SWE 4x **Übung zu Softwareentwicklung mit mit modernen Plattformen 4** **SS 2022, Übung 05**

Abgabe elektronisch bis Sa 8 Uhr in der KW 20

☐ ~~Gr. 1,~~ Winkler, BSc Msc **Name: Roman Kofler-Hofer** **Aufwand in h: 35**

☐ **Gr. 2,** Dr. Pitzer

Punkte: ________ **Kurzzeichen Tutor / Übungsleiter ____/____**

Beispiel	L Lösungsidee	I Implementierung	T Tests	S = L+I+T	Multiplikator	S*M
a	☒☒☒	☒☒☒☒	☒☒☒	10	5	50
b	☒☒☒	☒☒☒☒	☒☒☒	10	5	50
					Summe	**100**

1. Lösungsidee

1.1. Allgemein

Um das komplexe Problem besser strukturieren zu können und mir einen Überblick zu verschaffen, habe ich zuerst ein Klassendiagramm erstellt. Nach längerem Überlegen habe ich mich dazu entschlossen, dass es zwei separate Klassen für Bedarf und Hilfsgut benötigt. D.h. eine Annahmestelle meldet einen Bedarf für ein im System hinterlegten Hilfsgut an. Da es sich um eine M:N Beziehung handelt und weitere Infos je Bedarf gespeichert werden sollen (Menge), habe ich die Klasse "Bedarf" als Assoziationsklasse modelliert.

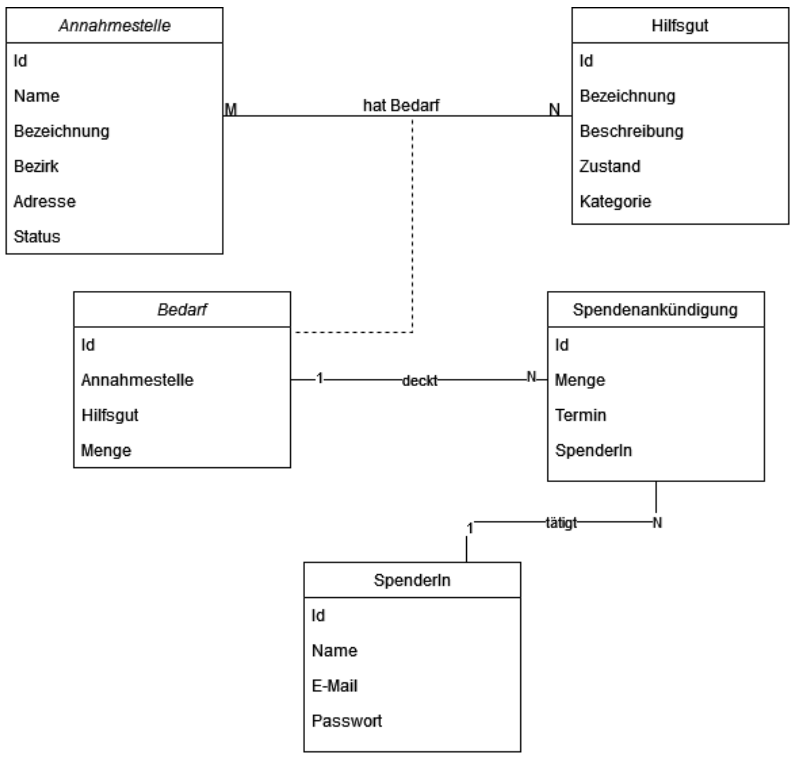
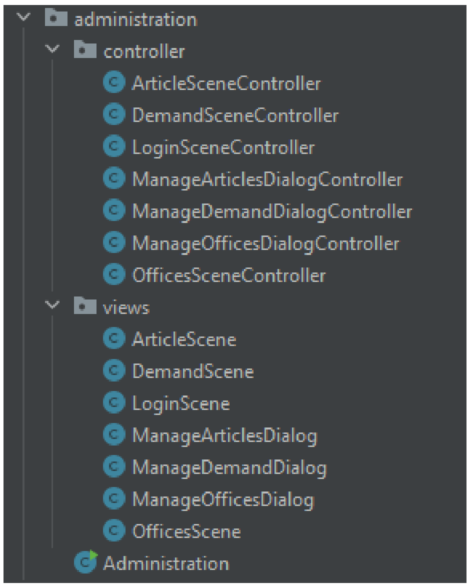


Abbildung 1: Klassendiagramm

Design / Programmstruktur:

Bezüglich des Designs habe ich versucht ein MVC-Pattern umzusetzen. Dabei wird versucht die Datenschicht von der Darstellung möglichst zu trennen. Der Controller interagiert als Vermittler zwischen View und Model (Model = Daten) und ist unter anderem für das Eventhandling verantwortlich. Das Model sind wie gesagt die Daten. Diese verwalte ich in einer Klasse "dbMock". Hier lege ich die Dummydaten an und mache diese mit Gettern und Methoden wie "addDemand", "deleteArticle" usw zugänglich. Beide Anwendungen greifen auf diese Klasse zu. In einer weiteren Ausbaustufe ist es mein Ziel, nur die Implementierung der dbMock-Klasse durch den tatsächlichen DB-Zugriff austauschen zu müssen.

Bei mir ist jedes Dialogfenster und jede Ansicht eine eigene Klasse und somit eine eigene View. Lt. MVC-Pattern gibt es für jede View einen Controller, was ich auch so umgesetzt habe. Der folgende Screenshot zeigt die vorhandenen Klassen für die Verwaltungsanwendung (8 Views und für jede View ein Controller).

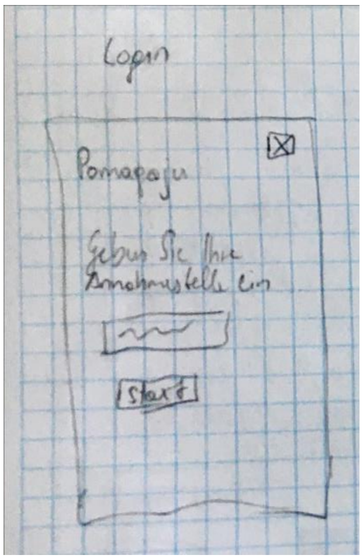


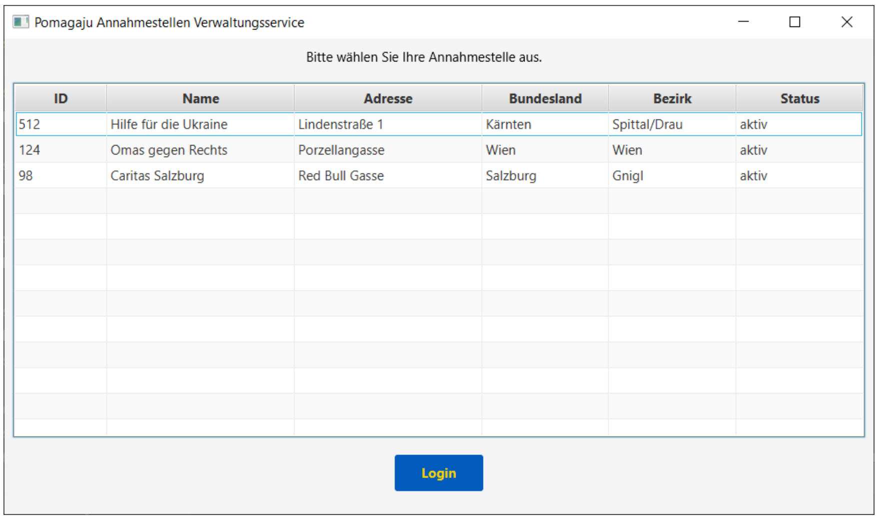
```
administration
  controller
    ArticleSceneController
    DemandSceneController
    LoginSceneController
    ManageArticlesDialogController
    ManageDemandDialogController
    ManageOfficesDialogController
    OfficesSceneController
  views
    ArticleScene
    DemandScene
    LoginScene
    ManageArticlesDialog
    ManageDemandDialog
    ManageOfficesDialog
    OfficesScene
  Administration
```

1.2. Verwaltungsanwendung:

Im Folgenden gebe ich einen Überblick über die Mockups und die tatsächlichen Screens der Verwaltungsanwendung. Während dem Programmieren haben sich die Mockups natürlich auch weiterentwickelt und verändert, da man immer wieder auf neue Punkte drauf gekommen ist, die ich zu Beginn nicht berücksichtigt habe.

Laut Angabe wurde gefordert, dass ein User zuerst eine/seine Annahmestelle auswählen muss und sich alle weiteren Operationen dann auf diese Annahmestelle beziehen sollen. Daher habe ich folgenden Startscreen für die Verwaltungsapp implementiert. Der User wählt seine Annahmestelle aus einer Liste. Im Vergleich zum Mock-Up habe ich mich für eine Tabelle und kein Drop-Down-Feld entschieden, da das etwas unübersichtlicher werden kann bei vielen Annahmestellen im System.





Pomagaju Annahmestellen Verwaltungsservice

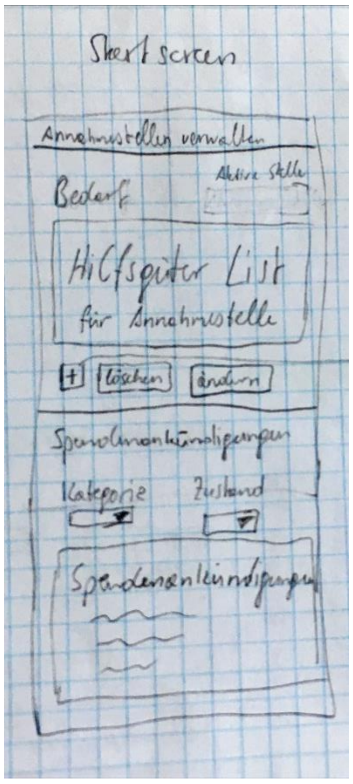
Bitte wählen Sie Ihre Annahmestelle aus.

