatchPass = BCrypt.checkpw(passwordStr, pass);
```

Die **checkpw** benutzt die BCrypt-Bibliothek, dementsprechend soll diese auch importiert werden. Wenn der Vergleich erfolgreich ist, wird eine Sitzung mit diesem Benutzer eröffnet:

```
1  session.setLogIn(true); // start the session
2  session.saveId(idInt);  // save user_ID
```

Beide Methoden *setLogIn* und *saveId* sind in der Klasse **Session.java** implementiert und benutzen **SharedPreferences**, um Benutzerdaten lokal auf dem Smartphone zu speichern.

3.3.4 Implementierung der Session

In dieser Klasse wird die Sitzung eines Benutzers implementiert. Um eine Session zu eröffnen, musste das folgende Zeile importiert werden:

```
1  import android.content.SharedPreferences;
```

SharedPreferences [15] wird die Rohdaten in der Form eines Schlüssels in der Datei der Mobile-App speichern. Es konnte den privaten Speicherungsmodus ausgewählt werden, damit die anderen Anwendungen in dieser Datei nicht zugreifen. Der Editor wurde sich um die sichere Änderungen der Informationen kümmern.

```
preferences = c.getSharedPreferences("KanAzubi", Context.MODE_PRIVATE);
      editor = preferences.edit();
      editor.apply();
```

In dem **SharedPreferences** werden die Email-Adresse, der Namen und die ID des Benutzers gespeichert. Mit den *getter*-Methoden dieser Klasse können diese Informationen immer abgerufen werden, wenn es eine aktive Sitzung gibt.

3.3.5 Implementierung der AdminPlattform

Diese Klasse wird es ermöglichen, dass der Administrator neue Aufgaben direkt von der Android-App in der Datenbank hinzufügen kann, indem er die Felder mit der Informationen für die Aufgabe ausfüllt. Diese Admin-Plattform kann nur mit der Email-Adresse *„admin@dataport.de“* erreicht werden. Die spezielle Implementierung hier ist die Methode *addTask* aus der *DatabaseHelper*-Klasse, welche die Informationen von dieser Klasse nimmt und die in der Datenbank hinzufügt.

```
public void addTask(..., String team) {
      db = this.getWritableDatabase();
      ContentValues cv = new ContentValues();
      String query = "SELECT * FROM " + TABLE_NAME2;
      Cursor crs = db.rawQuery(query, null);
      int count = crs.getCount() + 1;
      cv.put(COLUMB_ID, count);
      ...
      cv.put(COLUMB_STATUS, "Offen");
      cv.put(COLUMB_TEAM, team);
      cv.put(COLUMB_USERID, 0);
      db.insert(TABLE_NAME2, null, cv);
      crs.close();
      db.close();
  }
```

3.3.6 Implementierung der Menu

Beim Starten von dieser Aktivität wird geprüft, ob der angemeldeten Benutzer der Administrator ist. Wenn das der Fall ist, dann wird der *Suchen* Button in dem *Hinzufügen* Button geändert, welcher zu der *AdminPlattform* Aktivität weiterleitet.

```
if (session.getId() == 1 && session.getEmail().equals("admin@dataport.de")) {
    String addTask = "Hinzufügen";
      searchBtn.setText(addTask);
      ...
  }
```

Ansonsten wird die Klasse wie üblicherweise funktionieren und die Aktivität soll für Benutzer bereit sein. In dieser Klasse wurde auch der Abmeldungsbutton implementiert. Der sollte die aktive Sitzung beenden und den Benutzer abmelden. Beim Starten der Aktivität wird überprüft, ob eine Session vorhanden ist. Wenn eine solche nicht existiert, wird der Benutzer automatisch abgemeldet:

```
session = new Session(this);
  if (!session.loggedIn()) {
      logout();
  }
```

Wenn es aber eine aktive Sitzung gibt, dann kann beim Klicken von dem Abmeldungsbutton diese beendet werden. Dies ist nur mit einer Zeile möglich:

```
session.setLogIn(false);
```

In dieser Klasse gibt es noch zwei Spinner, welche die Suchkriterien entscheiden und an die nächsten Aktivität weiterleiten:

```
1     i.putExtra("status", statusStr);
2     i.putExtra("team", teamStr);
```

Und diese zwei Variablen soll in der Liste so behandelt werden, dass es nur eine Liste mit diesen Kriterien geöffnet wurde.

3.3.7 Implementierung der List

In dieser Klasse sollte eine Liste (ArrayList) mit den Ergebnissen von den Suchkriterien der *Menü*-Klasse erzeugt werden. In dem Code unten ist zu sehen, wie werden die durchgehende Werte mit dem „Intent“ für das Team und den Status funktioniern.

```
1   Intent i = getIntent();
2     Bundle bundle = i.getExtras();
3     if (bundle != null) {
4       status = (String) bundle.get("status"); //selected Status from Menu.java
5         team = (String) bundle.get("team"); //selected Team from Menu.java
6     }
```

Wenn es keine Ergebnisse gibt, sollte der Text „Die Liste ist leer“ anzeigen. Hier sollte auch die Benutzer ID geholt werden, damit die Liste nur für den richtigen Benutzer erstellt wird.

3.3.8 Implementierung der Tasks

In dieser Klasse sollen alle Informationen einer Aufgabe angezeigt werden. Jeder Benutzer soll auch die Möglichkeit haben, eine Aufgabe anzunehmen, wenn sie offen ist, oder als erledigt zu markieren, wenn diese schon von ihm angenommen wurde. Wenn eine Aufgabe „Erledigt“ ist, soll nur ihre Informationen zeigen, ohne einen Button unten. In dem Code unten sollte die richtige Aufgabe ID gefunden werden und diese auch in der Methode *addData* gesendet werden, damit die richtige Informationen für diese Tätigkeit ausgefüllt wird.

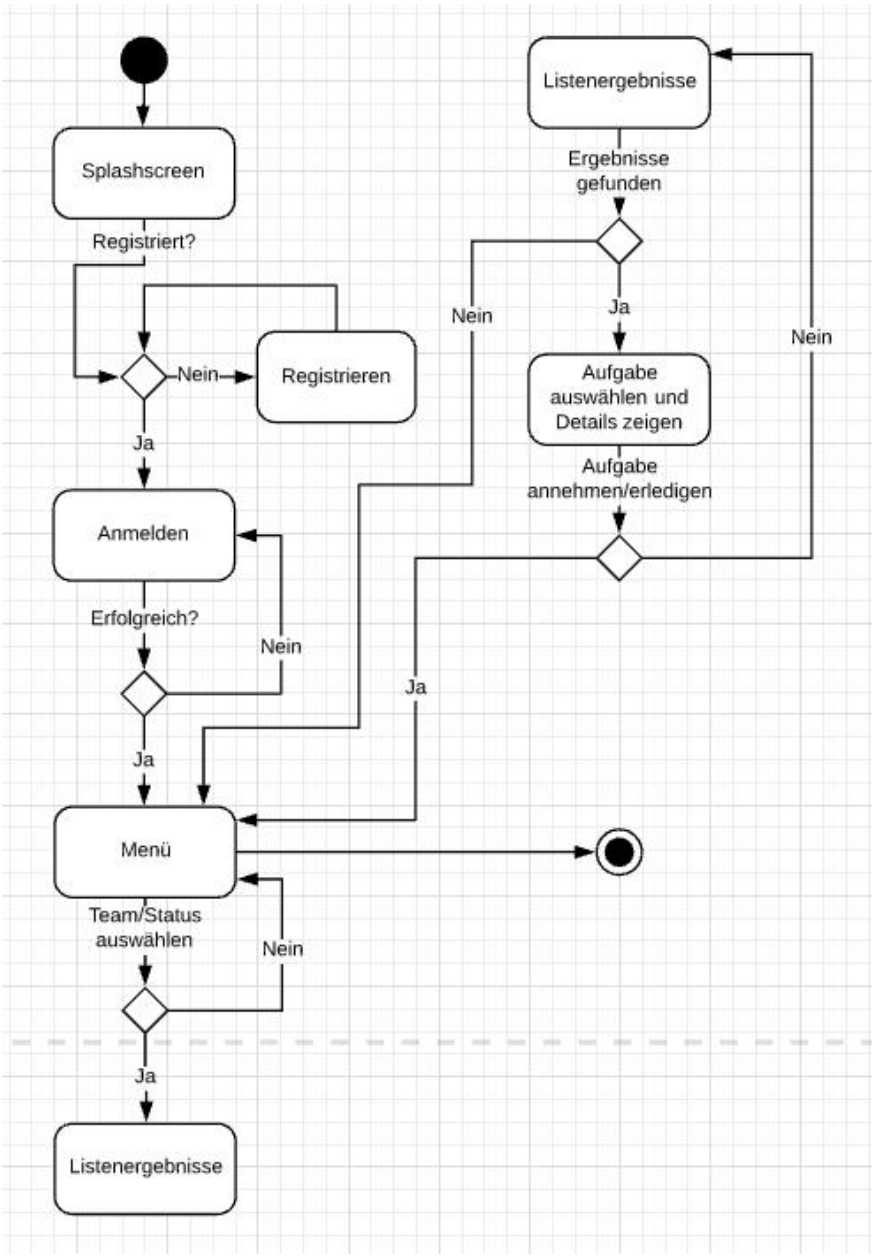
```
1 final Bundle bundle = getIntent().getExtras();
2     if (bundle != null) {
3         taskID = bundle.getInt("taskID");
4         listID = bundle.getIntegerArrayList("listID");
5     }
6     for(int i = 0; i < listID.size(); i++){
7          if(i == taskID){
8             taskNr = listID.get(i);
9           addData(taskNr);
10       }
11   }
```

Die anderen zwei Methoden, die sehr wichtig für diese Klasse sind, sind *acceptTask* und *doneTask*. Die werden in der *DatabaseHelper*-Klasse implementiert, da beide einige Werte in der Datenbank updaten müssen. Die SQL-Anfrage soll mit dem Wort „UPDATE“ anfangen und sie soll noch irgendwelche Bedingung in der „WHERE“-Anweisung hat, um nur ein Ergebnis zu aktualisieren. Siehe das Beispiel unten mit der erledigten Aufgabe:

```
1   String query = "UPDATE " + TABLE_NAME2 + " SET " +
2   COLUMB_STATUS + " = '" + doneTask +
3   "' WHERE " + COLUMB_ID2 + " = '" + taskNr + "'";
4   db.execSQL(query);
```

3.4 Entwurf der UML-Klassendiagramme

Vor der Implementierung der Klassen wurde ein UML-Klassendiagramm per Hand gezeichnet. Nach dem Entwurf aller Java Klassen mithilfe des Diagramms wurde ein Plugin **SimpleUML** [16] im Android Studio benutzt, um alle Variablen und Funktionen jeder Klasse darzustellen. Diese werden in Abbildung 11 gezeigt. Mithilfe eines Klassendiagramms konnte berücksichtigt werden, welche der geplanten Methoden implementiert und welche Variablen benötigt werden. Außerdem wurden noch ein Aktivitätsdiagramm (siehe Abbildung unten) und ein Anwendungsfalldiagramm Abbildung 10 erstellt, welche die Verbindung aller Aktionen und die Anwendungsfälle bzw. Akteure in diesem System darstellen. Diese werden mit der Webseite **Lucidchart** [17] erzeugt.



3.5 Entwickeln der automatisierten Tests

In diesem Ausschnitt wird die Erstellung der Tests betrachtet. Anhand des fertigen Designs und der implementierten Java Klassen konnten einige Tests entworfen werden. Es handelt sich um automa-

tisierte Tests, die nicht nur die Funktionalität der Oberfläche, sondern auch die Funktionalität der Methoden überprüfen können. Der Code-Anhang im Unterabschnitt C-9 zeigt, dass ein Test nicht nur einen Fehler auslösen kann, wenn ein Wert nicht korrekt ist, sondern auch die Richtigkeit der Funktion bei der Eingabe von korrekten Werten überprüfen kann. Einige Tests überprüfen, ob zum Beispiel die Anmeldung erfolgreich ist oder ob ein Benutzer sich mit einer gültigen Email-Adresse registrieren kann. Es wird auch geprüft, ob die Email-Adresse schon benutzt wurde oder ob das Passwort die Bedingungen erfüllt. Wenn ein Passwort nicht vorhanden ist oder weniger als sechs Zeichen hat, ist keine Registrierung möglich und es wird eine Fehlermeldung angezeigt. Beispiel: Anmelden und Abmelden als Administrator (siehe unten).

```
1        onView(withId(R.id.et_email)).perform(typeText("admin@dataport.de"));
2        closeSoftKeyboard();
3        onView(withId(R.id.et_password)).perform(typeText("123456"));
4        closeSoftKeyboard();
5        onView(withId(R.id.loginBtn)).perform(ViewActions.click());
6        onView(withText(R.string.successfulLogin))
7        .inRoot(withDecorView(not(is(loginActivity.getWindow().getDecorView()))))
8        .check(matches(isDisplayed()));
9        onView(withId(R.id.logoutBtn)).perform(ViewActions.click());
```

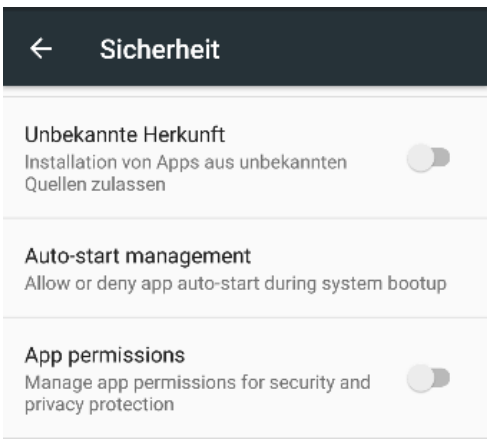
3.6 Entwickeln des JUnit Tests

Die automatisierten Tests haben die Funktionalität der Methoden in jeder Klasse geprüft. Es wurde eine Testsuite für einen **JUnit Test** [18] erstellt. Dieser Test soll prüfen, ob alle Emails, die für die Registrierung benutzt werden, gültig sind und ob die gestellten Bedingungen eingehalten werden. Er wird die *isEmailValid*-Funktion in der *Register*-Klasse validieren:

```
1  @Test
2  public void emailValidator_CorrectEmail() {
3    assertTrue(Register.isEmailValid("administrator@hotmail.co.uk"));
4    assertFalse(Register.isEmailValid("da$s?@a.da"));
5    ...
6  }
```

3.7 Deployen der Mobile-App

Um eine Android-App zu installieren und anwenden zu können, muss zuerst eine .apk-Datei erzeugt werden. Es gibt eine Release-Version und eine Debug-Version. In beiden Fällen wird Gradle als Buildsystem benutzt. Die App soll intern genutzt werden und von daher soll die Sicherheitseinstellung für unbekannte Quellen aktiviert werden. Siehe Abbildung unten.



4 Projektabschluss

4.1 Testdurchführung

Um die Tests durchzuführen muss ein Emulator installiert oder ein Android-Gerät angeschlossen sein. Alle Tests sind automatisiert und werden mithilfe des Espresso Frameworks durchgeführt. Diese sogenannten Instrumented Tests können mit dem Befehl **gradlew connectedAndroidTest** ausgeführt werden. Test-Ergebnisse werden mit dem Einsatz vom Gradle-Befehl produziert. Für die Tests wurde ein Emulator mit der Android-Version 7.0 („Nougat“) [19] verwendet. Auf der Konsole wurden alle Testszenarien und alle einzelnen Testfälle durchgeführt. Dort ist zu sehen welche und wie viele Tests erfolgreich bzw. fehlgeschlagen sind. Dazu werden die Tests nochmal auf einem Android-Handy mit der Version 6.0 („Marshmallow“) [20] durchgeführt. Auf der Abbildung 9 sind alle Tests zu sehen. Die Durchführung der Testszenarien verlief wie geplant. Alle 18 Tests sind automatisiert und waren auf Smartphone und Emulator 100% erfolgreich.

4.2 Soll-Ist-Vergleich

In der folgenden Tabelle wird der Verlauf des Projekts und die aufgewendeten Stunden mit der Planung verglichen. Der Teil der Projektplanung und der Dokumentation und Tests benötigten jeweils zwei Stunden mehr Zeit als gedacht, wodurch die Implementierung der App von 47 auf 43 Stunden gesunken ist. Diese Stunden wurden in der Durchführungsphase eingespart, da die Implementierung schnell und fehlerfrei verliefen. Entgegen der Planung im Projektantrag konnten die Funktionen Registrieren von Benutzern und Hinzufügen von Aufgaben (nur als Administrator) erfolgreich umgesetzt werden, da dieses den geplanten Zeitrahmen nicht überschritten hätte. Die Planung des Projektes wurde auch zeitlich länger als geplant sein, da das Soll-Konzept und die Kosten-Nutzen-Analyse mehrmals geändert wurden. Für die Dokumentation und die Tests wurde mehr Zeit genutzt, sodass die Implementierung gut beschrieben und getestet werden kann. Somit wurden die 70 Arbeitsstunden eingehalten.

Projektphase	Soll-Stunden	Ist-Stunden
Projektplanung	5	7
Projektdurchführung	47	43
Test und Dokumentation	18	20
Gesamtsumme	**70**	**70**

Tabelle 2: Soll-Ist-Vergleich

4.3 Fazit

Das Projekt „Entwicklung einer Mobile-App-Anwendung zum Annehmen von verschiedenen Aufgaben, die für die Auszubildenden im Unternehmen bereitgestellt werden“ konnte mit erhöhtem Leistungsumfang erfolgreich abgeschlossen werden. Als Resultat ist eine Android-App entstanden, mit dessen Hilfe viele Auszubildenden verschiedene Aufgaben annehmen und erledigen können.

A Glossar

.apk Die Installationsdatei für eine Android-Applikation.

Android Ein Betriebssystem für Smartphones.

Android SDK Android Entwicklerwerkzeug.

Aktivität Die Ansicht einer Android-Applikation.

Dependency Bibliotheks-Abhängigkeit für ein Framework.

Deployen Die Bereitstellung einer Android-Anwendung.

Full Service Provider Vollständige Übernahme von Geschäftsprozessen.

Framework Eine Art externer Bibliotheken.

Gradle Build-Management-Tool.

Integrationstests Eine Menge an Tests, die die Funktionalität einer Software als ganzes testen soll.

Java Eine höhere Programmiersprache
Kanban ist eine Methode, bei der die Anzahl gleichlaufender Arbeiten begrenzt und somit kürzere Durchlaufzeiten erreicht bzw. Hindernisse und Probleme schnell erkennbar gemacht werden sollen.
Linux Ein freies, unix-ähnliches Mehrbenutzer-Betriebssystem.

Mockup Ein Vorführmodell, das die Funktionen eines fertigen Produktes repräsentieren soll.

Mobile-App Eine Applikation für das Smartphone.

MS-Outlook Eine Software von Microsoft zum Verwalten von Terminen und Aufgaben, sowie zum Empfangen und Versenden von E-Mails.
Open Source Software Eine Software, deren Quelltext öffentlich zugreifbar ist und vom Dritter eingesehen und genutzt werden kann.

Splashscreen Eine Aktivität, die beim Starten einer Applikation dargestellt wird.

SQL ist eine Datenbanksprache zur Definition von Datenstrukturen in relationalen Datenbanken sowie zum Bearbeiten und Abfragen von darauf basierenden Datenbeständen.

UML-Aktivitätsdiagramm Ein Aktivitätsdiagramm stellt die Verbindung von elementaren Aktionen und deren Vernetzungen mit Kontroll- und Datenflüssengrafisch dar.

UML-Klassendiagramm Ein Klassendiagramm ist ein Strukturdiagramm der Unified Modeling Language (UML) zur grafischen Darstellung von Klassen und ihre Beziehungen.

UML-Anwendungsfalldiagramm Ein Anwendungsfalldiagramm stellt Anwendungsfälle und Akteure mit ihren jeweiligen Abhängigkeiten und Beziehungen dar.

Virtualisierung Abstraktion von physikalischen IT-Ressourcen in Form von Hardware, Software oder Netzwerken zu virtuellen Komponenten.

B Literaturverzeichnis

[1] D. AöR, "Dataport lösungen a-z." https://www.dataport.de/Seiten/L%C3%B6sungen/LC3%B6sungen-A-bis-Z.aspx. [Online, letzter Abruf: 02-November-2018].

[2] D. AöR, "Unternehmensportät." https://www.dataport.de/Seiten/Unternehmen/%C3%9Cber-uns.aspx. [Online, letzter Abruf: 02-November-2018].

[3] Google, "Android." https://www.android.com/. [Online, letzter Abruf: 11-Oktober-2018].

[4] "Android studio." https://developer.android.com/studio/index.html. [Online, letzter Abruf: 11-Oktober-2018].

[5] "Sqliteman." http://sqliteman.yarpen.cz/. [Online, letzter Abruf: 12-Oktober-2018].

[6] "Sqlite dokumentation." https://www.sqlite.org/docs.html. [Online, letzter Abruf: 17-Oktober-2018].

[7] G.Inc., "Gradle." https://gradle.org/. [Online, letzter Abruf: 13-Oktober-2018].

[8] Google, "Ui-tests mit espresso." https://developer.android.com/training/testing/espresso/index.html. [Online, letzter Abruf: 01-November-2018].

[9] "Adobe spark." https://spark.adobe.com/sp/. [Online, letzter Abruf: 11-Oktober-2018].

[10] "Background." http://bg.siteorigin.com/. [Online, letzter Abruf: 13-Oktober-2018].

[11] "Round icon." https://romannurik.github.io/AndroidAssetStudio/icons-launcher.html#foreground.type=clipart&foreground.clipart=assignment_turned_in&foreground.space.trim=1&foreground.space.pad=0.2&foreColor=rgba(161%2C%20184%2C%20194%2C%200)&backColor=rgb(148%2C%2068%2C%2068)&crop=1&backgroundShape=square&effects=shadow&name=ic_launcher. [Online, letzter Abruf: 01-November-2018].

[12] "bcrypt." https://www.mindrot.org/files/jBCrypt/jBCrypt-0.1-doc/BCrypt.html. [Online, letzter Abruf: 19-Oktober-2018].

[13] "Email validation." https://www.geeksforgeeks.org/check-email-address-valid-not-java/. [Online, letzter Abruf: 20-Oktober-2018].

[14] "Struktur einer email-adresse." https://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html. [Online, letzter Abruf: 16-Oktober-2018].

[15] "Shared preferences." https://developer.android.com/reference/android/content/SharedPreferences. [Online, letzter Abruf: 14-Oktober-2018].

[16] "Simpleuml." https://plugins.jetbrains.com/plugin/243-simpleuml. [Online, letzter Abruf: 31-Oktober-2018].

[17] "Lucidchart." https://www.lucidchart.com/. [Online, letzter Abruf: 12-Oktober-2018].

[18] "Junit tests." https://developer.android.com/training/testing/unit-testing/local-unit-tests. [Online, letzter Abruf: 28-Oktober-2018].

[19] Google, “Android 7.0 nougat.” https://www.android.com/intl/de_de/versions/nougat-7-0/. [Online, letzter Abruf: 27-Oktober-2018].

[20] Google, “Android 6.0 marshmallow.” https://www.android.com/intl/de_de/versions/marshmallow-6-0/. [Online, letzter Abruf: 27-Oktober-2018].

C Code-Anhänge

Android-Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.techgeek.kanazubi">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_icon"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_icon"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".Login"
            android:configChanges="orientation|keyboardHidden|screenSize"
            android:label="@string/app_name"
            android:screenOrientation="portrait"
            android:theme="@style/FullscreenTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".Menu"
            android:screenOrientation="portrait" />
        <activity
            android:name=".Register"
            android:screenOrientation="portrait" />
        <activity
            android:name=".List"
            android:screenOrientation="portrait" />
        <activity
            android:name=".Task"
            android:screenOrientation="portrait" />
        <activity
            android:name=".AdminPlattform"
            android:screenOrientation="portrait"/>
    </application>

</manifest>
```

Code-Anhang 1: Android-Manifest.xml

Shake.xml

```
<?xml version="1.0" encoding="utf-8"?>
<rotate xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="100"
    android:fromDegrees="-3"
    android:pivotX="50%"
    android:pivotY="50%"
    android:repeatCount="5"
```

```
    android:repeatMode="reverse"
    android:toDegrees="3" />
```

Code-Anhang 2: Shake.xml

Strings.xml

```
<resources>
    <string name="app_name">KanAzubi</string>

    <string name="iv_logo"> </string>

    <string name="title_activity_login">Login</string>
    <string name="tv_login">Anmelden</string>
    <string name="et_email">Email Adresse</string>
    <string name="et_password">Passwort</string>
    <string name="loginBtn">Anmelden</string>
    <string name="goToRegisterBtn">Registrieren</string>

    <string name="title_activity_register">Register</string>
    <string name="tv_register">Registrieren</string>
    <string name="addEmail">Email Adresse</string>
    <string name="addName">Name</string>
    <string name="addPassword">Passwort</string>
    <string name="confirmPassword">Passwort bestätigen</string>
    <string name="registerBtn">Registrieren</string>
    <string name="goToLoginBtn">Anmelden jetzt</string>

    <string name="title_activity_menu">Menü</string>
    <string name="tv_menu">Menü</string>
    <string name="idMenu">*ID*</string>
    <string name="nameMenu">*Name*</string>
    <string name="searchBtn">Suchen</string>
    <string name="logoutBtn">Abmelden</string>
    <string name="statusText">Status</string>
    <string-array name="status">
        <item>Offen</item>
        <item>In Bearbeitung</item>
        <item>Erledigt</item>
    </string-array>
    <string name="teamText">Team</string>
    <string-array name="team">
        <item>Technik</item>
        <item>Messe</item>
        <item>Marketing</item>
        <item>Software</item>
    </string-array>

    <string name="title_activity_list">Liste</string>
    <string name="emptyList">Die Liste ist leer</string>
    <string name="listResult">Liste</string>

    <string name="title_activity_task">Task</string>
    <string name="taskTitle">Titel</string>
    <string name="taskReleased">Veröffentlicht seit:..</string>
    <string name="taskDescr">Beschreibung</string>
    <string name="taskTeam">Team: Marketing</string>
    <string name="taskStatus">Status: In Bearbeitung</string>
    <string name="joinTaskBtn">Annehmen/In Bearbeitung/Erledigt</string>
```

```
53
54      <string name="title_activity_admin">Admin</string>
55      <string name="et_taskTitle">Titel</string>
56      <string name="et_taskReleased">Datum hier</string>
57      <string name="et_taskDescr">Beschreibung</string>
58      <string name="et_taskTeam">Team: ..</string>
59      <string name="et_taskStatus">Status: Offen</string>
60      <string name="addTaskBtn">Hinzufügen</string>
61
62      <string name="Tests">Tests</string>
63      <string name="successfulLogin">Erfolgreich angemeldet!</string>
64      <string name="incorrectEmail">Email ist invalid!</string>
65      <string name="incorrectPassword">Passwort ist falsch!</string>
66      <string name="emptyEmailPassword">Email/Passwort ist leer!</string>
67      <string name="notEnoughCharacters">Passwort ist falsch!</string>
68      <string name="successfulRegister">Benutzer wurde erstellt!</string>
69      <string name="registerEmailExists">Dieser Email wurde schon benutzt!</string>
70      <string name="registerInvalidEmail">Email ist nicht valid!</string>
71      <string name="registerNotMatchingPasswords">Passwörter stimmen nicht überein!</
            ↪ string>
72      <string name="registerEmptyFields">Nicht alle Informationen sind eingegeben!</
            ↪ string>
73      <string name="registerTooShortPassword">Passwort soll mindestens 6 Zeichen
            ↪ enthalten!</string>
74
75  </resources>
```

Code-Anhang 3: Strings.xml

Colors.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3      <color name="colorPrimary">#3F51B5</color>
4      <color name="colorPrimaryDark">#303F9F</color>
5      <color name="colorAccent">#FF4081</color>
6      <color name="black_overlay">#66000000</color>
7
8      <!-- added  -->
9      <color name="color_bg">#944444</color>
10     <color name="color_red">#973324</color>
11     <color name="color_darkgray">#594B4B</color>
12     <color name="color_cirlceBtn">#990909</color>
13     <color name="color_white">#fff</color>
14     <color name="color_almostwhite">#dedde5</color>
15     <color name="color_verified">#367820</color>
16
17
18 </resources>
```

Code-Anhang 4: Colors.xml

Styles.xml

```
1      <!-- Base application theme. -->
2      <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
```

```
3         <item name="android:windowActionBar">false</item>
4         <item name="android:windowNoTitle">true</item>
5         <item name="colorPrimary">@color/colorPrimary</item>
6         <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
7         <item name="colorAccent">@color/colorAccent</item>
8     </style>
```

Code-Anhang 5: Styles.xml

Rand der Informationen

```
1  <shape xmlns:android="http://schemas.android.com/apk/res/android"
2      android:shape="rectangle">
3
4
5      <corners
6          android:bottomLeftRadius="75dp"
7          android:bottomRightRadius="15dp"
8          android:topLeftRadius="15dp"
9          android:topRightRadius="75dp" />
10
11     <stroke
12         android:width="3dp"
13         android:color="@color/color_red" />
14
15 </shape>
```

Code-Anhang 6: Rand der Informationen

Rand der Listelementen

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <selector xmlns:android="http://schemas.android.com/apk/res/android">
3      <item><layer-list>
4          <item>
5              <shape>
6                  <solid
7                      android:color="@color/color_bg">
8                  </solid>
9
10                 <gradient android:angle="90" android:endColor="@color/color_bg"
                       ↪ android:startColor="@color/color_darkgray" android:type="
                       ↪ linear" />
11                 <corners android:radius="10dp" />
12                 <padding
13                     android:bottom="10dp"
14                     android:left="10dp"
15                     android:right="10dp"
16                     android:top="10dp" />
17                 <stroke
18                     android:width="5dp"
19                     android:color="@color/color_cirlceBtn"/>
20             </shape>
21         </item>
22     </layer-list></item>
23 </selector>
```

Code-Anhang 7: Rand der Listelementen

Rand der Anmelden EditTexts

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <corners
        android:bottomLeftRadius="20dp"
        android:bottomRightRadius="20dp"
        android:topLeftRadius="0dp"
        android:topRightRadius="20dp" />

    <stroke
        android:width="2dp"
        android:color="#fff" />

</shape>
```

Code-Anhang 8: Rand der Anmelden EditTexts

Button für das Anmelden

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <corners
        android:bottomLeftRadius="45dp"
        android:bottomRightRadius="45dp"
        android:topLeftRadius="45dp"
        android:topRightRadius="45dp" />

    <stroke
        android:width="5dp"
        android:color="@color/color_cirlceBtn" />

</shape>
```

Code-Anhang 9: Button für das Anmelden

Hintergrund-Bildschirm der Spinner

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item><layer-list>
        <item>
            <shape>
                <solid
                    android:color="@color/color_bg">
                </solid>

                <gradient android:angle="90" android:endColor="@color/color_bg"
                    ↪ android:startColor="@color/color_darkgray" android:type="
                    ↪ linear" />
                <corners android:radius="10dp" />
```

```
12                 <padding
13                     android:bottom="10dp"
14                     android:left="10dp"
15                     android:right="10dp"
16                     android:top="10dp" />
17                 <stroke
18                     android:width="5dp"
19                     android:color="@color/color_cirlceBtn"/>
20             </shape>
21         </item>
22         <item >
23             <bitmap android:gravity="center_horizontal|right"
24                 android:src="@drawable/baseline_expand_more_white_18dp" />
25         </item>
26     </layer-list></item>
27 </selector>
```

Code-Anhang 10: Hintergrund-Bildschirm der Spinner

activity-login.xml

```
1  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools"
3      android:id="@+id/title_activity_login"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:animateLayoutChanges="true"
7      android:background="@drawable/bg"
8      tools:context="com.example.techgeek.kanazubi.Login">
9
10     <RelativeLayout
11         android:layout_width="wrap_content"
12         android:layout_height="wrap_content"
13         android:layout_centerInParent="true"
14         android:layout_marginLeft="40dp"
15         android:layout_marginRight="40dp">
16
17         <ImageView
18             android:id="@+id/imgView_logo"
19             android:layout_width="240dp"
20             android:layout_height="90dp"
21             android:adjustViewBounds="true"
22             android:contentDescription="@+id/imgView_logo"
23             android:scaleType="fitCenter"
24             android:src="@drawable/logo_dp" />
25
26         <RelativeLayout
27             android:id="@+id/rellay1"
28             android:layout_width="wrap_content"
29             android:layout_height="wrap_content"
30             android:layout_below="@+id/imgView_logo"
31             android:visibility="gone">
32
33             <TextView
34                 android:id="@+id/tv_login"
35                 android:layout_width="wrap_content"
36                 android:layout_height="wrap_content"
37                 android:layout_marginTop="20dp"
38                 android:fontFamily="sans-serif-light"
```

```
39                  android:text="@string/tv_login"
40                  android:textColor="@color/color_white"
41                  android:textSize="30sp"
42                  android:textStyle="bold" />
43
44              <LinearLayout
45                  android:id="@+id/linlayLogin"
46                  android:layout_width="match_parent"
47                  android:layout_height="wrap_content"
48                  android:layout_below="@+id/tv_login"
49                  android:layout_marginTop="20dp"
50                  android:orientation="vertical">
51
52                  <LinearLayout
53                      android:layout_width="match_parent"
54                      android:layout_height="wrap_content"
55                      android:orientation="vertical">
56
57                      <TextView
58                          android:layout_width="wrap_content"
59                          android:layout_height="wrap_content"
60                          android:fontFamily="sans-serif-medium"
61                          android:text="@string/et_email"
62                          android:textAllCaps="true"
63                          android:textColor="@color/color_white"
64                          android:textSize="15sp" />
65
66                      <EditText
67                          android:id="@+id/et_email"
68                          android:layout_width="match_parent"
69                          android:layout_height="40dp"
70                          android:layout_marginTop="5dp"
71                          android:background="@drawable/et_loginfields"
72                          android:fontFamily="sans-serif-medium"
73                          android:maxLength="62"
74                          android:inputType="textEmailAddress"
75                          android:labelFor="@+id/et_email"
76                          android:paddingLeft="15dp"
77                          android:paddingRight="15dp"
78                          android:textColor="@color/color_white"
79                          android:textSize="15sp" />
80
81                  </LinearLayout>
82
83                  <LinearLayout
84                      android:layout_width="match_parent"
85                      android:layout_height="wrap_content"
86                      android:layout_marginTop="10dp"
87                      android:orientation="vertical">
88
89                      <TextView
90                          android:layout_width="wrap_content"
91                          android:layout_height="wrap_content"
92                          android:fontFamily="sans-serif-medium"
93                          android:text="@string/et_password"
94                          android:textAllCaps="true"
95                          android:textColor="@color/color_white"
96                          android:textSize="15sp" />
97
98                      <EditText
99                          android:id="@+id/et_password"
```

```
100                         android:layout_width="match_parent"
101                         android:layout_height="40dp"
102                         android:layout_marginTop="5dp"
103                         android:background="@drawable/et_loginfields"
104                         android:fontFamily="sans-serif-medium"
105                         android:inputType="textPassword"
106                         android:labelFor="@+id/et_password"
107                         android:maxLength="30"
108                         android:paddingLeft="15dp"
109                         android:paddingRight="15dp"
110                         android:textColor="@color/color_white"
111                         android:textSize="15sp" />
112
113                 </LinearLayout>
114
115             </LinearLayout>
116
117             <Button
118                 android:id="@+id/loginBtn"
119                 android:layout_width="wrap_content"
120                 android:layout_height="wrap_content"
121                 android:layout_below="@+id/linlayLogin"
122                 android:layout_centerHorizontal="true"
123                 android:layout_gravity="center"
124                 android:layout_marginTop="24dp"
125                 android:background="@drawable/signin_btn"
126                 android:fontFamily="sans-serif-light"
127                 android:paddingBottom="7dp"
128                 android:paddingLeft="30dp"
129                 android:paddingRight="30dp"
130                 android:paddingTop="7dp"
131                 android:text="@string/tv_login"
132                 android:textColor="@color/color_white"
133                 android:textSize="16sp"
134                 android:textStyle="bold" />
135
136             <Button
137                 android:id="@+id/goToRegisterBtn"
138                 android:layout_width="wrap_content"
139                 android:layout_height="wrap_content"
140                 android:layout_below="@+id/loginBtn"
141                 android:layout_centerHorizontal="true"
142                 android:layout_gravity="center"
143                 android:layout_marginTop="18dp"
144                 android:background="@drawable/signin_btn"
145                 android:fontFamily="sans-serif-light"
146                 android:paddingBottom="7dp"
147                 android:paddingLeft="30dp"
148                 android:paddingRight="30dp"
149                 android:paddingTop="7dp"
150                 android:text="@string/goToRegisterBtn"
151                 android:textColor="@color/color_white"
152                 android:textSize="16sp"
153                 android:textStyle="bold" />
154
155         </RelativeLayout>
156
157     </RelativeLayout>
158
159     <RelativeLayout
160         android:id="@+id/rellay2"
```

```
161         android:layout_width="match_parent"
162         android:layout_height="wrap_content"
163         android:layout_alignParentBottom="true"
164         android:layout_marginLeft="20dp"
165         android:layout_marginRight="20dp"
166         android:visibility="gone">
167
168     </RelativeLayout>
169
170 </RelativeLayout>
```

Code-Anhang 11: activity-login.xml

activity-register.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:background="@drawable/bg"
7      tools:context="com.example.techgeek.kanazubi.Register">
8
9      <RelativeLayout
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:layout_centerInParent="true"
13         android:layout_marginLeft="40dp"
14         android:layout_marginRight="40dp">
15
16         <ImageView
17             android:id="@+id/imgView_logo"
18             android:layout_width="240dp"
19             android:layout_height="90dp"
20             android:adjustViewBounds="true"
21             android:contentDescription="@+id/imgView_logo"
22             android:scaleType="fitCenter"
23             android:src="@drawable/logo_dp" />
24
25         <RelativeLayout
26             android:id="@+id/rellayReg1"
27             android:layout_width="wrap_content"
28             android:layout_height="wrap_content"
29             android:layout_below="@+id/imgView_logo"
30             android:visibility="gone">
31
32             <TextView
33                 android:id="@+id/tv_register"
34                 android:layout_width="wrap_content"
35                 android:layout_height="wrap_content"
36                 android:layout_marginTop="10dp"
37                 android:fontFamily="sans-serif-light"
38                 android:text="@string/tv_register"
39                 android:textColor="@color/color_white"
40                 android:textSize="30sp"
41                 android:textStyle="bold" />
42
43             <LinearLayout
44                 android:id="@+id/linlayLogin"
```

```
45                  android:layout_width="match_parent"
46                  android:layout_height="wrap_content"
47                  android:layout_below="@+id/tv_register"
48                  android:layout_marginTop="7dp"
49                  android:orientation="vertical">
50
51                  <LinearLayout
52                      android:layout_width="match_parent"
53                      android:layout_height="wrap_content"
54                      android:orientation="vertical">
55
56                      <TextView
57                          android:layout_width="wrap_content"
58                          android:layout_height="wrap_content"
59                          android:layout_marginTop="10dp"
60                          android:fontFamily="sans-serif-medium"
61                          android:text="@string/addEmail"
62                          android:textAllCaps="true"
63                          android:textColor="@color/color_white"
64                          android:textSize="15sp" />
65
66                      <EditText
67                          android:id="@+id/addEmail"
68                          android:layout_width="match_parent"
69                          android:layout_height="40dp"
70                          android:layout_marginTop="3dp"
71                          android:background="@drawable/et_loginfields"
72                          android:fontFamily="sans-serif-medium"
73                          android:maxLength="62"
74                          android:inputType="textEmailAddress"
75                          android:labelFor="@+id/addEmail"
76                          android:paddingLeft="15dp"
77                          android:paddingRight="15dp"
78                          android:textColor="@color/color_white"
79                          android:textSize="15sp"
80                          tools:ignore="LabelFor" />
81
82                  </LinearLayout>
83
84                  <LinearLayout
85                      android:layout_width="match_parent"
86                      android:layout_height="wrap_content"
87                      android:layout_marginTop="7dp"
88                      android:orientation="vertical">
89
90                      <TextView
91                          android:layout_width="wrap_content"
92                          android:layout_height="wrap_content"
93                          android:fontFamily="sans-serif-medium"
94                          android:text="@string/addName"
95                          android:textAllCaps="true"
96                          android:textColor="@color/color_white"
97                          android:textSize="15sp" />
98
99                      <EditText
100                         android:id="@+id/addName"
101                         android:layout_width="match_parent"
102                         android:layout_height="40dp"
103                         android:layout_marginTop="3dp"
104                         android:background="@drawable/et_loginfields"
105                         android:fontFamily="sans-serif-medium"
```

```
106                         android:maxLength="30"
107                         android:inputType="text"
108                         android:labelFor="@+id/addName"
109                         android:paddingLeft="15dp"
110                         android:paddingRight="15dp"
111                         android:textColor="@color/color_white"
112                         android:textSize="15sp"
113                         tools:ignore="LabelFor" />
114
115                 </LinearLayout>
116
117                 <LinearLayout
118                     android:layout_width="match_parent"
119                     android:layout_height="wrap_content"
120                     android:layout_marginTop="7dp"
121                     android:orientation="vertical">
122
123                     <TextView
124                         android:layout_width="wrap_content"
125                         android:layout_height="wrap_content"
126                         android:fontFamily="sans-serif-medium"
127                         android:text="@string/addPassword"
128                         android:textAllCaps="true"
129                         android:textColor="@color/color_white"
130                         android:textSize="15sp" />
131
132                     <EditText
133                         android:id="@+id/addPassword"
134                         android:layout_width="match_parent"
135                         android:layout_height="40dp"
136                         android:layout_marginTop="3dp"
137                         android:background="@drawable/et_loginfields"
138                         android:fontFamily="sans-serif-medium"
139                         android:maxLength="30"
140                         android:inputType="textPassword"
141                         android:labelFor="@+id/addPassword"
142                         android:paddingLeft="15dp"
143                         android:paddingRight="15dp"
144                         android:textColor="@color/color_white"
145                         android:textSize="15sp"
146                         tools:ignore="LabelFor" />
147
148                 </LinearLayout>
149
150                 <LinearLayout
151                     android:layout_width="match_parent"
152                     android:layout_height="wrap_content"
153                     android:layout_marginTop="7dp"
154                     android:orientation="vertical">
155
156                     <TextView
157                         android:layout_width="wrap_content"
158                         android:layout_height="wrap_content"
159                         android:fontFamily="sans-serif-medium"
160                         android:text="@string/confirmPassword"
161                         android:textAllCaps="true"
162                         android:textColor="@color/color_white"
163                         android:textSize="15sp" />
164
165                     <EditText
166                         android:id="@+id/confirmPassword"
```

```
167                         android:layout_width="match_parent"
168                         android:layout_height="40dp"
169                         android:layout_marginTop="3dp"
170                         android:background="@drawable/et_loginfields"
171                         android:fontFamily="sans-serif-medium"
172                         android:maxLength="30"
173                         android:inputType="textPassword"
174                         android:labelFor="@+id/confirmPassword"
175                         android:paddingLeft="15dp"
176                         android:paddingRight="15dp"
177                         android:textColor="@color/color_white"
178                         android:textSize="15sp"
179                         tools:ignore="LabelFor" />
180
181                 </LinearLayout>
182
183             </LinearLayout>
184
185             <Button
186                 android:id="@+id/registerBtn"
187                 android:layout_width="wrap_content"
188                 android:layout_height="wrap_content"
189                 android:layout_below="@+id/linlayLogin"
190                 android:layout_centerHorizontal="true"
191                 android:layout_gravity="center"
192                 android:layout_marginTop="24dp"
193                 android:background="@drawable/signin_btn"
194                 android:fontFamily="sans-serif-light"
195                 android:paddingBottom="7dp"
196                 android:paddingLeft="30dp"
197                 android:paddingRight="30dp"
198                 android:paddingTop="7dp"
199                 android:text="@string/registerBtn"
200                 android:textColor="@color/color_white"
201                 android:textSize="16sp"
202                 android:textStyle="bold" />
203
204             <Button
205                 android:id="@+id/goToLoginBtn"
206                 android:layout_width="wrap_content"
207                 android:layout_height="wrap_content"
208                 android:layout_below="@+id/registerBtn"
209                 android:layout_centerHorizontal="true"
210                 android:layout_gravity="center"
211                 android:layout_marginTop="18dp"
212                 android:background="@drawable/signin_btn"
213                 android:fontFamily="sans-serif-light"
214                 android:paddingBottom="7dp"
215                 android:paddingLeft="30dp"
216                 android:paddingRight="30dp"
217                 android:paddingTop="7dp"
218                 android:text="@string/goToLoginBtn"
219                 android:textColor="@color/color_white"
220                 android:textSize="16sp"
221                 android:textStyle="bold" />
222
223         </RelativeLayout>
224
225     </RelativeLayout>
226
227     <RelativeLayout
```

```
228         android:id="@+id/rellayReg2"
229         android:layout_width="match_parent"
230         android:layout_height="wrap_content"
231         android:layout_alignParentBottom="true"
232         android:layout_marginLeft="20dp"
233         android:layout_marginRight="20dp"
234         android:visibility="gone">
235
236     </RelativeLayout>
237
238 </RelativeLayout>
```

Code-Anhang 12: activity-register.xml

activity-menu.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:background="@drawable/bg"
7      tools:context="com.example.techgeek.kanazubi.Menu">
8
9      <RelativeLayout
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:layout_centerInParent="true"
13         android:layout_marginTop="10dp"
14         android:layout_marginBottom="10dp"
15         android:layout_marginLeft="10dp"
16         android:layout_marginRight="10dp">
17
18         <ImageView
19             android:id="@+id/imgView_logo"
20             android:layout_marginTop="10dp"
21             android:layout_width="240dp"
22             android:layout_height="75dp"
23             android:layout_centerHorizontal="true"
24             android:adjustViewBounds="true"
25             android:contentDescription="@+id/imgView_logo"
26             android:scaleType="fitCenter"
27             android:src="@drawable/logo_dp" />
28
29         <RelativeLayout
30             android:id="@+id/rellay1"
31             android:layout_width="wrap_content"
32             android:layout_height="wrap_content"
33             android:layout_below="@+id/imgView_logo"
34             android:layout_marginTop="15dp">
35             android:visibility="gone">
36
37             <LinearLayout
38                 android:id="@+id/linlay1"
39                 android:layout_width="match_parent"
40                 android:layout_height="70dp"
41                 android:orientation="horizontal"
42                 android:layout_marginTop="5dp"
43                 android:layout_marginBottom="5dp">
```

```
44
45                  <RelativeLayout
46                      android:id="@+id/rellay11"
47                      android:layout_width="match_parent"
48                      android:layout_height="match_parent"
49                      tools:ignore="UselessParent">
50
51                      <TextView
52                          android:id="@+id/tvName"
53                          android:layout_width="180dp"
54                          android:layout_height="50dp"
55                          android:maxHeight="30dp"
56                          android:layout_marginStart="10dp"
57                          android:textAlignment="center"
58                          android:layout_alignParentStart="true"
59                          android:layout_gravity="center_vertical"
60                          android:gravity="center"
61                          android:text="@string/nameMenu"
62                          android:textSize="19sp"
63                          android:maxLength="30"
64                          android:maxLines="2"
65                          android:textColor="@color/color_almostwhite"/>
66
67                      <Button
68                          android:id="@+id/logoutBtn"
69                          android:layout_width="wrap_content"
70                          android:layout_height="wrap_content"
71                          android:layout_gravity="center_vertical"
72                          android:layout_alignParentEnd="true"
73                          android:background="@drawable/signin_btn"
74                          android:fontFamily="sans-serif-light"
75                          android:paddingLeft="15dp"
76                          android:paddingRight="15dp"
77                          android:text="@string/logoutBtn"
78                          android:textColor="@color/color_white"
79                          android:textSize="16sp"
80                          android:textStyle="bold"
81                          tools:ignore="RelativeOverlap" />
82
83                  </RelativeLayout>
84
85              </LinearLayout>
86
87              <LinearLayout
88                  android:layout_width="match_parent"
89                  android:layout_height="match_parent"
90                  android:orientation="vertical"
91                  android:layout_below="@+id/linlay1">
92
93                  <RelativeLayout
94                      android:id="@+id/rellay22"
95                      android:layout_width="match_parent"
96                      android:layout_height="match_parent"
97                      tools:ignore="UselessParent">
98
99                      <TextView
100                         android:id="@+id/tv_menu"
101                         android:layout_width="wrap_content"
102                         android:layout_height="wrap_content"
103                         android:layout_centerHorizontal="true"
104                         android:layout_gravity="center_horizontal"
```

```
105                         android:text="@string/tv_menu"
106                         android:fontFamily="sans-serif-light"
107                         android:textColor="@color/color_white"
108                         android:textSize="27sp"
109                         android:textStyle="bold" />
110
111                     <TextView
112                         android:id="@+id/tv_status"
113                         android:text="@string/statusText"
114                         android:layout_width="wrap_content"
115                         android:layout_height="wrap_content"
116                         android:layout_marginStart="25dp"
117                         android:layout_marginTop="65dp"
118                         android:layout_alignParentStart="true"
119                         android:layout_below="@id/tv_menu"
120                         android:textSize="21sp"
121                         android:textColor="@color/color_almostwhite"/>
122
123                     <Spinner
124                         android:id="@+id/spinnerStatus"
125                         android:layout_width="fill_parent"
126                         android:layout_height="wrap_content"
127                         android:background="@drawable/spinner_bg"
128                         android:layout_marginStart="50dp"
129                         android:layout_alignBaseline="@id/tv_status"
130                         android:layout_toEndOf="@id/tv_status"
131                         android:drawSelectorOnTop="true"
132                         android:layout_below="@id/tv_status">
133
134                     </Spinner>
135
136                     <TextView
137                         android:id="@+id/tv_team"
138                         android:text="@string/teamText"
139                         android:layout_width="wrap_content"
140                         android:layout_height="wrap_content"
141                         android:layout_marginStart="25dp"
142                         android:layout_marginTop="35dp"
143                         android:layout_alignParentStart="true"
144                         android:layout_below="@id/spinnerStatus"
145                         android:textSize="21sp"
146                         android:textColor="@color/color_almostwhite"/>
147
148                     <Spinner
149                         android:id="@+id/spinnerTeam"
150                         android:layout_width="fill_parent"
151                         android:layout_height="wrap_content"
152                         android:background="@drawable/spinner_bg"
153                         android:layout_marginStart="50dp"
154                         android:layout_alignBaseline="@id/tv_team"
155                         android:layout_toEndOf="@id/tv_status"
156                         android:drawSelectorOnTop="true"
157                         android:layout_below="@id/tv_team">
158
159                     </Spinner>
160
161                     <Button
162                         android:id="@+id/searchBtn"
163                         android:layout_width="wrap_content"
164                         android:layout_height="wrap_content"
165                         android:layout_centerHorizontal="true"
```

```
                        android:layout_gravity="center_horizontal"
                        android:layout_marginTop="55dp"
                        android:layout_below="@id/spinnerTeam"
                        android:background="@drawable/signin_btn"
                        android:fontFamily="sans-serif-light"
                        android:paddingLeft="20dp"
                        android:paddingRight="20dp"
                        android:text="@string/searchBtn"
                        android:textColor="@color/color_white"
                        android:textSize="16sp"
                        android:textStyle="bold" />

                </RelativeLayout>

            </LinearLayout>

        </RelativeLayout>

    </RelativeLayout>

    <RelativeLayout
        android:id="@+id/rellay2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp"
        android:visibility="gone">

    </RelativeLayout>

</RelativeLayout>
```

Code-Anhang 13: activity-menu.xml

activity-admin-plattform.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bg"
    tools:context="com.example.techgeek.kanazubi.AdminPlattform">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_centerInParent="true"
        android:layout_marginBottom="5dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:layout_marginTop="10dp"
        tools:ignore="UselessParent">

        <ImageView
            android:id="@+id/imgView_logo"
            android:layout_width="240dp"
            android:layout_height="75dp"
```

```
23              android:layout_centerHorizontal="true"
24              android:layout_marginTop="10dp"
25              android:adjustViewBounds="true"
26              android:contentDescription="@+id/imgView_logo"
27              android:scaleType="fitCenter"
28              android:src="@drawable/logo_dp" />
29
30          <RelativeLayout
31              android:id="@+id/rellay1"
32              android:layout_width="match_parent"
33              android:layout_height="match_parent"
34              android:layout_below="@+id/imgView_logo"
35              android:layout_marginBottom="60dp"
36              android:layout_marginTop="15dp">
37
38              <EditText
39                  android:id="@+id/et_taskTitle"
40                  android:layout_width="match_parent"
41                  android:layout_height="40dp"
42                  android:background="@drawable/boarder_info_box"
43                  android:textAlignment="center"
44                  android:maxLength="60"
45                  android:maxLines="1"
46                  android:paddingEnd="4dp"
47                  android:paddingStart="4dp"
48                  android:text="@string/et_taskTitle"
49                  android:textColor="@color/color_almostwhite"
50                  android:textSize="16sp"
51                  android:textStyle="bold" />
52
53              <EditText
54                  android:id="@+id/et_taskReleased"
55                  android:layout_width="wrap_content"
56                  android:layout_height="40dp"
57                  android:layout_below="@+id/et_taskTitle"
58                  android:layout_centerInParent="true"
59                  android:layout_marginTop="7dp"
60                  android:background="@drawable/boarder_info_box"
61                  android:gravity="center"
62                  android:maxLength="10"
63                  android:paddingEnd="45dp"
64                  android:paddingStart="45dp"
65                  android:text="@string/et_taskReleased"
66                  android:textAlignment="center"
67                  android:textColor="@color/color_almostwhite"
68                  android:textSize="16sp"
69                  android:textStyle="bold" />
70
71              <RelativeLayout
72                  android:id="@+id/rellay"
73                  android:layout_width="match_parent"
74                  android:layout_height="wrap_content"
75                  android:layout_below="@id/et_taskReleased"
76                  android:layout_marginTop="7dp">
77
78                  <EditText
79                      android:id="@+id/et_taskTeam"
80                      android:layout_width="wrap_content"
81                      android:layout_height="50dp"
82                      android:layout_marginEnd="5dp"
83                      android:background="@drawable/boarder_info_box"
```

```
84                      android:gravity="center"
85                      android:maxWidth="150dp"
86                      android:paddingEnd="25dp"
87                      android:paddingStart="25dp"
88                      android:maxLength="10"
89                      android:text="@string/et_taskTeam"
90                      android:textAlignment="center"
91                      android:textColor="@color/color_almostwhite"
92                      android:textSize="16sp"
93                      android:textStyle="bold" />
94
95                  <TextView
96                      android:id="@+id/et_taskStatus"
97                      android:layout_width="wrap_content"
98                      android:layout_height="50dp"
99                      android:layout_alignParentEnd="true"
100                     android:layout_toStartOf="@id/et_taskTeam"
101                     android:background="@drawable/boarder_info_box"
102                     android:gravity="center"
103                     android:maxWidth="150dp"
104                     android:paddingEnd="25dp"
105                     android:paddingStart="25dp"
106                     android:text="@string/et_taskStatus"
107                     android:textAlignment="center"
108                     android:textColor="@color/color_almostwhite"
109                     android:textSize="16sp"
110                     android:textStyle="bold" />
111
112             </RelativeLayout>
113
114             <EditText
115                 android:id="@+id/et_taskDescr"
116                 android:layout_width="match_parent"
117                 android:layout_height="match_parent"
118                 android:layout_below="@+id/rellay"
119                 android:layout_marginTop="10dp"
120                 android:background="@drawable/boarder_info_box"
121                 android:paddingBottom="22dp"
122                 android:paddingEnd="27dp"
123                 android:paddingStart="27dp"
124                 android:paddingTop="18dp"
125                 android:scrollbars="vertical"
126                 android:text="@string/et_taskDescr"
127                 android:textAlignment="viewStart"
128                 android:textColor="@color/color_almostwhite"
129                 android:textSize="16sp" />
130
131         </RelativeLayout>
132
133         <Button
134             android:id="@+id/addTaskBtn"
135             android:layout_width="90dp"
136             android:layout_height="wrap_content"
137             android:layout_alignParentBottom="true"
138             android:layout_centerHorizontal="true"
139             android:layout_marginBottom="5dp"
140             android:background="@drawable/signin_btn"
141             android:gravity="center"
142             android:text="@string/addTaskBtn"
143             android:textColor="@color/color_almostwhite" />
144
```

```
145     </RelativeLayout>
146
147 </RelativeLayout>
```