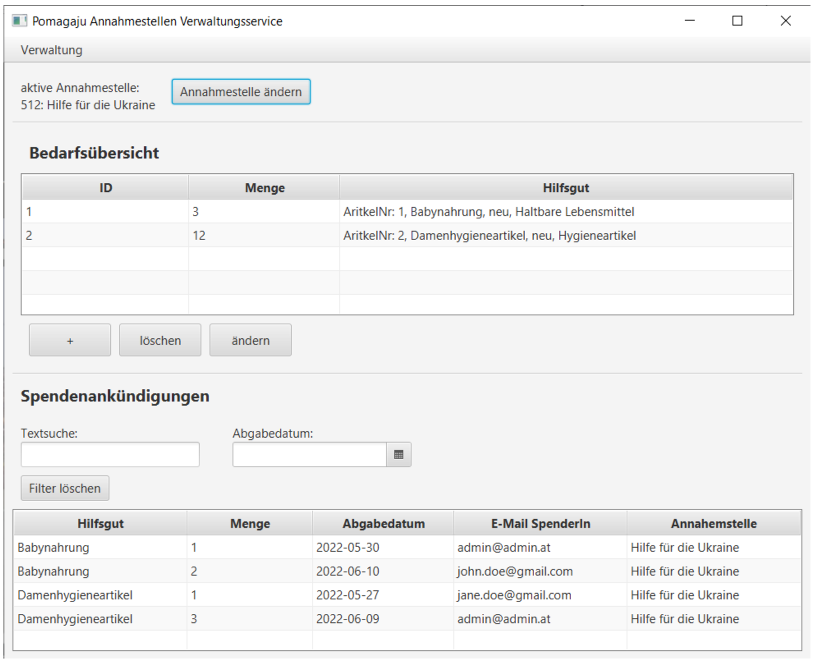
ID	Name	Adresse	Bundesland	Bezirk	Status
512	Hilfe für die Ukraine	Lindenstraße 1	Kärnten	Spittal/Drau	aktiv
124	Omas gegen Rechts	Porzellangasse	Wien	Wien	aktiv
98	Caritas Salzburg	Red Bull Gasse	Salzburg	Gnigl	aktiv

Sobald der User seine Annahmestelle ausgewählt hat, erhält er ein Fenster mit folgenden Inhalten:

- Top Navigation

- Info zur aktiven Annahmestelle und Möglichkeit diese zu ändern
- Tabelle mit den gemeldeten Bedarfsmeldungen dieser Annahmestelle
- Buttons, um den Bedarf zu verwalten (hinzufügen, löschen, ändern)
- Tabelle mit den Spendenankündigungen für diese Annahmestelle und Filtermöglichkeiten





aktive Annahmestelle:
512: Hilfe für die Ukraine

Bedarfsübersicht

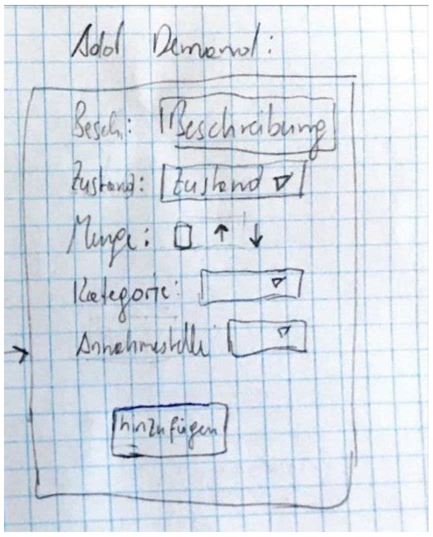
ID	Menge	Hilfsgut
1	3	AritkelNr: 1, Babynahrung, neu, Haltbare Lebensmittel
2	12	AritkelNr: 2, Damenhygieneartikel, neu, Hygieneartikel

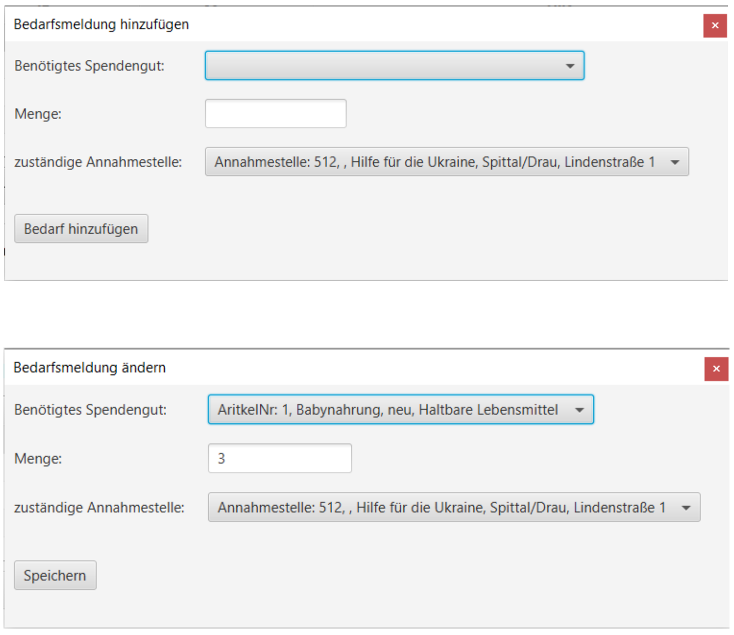
Spendenankündigungen

Textsuche:
Abgabedatum:

Hilfsgut	Menge	Abgabedatum	E-Mail SpenderIn	Annahemstelle
Babynahrung	1	2022-05-30	admin@admin.at	Hilfe für die Ukraine
Babynahrung	2	2022-06-10	john.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	1	2022-05-27	jane.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	3	2022-06-09	admin@admin.at	Hilfe für die Ukraine

Die Bedarfsmeldungen können über ein Dialogfeld verwaltet werden:




Bedarfsmeldung hinzufügen

Benötigtes Spendengut:
Menge:
zuständige Annahmestelle: Annahmestelle: 512, , Hilfe für die Ukraine, Spittal/Drau, Lindenstraße 1

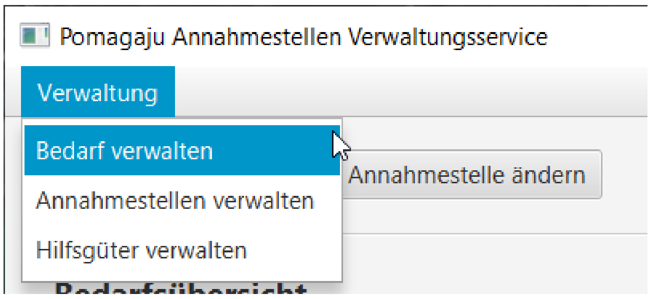
Bedarfsmeldung ändern

Benötigtes Spendengut: AritkelNr: 1, Babynahrung, neu, Haltbare Lebensmittel
Menge: 3
zuständige Annahmestelle: Annahmestelle: 512, , Hilfe für die Ukraine, Spittal/Drau, Lindenstraße 1

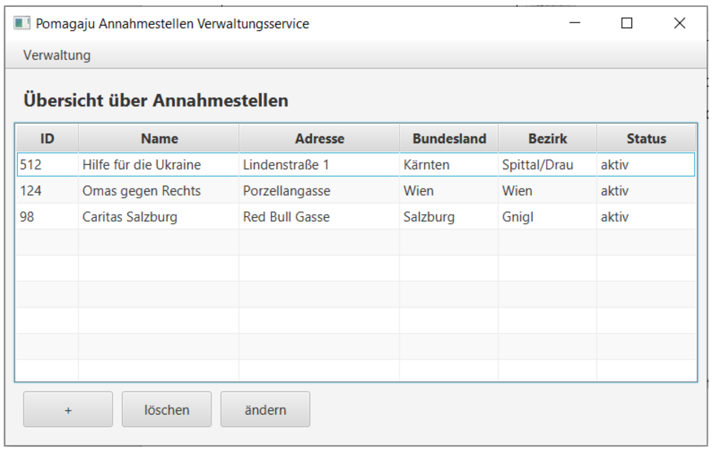
Über die Navigation-Bar können mehrere Submenüs ausgewählt werden

- Bedarf verwalten (das ist die oberhalb gesehen Ansicht ➔ also eigentlich die Hauptansicht)

- Annahmestellen vewalten ➔ hier können die Annahmestellen verwaltet werden
- Hilfsgüter verwalten ➔ zum Verwalten der Hilfsgüter



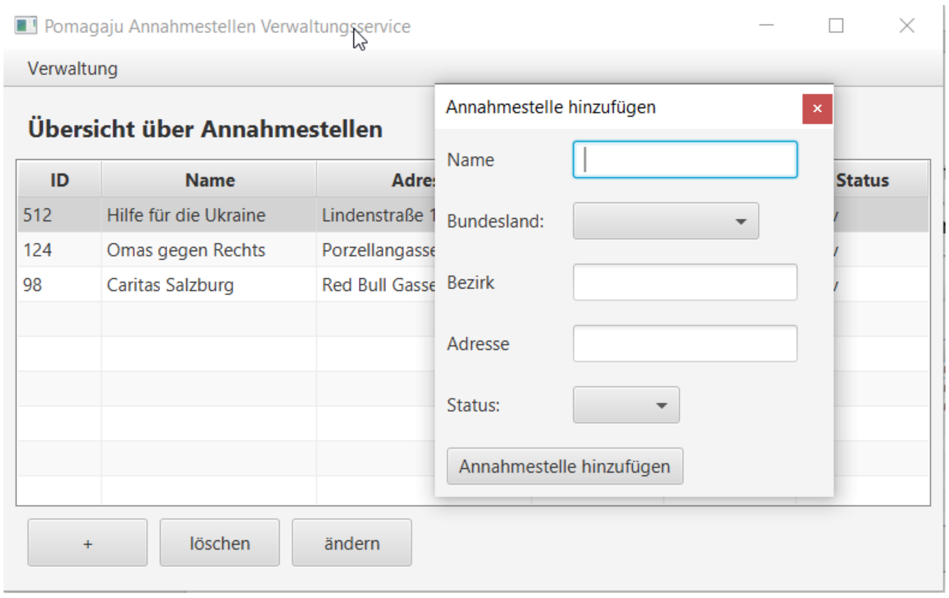
Die beiden Menüs "Annahemstellen verwalten" und "Hilfsguter verwalten" sind beide ident aufgebaut. In einer Tabelle werden die Infos dargestellt.



Übersicht über Annahmestellen

ID	Name	Adresse	Bundesland	Bezirk	Status
512	Hilfe für die Ukraine	Lindenstraße 1	Kärnten	Spittal/Drau	aktiv
124	Omas gegen Rechts	Porzellangasse	Wien	Wien	aktiv
98	Caritas Salzburg	Red Bull Gasse	Salzburg	Gnigl	aktiv

Über Buttons unterhalb der Tabelle können die Daten verwaltet werden. Dazu wird jeweils ein Dialogfenster geöffnet.



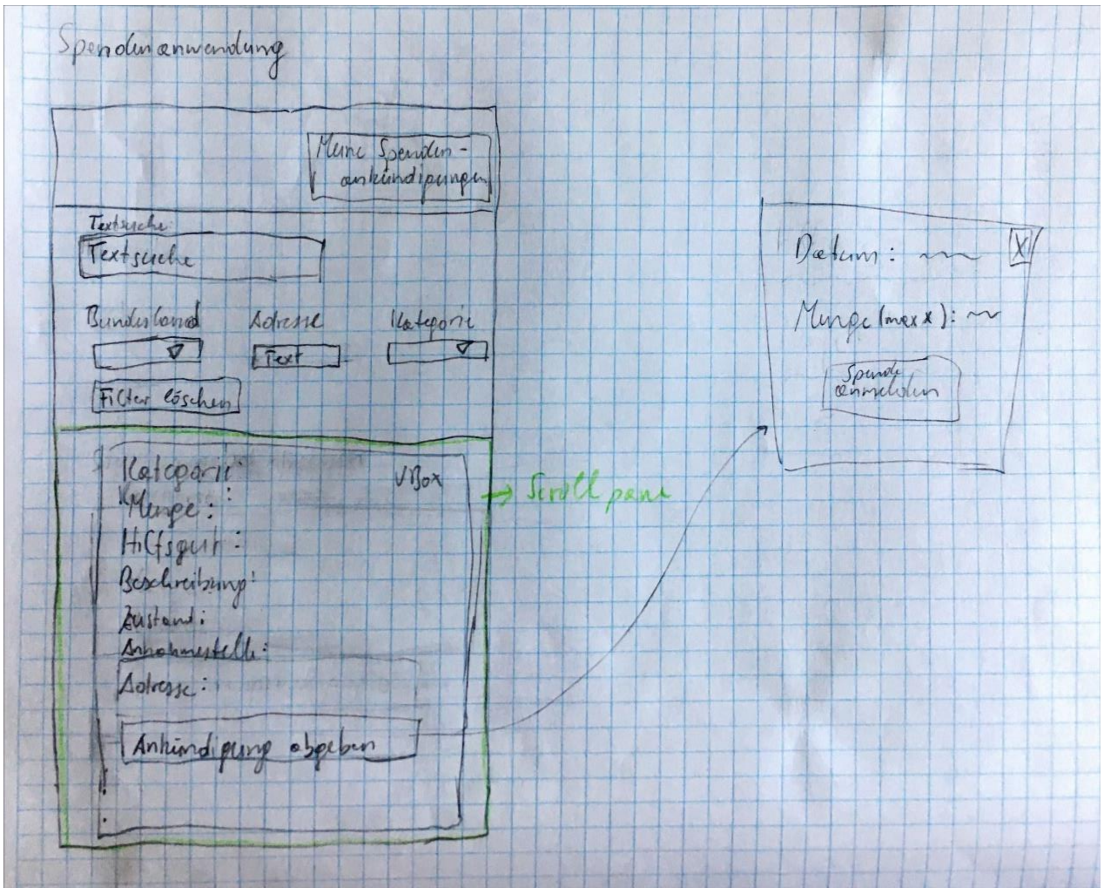
Annahmestelle hinzufügen

Name	
Bundesland:	
Bezirk	
Adresse	
Status:	