Code-Anhang 14: activity-admin-plattform.xml

activity-list.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:background="@drawable/bg"
7      tools:context="com.example.techgeek.kanazubi.List">
8
9
10     <RelativeLayout
11         android:layout_width="match_parent"
12         android:layout_height="match_parent"
13         android:layout_centerInParent="true"
14         android:layout_marginBottom="10dp"
15         android:layout_marginLeft="10dp"
16         android:layout_marginRight="10dp"
17         android:layout_marginTop="10dp"
18         tools:ignore="UselessParent">
19
20         <ImageView
21             android:id="@+id/imgView_logo"
22             android:layout_width="240dp"
23             android:layout_height="75dp"
24             android:layout_centerHorizontal="true"
25             android:layout_marginTop="10dp"
26             android:adjustViewBounds="true"
27             android:contentDescription="@+id/imgView_logo"
28             android:scaleType="fitCenter"
29             android:src="@drawable/logo_dp" />
30
31         <RelativeLayout
32             android:id="@+id/rellay1"
33             android:layout_width="wrap_content"
34             android:layout_height="wrap_content"
35             android:layout_below="@+id/imgView_logo"
36             android:layout_marginTop="5dp"
37             android:layout_marginBottom="5dp"
38             android:paddingTop="10dp"
39             android:paddingBottom="10dp">
40
41             <ListView
42                 android:id="@+id/listResult"
43                 android:layout_width="match_parent"
44                 android:layout_height="wrap_content">
45
46             </ListView>
47
48         </RelativeLayout>
49
50         <TextView
51             android:id="@+id/emptyList"
```

```
52              android:layout_width="match_parent"
53              android:layout_height="60dp"
54              android:layout_marginTop="45dp"
55              android:layout_below="@id/imgView_logo"
56              android:textAlignment="center"
57              android:text="@string/emptyList"
58              android:textSize="24sp"
59              android:textStyle="bold"
60              android:textColor="@color/color_almostwhite"/>
61
62      </RelativeLayout>
63
64  </RelativeLayout>
```

Code-Anhang 15: activity-list.xml

activity-task.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:background="@drawable/bg"
7      tools:context="com.example.techgeek.kanazubi.Task">
8
9      <RelativeLayout
10         android:layout_width="match_parent"
11         android:layout_height="match_parent"
12         android:layout_centerInParent="true"
13         android:layout_marginBottom="5dp"
14         android:layout_marginLeft="10dp"
15         android:layout_marginRight="10dp"
16         android:layout_marginTop="10dp"
17         tools:ignore="UselessParent">
18
19         <ImageView
20             android:id="@+id/imgView_logo"
21             android:layout_width="240dp"
22             android:layout_height="75dp"
23             android:layout_centerHorizontal="true"
24             android:layout_marginTop="10dp"
25             android:adjustViewBounds="true"
26             android:contentDescription="@+id/imgView_logo"
27             android:scaleType="fitCenter"
28             android:src="@drawable/logo_dp" />
29
30         <RelativeLayout
31             android:id="@+id/rellay1"
32             android:layout_width="match_parent"
33             android:layout_height="match_parent"
34             android:layout_marginBottom="60dp"
35             android:layout_below="@+id/imgView_logo"
36             android:layout_marginTop="15dp">
37
38             <HorizontalScrollView
39                 android:id="@+id/hsv"
40                 android:layout_width="match_parent"
41                 android:layout_height="45dp"
```

```
42                  android:layout_marginTop="7dp"
43                  android:layout_marginStart="10dp"
44                  android:layout_marginEnd="10dp">
45                  <TextView
46                      android:id="@+id/taskTitle"
47                      android:layout_width="wrap_content"
48                      android:layout_height="40dp"
49                      android:text="@string/taskTitle"
50                      android:maxLines="1"
51                      android:ellipsize="end"
52                      android:maxLength="60"
53                      android:maxEms="36"
54                      android:background="@drawable/boarder_info_box"
55                      android:paddingStart="12dp"
56                      android:paddingEnd="12dp"
57                      android:textSize="16sp"
58                      android:textStyle="bold"
59                      android:textColor="@color/color_almostwhite"
60                      android:textAlignment="center"
61                      android:gravity="center"/>
62
63              </HorizontalScrollView>
64
65              <TextView
66                  android:id="@+id/taskReleased"
67                  android:layout_width="wrap_content"
68                  android:layout_height="40dp"
69                  android:layout_marginTop="7dp"
70                  android:layout_centerInParent="true"
71                  android:layout_below="@+id/hsv"
72                  android:paddingStart="45dp"
73                  android:paddingEnd="45dp"
74                  android:textColor="@color/color_almostwhite"
75                  android:text="@string/taskReleased"
76                  android:textSize="16sp"
77                  android:textStyle="bold"
78                  android:textAlignment="center"
79                  android:gravity="center"
80                  android:background="@drawable/boarder_info_box"/>
81
82              <RelativeLayout
83                  android:id="@+id/rellay"
84                  android:layout_width="match_parent"
85                  android:layout_height="wrap_content"
86                  android:layout_marginTop="7dp"
87                  android:layout_below="@id/taskReleased">
88
89                  <TextView
90                      android:id="@+id/taskTeam"
91                      android:layout_width="wrap_content"
92                      android:layout_height="50dp"
93                      android:paddingStart="25dp"
94                      android:paddingEnd="25dp"
95                      android:maxWidth="150dp"
96                      android:gravity="center"
97                      android:textColor="@color/color_almostwhite"
98                      android:textAlignment="center"
99                      android:textSize="16sp"
100                     android:textStyle="bold"
101                     android:layout_marginEnd="5dp"
102                     android:text="@string/taskTeam"
```

```
                    android:background="@drawable/boarder_info_box"/>

                <TextView
                    android:id="@+id/taskStatus"
                    android:layout_width="wrap_content"
                    android:layout_height="50dp"
                    android:maxWidth="150dp"
                    android:paddingStart="25dp"
                    android:paddingEnd="25dp"
                    android:textColor="@color/color_almostwhite"
                    android:textAlignment="center"
                    android:gravity="center"
                    android:text="@string/taskStatus"
                    android:layout_toStartOf="@id/taskTeam"
                    android:textSize="16sp"
                    android:textStyle="bold"
                    android:layout_alignParentEnd="true"
                    android:background="@drawable/boarder_info_box"/>

            </RelativeLayout>

            <TextView
                android:id="@+id/taskDescr"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_below="@+id/rellay"
                android:layout_marginTop="10dp"
                android:paddingStart="27dp"
                android:paddingTop="18dp"
                android:paddingBottom="22dp"
                android:paddingEnd="27dp"
                android:textColor="@color/color_almostwhite"
                android:text="@string/taskDescr"
                android:textSize="16sp"
                android:textAlignment="viewStart"
                android:scrollbars = "vertical"
                android:background="@drawable/boarder_info_box"/>

        </RelativeLayout>

        <Button
            android:id="@+id/joinTaskBtn"
            android:layout_width="160dp"
            android:layout_height="wrap_content"
            android:layout_alignParentBottom="true"
            android:layout_centerHorizontal="true"
            android:layout_marginBottom="5dp"
            android:textColor="@color/color_almostwhite"
            android:background="@drawable/signin_btn"
            android:gravity="center"
            android:text="@string/joinTaskBtn" />

    </RelativeLayout>

</RelativeLayout>
```

Code-Anhang 16: activity-task.xml

spinner-items.xml

```xml
<?xml version="1.0" encoding="utf-8"?>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:singleLine="true"
    android:ellipsize="end"
    android:textColor="@color/color_almostwhite"
    android:textSize="19sp" />
```

Code-Anhang 17: spinner-items.xml

LoginTests.java

```java
package com.example.techgeek.kanazubi;

import android.support.test.espresso.action.ViewActions;
import android.support.test.espresso.matcher.ViewMatchers;
import android.support.test.rule.ActivityTestRule;
import android.view.View;

import org.junit.After;
import org.junit.Before;
import org.junit.Rule;
import org.junit.Test;

import static android.support.test.espresso.Espresso.closeSoftKeyboard;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.clearText;
import static android.support.test.espresso.action.ViewActions.typeText;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.RootMatchers.withDecorView;
import static android.support.test.espresso.matcher.ViewMatchers.isDisplayed;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
import static org.hamcrest.Matchers.is;
import static org.hamcrest.Matchers.not;
import static org.junit.Assert.assertNotNull;

public class LoginTest {

    @Rule
    public ActivityTestRule<Login> mLoginTestRule = new ActivityTestRule<>(Login.
        ↪ class);
    private Login loginActivity = null;

    @Before
    public void setUp() throws Exception {
        loginActivity = mLoginTestRule.getActivity();
    }

    @Test
    public void loginAndMore() {

        //check if all EditTexts or Buttons are not null
        View view1 = loginActivity.findViewById(R.id.et_email);
```

```
42          assertNotNull(view1);
43          View view2 = loginActivity.findViewById(R.id.et_password);
44          assertNotNull(view2);
45          View view3 = loginActivity.findViewById(R.id.loginBtn);
46          assertNotNull(view3);
47          View view4 = loginActivity.findViewById(R.id.goToRegisterBtn);
48          assertNotNull(view4);
49
50
51          try {
52              Thread.sleep(1800);
53          } catch (InterruptedException e) {
54              e.printStackTrace();
55          }
56
57          onView(withId(R.id.et_email)).perform(typeText("admin@dataport.de"));
58          closeSoftKeyboard();
59          onView(withId(R.id.et_password)).perform(typeText("123456"));
60          closeSoftKeyboard();
61          onView(withId(R.id.loginBtn)).perform(ViewActions.click());
62          onView(withText(R.string.successfulLogin)).inRoot(withDecorView(not(is(
                ↪ loginActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
63          onView(withId(R.id.logoutBtn)).perform(ViewActions.click());
64      }
65
66      @Test
67      public void enterIncorrectEmail() {
68
69          try {
70              Thread.sleep(1800);
71          } catch (InterruptedException e) {
72              e.printStackTrace();
73          }
74
75          onView(withId(R.id.et_email)).perform(typeText("adeqeacdde"));
76          closeSoftKeyboard();
77          onView(withId(R.id.et_password)).perform(typeText("123456"));
78          closeSoftKeyboard();
79          onView(withId(R.id.loginBtn)).perform(ViewActions.click());
80          onView(withText(R.string.incorrectEmail)).inRoot(withDecorView(not(is(
                ↪ loginActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
81      }
82
83      @Test
84      public void enterIncorrectPassword() {
85
86          try {
87              Thread.sleep(1800);
88          } catch (InterruptedException e) {
89              e.printStackTrace();
90          }
91
92          onView(withId(R.id.et_email)).perform(typeText("admin@dataport.de"));
93          closeSoftKeyboard();
94          onView(withId(R.id.et_password)).perform(typeText("12345"));
95          closeSoftKeyboard();
96          onView(withId(R.id.loginBtn)).perform(ViewActions.click());
97          onView(withText(R.string.incorrectPassword)).inRoot(withDecorView(not(is(
                ↪ loginActivity.getWindow().getDecorView())))).check(matches(
```

```
            ↪ isDisplayed()));
98      }
99
100     @Test
101     public void enterEmptyEmailOrPassword() {
102
103         try {
104             Thread.sleep(1800);
105         } catch (InterruptedException e) {
106             e.printStackTrace();
107         }
108
109         onView(withId(R.id.et_email)).perform(typeText("admin@dataport.de"));
110         closeSoftKeyboard();
111         onView(withId(R.id.et_password)).perform(typeText(""));
112         closeSoftKeyboard();
113         onView(withId(R.id.loginBtn)).perform(ViewActions.click());
114         onView(withText(R.string.emptyEmailPassword)).inRoot(withDecorView(not(is(
            ↪ loginActivity.getWindow().getDecorView())))).check(matches(
            ↪ isDisplayed()));
115         onView(withId(R.id.et_email)).perform(clearText());
116
117         onView(withId(R.id.et_email)).perform(typeText(""));
118         closeSoftKeyboard();
119         onView(withId(R.id.et_password)).perform(typeText("123456"));
120         closeSoftKeyboard();
121         onView(ViewMatchers.withId(R.id.loginBtn)).perform(ViewActions.click());
122         onView(withText(R.string.incorrectEmail)).inRoot(withDecorView(not(is(
            ↪ loginActivity.getWindow().getDecorView())))).check(matches(
            ↪ isDisplayed()));
123         onView(withId(R.id.et_password)).perform(clearText());
124
125         onView(withId(R.id.et_email)).perform(typeText(""));
126         closeSoftKeyboard();
127         onView(withId(R.id.et_password)).perform(typeText(""));
128         closeSoftKeyboard();
129         onView(ViewMatchers.withId(R.id.loginBtn)).perform(ViewActions.click());
130         onView(withText(R.string.incorrectEmail)).inRoot(withDecorView(not(is(
            ↪ loginActivity.getWindow().getDecorView())))).check(matches(
            ↪ isDisplayed()));
131     }
132
133     @Test
134     public void goToRegistrationAndBack() {
135         try {
136             Thread.sleep(1800);
137         } catch (InterruptedException e) {
138             e.printStackTrace();
139         }
140         onView(withId(R.id.goToRegisterBtn)).perform(ViewActions.click());
141
142         try {
143             Thread.sleep(1800);
144         } catch (InterruptedException e) {
145             e.printStackTrace();
146         }
147         onView(withId(R.id.goToLoginBtn)).perform(ViewActions.click());
148
149         try {
150             Thread.sleep(1800);
151         } catch (InterruptedException e) {
```

```
152             e.printStackTrace();
153         }
154         onView(withId(R.id.loginBtn)).perform(ViewActions.click());
155         onView(withText(R.string.incorrectEmail)).inRoot(withDecorView(not(is(
                ↪ loginActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
156     }
157
158     @After
159     public void tearDown() throws Exception {
160         loginActivity = null;
161     }
162 }
```

Code-Anhang 18: LoginTests.java

RegisterTests.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.support.test.espresso.action.ViewActions;
4  import android.support.test.rule.ActivityTestRule;
5  import android.view.View;
6
7  import org.junit.After;
8  import org.junit.Before;
9  import org.junit.Rule;
10 import org.junit.Test;
11
12 import static android.support.test.espresso.Espresso.closeSoftKeyboard;
13 import static android.support.test.espresso.Espresso.onView;
14 import static android.support.test.espresso.action.ViewActions.clearText;
15 import static android.support.test.espresso.action.ViewActions.typeText;
16 import static android.support.test.espresso.assertion.ViewAssertions.matches;
17 import static android.support.test.espresso.matcher.RootMatchers.withDecorView;
18 import static android.support.test.espresso.matcher.ViewMatchers.isDisplayed;
19 import static android.support.test.espresso.matcher.ViewMatchers.withId;
20 import static android.support.test.espresso.matcher.ViewMatchers.withText;
21 import static org.hamcrest.Matchers.is;
22 import static org.hamcrest.Matchers.not;
23 import static org.junit.Assert.assertNotNull;
24
25 public class RegistrationTest {
26
27     @Rule
28     public ActivityTestRule<Register> mRegTestRule = new ActivityTestRule<>(Register
            ↪ .class);
29     private Register regActivity = null;
30
31     @Before
32     public void setUp() throws Exception {
33         regActivity = mRegTestRule.getActivity();
34     }
35
36     @Test
37     public void registerAndMore() {
38
39         //check if all EditTexts or Buttons are not null
40         View view1 = regActivity.findViewById(R.id.addEmail);
```

```
41          assertNotNull(view1);
42          View view2 = regActivity.findViewById(R.id.addName);
43          assertNotNull(view2);
44          View view3 = regActivity.findViewById(R.id.addPassword);
45          assertNotNull(view3);
46          View view4 = regActivity.findViewById(R.id.confirmPassword);
47          assertNotNull(view4);
48          View view5 = regActivity.findViewById(R.id.registerBtn);
49          assertNotNull(view5);
50
51          try {
52              Thread.sleep(1800);
53          } catch (InterruptedException e) {
54              e.printStackTrace();
55          }
56          onView(withId(R.id.addEmail)).perform(typeText("test@dataport.de"));
57          closeSoftKeyboard();
58          onView(withId(R.id.addName)).perform(typeText("Testing Registration"));
59          closeSoftKeyboard();
60          onView(withId(R.id.addPassword)).perform(typeText("9a$$w0rD"));
61          closeSoftKeyboard();
62          onView(withId(R.id.confirmPassword)).perform(typeText("9a$$w0rD"));
63          closeSoftKeyboard();
64          onView(withId(R.id.registerBtn)).perform(ViewActions.click());
65          onView(withText(R.string.successfulRegister)).inRoot(withDecorView(not(is(
                ↪ regActivity.getWindow().getDecorView())))).check(matches(isDisplayed
                ↪ ())));
66
67          try {
68              Thread.sleep(1800);
69          } catch (InterruptedException e) {
70              e.printStackTrace();
71          }
72          onView(withId(R.id.goToRegisterBtn)).perform(ViewActions.click());
73
74          try {
75              Thread.sleep(1800);
76          } catch (InterruptedException e) {
77              e.printStackTrace();
78          }
79          onView(withId(R.id.addEmail)).perform(typeText("test@dataport.de"));
80          onView(withId(R.id.addEmail)).perform(clearText());
81      }
82
83      @Test
84      public void registerInvalidEmail() {
85
86          try {
87              Thread.sleep(1800);
88          } catch (InterruptedException e) {
89              e.printStackTrace();
90          }
91
92          onView(withId(R.id.addEmail)).perform(typeText("adsaefsa"));
93          closeSoftKeyboard();
94          onView(withId(R.id.addName)).perform(typeText("Testing Registration"));
95          closeSoftKeyboard();
96          onView(withId(R.id.addPassword)).perform(typeText("9a$$w0rD"));
97          closeSoftKeyboard();
98          onView(withId(R.id.confirmPassword)).perform(typeText("9a$$w0rD"));
99          closeSoftKeyboard();
```