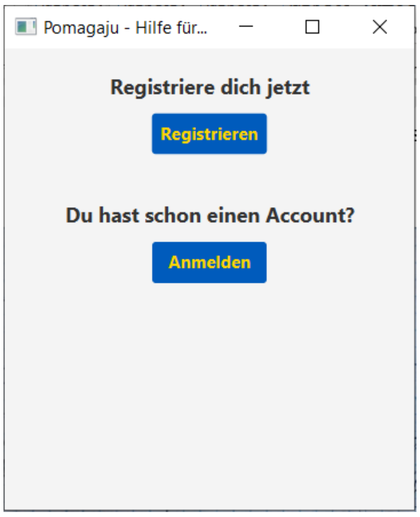
Annahmestelle hinzufügen

1.3. Spendenanwendung:

In der Spendenanwendung sollen die Bedarfsmeldungen der Annahmestellen angezeigt werden. Der User soll die Möglichkeit haben die Bedarfsmeldungen nach verschiedenen Kriterien filtern zu können. Ich habe versucht die Bedarfsmeldungen kompakter und eher im Hochformat darzustellen, damit diese Darstellungen auch auf Mobilgeräte übertragen werden könnte. Die Ankündigung einer Spende funktioniert wieder über eine Pop-Up in welchem nach Abschluss des Prozesses auch der Token angezeigt wird.



Login mit "admin" und "admin":

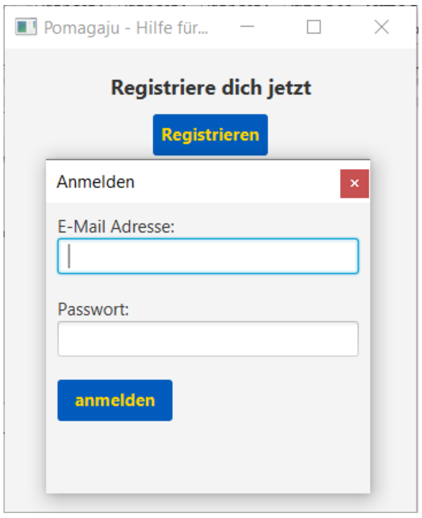

Pomagaju - Hilfe für...

Registriere dich jetzt

Registrieren

Du hast schon einen Account?

Anmelden


Pomagaju - Hilfe für...

Registriere dich jetzt

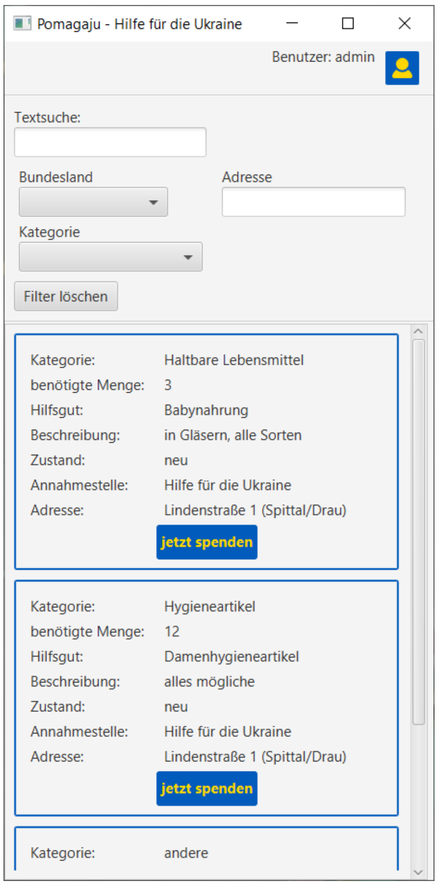
Registrieren

Anmelden

E-Mail Adresse:

Passwort:

anmelden


Pomagaju - Hilfe für die Ukraine

Benutzer: admin

Textsuche:

Bundesland

Adresse

Kategorie

Filter löschen

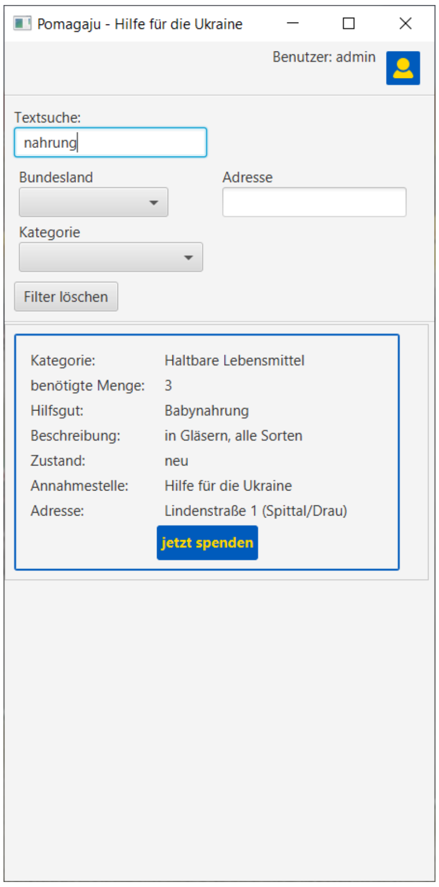
Kategorie:	Haltbare Lebensmittel
benötigte Menge:	3
Hilfsgut:	Babynahrung
Beschreibung:	in Gläsern, alle Sorten
Zustand:	neu
Annahmestelle:	Hilfe für die Ukraine
Adresse:	Lindenstraße 1 (Spittal/Drau)

jetzt spenden

Kategorie:	Hygieneartikel
benötigte Menge:	12
Hilfsgut:	Damenhygieneartikel
Beschreibung:	alles mögliche
Zustand:	neu
Annahmestelle:	Hilfe für die Ukraine
Adresse:	Lindenstraße 1 (Spittal/Drau)

jetzt spenden

Kategorie:	andere


Pomagaju - Hilfe für die Ukraine

Benutzer: admin

Textsuche:

nahrung

Bundesland

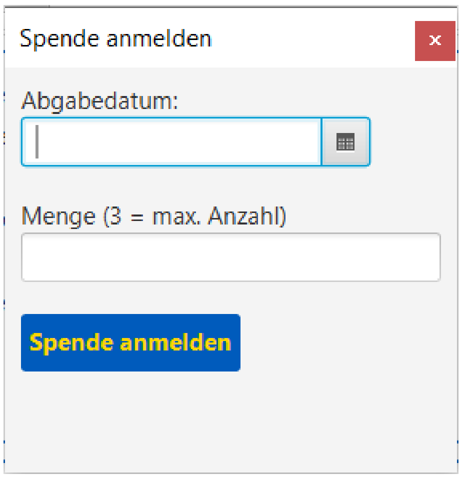
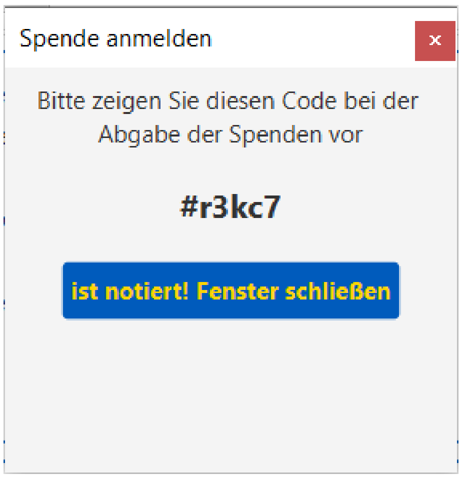
Adresse

Kategorie

Filter löschen

Kategorie:	Haltbare Lebensmittel
benötigte Menge:	3
Hilfsgut:	Babynahrung
Beschreibung:	in Gläsern, alle Sorten
Zustand:	neu
Annahmestelle:	Hilfe für die Ukraine
Adresse:	Lindenstraße 1 (Spittal/Drau)

jetzt spenden

 Spende anmelden Abgabedatum: Menge (3 = max. Anzahl)	 Spende anmelden Bitte zeigen Sie diesen Code bei der Abgabe der Spenden vor **#r3kc7**

Laut Angabe ist gefordert, dass der User neben dem Datum und der Menge auch die Annahmestelle auswählen kann. Nachdem eine Bedarfsmeldung aber zu genau einer Annahmestelle gehört, erscheint es mir unsinnig, dass der User eine beliebige Stelle auswählen kann.

2. Quellcode

2.1. Datenmodell (Model):

```java
package at.fhooe.swe4.model.enums;
//File: Category.java

public enum Category {
  CLOTHS("Kleidung"),
  ELECTRONICS("Elektronikgeräte"),
  FOOD("Haltbare Lebensmittel"),
  HYGIENE("Hygieneartikel"),
  OTHER("andere");

  private String stringRepresentation;

  public String getCategory() {
    return stringRepresentation;
  }

  @Override
  public String toString() {
    return stringRepresentation;
  }

  private Category(String category) {
    this.stringRepresentation = category;
  }

}
```

```java
package at.fhooe.swe4.model.enums;
//File: Condition.java

public enum Condition {
  NEW("neu"),
  SLIGHTLYUSED("wenig benutzt"),
  HEAVILYUSED("stark benutzt");

  private String stringRepresentation;
  private Condition(String condition) {
    this.stringRepresentation = condition;
  }

  @Override
  public String toString() {
    return stringRepresentation;
  }
}


package at.fhooe.swe4.model.enums;
//File: FederalSate.java

public enum FederalState {
  VORARLBERG("Vorarlberg"),
  TYROL("Tirol"),
  SALZBURG("Salzburg"),
  CARINTHIA("Kärnten"),
  LOWERAUSTRIA("Niederösterreich"),
  UPPERAUSTRIA("Oberösterreich"),
  STYRIA("Steiermark"),
  BURGENLAND("Burgenland"),
  VIEANNA("Wien");

  private String stringRepresentation;

  private FederalState(String federalState) {
    this.stringRepresentation = federalState;
  }

  @Override
  public String toString() {
    return stringRepresentation;
  }
}


package at.fhooe.swe4.model.enums;
//File: Status.java
```

```
public enum Status {
  ACTIVE("aktiv"),
  INACTIVE("inaktiv");

  private String stringRepresentation;

  private Status(String status) {
    this.stringRepresentation = status;
  }

  @Override
  public String toString() {
    return stringRepresentation;
  }
}
```

```java
package at.fhooe.swe4.model;
//File: Article.java

import at.fhooe.swe4.model.enums.Category;
import at.fhooe.swe4.model.enums.Condition;

public class Article {
  private Integer id;
  private String name;
  private String description;
  private Condition condition;
  private Category category;

  public Article(Integer id, String name, String description, Condition condition, Category category) {
    this.id = id;
    this.name = name;
    this.description = description;
    this.condition = condition;
    this.category = category;
  }

  @Override
  public String toString() {
    return
            "AritkelNr: " + id + ", " + name +  ", " + condition + ", " + category;
  }

  public Integer getId() {
    return id;
  }

  public void setId(Integer id) {
    this.id = id;
  }

  public String getName() {
    return name;
  }

  public void setName(String name) {
    this.name = name;
  }

  public String getDescription() {
    return description;
  }

  public void setDescription(String description) {
    this.description = description;
  }

  public Condition getCondition() {
    return condition;
  }

  public void setCondition(Condition condition) {
    this.condition = condition;
  }

  public Category getCategory() {
    return category;
  }

  public void setCategory(Category category) {
    this.category = category;
  }
}


package at.fhooe.swe4.model;
//File: DemandItem.java

import javafx.beans.property.IntegerProperty;
import javafx.beans.property.SimpleIntegerProperty;

public class DemandItem {
  private Integer id;
  private Article relatedArticle;
  private ReceivingOffice relatedOffice;
  private IntegerProperty amount;

  public Integer getId() {
```

```java
    return id;
  }

  public void setId(Integer id) {
    this.id = id;
  }

  public DemandItem(Integer id, Article relatedGood, ReceivingOffice relatedOffice, Integer amount) {
    this.id = id;
    this.relatedArticle = relatedGood;
    this.relatedOffice = relatedOffice;
    this.amount = new SimpleIntegerProperty();
    this.amount.setValue(amount);
  }

  public IntegerProperty getObsAmount() {
    return amount;
  }

  @Override
  public String toString() {
    return  id + ", " + relatedArticle + ", " + relatedOffice + ", " + amount;
  }

  public Article getRelatedArticle() {
    return relatedArticle;
  }

  public void setRelatedArticle(Article relatedArticle) {
    this.relatedArticle = relatedArticle;
  }

  public ReceivingOffice getRelatedOffice() {
    return relatedOffice;
  }

  public void setRelatedOffice(ReceivingOffice relatedOffice) {
    this.relatedOffice = relatedOffice;
  }

  public Integer getAmount() {
    return amount.getValue();
  }

  public void setAmount(Integer amount) {
    this.amount.setValue(amount);
  }
}


package at.fhooe.swe4.model;
//File: Donation.java

import java.time.LocalDate;

public class Donation {
  private DemandItem relatedDemand;
  private LocalDate date;
  private User user;
  private Integer amount;

  public Donation(DemandItem relatedDemand, LocalDate date, User user, Integer amount) {
    this.relatedDemand = relatedDemand;
    this.date = date;
    this.user = user;
    this.amount = amount;
  }

  @Override
  public String toString() {
    return relatedDemand +", " + date +
           ", " + user +
           ", " + amount;
  }

  public DemandItem getRelatedDemand() {
    return relatedDemand;
  }

  public void setRelatedDemand(DemandItem relatedDemand) {
    this.relatedDemand = relatedDemand;
  }
  public LocalDate getDate() {
    return date;
  }

  public void setDate(LocalDate date) {
    this.date = date;
  }

  public User getUser() {
    return user;
  }

  public void setUser(User user) {
    this.user = user;
  }

  public Integer getAmount() {
    return amount;
  }

  public void setAmount(Integer amount) {
    this.amount = amount;
  }
}


package at.fhooe.swe4.model;
//File: ReceivingOffice.java

import at.fhooe.swe4.model.enums.FederalState;
import at.fhooe.swe4.model.enums.Status;

import java.util.Objects;

public class ReceivingOffice {
  private Integer id;
  private String name;
  private FederalState federalState;
  private String district;
  private String address;
  private Status status;

  public ReceivingOffice(Integer id, String name, FederalState federalState, String district,
                         String address, Status status) {
    this.id = id;
    this.name = name;
    this.federalState = federalState;
    this.district = district;
    this.address = address;
    this.status = status;
  }

  @Override
  public String toString() {
    return "Annahmestelle: " + id + ", "
            + ", "+ name +", " + district + ", " + address;
  }

  @Override
  public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    ReceivingOffice office = (ReceivingOffice) o;
    return id.equals(office.id);
  }

  @Override
  public int hashCode() {
    return Objects.hash(id);
  }

  public Integer getId() {
    return id;
  }

  public void setId(Integer id) {
    this.id = id;
  }

  public String getName() {
    return name;
  }

  public void setName(String name) {
    this.name = name;
  }

  public FederalState getFederalState() {
    return federalState;
  }

  public void setFederalState(FederalState federalState) {
    this.federalState = federalState;
  }

  public String getDistrict() {
    return district;
  }

  public void setDistrict(String district) {
    this.district = district;
  }

  public String getAddress() {
    return address;
  }

  public void setAddress(String address) {
    this.address = address;
  }

  public Status getStatus() {
    return status;
  }

  public void setStatus(Status status) {
    this.status = status;
  }
}
```

```
package at.fhooe.swe4.model;
```

```java
//File: User.java

import java.util.Objects;

public class User {
  private String name;
  private String password;
  private String email;

  public User(String name, String password, String email) {
    this.name = name;
    this.password = password;
    this.email = email;
  }

  @Override
  public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    User user = (User) o;
    return Objects.equals(name, user.name) && Objects.equals(password, user.password) && Objects.equals(email, user.email);
  }

  @Override
  public int hashCode() {
    return Objects.hash(name, password, email);
  }

  @Override
  public String toString() {
    return  name + ", " + email;
  }

  public String getEmail() {
    return email;
  }

  public void setEmail(String email) {
    this.email = email;
  }
  public String getName() {
    return name;
  }

  public void setName(String name) {
    this.name = name;
  }

  public String getPassword() {
    return password;
  }

  public void setPassword(String password) {
    this.password = password;
  }
}
```

```java
package at.fhooe.swe4.model;
//File: dbMock.java

import at.fhooe.swe4.model.enums.Category;
import at.fhooe.swe4.model.enums.Condition;
import at.fhooe.swe4.model.enums.FederalState;
import at.fhooe.swe4.model.enums.Status;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;

import java.time.LocalDate;
import java.util.HashMap;

public class dbMock {
  private static dbMock obj;

  private static final ObservableList<DemandItem> demand =
          FXCollections.observableArrayList();

  private static final ObservableList<Donation> donations =
          FXCollections.observableArrayList();

  private static final ObservableList<Article> articles =
          FXCollections.observableArrayList();

  private static final ObservableList<ReceivingOffice> offices =
          FXCollections.observableArrayList();
  private static ReceivingOffice activeOffice;

  //key = email (String); value = user object
  private static HashMap<String, User> users = new HashMap<>();
  private static User currentUser;

  private dbMock() {};

  public static dbMock getInstance() {
    if(obj == null) {
      obj = new dbMock();
      obj.initWithTestData();
    }
    return obj;
  }

  private void initWithTestData(){

    //articles
    Article temp = new Article(1, "Babynahrung", "in Gläsern, alle Sorten", Condition.NEW, Category.FOOD);
    Article temp2 = new Article(2, "Damenhygieneartikel", "alles mögliche", Condition.NEW, Category.HYGIENE);
    Article temp3 = new Article(3, "Spielzeug", "für alle Altersstufen", Condition.NEW, Category.OTHER);
    Article temp4 = new Article(4, "Handys", "frei, müssen keine Smartphones sein", Condition.NEW, Category.ELECTRONICS);
    articles.addAll(temp, temp2, temp3, temp4);

    //Offices
    offices.add(new ReceivingOffice(512, "Hilfe für die Ukraine", FederalState.CARINTHIA,
            "Spittal/Drau", "Lindenstraße 1", Status.ACTIVE));
    offices.add(new ReceivingOffice(124, "Omas gegen Rechts", FederalState.VIEANNA,
            "Wien", "Porzellangasse", Status.ACTIVE));
    offices.add(new ReceivingOffice(98, "Caritas Salzburg", FederalState.SALZBURG,
            "Gnigl", "Red Bull Gasse", Status.ACTIVE));

    //demand items
    demand.add(new DemandItem(1, articles.get(0), offices.get(0), 3));
    demand.add(new DemandItem(2, articles.get(1), offices.get(0), 12));
    demand.add(new DemandItem(3, articles.get(2), offices.get(1), 4));

    //users
    users.put("admin", new User("admin", "admin", "admin@admin.at"));
    users.put("john.doe@gmail.com", new User("Jon Doe", "1234", "john.doe@gmail.com"));
    users.put("jane.doe@gmail.com", new User("Jane Doe", "save", "jane.doe@gmail.com"));

    //donations
    Donation d1 = new Donation(demand.get(0), LocalDate.of(2022,5,30), getUserByEMail("admin"),1);
    Donation d2 = new Donation(demand.get(0), LocalDate.of(2022,6,10), getUserByEMail("john.doe@gmail.com"),2);
    Donation d3 = new Donation(demand.get(1), LocalDate.of(2022,5,27), getUserByEMail("jane.doe@gmail.com"),1);
    Donation d4 = new Donation(demand.get(1), LocalDate.of(2022,6,9), getUserByEMail("admin"),3);
    Donation d5 = new Donation(demand.get(2), LocalDate.of(2022,6,7), getUserByEMail("john.doe@gmail.com"),4);
    Donation d6 = new Donation(demand.get(2), LocalDate.of(2022,6,10), getUserByEMail("jane.doe@gmail.com"),2);
    donations.addAll(d1,d2,d3,d4,d5,d6);
  }

  /*
  ------------------------Demand data------------------------
```

```
 */

public ObservableList<DemandItem> getDemandItems() {return demand;}

public boolean demandIsLinkedToReceivingOffice(ReceivingOffice office) {
  for(int i = 0; i < demand.size(); i++) {
    DemandItem d = demand.get(i);
    ReceivingOffice currentOffice = d.getRelatedOffice();
    if(currentOffice.equals(office)) {
      return true;
    }
  }
  return false;
}

public void addDemand(DemandItem d) {demand.add(d);}
public void deleteDemand(DemandItem d) {demand.remove(d);}
public void updateArticle(DemandItem d, Article article, ReceivingOffice relatedOffice, Integer amount) {
  d.setRelatedArticle(article);
  d.setRelatedOffice(relatedOffice);
  d.setAmount(amount);
}

public void reduceDemand(DemandItem d, Integer amount) {
  d.setAmount(d.getAmount()-amount);
}


/*
------------------------Donation data------------------------
 */
public ObservableList<Donation> getDonations() {return donations;}

public void addDonation(Donation d) {
  donations.add(d);
}


  /*
------------------------Article data------------------------
 */
```

```
public ObservableList<Article> getArticles() {
  return articles;
}

public void addArticle(Article a) {
  articles.add(a);
}

public void deleteArticle(Article a) {
  articles.remove(a);
}

public void updateArticle(Article a, String name, String description, Condition condition, Category category) {
  a.setName(name);
  a.setDescription(description);
  a.setCondition(condition);
  a.setCategory(category);
}

  /*
------------------------Offices data------------------------
*/
public ObservableList<ReceivingOffice> getOffices() {return offices;}

public void setActiveOffice(ReceivingOffice o) {
  activeOffice = o;
}

public ReceivingOffice getActiveOffice() {
  return  activeOffice;
}

public void addOffice(ReceivingOffice o) {offices.add(o);}
public void deleteOffice(ReceivingOffice o) {offices.remove(o);}
public void updateOffice(ReceivingOffice o, String name, FederalState federalState, String district,
                         String address, Status status) {
  o.setName(name);
  o.setFederalState(federalState);
  o.setDistrict(district);
  o.setAddress(address);
```

```
    o.setStatus(status);
  }

    /*
  ------------------------User data------------------------
  */
  public static HashMap<String, User> getUsers() {
    return users;
  }

  public void addUser(String key, User user) {
    users.put(key, user);
  }

  public User getUserByEMail(String email) {
    return users.get(email);
  }

  public User getCurrentUser() {
    return currentUser;
  }

  public void setCurrentUser(User user) {
    currentUser = user;
  }

}
```