```
100         onView(withId(R.id.registerBtn)).perform(ViewActions.click());
101         onView(withText(R.string.registerInvalidEmail)).inRoot(withDecorView(not(is(
                ↪ regActivity.getWindow().getDecorView())))).check(matches(isDisplayed
                ↪ ())));
102     }
103
104     @Test
105     public void registerEmailExists() {
106
107         try {
108             Thread.sleep(1800);
109         } catch (InterruptedException e) {
110             e.printStackTrace();
111         }
112
113         onView(withId(R.id.addEmail)).perform(typeText("test@dataport.de"));
114         closeSoftKeyboard();
115         onView(withId(R.id.addName)).perform(typeText("Testing Registration"));
116         closeSoftKeyboard();
117         onView(withId(R.id.addPassword)).perform(typeText("9a$$w0rD"));
118         closeSoftKeyboard();
119         onView(withId(R.id.confirmPassword)).perform(typeText("9a$$w0rD"));
120         closeSoftKeyboard();
121         onView(withId(R.id.registerBtn)).perform(ViewActions.click());
122         onView(withText(R.string.registerEmailExists)).inRoot(withDecorView(not(is(
                ↪ regActivity.getWindow().getDecorView())))).check(matches(isDisplayed
                ↪ ())));
123     }
124
125     @Test
126     public void registerPasswordsNotMatching() {
127
128         try {
129             Thread.sleep(1800);
130         } catch (InterruptedException e) {
131             e.printStackTrace();
132         }
133
134         onView(withId(R.id.addEmail)).perform(typeText("testing@dataport.de"));
135         closeSoftKeyboard();
136         onView(withId(R.id.addName)).perform(typeText("Testing Registration"));
137         closeSoftKeyboard();
138         onView(withId(R.id.addPassword)).perform(typeText("9a$$w0rD_&2"));
139         closeSoftKeyboard();
140         onView(withId(R.id.confirmPassword)).perform(typeText("9a$$w0rD"));
141         closeSoftKeyboard();
142         onView(withId(R.id.registerBtn)).perform(ViewActions.click());
143         onView(withText(R.string.registerNotMatchingPasswords)).inRoot(withDecorView
                ↪ (not(is(regActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
144     }
145
146     @Test
147     public void registerEmptyFields() {
148
149         try {
150             Thread.sleep(1800);
151         } catch (InterruptedException e) {
152             e.printStackTrace();
153         }
154
```

```
155         onView(withId(R.id.addEmail)).perform(typeText("testing@dataport.de"));
156         closeSoftKeyboard();
157         onView(withId(R.id.addName)).perform(typeText(""));
158         closeSoftKeyboard();
159         onView(withId(R.id.addPassword)).perform(typeText("9a$$w0rD"));
160         closeSoftKeyboard();
161         onView(withId(R.id.confirmPassword)).perform(typeText("9a$$w0rD"));
162         closeSoftKeyboard();
163         onView(withId(R.id.registerBtn)).perform(ViewActions.click());
164         onView(withText(R.string.registerEmptyFields)).inRoot(withDecorView(not(is(
                ↪ regActivity.getWindow().getDecorView())))).check(matches(isDisplayed
                ↪ ()));
165     }
166
167     @Test
168     public void registerPasswordLessThan6Chars() {
169
170         try {
171             Thread.sleep(1800);
172         } catch (InterruptedException e) {
173             e.printStackTrace();
174         }
175
176         onView(withId(R.id.addEmail)).perform(typeText("testing@dataport.de"));
177         closeSoftKeyboard();
178         onView(withId(R.id.addName)).perform(typeText("Testing Registration"));
179         closeSoftKeyboard();
180         onView(withId(R.id.addPassword)).perform(typeText("9a$$w"));
181         closeSoftKeyboard();
182         onView(withId(R.id.confirmPassword)).perform(typeText("9a$$w"));
183         closeSoftKeyboard();
184         onView(withId(R.id.registerBtn)).perform(ViewActions.click());
185         onView(withText(R.string.registerTooShortPassword)).inRoot(withDecorView(not
                ↪ (is(regActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
186     }
187
188     @After
189     public void tearDown() throws Exception {
190         regActivity = null;
191     }
192 }
```

Code-Anhang 19: RegisterTests.java

SharedPreferencesTests.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.content.Context;
4  import android.content.SharedPreferences;
5  import android.support.test.InstrumentationRegistry;
6
7  import org.junit.After;
8  import org.junit.Before;
9  import org.junit.Test;
10
11 import static org.junit.Assert.assertEquals;
12
```

```
13 public class SharedPreferencesTest {
14     private static final String PREFS_NAME = "KanAzubi";
15     private static final String KEY_PREF = "prefs";
16     private SharedPreferences sharedPreferences;
17
18     @Before
19     public void before() {
20         Context context = InstrumentationRegistry.getTargetContext();
21         sharedPreferences = context.getSharedPreferences(PREFS_NAME, Context.
               ↪ MODE_PRIVATE);
22     }
23
24     @Test
25     public void putAndGetID() throws Exception {
26         int putInt = 3;
27         sharedPreferences.edit().putInt(KEY_PREF, putInt).apply();
28         int getInt = sharedPreferences.getInt(KEY_PREF, 0);
29         assertEquals(putInt, getInt);
30     }
31
32     @Test
33     public void putAndGetEmail() throws Exception {
34         String putString = "test@dataport.de";
35         sharedPreferences.edit().putString(KEY_PREF, putString).apply();
36         String getString = sharedPreferences.getString(KEY_PREF, "");
37         assertEquals(putString, getString);
38     }
39
40     @Test
41     public void putAndGetName() throws Exception {
42         String putString = "testMe";
43         sharedPreferences.edit().putString(KEY_PREF, putString).apply();
44         String getString = sharedPreferences.getString(KEY_PREF, "");
45         assertEquals(putString, getString);
46     }
47
48     @After
49     public void after() {
50         sharedPreferences.edit().putString(KEY_PREF, null).apply();
51     }
52 }
```

Code-Anhang 20: SharedPreferencesTests.java

AdminPlattformTests.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.support.test.espresso.action.ViewActions;
4  import android.support.test.rule.ActivityTestRule;
5
6  import org.junit.After;
7  import org.junit.Before;
8  import org.junit.Rule;
9  import org.junit.Test;
10
11 import static android.support.test.espresso.Espresso.closeSoftKeyboard;
12 import static android.support.test.espresso.Espresso.onData;
13 import static android.support.test.espresso.Espresso.onView;
```

```
14 import static android.support.test.espresso.action.ViewActions.clearText;
15 import static android.support.test.espresso.action.ViewActions.replaceText;
16 import static android.support.test.espresso.action.ViewActions.typeText;
17 import static android.support.test.espresso.assertion.ViewAssertions.matches;
18 import static android.support.test.espresso.matcher.RootMatchers.withDecorView;
19 import static android.support.test.espresso.matcher.ViewMatchers.isDisplayed;
20 import static android.support.test.espresso.matcher.ViewMatchers.isRoot;
21 import static android.support.test.espresso.matcher.ViewMatchers.withId;
22 import static android.support.test.espresso.matcher.ViewMatchers.withText;
23 import static org.hamcrest.CoreMatchers.allOf;
24 import static org.hamcrest.CoreMatchers.anything;
25 import static org.hamcrest.CoreMatchers.instanceOf;
26 import static org.hamcrest.Matchers.is;
27 import static org.hamcrest.Matchers.not;
28 import static org.hamcrest.core.StringStartsWith.startsWith;
29
30 public class AdminPlattformTest {
31
32     @Rule
33     public ActivityTestRule<Login> mLoginTestRule = new ActivityTestRule<>(Login.
         ↪ class);
34     private Login loginActivity = null;
35
36     @Before
37     public void setUp() throws Exception {
38         loginActivity = mLoginTestRule.getActivity();
39     }
40
41     @Test
42     public void adminPlattformTest() {
43         try {
44             Thread.sleep(1800);
45         } catch (InterruptedException e) {
46             e.printStackTrace();
47         }
48         onView(withId(R.id.et_email)).perform(typeText("admin@dataport.de"));
49         closeSoftKeyboard();
50         onView(withId(R.id.et_password)).perform(typeText("123456"));
51         closeSoftKeyboard();
52         onView(withId(R.id.loginBtn)).perform(ViewActions.click());
53         onView(withText(R.string.successfulLogin)).inRoot(withDecorView(not(is(
             ↪ loginActivity.getWindow().getDecorView())))).check(matches(
             ↪ isDisplayed()));
54
55         try {
56             Thread.sleep(1800);
57         } catch (InterruptedException e) {
58             e.printStackTrace();
59         }
60         onView(withId(R.id.searchBtn)).perform(ViewActions.click());
61
62         onView(withId(R.id.et_taskTitle)).perform(clearText());
63         onView(withId(R.id.et_taskTitle)).perform(replaceText("Test Titel hinzufügen
             ↪ "));
64         closeSoftKeyboard();
65         onView(withId(R.id.et_taskDescr)).perform(clearText());
66         onView(withId(R.id.et_taskDescr)).perform(replaceText("Test ist sehr wichtig
             ↪ !"));
67         closeSoftKeyboard();
68         onView(withId(R.id.et_taskReleased)).perform(clearText());
69         onView(withId(R.id.et_taskReleased)).perform(replaceText("01.01.2004"));
```

```
70          closeSoftKeyboard();
71          onView(withId(R.id.et_taskTeam)).perform(clearText());
72          onView(withId(R.id.et_taskTeam)).perform(replaceText("Software"));
73          closeSoftKeyboard();
74
75          onView(withId(R.id.addTaskBtn)).perform(ViewActions.click());
76          onView(withId(R.id.logoutBtn)).perform(ViewActions.click());
77
78          try {
79              Thread.sleep(1800);
80          } catch (InterruptedException e) {
81              e.printStackTrace();
82          }
83          onView(withId(R.id.et_email)).perform(typeText("tony@gmail.com"));
84          closeSoftKeyboard();
85          onView(withId(R.id.et_password)).perform(typeText("qwerty"));
86          closeSoftKeyboard();
87          onView(withId(R.id.loginBtn)).perform(ViewActions.click());
88          onView(withText(R.string.successfulLogin)).inRoot(withDecorView(not(is(
                ↪ loginActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
89
90          try {
91              Thread.sleep(1800);
92          } catch (InterruptedException e) {
93              e.printStackTrace();
94          }
95
96          onView(withId(R.id.spinnerTeam)).perform(ViewActions.click());
97          onData(allOf(is(instanceOf(String.class)), is("Software"))).perform(
                ↪ ViewActions.click());
98          onView(withId(R.id.spinnerStatus)).perform(ViewActions.click());
99          onData(allOf(is(instanceOf(String.class)), is("Offen"))).perform(ViewActions
                ↪ .click());
100         onView(withId(R.id.searchBtn)).perform(ViewActions.click());
101
102         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(1).
                ↪ check(matches(withText(startsWith("Test Titel hinzufügen"))));
103         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(1).
                ↪ perform(ViewActions.click());
104         onView(withId(R.id.taskDescr)).check(matches(withText("Test ist sehr wichtig
                ↪ !")));
105         onView(withId(R.id.taskReleased)).check(matches(withText("Veröffentlicht
                ↪ seit: 01.01.2004")));
106         onView(withId(R.id.joinTaskBtn)).perform(ViewActions.click());
107
108         onView(withId(R.id.spinnerTeam)).perform(ViewActions.click());
109         onData(allOf(is(instanceOf(String.class)), is("Software"))).perform(
                ↪ ViewActions.click());
110         onView(withId(R.id.spinnerStatus)).perform(ViewActions.click());
111         onData(allOf(is(instanceOf(String.class)), is("In Bearbeitung"))).perform(
                ↪ ViewActions.click());
112         onView(withId(R.id.searchBtn)).perform(ViewActions.click());
113
114         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(1).
                ↪ check(matches(withText(startsWith("Test Titel hinzufügen"))));
115         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(1).
                ↪ perform(ViewActions.click());
116
117         onView(isRoot()).perform(ViewActions.pressBack());
118         onView(isRoot()).perform(ViewActions.pressBack());
```

```
119         onView(withId(R.id.logoutBtn)).perform(ViewActions.click());
120     }
121
122     @After
123     public void tearDown() throws Exception {
124         loginActivity = null;
125     }
126
127 }
```

Code-Anhang 21: AdminPlattformTests.java

MenuTests.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.support.test.espresso.action.ViewActions;
4  import android.support.test.rule.ActivityTestRule;
5
6  import org.junit.After;
7  import org.junit.Before;
8  import org.junit.Rule;
9  import org.junit.Test;
10
11 import static android.support.test.espresso.Espresso.closeSoftKeyboard;
12 import static android.support.test.espresso.Espresso.onData;
13 import static android.support.test.espresso.Espresso.onView;
14 import static android.support.test.espresso.action.ViewActions.typeText;
15 import static android.support.test.espresso.assertion.ViewAssertions.matches;
16 import static android.support.test.espresso.matcher.RootMatchers.withDecorView;
17 import static android.support.test.espresso.matcher.ViewMatchers.isDisplayed;
18 import static android.support.test.espresso.matcher.ViewMatchers.isRoot;
19 import static android.support.test.espresso.matcher.ViewMatchers.withId;
20 import static android.support.test.espresso.matcher.ViewMatchers.withText;
21 import static org.hamcrest.CoreMatchers.allOf;
22 import static org.hamcrest.CoreMatchers.anything;
23 import static org.hamcrest.CoreMatchers.instanceOf;
24 import static org.hamcrest.Matchers.is;
25 import static org.hamcrest.Matchers.not;
26 import static org.hamcrest.core.StringStartsWith.startsWith;
27
28 public class MenuTest {
29
30     @Rule
31     public ActivityTestRule<Login> mLoginTestRule = new ActivityTestRule<>(Login.
       ↪ class);
32     private Login loginActivity = null;
33
34     @Before
35     public void setUp() throws Exception {
36         loginActivity = mLoginTestRule.getActivity();
37
38         try {
39             Thread.sleep(1800);
40         } catch (InterruptedException e) {
41             e.printStackTrace();
42         }
43         onView(withId(R.id.et_email)).perform(typeText("tony@gmail.com"));
44         closeSoftKeyboard();
```

```
45          onView(withId(R.id.et_password)).perform(typeText("qwerty"));
46          closeSoftKeyboard();
47          onView(withId(R.id.loginBtn)).perform(ViewActions.click());
48          onView(withText(R.string.successfulLogin)).inRoot(withDecorView(not(is(
                ↪ loginActivity.getWindow().getDecorView())))).check(matches(
                ↪ isDisplayed()));
49      }
50
51      @Test
52      public void menuSpinnerAndListResultTest() {
53
54          try {
55              Thread.sleep(1800);
56          } catch (InterruptedException e) {
57              e.printStackTrace();
58          }
59
60          onView(withId(R.id.spinnerTeam)).perform(ViewActions.click());
61          onData(allOf(is(instanceOf(String.class)), is("Software"))).perform(
                ↪ ViewActions.click());
62
63          onView(withId(R.id.spinnerStatus)).perform(ViewActions.click());
64          onData(allOf(is(instanceOf(String.class)), is("In Bearbeitung"))).perform(
                ↪ ViewActions.click());
65
66          onView(withId(R.id.searchBtn)).perform(ViewActions.click());
67
68          onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(0).
                ↪ check(matches(withText(startsWith("Test"))));
69          onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(0).
                ↪ perform(ViewActions.click());
70          onView(withId(R.id.taskReleased)).check(matches(withText("Veröffentlicht
                ↪ seit: 23.08.2014")));
71
72          onView(isRoot()).perform(ViewActions.pressBack());
73          onView(isRoot()).perform(ViewActions.pressBack());
74          onView(withId(R.id.logoutBtn)).perform(ViewActions.click());
75      }
76
77      @Test
78      public void joinTask_AcceptReadyDone_Test() {
79
80          try {
81              Thread.sleep(1800);
82          } catch (InterruptedException e) {
83              e.printStackTrace();
84          }
85
86          onView(withId(R.id.spinnerTeam)).perform(ViewActions.click());
87          onData(allOf(is(instanceOf(String.class)), is("Marketing"))).perform(
                ↪ ViewActions.click());
88          onView(withId(R.id.spinnerStatus)).perform(ViewActions.click());
89          onData(allOf(is(instanceOf(String.class)), is("Offen"))).perform(ViewActions
                ↪ .click());
90          onView(withId(R.id.searchBtn)).perform(ViewActions.click());
91
92          onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(1).
                ↪ check(matches(withText(startsWith("Test2"))));
93          onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(1).
                ↪ perform(ViewActions.click());
94          onView(withId(R.id.taskTitle)).check(matches(withText("Test2")));
```

```
95          onView(withId(R.id.joinTaskBtn)).perform(ViewActions.click());
96
97          onView(withId(R.id.spinnerTeam)).perform(ViewActions.click());
98          onData(allOf(is(instanceOf(String.class)), is("Marketing"))).perform(
                ↪ ViewActions.click());
99          onView(withId(R.id.spinnerStatus)).perform(ViewActions.click());
100         onData(allOf(is(instanceOf(String.class)), is("In Bearbeitung"))).perform(
                ↪ ViewActions.click());
101         onView(withId(R.id.searchBtn)).perform(ViewActions.click());
102
103         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(0).
                ↪ check(matches(withText(startsWith("Test2"))));
104         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(0).
                ↪ perform(ViewActions.click());
105         onView(withId(R.id.taskDescr)).check(matches(withText("Test In Bearbeitung")
                ↪ ));
106
107         onView(withId(R.id.joinTaskBtn)).perform(ViewActions.click());
108
109         onView(withId(R.id.spinnerTeam)).perform(ViewActions.click());
110         onData(allOf(is(instanceOf(String.class)), is("Marketing"))).perform(
                ↪ ViewActions.click());
111         onView(withId(R.id.spinnerStatus)).perform(ViewActions.click());
112         onData(allOf(is(instanceOf(String.class)), is("Erledigt"))).perform(
                ↪ ViewActions.click());
113         onView(withId(R.id.searchBtn)).perform(ViewActions.click());
114
115         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(0).
                ↪ check(matches(withText(startsWith("Test2"))));
116         onData(anything()).inAdapterView(withId(R.id.listResult)).atPosition(0).
                ↪ perform(ViewActions.click());
117         onView(withId(R.id.taskReleased)).check(matches(withText("Veröffentlicht
                ↪ seit: 14.02.2016")));
118
119         onView(isRoot()).perform(ViewActions.pressBack());
120         onView(isRoot()).perform(ViewActions.pressBack());
121         onView(withId(R.id.logoutBtn)).perform(ViewActions.click());
122     }
123
124     @After
125     public void tearDown() throws Exception {
126         loginActivity = null;
127     }
128 }
```