2.2. Verwaltungsanwendung

```java
package at.fhooe.swe4.administration;
//File Administration.java

import at.fhooe.swe4.administration.views.LoginScene;
import javafx.application.Application;
import javafx.stage.Stage;

public class Administration extends Application {

  private Stage window;

  @Override
  public void start(Stage primaryStage) throws Exception {
    window = primaryStage;

    LoginScene loginScene = new LoginScene(window);

    window.setScene(loginScene.getLoginScene());
    window.setMinWidth(400);
    window.setWidth(1000);
    window.setHeight(800);
    window.setMinHeight(400);

    window.setTitle("Pomagaju Annahmestellen Verwaltungsservice");
    window.show();
  }

  public static void main(String[] args) {
    launch(args);
  }
}
```

```java
package at.fhooe.swe4.administration.views;
//file ArticleScene.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.administration.controller.ArticleSceneController;
import at.fhooe.swe4.model.Article;
import at.fhooe.swe4.model.dbMock;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Pane;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

public class ArticleScene {

  //global scene management
  private  Stage window;
  private final Scene manageArticleScene;
  private  Pane articlesPane;

  //controller for this scene
  ArticleSceneController controller;

  //View nodes
  private Button addArticleBtn;
  private Button deleteArticleBtn;
  private Button editArticleBtn;

  public Button getAddArticleBtn() {return addArticleBtn;}
  public Button getDeleteArticleBtn() {return deleteArticleBtn;}
  private TableView<Article> articleTable;

  //getter
  public Button getEditArticleBtn() {return editArticleBtn;}
  public Stage getWindow() {return window;}
  public TableView<Article> getArticleTable() {return articleTable;}

  public Scene getManageArticleScene() {return this.manageArticleScene;}

  public ArticleScene(Stage window) {
    this.window = window;

    articlesPane = new VBox();
    articlesPane.getChildren().add(Utilities.createMenuBar(window));
    articlesPane.getChildren().add(createMainPane());
    articlesPane.setId("articles-pane");

    manageArticleScene = new Scene(articlesPane, 600,600);
    manageArticleScene.getStylesheets().add(getClass().getResource("/administration.css").toString());

    controller = new ArticleSceneController(this);
  }

  private TableView<Article> createArticleTable() {
    TableView<Article> articleTable = new TableView<>();
    articleTable.setId("article-table");
    articleTable.setColumnResizePolicy(TableView.CONSTRAINED_RESIZE_POLICY);
    articleTable.setItems(dbMock.getInstance().getArticles());

    TableColumn<Article, Integer> idCol = new TableColumn<>("ID");
    TableColumn<Article, String> nameCol = new TableColumn<>("Name");
    TableColumn<Article, String> descCol = new TableColumn<>("Beschreibung");
    TableColumn<Article, String> condCol = new TableColumn<>("Zustand");
    TableColumn<Article, String> catCol = new TableColumn<>("Kategorie");

    idCol.setCellValueFactory(new PropertyValueFactory<>("id"));
    nameCol.setCellValueFactory(new PropertyValueFactory<>("name"));
    descCol.setCellValueFactory(new PropertyValueFactory<>("description"));
    condCol.setCellValueFactory(new PropertyValueFactory<>("condition"));
    catCol.setCellValueFactory(new PropertyValueFactory<>("category"));

    idCol.setMinWidth(40);
    nameCol.setMinWidth(80);
    descCol.setMinWidth(120);
    condCol.setMinWidth(80);
    catCol.setMinWidth(80);
```

```
    articleTable.getColumns().add(idCol);
    articleTable.getColumns().add(nameCol);
    articleTable.getColumns().add(descCol);
    articleTable.getColumns().add(condCol);
    articleTable.getColumns().add(catCol);

    return articleTable;
  }

  protected Pane createMainPane() {
    Label articlesHL = new Label("Übersicht über Hilfsgüter");
    articlesHL.setId("headline");
    articleTable = createArticleTable();

    addArticleBtn = Utilities.createTextButton("add-article", "+");
    addArticleBtn.setId("standard-button");
    deleteArticleBtn = Utilities.createTextButton("delete-article", "löschen");
    deleteArticleBtn.setId("standard-button");
    editArticleBtn = Utilities.createTextButton("edit-article", "ändern");
    editArticleBtn.setId("standard-button");

    HBox articleButtonPane = new HBox(addArticleBtn, deleteArticleBtn, editArticleBtn);
    articleButtonPane.setId("article-buttons-pane");

    VBox articlesPane = new VBox(articlesHL, articleTable, articleButtonPane);
    articlesPane.setId("articles-pane-content");

    return articlesPane;
  }
}
```

```
package at.fhooe.swe4.administration.controller;
//file ArticleSceneController.java

import at.fhooe.swe4.administration.views.ArticleScene;
import at.fhooe.swe4.administration.views.ManageArticlesDialog;
import at.fhooe.swe4.model.Article;
import at.fhooe.swe4.model.dbMock;
```

```java
import javafx.event.ActionEvent;

public class ArticleSceneController {
  private ArticleScene view;

  public ArticleSceneController(ArticleScene view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getAddArticleBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleAddArticleEvent(e));
    view.getDeleteArticleBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleDeleteArticleEvent(e));
    view.getEditArticleBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleEditArticleEvent(e));
  }

  private void handleAddArticleEvent(ActionEvent e) {
    ManageArticlesDialog dialog = new ManageArticlesDialog(view.getWindow(), view.getArticleTable());
    dialog.showAddDialog();
  }

  private void handleEditArticleEvent(ActionEvent e) {
    Article selectedArticle = view.getArticleTable().getSelectionModel().getSelectedItem();

    if (selectedArticle != null) {
      ManageArticlesDialog dialog = new ManageArticlesDialog(view.getWindow(), view.getArticleTable());
      dialog.showEditDialog(selectedArticle);
    }
  }

  private void handleDeleteArticleEvent(ActionEvent e) {
    Article selectedArticle = view.getArticleTable().getSelectionModel().getSelectedItem();
    if (selectedArticle != null) {
      dbMock.getInstance().deleteArticle(selectedArticle);
    }
  }

}
```

```java
package at.fhooe.swe4.administration.views;
//file DemandScene.java

import at.fhooe.swe4.administration.controller.DemandSceneController;
import at.fhooe.swe4.model.*;
import at.fhooe.swe4.Utilities;
import javafx.beans.property.SimpleStringProperty;
import javafx.beans.value.ObservableValue;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Pane;
import javafx.scene.layout.TilePane;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;
import javafx.util.Callback;

import java.time.LocalDate;

public class DemandScene {

  //global scene management
  private Stage window;
  private final Scene mainScene;
  private Pane mainPane;

  //controller for this scene
  DemandSceneController controller;

  //View nodes

  //Tables
  private TableView<DemandItem> demandTable;
  private TableView<Donation> donationsTable;


  //Buttons and input fields
  private Button addDemandBtn;
  private Button deleteDemandBtn;
  private Button editDemandBtn;
  private Button deleteFiltersBtn;
  private Button changeActiveOfficeBtn;
  private TextField textSearchField;
  private DatePicker datePicker;

  public Stage getWindow() {return window;}

  public TableView<DemandItem> getDemandTable() {
    return demandTable;
  }
  public TableView<Donation> getDonationsTable() {
    return donationsTable;
  }

  public Button getAddDemandBtn() {
    return addDemandBtn;
  }
  public Button getDeleteDemandBtn() {
    return deleteDemandBtn;
  }
  public Button getEditDemandBtn() {
    return editDemandBtn;
  }
  public Button getChangeActiveOfficeBtn() {return changeActiveOfficeBtn;}
  public Button getDeleteFiltersBtn() {return deleteFiltersBtn;}
  public TextField getTextSearchField() {return textSearchField;}
  public DatePicker getDatePicker() {return datePicker;}

  public DemandScene(Stage window) {
    this.window = window;

    mainPane = new VBox();
    mainPane.getChildren().add(Utilities.createMenuBar(window));
    mainPane.getChildren().add(createMainPane());
    mainPane.setId("demand-pane");

    mainScene = new Scene(mainPane, 600,600);
    mainScene.getStylesheets().add(getClass().getResource("/administration.css").toString());

    controller = new DemandSceneController(this);
  }
```

```
public Scene getMainScene() {
  return mainScene;
}

private TableView<DemandItem> createDemandTable() {
  TableView<DemandItem> demandTable = new TableView<>();
  demandTable.setId("demand-table");
  demandTable.setColumnResizePolicy(TableView.CONSTRAINED_RESIZE_POLICY);

  TableColumn<DemandItem, Integer> idCol = new TableColumn<>("ID");
  TableColumn<DemandItem, String> articleCol = new TableColumn<>("Hilfsgut");
  TableColumn<DemandItem, Integer> amountCol = new TableColumn<>("Menge");

  idCol.setCellValueFactory(new PropertyValueFactory<>("id"));
  articleCol.setCellValueFactory(new PropertyValueFactory<>("relatedArticle"));
  amountCol.setCellValueFactory(new PropertyValueFactory<>("amount"));

  idCol.setMinWidth(40);
  amountCol.setMinWidth(20);
  articleCol.setMinWidth(400);

  demandTable.getColumns().add(idCol);
  demandTable.getColumns().add(amountCol);
  demandTable.getColumns().add(articleCol);

  return demandTable;
}

private HBox createActiveOfficePane() {
  Label activeOfficeLabel = new Label("aktive Annahmestelle:");
  Label activeOffice = new Label(dbMock.getInstance().getActiveOffice().getId().toString() + ": " +
          dbMock.getInstance().getActiveOffice().getName());

  VBox activeOfficeInfo = new VBox(activeOfficeLabel, activeOffice);
  activeOfficeInfo.setId("active-office-info");

  changeActiveOfficeBtn = new Button("Annahmestelle ändern");
  changeActiveOfficeBtn.setId("standard-button");

  HBox activeOfficeControlBox = new HBox(activeOfficeInfo, changeActiveOfficeBtn);
```

```
    return activeOfficeControlBox;
  }

  private VBox createDemandOverview() {
    Label demandHL = new Label("Bedarfsübersicht");
    demandHL.setId("headline");

    demandTable = createDemandTable();

    addDemandBtn = Utilities.createTextButton("add-demand", "+");
    addDemandBtn.setId("standard-button");
    deleteDemandBtn = Utilities.createTextButton("delete-demand", "löschen");
    deleteDemandBtn.setId("standard-button");
    editDemandBtn = Utilities.createTextButton("edit-demand", "ändern");
    editDemandBtn.setId("standard-button");

    HBox demandButtonsPane = new HBox(addDemandBtn, deleteDemandBtn, editDemandBtn);
    demandButtonsPane.setId("demand-buttons-pane");

    VBox demandPane = new VBox(demandHL, demandTable, demandButtonsPane);
    demandPane.setId("demand-pane-content");
    return demandPane;
  }

  private VBox createDonationsOverview(){
    Label donationsHL = new Label("Spendenankündigungen");
    donationsHL.setId("headline");

    VBox filters = createDonationsFilterPane();
    donationsTable = createDonationsTable();
    VBox donationsBox = new VBox(donationsHL, filters, donationsTable);
    donationsBox.setId("donations-pane-content");

    return donationsBox;
  }

  private TableView<Donation> createDonationsTable() {
    TableView<Donation> donationTable = new TableView<>();
    donationTable.setId("donations-table");
    donationTable.setColumnResizePolicy(TableView.CONSTRAINED_RESIZE_POLICY);
    TableColumn<Donation, Article> articleCol = new TableColumn("Hilfsgut");
    TableColumn<Donation, Integer> amountCol = new TableColumn("Menge");
    TableColumn<Donation, LocalDate> dateCol = new TableColumn("Abgabedatum");
    TableColumn<Donation, User> userCol = new TableColumn<>("E-Mail SpenderIn");
    TableColumn<Donation, ReceivingOffice> officeCol = new TableColumn<>("Annahemstelle");

    articleCol.setCellValueFactory(new Callback<TableColumn.CellDataFeatures<Donation, Article>, ObservableValue<Ar-
ticle>>() {
      @Override
      public ObservableValue call(TableColumn.CellDataFeatures<Donation, Article> param) {
        return new SimpleStringProperty(param.getValue().getRelatedDemand().getRelatedArticle().getName());
      }
    });

    amountCol.setCellValueFactory(new PropertyValueFactory<>("amount"));
    dateCol.setCellValueFactory(new PropertyValueFactory<>("date"));

    userCol.setCellValueFactory(new Callback<TableColumn.CellDataFeatures<Donation, User>, ObservableValue<User>>() {
      @Override
      public ObservableValue call(TableColumn.CellDataFeatures<Donation, User> param) {
        return new SimpleStringProperty(param.getValue().getUser().getEmail());
      }
    });

    officeCol.setCellValueFactory(new Callback<TableColumn.CellDataFeatures<Donation, ReceivingOffice>, ObservableValue<Re-
ceivingOffice>>() {
      @Override
      public ObservableValue call(TableColumn.CellDataFeatures<Donation, ReceivingOffice> param) {
        return new SimpleStringProperty(param.getValue().getRelatedDemand().getRelatedOffice().getName());
      }
    });

    articleCol.setMinWidth(100);
    amountCol.setMinWidth(40);
    dateCol.setMinWidth(60);
    userCol.setMinWidth(100);
    officeCol.setMinWidth(100);

    donationTable.getColumns().addAll(articleCol, amountCol, dateCol, userCol, officeCol);

    return donationTable;
```

```
  }

  private VBox createDonationsFilterPane() {
    textSearchField = new TextField();
    textSearchField.setId("text-search-field");
    VBox textSearch = new VBox(new Label("Textsuche:"), textSearchField);

    datePicker = new DatePicker();
    VBox datePickerFilter = new VBox(new Label("Abgabedatum:"), datePicker);
    datePickerFilter.setId("date-search-field");

    TilePane filterSelection = new TilePane(textSearch, datePickerFilter);
    filterSelection.setId("filter-tile-pane");

    deleteFiltersBtn = new Button("Filter löschen");
    deleteFiltersBtn.setId("standard-button");

    VBox filters = new VBox(filterSelection, deleteFiltersBtn);
    filters.setId("filters-pane");

    return filters;
  }

  private Pane createMainPane() {
    HBox activeOfficeControlBox = createActiveOfficePane();
    activeOfficeControlBox.setId("activeOfficeControl-hbox");

    VBox demandPane = createDemandOverview();
    VBox donationsPane = createDonationsOverview();

    VBox mainPane = new VBox(activeOfficeControlBox, new Separator(), demandPane, new Separator(), donationsPane);
    mainPane.setId("demand-pane-content");

    return mainPane;
  }
}


package at.fhooe.swe4.administration.controller;
//file DemandSceneController.java
```

```java
import at.fhooe.swe4.administration.views.DemandScene;