Code-Anhang 22: MenuTests.java

ValidEmailTest.java

```
1  package com.example.techgeek.kanazubi;
2
3  import org.junit.Test;
4
5  import static org.junit.Assert.*;
6
7  public class ValidEmailTest {
8      @Test
9      public void emailValidator_CorrectEmail() {
10         assertTrue(Register.isEmailValid("name@email.com"));
```

```
11          assertTrue(Register.isEmailValid("admin@dataport.de"));
12          assertTrue(Register.isEmailValid("administrator@hotmail.co.uk"));
13          assertFalse(Register.isEmailValid("dadfgrefasdas.adada"));
14          assertFalse(Register.isEmailValid("da$s?@a.da"));
15          assertFalse(Register.isEmailValid("dada"));
16      }
17  }
```

Code-Anhang 23: ValidEmailTest.java

User.java

```
1  package com.example.techgeek.kanazubi.Model;
2
3  public class User {
4
5      private String email;
6      private String password;
7      private String name;
8
9      public String getEmail() {
10         return email;
11     }
12
13     public void setEmail(String email) {
14         this.email = email;
15     }
16
17     public String getPassword() {
18         return password;
19     }
20
21     public void setPassword(String password) {
22         this.password = password;
23     }
24
25     public String getName() {
26         return name;
27     }
28
29     public void setName(String name) {
30         this.name = name;
31     }
32 }
```

Code-Anhang 24: User.java

Session.java

```
1  package com.example.techgeek.kanazubi.Session;
2
3  import android.content.Context;
4  import android.content.SharedPreferences;
5
6  public class Session {
7      private SharedPreferences preferences;
```

```
8     private SharedPreferences.Editor editor;
9     private Context c;
10
11    public Session(Context c){
12        this.c = getContext();
13        preferences = c.getSharedPreferences("KanAzubi", Context.MODE_PRIVATE);
14        editor = preferences.edit();
15        editor.apply();
16    }
17
18    public void saveEmail(String email){
19        preferences.edit().putString("email", email).apply();
20    }
21
22    public String getEmail(){
23        return preferences.getString("email", "");
24    }
25
26    public void saveUserName(String name){
27        preferences.edit().putString("userName", name).apply();
28    }
29
30    public String getUserName(){
31        return preferences.getString("userName", "");
32    }
33
34    public void saveId(int id){
35        preferences.edit().putInt("userID", id).apply();
36    }
37
38    public int getId(){
39        return preferences.getInt("userID", 0);
40    }
41
42    public void setLogIn(boolean logIn){
43        editor.putBoolean("loggedIn", logIn);
44        editor.commit();
45    }
46
47    public boolean loggedIn(){
48        return preferences.getBoolean("loggedIn", false);
49    }
50
51    private Context getContext() {
52        return c;
53    }
54 }
```

Code-Anhang 25: Session.java

DatabaseHelper.java

```
1 package com.example.techgeek.kanazubi.SQL;
2
3 import android.annotation.SuppressLint;
4 import android.content.ContentValues;
5 import android.content.Context;
6 import android.database.Cursor;
7 import android.database.sqlite.SQLiteDatabase;
```

```
 8 import android.database.sqlite.SQLiteOpenHelper;
 9
10 import com.example.techgeek.kanazubi.Model.User;
11
12 public class DatabaseHelper extends SQLiteOpenHelper {
13
14     private SQLiteDatabase db;
15     private static final int DB_VERSION = 1;
16     //database with both tables
17     private static final String DB_NAME = "KanAzubiDB.db";
18     //users table
19     private static final String TABLE_NAME = "users";
20     private static final String COLUMB_ID = "ID";
21     private static final String COLUMB_EMAIL = "Email";
22     private static final String COLUMB_PASSWORD = "Password";
23     private static final String COLUMB_NAME = "Name";
24     //tasks table
25     private static final String TABLE_NAME2 = "tasks";
26     private static final String COLUMB_ID2 = "ID";
27     private static final String COLUMB_TITLE = "Title";
28     private static final String COLUMB_DESCR = "Description";
29     private static final String COLUMB_RELEASED = "Released";
30     private static final String COLUMB_TEAM = "Team";
31     private static final String COLUMB_STATUS = "Status";
32     private static final String COLUMB_USERID = "User_ID";
33
34     public DatabaseHelper(Context context) {
35         super(context, DB_NAME, null, DB_VERSION);
36     }
37
38     @Override
39     public void onCreate(SQLiteDatabase db) {
40         String tableCreate = "create table " + TABLE_NAME + " (" + COLUMB_ID + "
               ↪ integer primary key not null, " + COLUMB_EMAIL + " text not null, " +
               ↪  COLUMB_PASSWORD + " text not null, " + COLUMB_NAME + " text not null
               ↪ );";
41         db.execSQL(tableCreate);
42         this.db = db;
43     }
44
45     public boolean checkIfExist(String email) {
46         boolean exist = false;
47         db = this.getReadableDatabase();
48         String queryPass = "select " + COLUMB_EMAIL + " from " + TABLE_NAME;
49         @SuppressLint("Recycle")
50         Cursor crs = db.rawQuery(queryPass, null);
51         String emailStr;
52         if (crs.moveToFirst()) {
53             do {
54                 emailStr = crs.getString(0);
55                 if (emailStr.equals(email)) {
56                     exist = true;
57                 }
58             } while (crs.moveToNext());
59         }
60         crs.close();
61         db.close();
62         return exist;
63     }
64
65     public int searchForID(String email) {
```

```
66          db = this.getReadableDatabase();
67          String queryPass = "select " + COLUMB_ID + " , " + COLUMB_EMAIL + " from " +
                ↪  TABLE_NAME;
68          @SuppressLint("Recycle")
69          Cursor crs = db.rawQuery(queryPass, null);
70          String emailStr;
71          int idStr = 0;
72          if (crs.moveToFirst()) {
73              do {
74                  emailStr = crs.getString(1);
75                  if (emailStr.equals(email)) {
76                      idStr = crs.getInt(0);
77                      break;
78                  }
79
80              } while (crs.moveToNext());
81          }
82          crs.close();
83          db.close();
84          return idStr;
85      }
86
87      public String searchForName(String email) {
88          db = this.getReadableDatabase();
89          String queryPass = "select " + COLUMB_EMAIL + " , " + COLUMB_NAME + " from "
                ↪  + TABLE_NAME;
90          @SuppressLint("Recycle")
91          Cursor crs = db.rawQuery(queryPass, null);
92          String emailStr, nameStr = "";
93          if (crs.moveToFirst()) {
94              do {
95                  emailStr = crs.getString(0);
96                  if (emailStr.equals(email)) {
97                      nameStr = crs.getString(1);
98                      break;
99                  }
100
101             } while (crs.moveToNext());
102         }
103         crs.close();
104         db.close();
105         return nameStr;
106     }
107
108     public String searchForPass(String email) {
109         db = this.getReadableDatabase();
110         String queryPass = "select " + COLUMB_EMAIL + ", " + COLUMB_PASSWORD + "
                ↪ from " + TABLE_NAME;
111         @SuppressLint("Recycle")
112         Cursor crs = db.rawQuery(queryPass, null);
113         String emailStr, passwordStr = "";
114         if (crs.moveToFirst()) {
115             do {
116                 emailStr = crs.getString(0);
117                 if (emailStr.equals(email)) {
118                     passwordStr = crs.getString(1);
119                     break;
120                 }
121             } while (crs.moveToNext());
122         }
123         crs.close();
```

```
124         db.close();
125         return passwordStr;
126     }
127 
128     public void insertUser(User user) {
129         db = this.getWritableDatabase();
130         ContentValues cv = new ContentValues();
131         String query = "SELECT * FROM " + TABLE_NAME;
132         @SuppressLint("Recycle")
133         Cursor crs = db.rawQuery(query, null);
134         int count = crs.getCount() + 1; //start autoincrement id from '1'! ('0' is
                ↪ the default value for open tasks, which are not related to any user.)
135         cv.put(COLUMB_ID, count);
136         cv.put(COLUMB_EMAIL, user.getEmail());
137         cv.put(COLUMB_NAME, user.getName());
138         cv.put(COLUMB_PASSWORD, user.getPassword());
139         db.insert(TABLE_NAME, null, cv);
140         crs.close();
141         db.close();
142     }
143 
144     public void addTask(String title, String descr, String released, String team) {
145         db = this.getWritableDatabase();
146         ContentValues cv = new ContentValues();
147         String query = "SELECT * FROM " + TABLE_NAME2;
148         @SuppressLint("Recycle")
149         Cursor crs = db.rawQuery(query, null);
150         int count = crs.getCount() + 1;
151         cv.put(COLUMB_ID, count);
152         cv.put(COLUMB_TITLE, title);
153         cv.put(COLUMB_DESCR, descr);
154         cv.put(COLUMB_RELEASED, released);
155         cv.put(COLUMB_STATUS, "Offen");
156         cv.put(COLUMB_TEAM, team);
157         cv.put(COLUMB_USERID, 0);
158         db.insert(TABLE_NAME2, null, cv);
159         crs.close();
160         db.close();
161     }
162 
163     public Cursor queryList() {
164         db = this.getReadableDatabase();
165         //alternative: (select * from) without changing column position
166         String query = "select " + COLUMB_ID2 + ", " + COLUMB_TITLE + ", " +
                ↪ COLUMB_TEAM + ", " + COLUMB_STATUS + ", " +
167                 COLUMB_USERID + ", " + COLUMB_RELEASED + ", " + COLUMB_DESCR + "
                    ↪ from " + TABLE_NAME2;
168         return db.rawQuery(query, null);
169     }
170 
171     public void acceptTask(int userID, int taskNr) {
172         db = this.getWritableDatabase();
173         String acceptTask = "In Bearbeitung";
174         String query = "UPDATE " + TABLE_NAME2 + " SET " + COLUMB_STATUS + " = '" +
175                 acceptTask + "' " + ", " + COLUMB_USERID +
176                 " = '" + userID + "'" + " WHERE " + COLUMB_ID2 + " = '" + taskNr + "
                    ↪ '";
177         db.execSQL(query);
178         db.close();
179     }
180 
```

```
181     public void doneTask(int taskNr) {
182         db = this.getWritableDatabase();
183         String doneTask = "Erledigt";
184         String query = "UPDATE " + TABLE_NAME2 + " SET " + COLUMB_STATUS + " = '" +
185                 doneTask + "' WHERE " + COLUMB_ID2 + " = '" + taskNr + "'";
186         db.execSQL(query);
187         db.close();
188     }
189 
190     @Override
191     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
192         String tableDrop = "DROP TABLE IF EXISTS " + TABLE_NAME;
193         db.execSQL(tableDrop);
194         this.onCreate(db);
195     }
196 }
```

Code-Anhang 26: DatabaseHelper.java

Login.java

```
1  package com.example.techgeek.kanazubi;
2  
3  import android.content.Intent;
4  import android.os.Handler;
5  import android.support.v7.app.AppCompatActivity;
6  import android.os.Bundle;
7  import android.view.View;
8  import android.widget.Button;
9  import android.widget.EditText;
10 import android.widget.RelativeLayout;
11 import android.widget.Toast;
12 
13 import com.example.techgeek.kanazubi.SQL.DatabaseHelper;
14 import com.example.techgeek.kanazubi.Session.Session;
15 
16 import org.mindrot.jbcrypt.BCrypt;
17 
18 public class Login extends AppCompatActivity {
19 
20     RelativeLayout rellay1, rellay2;
21     Button loginBtn, goToRegisterBtn;
22     EditText email, password;
23     private DatabaseHelper dbhelper;
24     private Session session;
25 
26     Handler handler = new Handler();
27     Runnable runnable = new Runnable() {
28         @Override
29         public void run() {
30             rellay1.setVisibility(View.VISIBLE);
31             rellay2.setVisibility(View.VISIBLE);
32         }
33     };
34 
35     @Override
36     protected void onCreate(Bundle savedInstanceState) {
37         super.onCreate(savedInstanceState);
38         setContentView(R.layout.activity_login);
```

```
39
40          handler.postDelayed(runnable, 1600);
41
42          dbhelper = new DatabaseHelper(this);
43          session = new Session(this);
44
45          rellay1 = findViewById(R.id.rellay1);
46          rellay2 = findViewById(R.id.rellay2);
47          email = findViewById(R.id.et_email);
48          password = findViewById(R.id.et_password);
49          loginBtn = findViewById(R.id.loginBtn);
50          goToRegisterBtn = findViewById(R.id.goToRegisterBtn);
51
52          goToRegisterBtn.setOnClickListener(new View.OnClickListener() {
53              @Override
54              public void onClick(View v) {
55                  Intent in = new Intent(Login.this, Register.class);
56                  Login.this.startActivity(in);
57              }
58          });
59
60          loginBtn.setOnClickListener(new View.OnClickListener() {
61              @Override
62              public void onClick(View v) {
63                  loginUser();
64              }
65          });
66
67          if(session.loggedIn()){ //check if a session already exists
68              Intent i = new Intent(Login.this, Menu.class);
69              startActivity(i);
70          }
71      }
72
73      private void loginUser() {
74          String emailStr = email.getText().toString();
75          if (dbhelper.checkIfExist(emailStr)) {
76              String passwordStr = password.getText().toString();
77              String pass = dbhelper.searchForPass(emailStr);
78              String nameStr = dbhelper.searchForName(emailStr);
79              int idInt = dbhelper.searchForID(emailStr);
80
81              boolean ifMatchPass = BCrypt.checkpw(passwordStr, pass);
82
83              if (!emailStr.equals("") && !passwordStr.equals("")) {
84                  if (ifMatchPass) {
85                      Toast.makeText(Login.this, "Erfolgreich angemeldet!", Toast.
                            ↪ LENGTH_SHORT).show();
86                      Intent i = new Intent(Login.this, Menu.class);
87                      session.setLogIn(true); //create a session!
88                      session.saveId(idInt); //save userID for the session!
89                      session.saveEmail(emailStr); //save email for the session!
90                      session.saveUserName(nameStr); //save user's name for the
                            ↪ session!
91                      startActivity(i);
92                      finish();
93                  } else {
94                      Toast.makeText(Login.this, "Passwort ist falsch!", Toast.
                            ↪ LENGTH_SHORT).show();
95                  }
96              } else {
```

```
97                  Toast.makeText(Login.this, "Email/Passwort ist leer!", Toast.
                        ↪ LENGTH_SHORT).show();
98              }
99          } else {
100             Toast.makeText(Login.this, "Email ist invalid!", Toast.LENGTH_SHORT).
                    ↪ show();
101         }
102
103     }
104 }
```

Code-Anhang 27: Login.java

Register.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.content.Intent;
4  import android.os.Handler;
5  import android.support.v7.app.AppCompatActivity;
6  import android.os.Bundle;
7  import android.view.View;
8  import android.widget.Button;
9  import android.widget.EditText;
10 import android.widget.RelativeLayout;
11 import android.widget.Toast;
12
13 import com.example.techgeek.kanazubi.Model.User;
14 import com.example.techgeek.kanazubi.SQL.DatabaseHelper;
15
16 import org.mindrot.jbcrypt.BCrypt;
17
18 import java.util.regex.Pattern;
19
20 public class Register extends AppCompatActivity {
21
22     RelativeLayout rellayReg1, rellayReg2;
23     Button goToLogin, registerBtn;
24     EditText email, password, confirmPassword, name;
25     private DatabaseHelper dbhelper;
26
27     Handler handler = new Handler();
28     Runnable runnable = new Runnable() {
29         @Override
30         public void run() {
31             rellayReg1.setVisibility(View.VISIBLE);
32             rellayReg2.setVisibility(View.VISIBLE);
33         }
34     };
35
36     @Override
37     protected void onCreate(Bundle savedInstanceState) {
38         super.onCreate(savedInstanceState);
39         setContentView(R.layout.activity_register);
40
41         dbhelper = new DatabaseHelper(this);
42
43         rellayReg1 = findViewById(R.id.rellayReg1);
44         rellayReg2 = findViewById(R.id.rellayReg2);
```

```
45          registerBtn = findViewById(R.id.registerBtn);
46          goToLogin = findViewById(R.id.goToLoginBtn);
47
48          handler.postDelayed(runnable, 1600);
49
50          registerBtn.setOnClickListener(new View.OnClickListener() {
51              @Override
52              public void onClick(View v) {
53                  registerBtn.setEnabled(false);
54                  registerUser();
55                  registerBtn.setEnabled(true);
56              }
57          });
58
59          goToLogin.setOnClickListener(new View.OnClickListener() {
60              @Override
61              public void onClick(View v) {
62                  goToLogin.setEnabled(false);
63                  Intent in = new Intent(Register.this, Login.class);
64                  Register.this.startActivity(in);
65                  goToLogin.setEnabled(true);
66              }
67          });
68      }
69
70      //check the structure of the entered email and returns true if it's valid
71      public static boolean isEmailValid(String email) {
72          String emailPattern = "^[a-zA-Z0-9_*-]+(?:\\." +
73                  "[a-zA-Z0-9_+&*-]+)*@" +
74                  "(?:[a-zA-Z0-9-]+\\.)+[a-z" +
75                  "A-Z]{2,7}$";
76          Pattern pat = Pattern.compile(emailPattern);
77          return email != null && pat.matcher(email).matches();
78      }
79
80      public void registerUser() {
81          email = findViewById(R.id.addEmail);
82          name = findViewById(R.id.addName);
83          password = findViewById(R.id.addPassword);
84          confirmPassword = findViewById(R.id.confirmPassword);
85
86          String emailStr = email.getText().toString().trim();
87          String nameStr = name.getText().toString().trim();
88          String passwordStr = password.getText().toString().trim();
89          String confirmPasswordStr = confirmPassword.getText().toString().trim();
90
91          if(dbhelper.checkIfExist(emailStr)) {
92              Toast.makeText(Register.this, "Dieser Email wurde schon benutzt!", Toast
                    ↪ .LENGTH_SHORT).show();
93          }else if(!isEmailValid(emailStr)){
94              Toast.makeText(Register.this, "Email ist nicht valid!", Toast.
                    ↪ LENGTH_SHORT).show();
95          }else if (!passwordStr.equals(confirmPasswordStr)) {
96              Toast.makeText(Register.this, "Passwörter stimmen nicht überein!", Toast
                    ↪ .LENGTH_SHORT).show();
97          }else if(emailStr.equals("") || nameStr.equals("") || passwordStr.equals("")
                ↪  || confirmPasswordStr.equals("")) {
98              Toast.makeText(Register.this, "Nicht alle Informationen sind eingegeben!
                    ↪ ", Toast.LENGTH_SHORT).show();
99          }else if(passwordStr.length() < 6){
100             Toast.makeText(Register.this, "Passwort soll mindestens 6 Zeichen
```

```
                ↪ enthalten!", Toast.LENGTH_SHORT).show();
101         }else {
102             String generatedPasswordHash = BCrypt.hashpw(passwordStr, BCrypt.gensalt
                    ↪ (12));
103 
104             User user = new User();
105             user.setEmail(emailStr);
106             user.setPassword(generatedPasswordHash);
107             user.setName(nameStr);
108 
109             dbhelper.insertUser(user);
110             Toast.makeText(Register.this, "Benutzer wurde erstellt!", Toast.
                    ↪ LENGTH_SHORT).show();
111             Intent in = new Intent(Register.this, Login.class);
112             Register.this.startActivity(in);
113         }
114     }
115 }
```

Code-Anhang 28: Register.java

AdminPlattform.java

```
1  package com.example.techgeek.kanazubi;
2  
3  import android.content.Intent;
4  import android.support.v7.app.AppCompatActivity;
5  import android.os.Bundle;
6  import android.view.View;
7  import android.widget.Button;
8  import android.widget.TextView;
9  
10 import com.example.techgeek.kanazubi.SQL.DatabaseHelper;
11 
12 public class AdminPlattform extends AppCompatActivity {
13 
14     TextView et_taskTitle, et_taskDescr, et_taskReleased, et_taskTeam;
15     Button addTaskBtn;
16     private String title, descr, released, team;
17     private DatabaseHelper dbhelper;
18 
19     @Override
20     protected void onCreate(Bundle savedInstanceState) {
21         super.onCreate(savedInstanceState);
22         setContentView(R.layout.activity_admin_plattform);
23 
24         dbhelper = new DatabaseHelper(this);
25 
26         et_taskTitle = findViewById(R.id.et_taskTitle);
27         et_taskDescr = findViewById(R.id.et_taskDescr);
28         et_taskReleased = findViewById(R.id.et_taskReleased);
29         et_taskTeam = findViewById(R.id.et_taskTeam);
30         addTaskBtn = findViewById(R.id.addTaskBtn);
31 
32         addTaskBtn.setOnClickListener(new View.OnClickListener() {
33             @Override
34             public void onClick(View v) {
35                 title = et_taskTitle.getText().toString();
36                 descr = et_taskDescr.getText().toString();
```

```
37                 released = et_taskReleased.getText().toString();
38                 team = et_taskTeam.getText().toString();
39                 dbhelper.addTask(title,descr,released,team);
40                 Intent i = new Intent(AdminPlattform.this, Menu.class);
41                 startActivity(i);
42                 finish();
43             }
44         });
45     }
46 }
```

Code-Anhang 29: AdminPlattform.java

Menu.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.content.Intent;
4  import android.os.Handler;
5  import android.support.v7.app.AppCompatActivity;
6  import android.os.Bundle;
7  import android.view.View;
8  import android.widget.AdapterView;
9  import android.widget.ArrayAdapter;
10 import android.widget.Button;
11 import android.widget.RelativeLayout;
12 import android.widget.Spinner;
13 import android.widget.TextView;
14
15 import com.example.techgeek.kanazubi.Session.Session;
16
17 public class Menu extends AppCompatActivity {
18
19     RelativeLayout rellay1, rellay2;
20     TextView tvName;
21     Spinner status, team;
22     Button logoutBtn, searchBtn;
23     private String statusStr, teamStr;
24     private Session session;
25
26     Handler handler = new Handler();
27     Runnable runnable = new Runnable() {
28         @Override
29         public void run() {
30             rellay1.setVisibility(View.VISIBLE);
31             rellay2.setVisibility(View.VISIBLE);
32         }
33     };
34
35     @Override
36     protected void onCreate(Bundle savedInstanceState) {
37         super.onCreate(savedInstanceState);
38         setContentView(R.layout.activity_menu);
39
40         rellay1 = findViewById(R.id.rellay1);
41         rellay2 = findViewById(R.id.rellay2);
42         logoutBtn = findViewById(R.id.logoutBtn);
43         searchBtn = findViewById(R.id.searchBtn);
44         tvName = findViewById(R.id.tvName);
```

```
45
46          handler.postDelayed(runnable, 1240);
47
48          session = new Session(this);
49          if (!session.loggedIn()) {
50              logout();
51          }
52          String userName = session.getUserName();
53          tvName.setText(userName);
54
55          logoutBtn.setOnClickListener(new View.OnClickListener() {
56              @Override
57              public void onClick(View v) {
58                  logout();
59              }
60          });
61
62          status = findViewById(R.id.spinnerStatus);
63          ArrayAdapter<String> aaStatus = new ArrayAdapter<>(Menu.this, R.layout.
                ↪ spinner_items, getResources().getStringArray(R.array.status));
64          aaStatus.setDropDownViewResource(android.R.layout.
                ↪ simple_spinner_dropdown_item);
65          status.setAdapter(aaStatus);
66
67          status.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
68              @Override
69              public void onItemSelected(AdapterView<?> parent, View view, int
                    ↪ position, long id) {
70                  statusStr = (String) parent.getItemAtPosition(position);
71              }
72
73              @Override
74              public void onNothingSelected(AdapterView<?> parent) {
75
76              }
77          });
78
79          team = findViewById(R.id.spinnerTeam);
80          ArrayAdapter<String> aaTeam = new ArrayAdapter<>(Menu.this, R.layout.
                ↪ spinner_items, getResources().getStringArray(R.array.team));
81          aaTeam.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item
                ↪ );
82          team.setAdapter(aaTeam);
83
84          team.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
85              @Override
86              public void onItemSelected(AdapterView<?> parent, View view, int
                    ↪ position, long id) {
87                  teamStr = (String) parent.getItemAtPosition(position);
88              }
89
90              @Override
91              public void onNothingSelected(AdapterView<?> parent) {
92
93              }
94          });
95
96          if (session.getId() == 1 && session.getEmail().equals("admin@dataport.de"))
                ↪ {
97              String addTask = "Hinzufügen";
98              searchBtn.setText(addTask);
```

```
99              status.setEnabled(false);
100             status.setAlpha(.75f);
101             team.setEnabled(false);
102             team.setAlpha(.75f);
103         }
104
105         searchBtn.setOnClickListener(new View.OnClickListener() {
106             @Override
107             public void onClick(View v) {
108                 if (session.getId() == 1 && session.getEmail().equals("
                        ↪ admin@dataport.de")) { // if user_ID = 1 AND email = "
                        ↪ admin@dataport.de", then open admin plattform for adding new
                        ↪ tasks.
109                     Intent i = new Intent(Menu.this, AdminPlattform.class);
110                     startActivity(i);
111                 } else {
112                     Intent i = new Intent(Menu.this, List.class);
113                     i.putExtra("status", statusStr);
114                     i.putExtra("team", teamStr);
115                     startActivity(i);
116                 }
117             }
118         });
119     }
120
121     private void logout() {
122         session.setLogIn(false);
123         finish();
124         Intent i = new Intent(Menu.this, Login.class);
125         startActivity(i);
126     }
127
128     public void onBackPressed() {
129         //deactivate back button on menu activity
130     }
131 }
```

Code-Anhang 30: Menu.java

List.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.content.Intent;
4  import android.database.Cursor;
5  import android.support.annotation.NonNull;
6  import android.support.annotation.Nullable;
7  import android.support.v7.app.AppCompatActivity;
8  import android.os.Bundle;
9  import android.view.View;
10 import android.view.ViewGroup;
11 import android.view.animation.Animation;
12 import android.view.animation.AnimationUtils;
13 import android.widget.AdapterView;
14 import android.widget.ArrayAdapter;
15 import android.widget.ListView;
16 import android.widget.TextView;
17
18 import com.example.techgeek.kanazubi.SQL.DatabaseHelper;
```

```
19 import com.example.techgeek.kanazubi.Session.Session;
20
21 import java.util.ArrayList;
22
23 public class List extends AppCompatActivity {
24
25     TextView emptyList;
26     ListView listResult;
27     private int userID;
28     private String status, team;
29     private ArrayList<Integer> listID;
30     private ArrayList<String> listTitle;
31     private DatabaseHelper dbhelper;
32
33     @Override
34     protected void onCreate(Bundle savedInstanceState) {
35         super.onCreate(savedInstanceState);
36         setContentView(R.layout.activity_list);
37
38         dbhelper = new DatabaseHelper(this);
39
40         Session session = new Session(this);
41         userID = session.getId();           //userID
42
43         listResult = findViewById(R.id.listResult);
44         emptyList = findViewById(R.id.emptyList);
45         emptyList.setVisibility(View.GONE);
46         listID = new ArrayList<>();
47         listTitle = new ArrayList<>();
48
49         Intent i = getIntent();
50         Bundle bundle = i.getExtras();
51         if (bundle != null) {
52             status = (String) bundle.get("status");    //selected Status
53         }
54         if (bundle != null) {
55             team = (String) bundle.get("team");        //selected Team
56         }
57
58         createList();
59
60         if (listID.size() == 0) {
61             emptyList.setVisibility(View.VISIBLE);
62             listResult.setVisibility(View.GONE);
63         }
64
65         listResult.setOnItemClickListener(new AdapterView.OnItemClickListener() {
66             @Override
67             public void onItemClick(AdapterView<?> parent, View view, int position,
                   ↪ long id) {
68                 Intent i = new Intent(List.this, Task.class);
69                 i.putExtra("listID", listID);
70                 i.putExtra("taskID", position);
71                 Animation animation = AnimationUtils.loadAnimation(getBaseContext(),
                       ↪ R.anim.shake);
72                 animation.setDuration(75);
73                 view.startAnimation(animation);
74                 startActivity(i);
75             }
76         });
77
```

```
78      }
79
80      private void createList() {
81          Cursor crs = dbhelper.queryList();
82          while (crs.moveToNext()) {
83              if ((crs.getInt(4) == 0 || crs.getInt(4) == userID) && crs.getString(2).
                    ↪ equals(team) && crs.getString(3).equals(status)) {
84                  listID.add(crs.getInt(0));
85                  listTitle.add(crs.getString(1));
86              }
87          }
88          ArrayAdapter adapter = new ArrayAdapter<String>(this, R.layout.spinner_items
                ↪ , listTitle) {
89              @NonNull
90              @Override
91              public View getView(int position, @Nullable View convertView, @NonNull
                    ↪ ViewGroup parent) {
92                  View view = super.getView(position, convertView, parent);
93                  view.setBackground(getContext().getDrawable(R.drawable.
                        ↪ boarder_row_list)); //set boarder to every row in the list
94                  ViewGroup.LayoutParams params = view.getLayoutParams();
95                  params.height = 140; //height for every row in List activity
96                  view.setLayoutParams(params);
97                  return view;
98              }
99          };
100         listResult.setAdapter(adapter);
101
102         crs.close();
103         dbhelper.close();
104     }
105 }
```

Code-Anhang 31: List.java

Task.java

```
1  package com.example.techgeek.kanazubi;
2
3  import android.annotation.SuppressLint;
4  import android.content.Intent;
5  import android.database.Cursor;
6  import android.support.v7.app.AppCompatActivity;
7  import android.os.Bundle;
8  import android.text.method.ScrollingMovementMethod;
9  import android.view.View;
10 import android.widget.Button;
11 import android.widget.TextView;
12 import android.widget.Toast;
13
14 import com.example.techgeek.kanazubi.SQL.DatabaseHelper;
15 import com.example.techgeek.kanazubi.Session.Session;
16
17 import java.util.ArrayList;
18
19 public class Task extends AppCompatActivity {
20
21     TextView taskTitle, taskReleased, taskTeam, taskStatus, taskDescr;
22     Button joinTaskBtn;
```

```
23      private ArrayList<Integer> listID;
24      private int taskID, taskNr, userID;
25      private String openTask = "Offen", inProgressTask = "In Bearbeitung", buttonText
            ↪ ;
26      private DatabaseHelper dbhelper;
27
28      @Override
29      protected void onCreate(Bundle savedInstanceState) {
30          super.onCreate(savedInstanceState);
31          setContentView(R.layout.activity_task);
32
33          dbhelper = new DatabaseHelper(this);
34
35          Session session = new Session(this);
36          userID = session.getId();
37
38          taskTitle = findViewById(R.id.taskTitle);
39          taskReleased = findViewById(R.id.taskReleased);
40          taskTeam = findViewById(R.id.taskTeam);
41          taskStatus = findViewById(R.id.taskStatus);
42          taskDescr = findViewById(R.id.taskDescr);
43          taskDescr.setMovementMethod(new ScrollingMovementMethod());
44          joinTaskBtn = findViewById(R.id.joinTaskBtn);
45
46          final Bundle bundle = getIntent().getExtras();
47          if (bundle != null) {
48              taskID = bundle.getInt("taskID");
49              listID = bundle.getIntegerArrayList("listID");
50          }
51
52          assert listID != null;
53          for(int i = 0; i < listID.size(); i++){
54              if(i == taskID){
55                  taskNr = listID.get(i);
56                  addData(taskNr);
57              }
58          }
59
60          joinTaskBtn.setOnClickListener(new View.OnClickListener() {
61              @Override
62              public void onClick(View v) {
63                  if (buttonText.equals(openTask)) {
64                      dbhelper.acceptTask(userID, taskNr);
65                  } else if(buttonText.equals(inProgressTask)){
66                      dbhelper.doneTask(taskNr);
67                  }
68                  Intent i = new Intent(Task.this, Menu.class);
69                  startActivity(i);
70                  finish();
71              }
72          });
73
74      }
75
76      @SuppressLint("SetTextI18n")
77      private void addData(final int taskNr) {
78          Cursor crs = dbhelper.queryList();
79          if (crs.getCount() == 0) {
80              Toast.makeText(this, "Die Liste ist leer", Toast.LENGTH_LONG).show();
81          } else {
82              while (crs.moveToNext()) {
```

```
83                  if(taskNr == crs.getInt(0)){
84                      taskTitle.setText(crs.getString(1));
85                      taskReleased.setText("Veröffentlicht seit: " + crs.getString(5))
                            ↪ ;
86                      taskTeam.setText("Team: " + crs.getString(2));
87                      taskStatus.setText("Status: " + crs.getString(3));
88                      buttonText = crs.getString(3);
89                      taskDescr.setText(crs.getString(6));
90
91                      String doneTask = "Erledigt";
92                      if (buttonText.equals(openTask)) {
93                          joinTaskBtn.setText("Annehmen");
94                      } else if (buttonText.equals(inProgressTask)) {
95                          joinTaskBtn.setText("Erledigen");
96                      } else if(buttonText.equals(doneTask)){
97                          joinTaskBtn.setEnabled(false);
98                          joinTaskBtn.setVisibility(View.GONE);
99                      }
100                 }
101             }
102         }
103         crs.close();
104         dbhelper.close();
105     }
106 }
```

Code-Anhang 32: Task.java

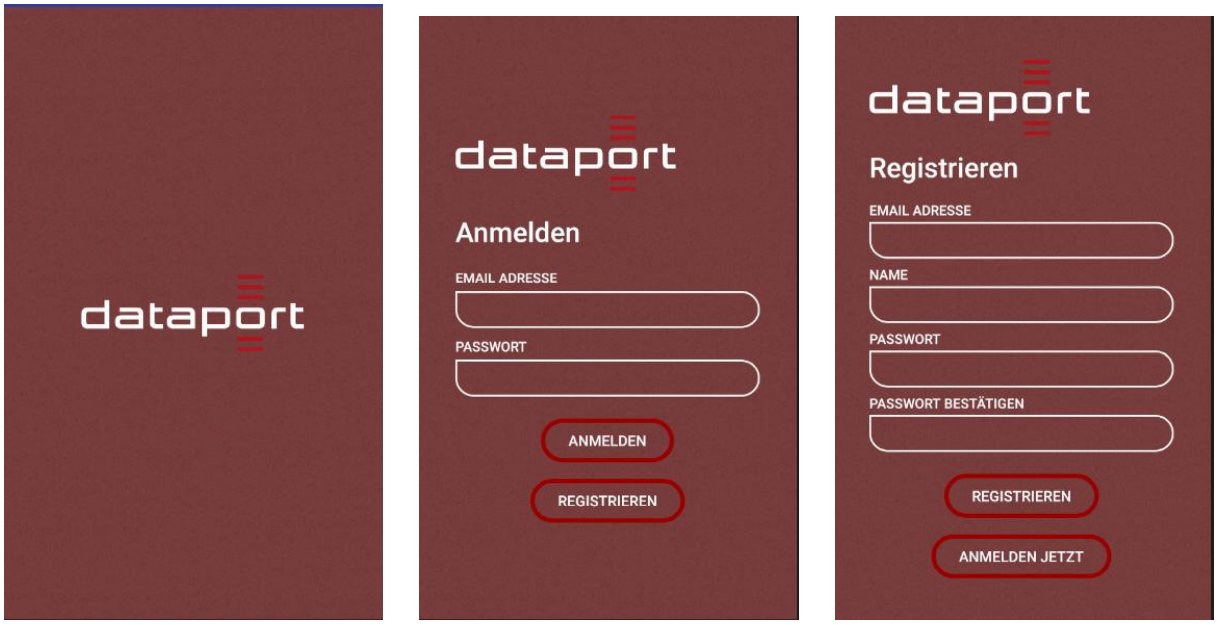
GradleBuild