import at.fhooe.swe4.administration.views.LoginScene;
import at.fhooe.swe4.administration.views.ManageDemandDialog;
import at.fhooe.swe4.model.DemandItem;
import at.fhooe.swe4.model.Donation;
import at.fhooe.swe4.model.dbMock;
import javafx.collections.transformation.FilteredList;
import javafx.event.ActionEvent;

import java.time.LocalDate;
import java.util.function.Predicate;

public class DemandSceneController {
  private DemandScene view;

  private FilteredList<DemandItem> filteredDemand = new FilteredList<>(dbMock.getInstance().getDemandItems());
  private FilteredList<Donation> filteredDonations = new FilteredList<>(dbMock.getInstance().getDonations());

  private LocalDate dateFilter;
  private String textFilter;

  public DemandSceneController(DemandScene view) {
    this.view = view;
    setView();
  }

  public void setView() {
    view.getDemandTable().setItems(filteredDemand);
    view.getDonationsTable().setItems(filteredDonations);

    filteredDemand.setPredicate(
            demandItem -> demandItem.getRelatedOffice().equals(dbMock.getInstance().getActiveOffice())
    );

    filteredDonations.setPredicate(
            donationItem -> donationItem.getRelatedDemand().getRelatedOffice().equals(dbMock.getInstance().getActive-
Office())
    );

    view.getAddDemandBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleAddDemandEvent(e));
    view.getDeleteDemandBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleDeleteDemandEvent(e));
    view.getEditDemandBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleEditDemandEvent(e));
    view.getChangeActiveOfficeBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleChangeActiveOfficeEvent(e));
    view.getDeleteFiltersBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleDeleteFiltersEvent(e));

    view.getTextSearchField().textProperty().addListener ((observable, oldValue, newValue)-> {
      textFilter = newValue.toLowerCase();
      filterDonations();
    });

    view.getDatePicker().addEventHandler(ActionEvent.ACTION, (e) -> handlePickDateEvent(e));
  }

  private void filterDonations() {
    Predicate<Donation> stringPredicate = donationItem -> textFilter.equals("") || donationItem.toString().toLower-
Case().contains(textFilter);
    Predicate<Donation> datePredicate = donationItem -> dateFilter == null || donationItem.getDate().equals(dateFilter);
    Predicate<Donation> officePredicate = donationItem -> donationItem.getRelatedDemand().getRelate-
dOffice().equals(dbMock.getInstance().getActiveOffice());

    filteredDonations.setPredicate(datePredicate.and(stringPredicate).and(officePredicate));
  }

  private void handlePickDateEvent(ActionEvent e) {
      dateFilter = view.getDatePicker().getValue();
      filterDonations();
  }

  private void handleDeleteFiltersEvent(ActionEvent e) {
      view.getTextSearchField().clear();
      view.getDatePicker().setValue(null);
      filteredDonations.setPredicate(null);
      filteredDonations.setPredicate(
              donationItem -> donationItem.getRelatedDemand().getRelatedOffice().equals(dbMock.getInstance().getActive-
Office())
      );
  }

  private void handleChangeActiveOfficeEvent(ActionEvent e) {
      double x = view.getWindow().getWidth();
      double y = view.getWindow().getHeight();
      view.getWindow().setWidth(x);
      view.getWindow().setHeight(y);

      LoginScene loginScene = new LoginScene(view.getWindow());
      view.getWindow().setScene(loginScene.getLoginScene());
  }

  private void handleDeleteDemandEvent(ActionEvent e) {
    DemandItem demandI = view.getDemandTable().getSelectionModel().getSelectedItem();
    if (demandI != null) {
      dbMock.getInstance().deleteDemand(demandI);
    }
  }

  private void handleAddDemandEvent(ActionEvent e) {
    ManageDemandDialog dialog = new ManageDemandDialog(view.getWindow(), view.getDemandTable());
    dialog.showAddDialog();
  }

  private void handleEditDemandEvent(ActionEvent e) {
    DemandItem demandI = view.getDemandTable().getSelectionModel().getSelectedItem();
    if (demandI != null) {
      ManageDemandDialog dialog = new ManageDemandDialog(view.getWindow(), view.getDemandTable());
      dialog.showEditDialog(demandI);
    }
  }
}




package at.fhooe.swe4.administration.views;
//file LoginScene.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.administration.controller.LoginSceneController;
import at.fhooe.swe4.model.ReceivingOffice;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.Pane;
import javafx.scene.layout.VBox;
```

```java
import javafx.scene.text.Text;
import javafx.stage.Stage;

public class LoginScene {

  private Stage window;
  private Scene loginScene;

  private LoginSceneController controller;

  private TableView<ReceivingOffice> officeTable;
  private Button startButton;

  public Stage getWindow() {
    return window;
  }

  public Button getStartButton() {
    return startButton;
  }

  public TableView<ReceivingOffice> getOfficeTable() {
    return officeTable;
  }

  public Scene getLoginScene() {
    return loginScene;
  }

  public LoginScene(Stage window) {
    this.window = window;

    Pane loginPane = new VBox(createLoginPane());
    loginPane.setId("login-pane");

    loginScene = new Scene(loginPane, 600,600);
    loginScene.getStylesheets().add(getClass().getResource("/administration.css").toString());
    controller = new LoginSceneController(this);
  }
  private Pane createLoginPane(){
    Text loginInfo = new Text("Bitte wählen Sie Ihre Annahmestelle aus.");
    loginInfo.setId("login-pane-info-text");
    loginInfo.setWrappingWidth(300);

    startButton = Utilities.createTextButton("button-login", "Login");
    officeTable = Utilities.createOfficesTable();

    VBox startPane = new VBox(loginInfo, officeTable, startButton);
    startPane.setId("login-screen-start-pane");

    return startPane;
  }

}


package at.fhooe.swe4.administration.controller;
//file LoginSceneController.java

import at.fhooe.swe4.administration.views.DemandScene;
import at.fhooe.swe4.administration.views.LoginScene;
import at.fhooe.swe4.model.ReceivingOffice;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;

public class LoginSceneController {
  private LoginScene view;

  public LoginSceneController(LoginScene view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getStartButton().addEventHandler(ActionEvent.ACTION, (e) -> handleLoginButtonEvent(e));
  }

  private void handleLoginButtonEvent(ActionEvent e) {

    ReceivingOffice selectedOffice = view.getOfficeTable().getSelectionModel().getSelectedItem();
```

```
      if(selectedOffice != null) {
        dbMock.getInstance().setActiveOffice(selectedOffice);

        double x = view.getWindow().getWidth();
        double y = view.getWindow().getHeight();
        view.getWindow().setWidth(x);
        view.getWindow().setHeight(y);

        DemandScene demandScene = new DemandScene(view.getWindow());
        view.getWindow().setScene(demandScene.getMainScene());
      }
    }
  }
```

```
package at.fhooe.swe4.administration.views;
//file ManageArticlesDialog.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.administration.controller.ManageArticlesDialogController;
import at.fhooe.swe4.model.Article;
import at.fhooe.swe4.model.enums.Category;
import at.fhooe.swe4.model.enums.Condition;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
import javafx.stage.Window;

public class ManageArticlesDialog {
  private Window owner;
  private Stage dialogStage;

  private ManageArticlesDialogController controller;

  private TableView<Article> articlesTable;
  private TextField nameInput;
  private TextArea descriptionInput;
  private ChoiceBox conditionInput;
  private ChoiceBox categoryInput;

  Button addArticleBtn = new Button("Hilfsgut hinzufügen");
  Button editArticlesDialog = new Button("Speichern");

  public TextField getNameInput() {return nameInput;}
  public TextArea getDescriptionInput() {return descriptionInput;}
  public ChoiceBox getConditionInput() {return conditionInput;}
  public ChoiceBox getCategoryInput() {return categoryInput;}

  public Button getAddArticleBtn() {return addArticleBtn;}
  public Button getEditArticlesDialog() {return editArticlesDialog;}

  public TableView<Article> getArticlesTable() {return articlesTable;}

  public ManageArticlesDialog(Window owner, TableView<Article> articlesTable) {
    dialogStage = new Stage();
    this.owner = owner;
    this.articlesTable = articlesTable;
  }

  private GridPane createInputGrid() {
    GridPane inputGrid = new GridPane();
    inputGrid.setId("article-input-grid");

    inputGrid.add(new Label("Name: "),0,0);
    nameInput = new TextField();
    inputGrid.add(nameInput,1,0);

    inputGrid.add(new Label("Beschreibung: "),0,1);
    descriptionInput = new TextArea();
    inputGrid.add(descriptionInput,1,1);

    inputGrid.add(new Label("Zustand: "),0,2);
    conditionInput = new ChoiceBox();
    conditionInput.getItems().addAll(Condition.values());
    inputGrid.add(conditionInput,1,2);

    inputGrid.add(new Label("Kategorie: "),0,3);
    categoryInput = new ChoiceBox();
```

```
    categoryInput.getItems().addAll(Category.values());
    inputGrid.add(categoryInput,1,3);

    return inputGrid;
  }

  private void addArticleDialog() {
    addArticleBtn.setId("button-add-article");

    HBox buttonBar = new HBox(20);
    buttonBar.setId("button-bar");
    buttonBar.getChildren().add(addArticleBtn);

    GridPane inputGrid = createInputGrid();
    inputGrid.add(buttonBar,0,4,3,1);

    Scene dialogScene = new Scene(inputGrid);
    dialogScene.getStylesheets().add(getClass().getResource("/administration.css").toExternalForm());
    Utilities.sceneSetup(owner, dialogStage, dialogScene, "Hilfsgut hinzufügen");
    controller = new ManageArticlesDialogController(this);
  }

  private void editArticlesDialog(Article selectedItem){
    editArticlesDialog.setId(("button-save"));

    HBox buttonBar = new HBox(20);
    buttonBar.setId("button-bar");
    buttonBar.getChildren().add(editArticlesDialog);

    GridPane inputGrid = createInputGrid();
    inputGrid.add(buttonBar,0,4,3,1);

    nameInput.setText(selectedItem.getName());
    descriptionInput.setText(selectedItem.getDescription());
    conditionInput.setValue(selectedItem.getCondition());
    categoryInput.setValue(selectedItem.getCategory());

    Scene dialogScene = new Scene(inputGrid);
    dialogScene.getStylesheets().add(getClass().getResource("/administration.css").toExternalForm());
    Utilities.sceneSetup(owner, dialogStage, dialogScene, "Hilfsgut verwalten");
    controller = new ManageArticlesDialogController(this);
```

```java
  }

  public void showAddDialog() {
    this.addArticleDialog();
    dialogStage.show();
  }

  public void showEditDialog(Article selectedItem) {
    this.editArticlesDialog(selectedItem);
    dialogStage.show();
  }

}


package at.fhooe.swe4.administration.controller;
//file ManageArticlesDialogController.java

import at.fhooe.swe4.administration.views.ManageArticlesDialog;
import at.fhooe.swe4.model.Article;
import at.fhooe.swe4.model.dbMock;
import at.fhooe.swe4.model.enums.Category;
import at.fhooe.swe4.model.enums.Condition;
import javafx.event.ActionEvent;

import java.util.Random;

public class ManageArticlesDialogController {
  private ManageArticlesDialog view;

  public ManageArticlesDialogController(ManageArticlesDialog view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getAddArticleBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleAddArticle(e));
    view.getEditArticlesDialog().addEventHandler(ActionEvent.ACTION, (e) -> handleEditArticle(e));
  }

  private void handleEditArticle(ActionEvent e) {
    Article selectedItem = view.getArticlesTable().getSelectionModel().getSelectedItem();

    dbMock.getInstance().updateArticle(
            selectedItem, view.getNameInput().getText(),
            view.getDescriptionInput().getText(),(Condition) view.getConditionInput().getValue(),
            (Category) view.getCategoryInput().getValue());

    view.getArticlesTable().refresh();
  }

  private void handleAddArticle(ActionEvent e) {
    String name = view.getNameInput().getText();
    String desc = view.getDescriptionInput().getText();
    Condition cond = (Condition) view.getConditionInput().getValue();
    Category categ = (Category) view.getCategoryInput().getValue();

    if(name.length() > 0 && desc.length() > 0 && cond != null && categ != null) {
      Random rand = new Random();
      Integer id = rand.nextInt(10000);
      dbMock.getInstance().addArticle(new Article(id, name, desc, cond, categ));

      view.getNameInput().clear();
      view.getDescriptionInput().clear();
      view.getConditionInput().setValue(null);
      view.getCategoryInput().setValue(null);
    }
  }
}
```

```java
package at.fhooe.swe4.administration.views;
//file ManageDemandDialog.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.administration.controller.ManageDemandDialogController;
import at.fhooe.swe4.model.DemandItem;
import at.fhooe.swe4.model.dbMock;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
import javafx.stage.Window;

public class ManageDemandDialog {
  private Window owner;
  private Stage dialogStage;

  private ManageDemandDialogController dialogController;

  private ChoiceBox articleDropDown;
  private ChoiceBox officeDropDown;
  private TextField inputAmount;
  private TableView<DemandItem> demandTable;

  private Button addDemandBtn = new Button("Bedarf hinzufügen");;
  private Button editDemandBtn = new Button("Speichern");

  public Button getAddDemandBtn() {return addDemandBtn;}
  public ChoiceBox getArticleDropDown() {return articleDropDown;}
  public TextField getInputAmount() {return inputAmount;}
  public ChoiceBox getOfficeDropDown() {return officeDropDown;}
  public Button getEditDemandBtn() {return editDemandBtn;}

  public TableView<DemandItem> getDemandTable() {return demandTable;}

  public ManageDemandDialog(Window owner, TableView<DemandItem> demandTable) {
    dialogStage = new Stage();
    this.owner = owner;
    this.demandTable = demandTable;
  }

    private GridPane createInputGrid() {
    GridPane inputGrid = new GridPane();
    inputGrid.setId("add-demand-input-grid");

    inputGrid.add(new Label("Benötigtes Spendengut: "),0,0);
    articleDropDown = new ChoiceBox();
    articleDropDown.getItems().addAll(dbMock.getInstance().getArticles());
    inputGrid.add(articleDropDown,1,0);
```

```
        inputGrid.add(new Label("Menge: "),0,1);
        inputAmount = new TextField();
        inputAmount.setId("manage-demand-input-amount");