```
1  dependencies {
2      implementation fileTree(dir: 'libs', include: ['*.jar'])
3      implementation 'com.android.support:appcompat-v7:27.1.1'
4      implementation 'com.android.support:support-v4:27.1.1'
5      implementation 'com.android.support:design:27.1.1'
6      implementation 'com.android.support.constraint:constraint-layout:1.1.3'
7      implementation "org.mindrot:jbcrypt:0.4"
8      testImplementation 'junit:junit:4.12'
9      androidTestImplementation 'com.android.support.test:runner:1.0.2'
10     androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2
           ↪ '
11     compile 'com.android.support:support-annotations:28.0.0'
12     androidTestImplementation 'com.android.support.test:rules:1.0.2'
13 }
```

Code-Anhang 33: GradleBuild

D Abbildungen

Abbildung 2: Screenshots: Splashscreen, Anmelden, Registrieren

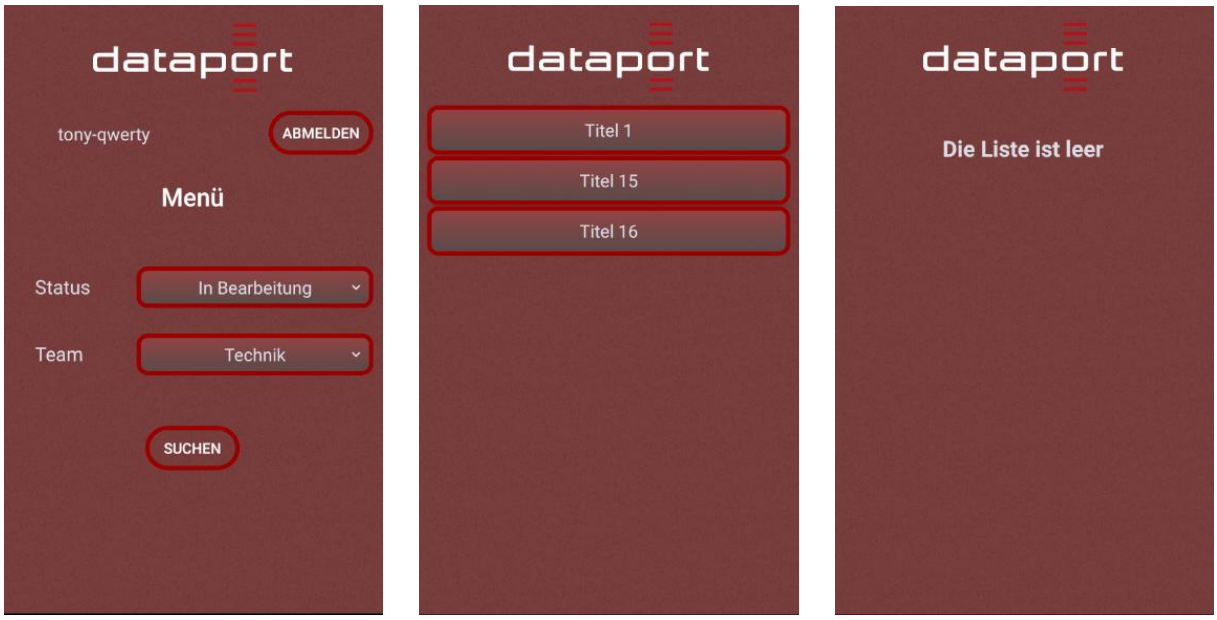


(a) Splashscreen (b) Anmelden (c) Registrieren

Abbildung 3: Screenshots: Admin, Admin-Plattform

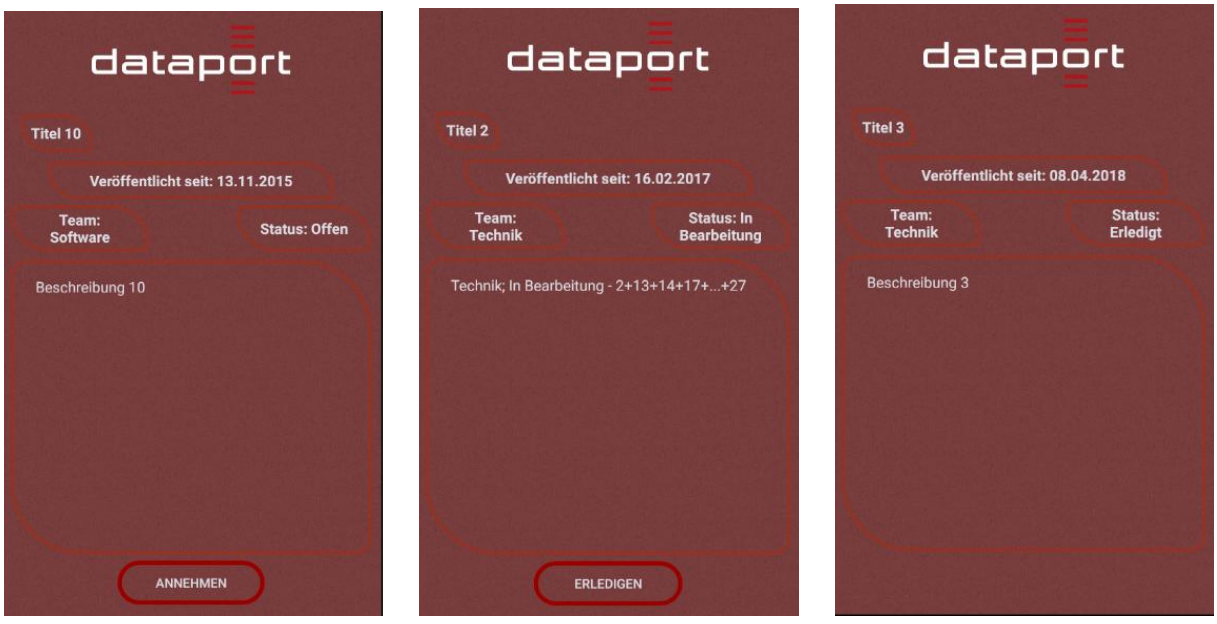
(a) Admin (b) Admin-Plattform

Abbildung 4: Screenshots: Benutzer, Listenergebnisse, Leere Liste



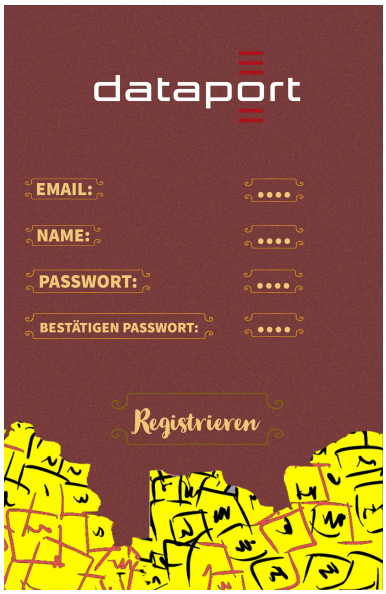
(a) Benutzer (b) Listenergebnisse (c) Leere Liste

Abbildung 5: Screenshots: Offen, In Bearbeitung, Erledigt

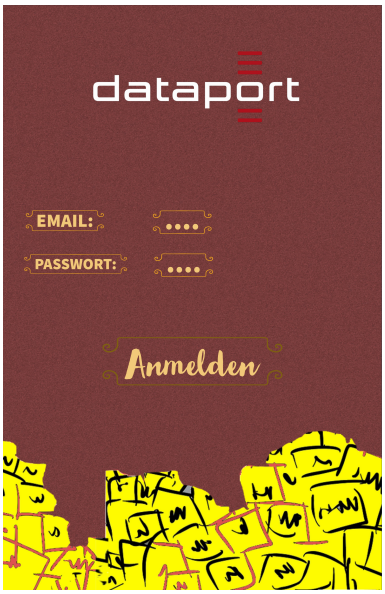


(a) Offen (b) In Bearbeitung (c) Erledigt

Abbildung 6: Screenshots: Menu, Mockup Beispiel, Mockup Realisierung



(a) Registrieren



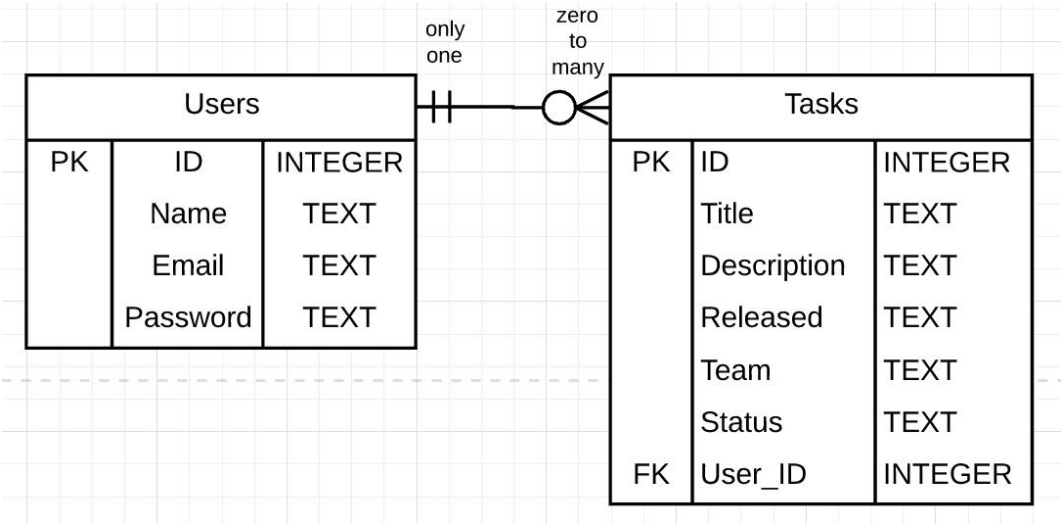
(b) Mockup Anmeldung

(c) Mockup Realisierung

Abbildung 7: Rund Icon

(a) Rund Icon

Abbildung 8: ERD, Tabelle users, Tabelle tasks



(a) ERD

	ID	Email	Password	Name
1	1	admin@dataport.de	$2a$12$RioU.yn/Qu4WBYhIwKZ18uadT2bYyO6JF1O5XXuIyjOWqgyKTphKS	admin-123456
2	2	tony@gmail.com	$2a$12$zkTNjGy8UDWOlUeUIy75GOIloUXjSkQhvEPZP1pQvbwlmXpGV/0SO	tony-qwerty
3	3	tonkata@gmx.de	$2a$12$HoSs/eXOtVc3zEbIkuUY2OullVAdN0rE1te1IqX6GU0PZVRmTXbgi	tonkata-123456

(b) Tabelle users

	ID	Title	Description	Released	Team	Status	User_ID
9	9	Titel 9	Beschreibung 9	06.04.2018	Marketing	Erledigt	3
10	10	Titel 10	Beschreibung 10	13.11.2015	Software	Offen	0
11	11	Titel 11	Beschreibung 11	23.08.2014	Software	In Bearbeitung	3
12	12	Titel 12	Beschreibung 12	09.09.2018	Software	Erledigt	2
13	13	Titel 13	Technik; In Bearbeitung - 2,13,14	17.02.2017	Technik	In Bearbeitung	2
14	14	Titel 14	Technik; In Bearbeitung - 2,13,14	07.09.2017	Technik	In Bearbeitung	2
15	15	Titel 15	Technik; Offen - 1+15+16	14.12.2017	Technik	Offen	0
16	16	Titel 16	Technik; Offen - 1+15+16	12.01.2016	Technik	Offen	0

(c) Tabelle tasks

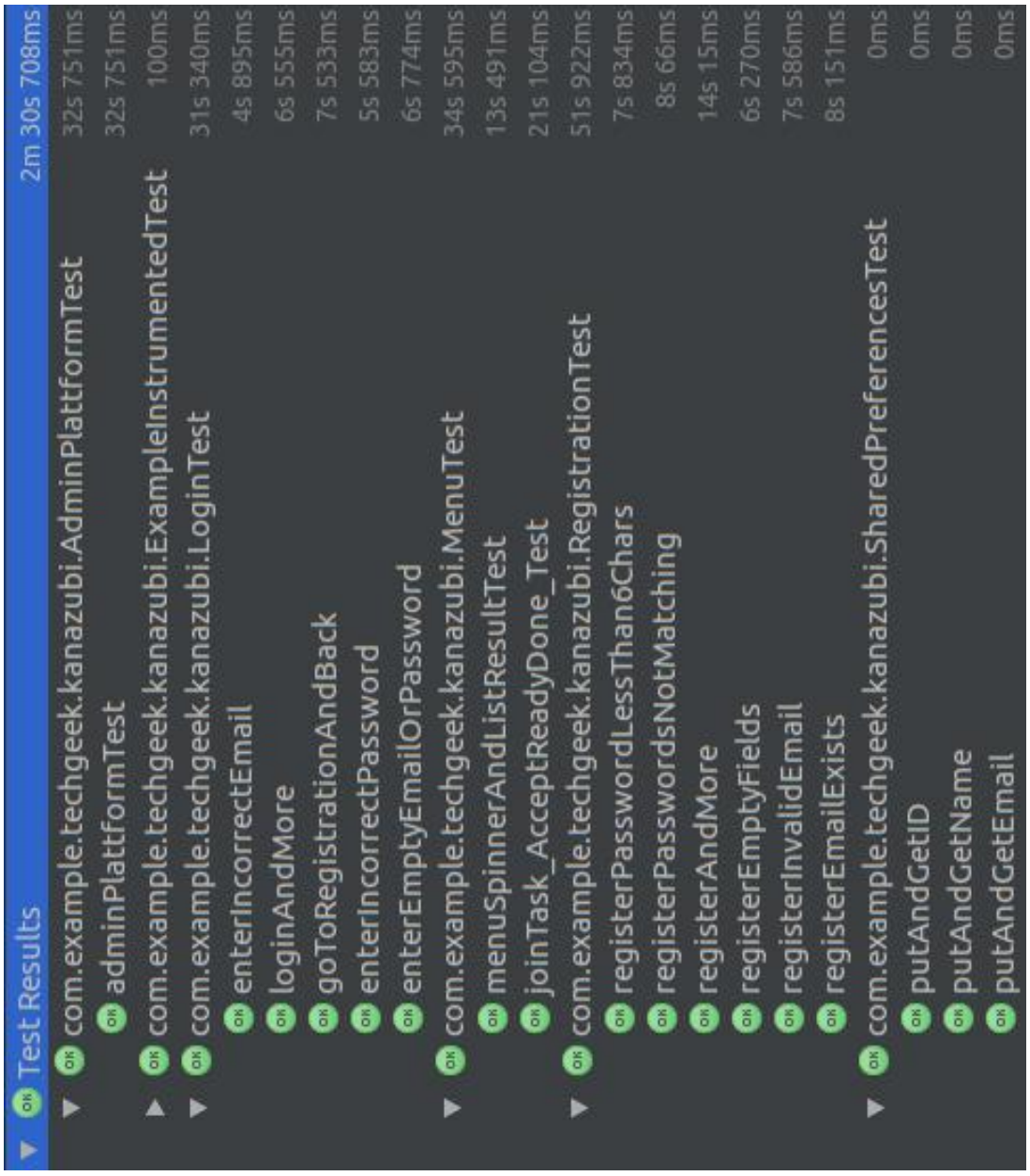


Abbildung 9: Alle automatisierte Tests

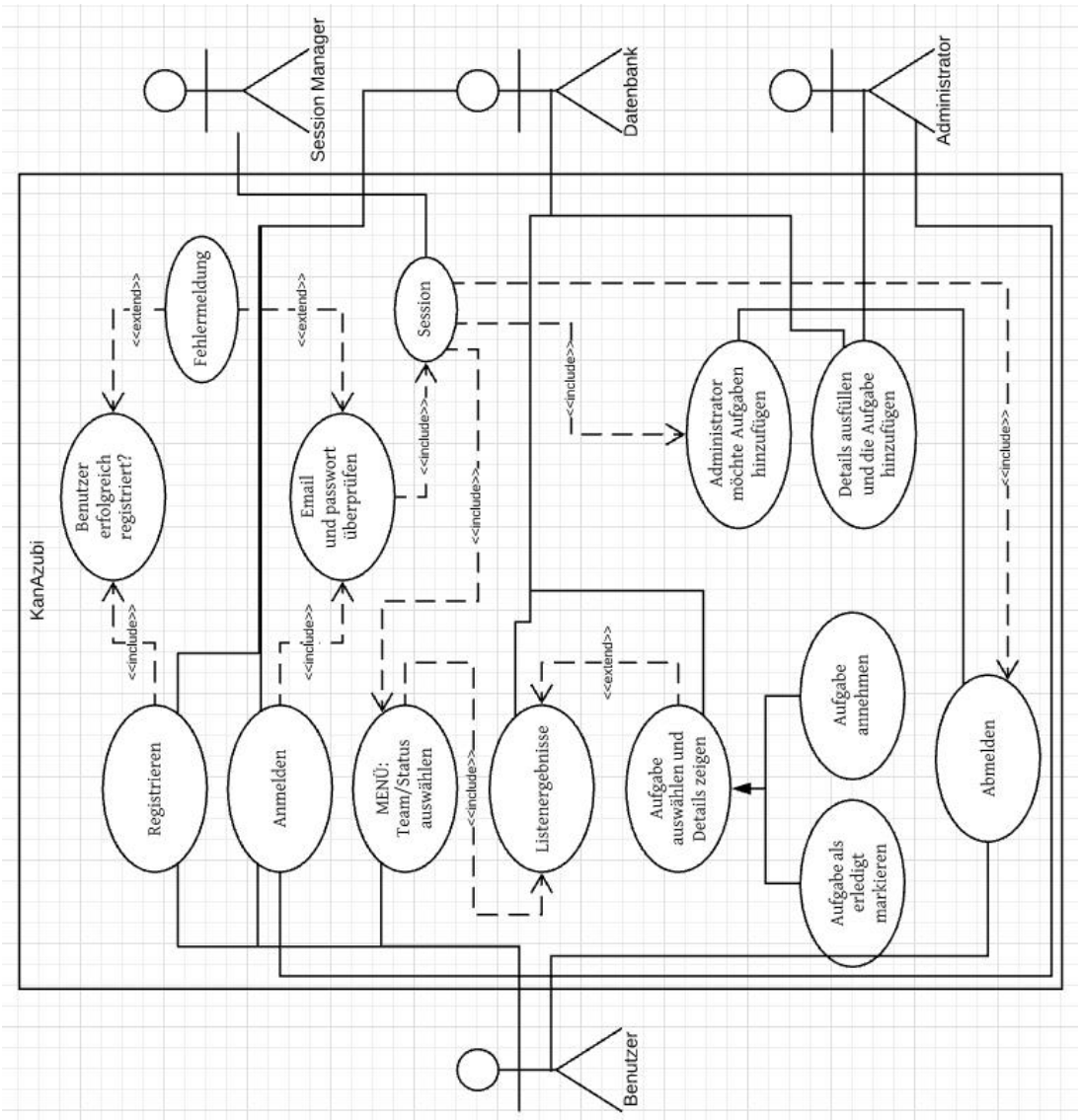


Abbildung 10: Anwendungsfalldiagramm

Abbildung 11: Alle Klassen, Funktionen und Variablen