        inputGrid.add(inputAmount, 1,1);

        inputGrid.add(new Label("zuständige Annahmestelle: "),0,2);
        officeDropDown = new ChoiceBox();
        officeDropDown.getItems().addAll(dbMock.getInstance().getOffices());
        officeDropDown.setValue(dbMock.getInstance().getActiveOffice());
        inputGrid.add(officeDropDown,1,2);

        return inputGrid;
    }

    private void addDemandDialog() {
        addDemandBtn.setId("button-add-demand");

        HBox buttonBar = new HBox(20);
        buttonBar.setId("button-bar");
        buttonBar.getChildren().add(addDemandBtn);

        GridPane inputGrid = createInputGrid();
        inputGrid.add(buttonBar,0,4,3,1);

        Scene dialogScene = new Scene(inputGrid);
        dialogScene.getStylesheets().add(getClass().getResource("/administration.css").toExternalForm());
        Utilities.sceneSetup(owner, dialogStage, dialogScene, "Bedarfsmeldung hinzufügen");
        dialogController = new ManageDemandDialogController(this);
    }

    private void editDemandDialog(DemandItem selectedItem){
        editDemandBtn.setId(("button-save"));

        HBox buttonBar = new HBox(20);
        buttonBar.setId("button-bar");
        buttonBar.getChildren().add(editDemandBtn);

        GridPane inputGrid = createInputGrid();
        inputGrid.add(buttonBar,0,4,3,1);
        articleDropDown.setValue(selectedItem.getRelatedArticle());
        inputAmount.setText(selectedItem.getAmount().toString());
        officeDropDown.setValue(selectedItem.getRelatedOffice());

        Scene dialogScene = new Scene(inputGrid);
        dialogScene.getStylesheets().add(getClass().getResource("/administration.css").toExternalForm());
        Utilities.sceneSetup(owner, dialogStage, dialogScene, "Bedarfsmeldung ändern");
        dialogController = new ManageDemandDialogController(this);
      }

      public void showAddDialog() {
        this.addDemandDialog();
        dialogStage.show();
      }

      public void showEditDialog(DemandItem selectedItem) {
        this.editDemandDialog(selectedItem);
        dialogStage.show();
      }
    }


    package at.fhooe.swe4.administration.controller;
    //file ManageDemandDialogController.java

    import at.fhooe.swe4.administration.views.ManageDemandDialog;
    import at.fhooe.swe4.model.Article;
    import at.fhooe.swe4.model.DemandItem;
    import at.fhooe.swe4.model.ReceivingOffice;
    import at.fhooe.swe4.model.dbMock;
    import javafx.event.ActionEvent;

    import java.util.Random;

    public class ManageDemandDialogController {
      private ManageDemandDialog view;

      public ManageDemandDialogController(ManageDemandDialog view) {
        this.view = view;
        setView();
      }
```

```java
public void setView() {
  view.getAddDemandBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleAddDemand(e));
  view.getEditDemandBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleEditDemand(e));
}

private void handleEditDemand(ActionEvent e) {
  DemandItem selectedItem = view.getDemandTable().getSelectionModel().getSelectedItem();

  String amount = view.getInputAmount().getText();
  Integer intAmount = Integer.parseInt(amount);

  selectedItem.setRelatedArticle((Article) view.getArticleDropDown().getValue());
  selectedItem.setAmount(intAmount);
  selectedItem.setRelatedOffice((ReceivingOffice)view.getOfficeDropDown().getValue());
  view.getDemandTable().refresh();
}

private void handleAddDemand(ActionEvent e) {
  Article article = (Article)view.getArticleDropDown().getValue();
  ReceivingOffice office = (ReceivingOffice)view.getOfficeDropDown().getValue();

  String amount = view.getInputAmount().getText();

  try {
    Integer intAmount = Integer.parseInt(amount);
    if(article != null && intAmount > 0) {
      Random rand = new Random();
      Integer id = rand.nextInt(10000);
      dbMock.getInstance().addDemand(new DemandItem(id, article, office, intAmount));
      view.getInputAmount().setStyle(null);

      view.getInputAmount().clear();
      view.getArticleDropDown().setValue(null);

    }

  } catch(NumberFormatException error) {
    view.getInputAmount().setStyle("-fx-border-color: red; -fx-border-width: 1;");
  }
}
```

```
}



package at.fhooe.swe4.administration.views;
//file ManageOfficesDialog.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.administration.controller.ManageOfficesDialogController;
import at.fhooe.swe4.model.enums.FederalState;
import at.fhooe.swe4.model.enums.Status;
import at.fhooe.swe4.model.ReceivingOffice;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
import javafx.stage.Window;

public class ManageOfficesDialog {
  private Window owner;
  private Stage dialogStage;

  private ManageOfficesDialogController controller;

  private TextField nameInput;
  private ChoiceBox fedStateInput;
  private TextField distInput;
  private TextField addressInput;
  private ChoiceBox statInput;
  private TableView<ReceivingOffice> officesTable;
  Button addOfficeBtn = new Button("Annahmestelle hinzufügen");
  Button editOfficeBtn = new Button("Speichern");

  public ChoiceBox getFedStateInput() {
    return fedStateInput;
  }

  public TextField getNameInput() {
    return nameInput;
  }

  public TextField getDistInput() {
    return distInput;
  }

  public TextField getAddressInput() {
    return addressInput;
  }

  public ChoiceBox getStatInput() {
    return statInput;
  }

  public TableView<ReceivingOffice> getOfficesTable() {
    return officesTable;
  }

  public Button getAddOfficeBtn() {
    return addOfficeBtn;
  }

  public Button getEditOfficeBtn() {
    return editOfficeBtn;
  }

  public ManageOfficesDialog(Window owner, TableView<ReceivingOffice> officesTable) {
    this.owner = owner;
    this.dialogStage = new Stage();
    this.officesTable = officesTable;
  }

  private GridPane createInputGrid() {
    GridPane inputGrid = new GridPane();
    inputGrid.setId("offices-input-grid");

    inputGrid.add(new Label("Name"), 0, 0);
    nameInput = new TextField();
    inputGrid.add(nameInput,1,0);

    inputGrid.add(new Label("Bundesland: "),0,1);
```

```
        fedStateInput = new ChoiceBox();
        fedStateInput.getItems().addAll(FederalState.values());
        inputGrid.add(fedStateInput,1,1);

        inputGrid.add(new Label("Bezirk"), 0, 2);
        distInput = new TextField();
        inputGrid.add(distInput,1,2);

        inputGrid.add(new Label("Adresse"), 0, 3);
        addressInput = new TextField();
        inputGrid.add(addressInput,1,3);

        inputGrid.add(new Label("Status: "),0,4);
        statInput = new ChoiceBox();
        statInput.getItems().addAll(Status.values());
        inputGrid.add(statInput,1,4);

        return inputGrid;
    }

    private void addOfficeDialog() {
        addOfficeBtn.setId("button-add-office");

        HBox buttonBar = new HBox(20);
        buttonBar.setId("button-bar");
        buttonBar.getChildren().add(addOfficeBtn);

        GridPane inputGrid = createInputGrid();
        inputGrid.add(buttonBar,0,5,3,1);

        Scene dialogScene = new Scene(inputGrid);
        dialogScene.getStylesheets().add(getClass().getResource("/administration.css").toExternalForm());
        Utilities.sceneSetup(owner, dialogStage, dialogScene, "Annahmestelle hinzufügen");
        controller = new ManageOfficesDialogController(this);
    }

    private void editDemandDialog(ReceivingOffice selectedItem){
        editOfficeBtn.setId(("button-save"));

        HBox buttonBar = new HBox(20);
        buttonBar.setId("button-bar");
        buttonBar.getChildren().add(editOfficeBtn);

        GridPane inputGrid = createInputGrid();

        inputGrid.add(buttonBar,0,5,3,1);

        nameInput.setText(selectedItem.getName());
        fedStateInput.setValue(selectedItem.getFederalState());
        distInput.setText(selectedItem.getDistrict());
        addressInput.setText(selectedItem.getAddress());
        statInput.setValue(selectedItem.getStatus());

        Scene dialogScene = new Scene(inputGrid);
        dialogScene.getStylesheets().add(getClass().getResource("/administration.css").toExternalForm());
        Utilities.sceneSetup(owner, dialogStage, dialogScene, "Annahmestelle verwalten");
        controller = new ManageOfficesDialogController(this);
      }

      public void showAddDialog() {
        this.addOfficeDialog();
        dialogStage.show();
      }

      public void showEditDialog(ReceivingOffice selectedItem) {
        this.editDemandDialog(selectedItem);
        dialogStage.show();
      }

    }


    package at.fhooe.swe4.administration.controller;
    //file ManageOfficesDialogController.java

    import at.fhooe.swe4.administration.views.ManageOfficesDialog;
    import at.fhooe.swe4.model.ReceivingOffice;
    import at.fhooe.swe4.model.dbMock;
    import at.fhooe.swe4.model.enums.FederalState;
    import at.fhooe.swe4.model.enums.Status;
    import javafx.event.ActionEvent;
```

```java
import java.util.Random;

public class ManageOfficesDialogController {
  private ManageOfficesDialog view;

  public ManageOfficesDialogController(ManageOfficesDialog view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getAddOfficeBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleAddOffice(e));
    view.getEditOfficeBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleEditOffice(e));
  }

  private void handleAddOffice(ActionEvent e) {
    String name = view.getNameInput().getText();
    FederalState fedState = (FederalState)view.getFedStateInput().getValue();
    String dist = view.getDistInput().getText();
    String addr = view.getAddressInput().getText();
    Status stat = (Status)view.getStatInput().getValue();

    if(name.length()>0 && fedState != null && dist.length()>0 && addr.length()>0 && stat != null) {
      Random rand = new Random();
      Integer id = rand.nextInt(10000);
      dbMock.getInstance().addOffice(new ReceivingOffice(id, name, fedState, dist, addr, stat));

      view.getNameInput().clear();
      view.getFedStateInput().setValue(null);
      view.getDistInput().clear();
      view.getAddressInput().clear();
      view.getStatInput().setValue(null);
    }
  }

  private void handleEditOffice(ActionEvent e) {
    ReceivingOffice selectedItem = view.getOfficesTable().getSelectionModel().getSelectedItem();
    dbMock.getInstance().updateOffice(
            selectedItem, view.getNameInput().getText(),
            (FederalState)view.getFedStateInput().getValue(), view.getDistInput().getText(),
            view.getAddressInput().getText(), (Status)view.getStatInput().getValue());
    view.getOfficesTable().refresh();
  }
}
```

```
package at.fhooe.swe4.administration.views;
//file OfficesScene.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.administration.controller.OfficesSceneController;
import at.fhooe.swe4.model.ReceivingOffice;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TableView;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Pane;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

public class OfficesScene {

  private Stage window;
  private final Scene manageOfficesScene;
  private Pane mainPane;

  //controller for scene
  private OfficesSceneController controller;

  private TableView<ReceivingOffice> officesTable;
  private Button addOfficeBtn;
  private Button editOfficeBtn;
  private Button deleteOfficeBtn;

  public Stage getWindow() {return window;}
  public Button getAddOfficeBtn() {return addOfficeBtn;}
  public Button getEditOfficeBtn() {return editOfficeBtn;}
  public Button getDeleteOfficeBtn() {return deleteOfficeBtn;}
  public TableView<ReceivingOffice> getOfficesTable() {return officesTable;}

  public Scene getOfficesScene() {return this.manageOfficesScene;}

  public OfficesScene(Stage window) {
    this.window = window;

    mainPane = new VBox();
    mainPane.getChildren().add(Utilities.createMenuBar(window));
    mainPane.getChildren().add(createMainPane());
    mainPane.setId("offices-pane");

    manageOfficesScene = new Scene(mainPane, 600,600);
    manageOfficesScene.getStylesheets().add(getClass().getResource("/administration.css").toString());
    controller = new OfficesSceneController(this);
  }

  protected Pane createMainPane() {
    Label articlesHL = new Label("Übersicht über Annahmestellen");
    articlesHL.setId("headline");
    officesTable = Utilities.createOfficesTable();

    addOfficeBtn = Utilities.createTextButton("add-office", "+");
    addOfficeBtn.setId("standard-button");
    editOfficeBtn = Utilities.createTextButton("edit-office", "ändern");
    editOfficeBtn.setId("standard-button");
    deleteOfficeBtn = Utilities.createTextButton("delete-office", "löschen");
    deleteOfficeBtn.setId("standard-button");

    HBox officesButtonPane = new HBox(addOfficeBtn, deleteOfficeBtn, editOfficeBtn);
    officesButtonPane.setId("offices-buttons-pane");

    VBox articlesPane = new VBox(articlesHL, officesTable, officesButtonPane);
    articlesPane.setId("offices-pane-content");

    return articlesPane;
  }
}


package at.fhooe.swe4.administration.controller;
```

```java
//file OfficesSceneController.java

import at.fhooe.swe4.administration.views.ManageOfficesDialog;
import at.fhooe.swe4.administration.views.OfficesScene;
import at.fhooe.swe4.model.ReceivingOffice;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;

public class OfficesSceneController {
  private OfficesScene view;

  public OfficesSceneController(OfficesScene view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getAddOfficeBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleAddOfficeEvent(e));
    view.getEditOfficeBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleEditOfficeEvent(e));
    view.getDeleteOfficeBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleDeleteOfficeEvent(e));
  }

  private void handleAddOfficeEvent(ActionEvent e) {
    ManageOfficesDialog dialog = new ManageOfficesDialog(view.getWindow(), view.getOfficesTable());
    dialog.showAddDialog();
  }

  private void handleDeleteOfficeEvent(ActionEvent e) {
    ReceivingOffice office = view.getOfficesTable().getSelectionModel().getSelectedItem();
    if (office != null && !dbMock.getInstance().demandIsLinkedToReceivingOffice(office)) {
      dbMock.getInstance().deleteOffice(office);
    }
  }

  private void handleEditOfficeEvent(ActionEvent e) {
    ReceivingOffice office = view.getOfficesTable().getSelectionModel().getSelectedItem();
    if (office != null) {
      ManageOfficesDialog dialog = new ManageOfficesDialog(view.getWindow(), view.getOfficesTable());
      dialog.showEditDialog(office);
    }
  }
}
```

2.3. Spendenanwendung

```java
package at.fhooe.swe4.donationsApp;
//File DonationsApp.java

import at.fhooe.swe4.donationsApp.views.LoginSceneDonations;
import javafx.application.Application;
import javafx.stage.Stage;

public class DonationsApp extends Application {
  private Stage window;

  @Override
  public void start(Stage primaryStage) throws Exception {
    window = primaryStage;

    LoginSceneDonations LoginSceneDonations = new LoginSceneDonations(window);
    window.setScene(LoginSceneDonations.getLoginScene());

    window.setMinWidth(200);
    window.setMinHeight(450);
    window.setWidth(470);
    window.setHeight(750);
    window.setTitle("Pomagaju - Hilfe für die Ukraine");
    window.show();
  }

  public static void main(String[] args) {
    launch(args);
  }
}


package at.fhooe.swe4.donationsApp.views;
//File: DonationsScene.java

import at.fhooe.swe4.donationsApp.controller.DonationSceneController;
import at.fhooe.swe4.model.enums.Category;
import at.fhooe.swe4.model.enums.FederalState;
import at.fhooe.swe4.model.DemandItem;
import at.fhooe.swe4.model.dbMock;
import javafx.collections.transformation.FilteredList;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.stage.Stage;

import java.util.HashMap;

public class DonationsScene {

  private Stage window;
  private final Scene donationScene;
  private Pane mainPane;

  private DonationSceneController controller;

  private FilteredList<DemandItem> filteredDemand = new FilteredList<>(dbMock.getInstance().getDemandItems());
  private  VBox filteredResults = new VBox();

  private TextField textSearchField;
  private ChoiceBox federalStatesDropDown;
  private TextField addressSearchField;
  private  ChoiceBox categoryDropDown;
  private Button addDonationButton;
  private Button deleteFiltersBtn;


  public Stage getWindow() {
    return window;
  }

  public Button getDeleteFiltersBtn() {
    return deleteFiltersBtn;
  }

  public FilteredList<DemandItem> getFilteredDemand() {
    return filteredDemand;
  }

  public TextField getTextSearchField() {
    return textSearchField;
  }

  public ChoiceBox getFederalStatesDropDown() {
    return federalStatesDropDown;
  }

  public TextField getAddressSearchField() {
    return addressSearchField;
  }

  public ChoiceBox getCategoryDropDown() {
    return categoryDropDown;
  }

  public Button getAddDonationButton() {
    return addDonationButton;
  }

  public DonationsScene(Stage window) {
    this.window = window;
    mainPane = new VBox(createMainPane());
    mainPane.setId("donations-pane");

    filteredResults.setId("filtered-results-vbox");
    updateResults();

    donationScene = new Scene(mainPane, 270,550);
    donationScene.getStylesheets().add(DonationsScene.class.getResource("/donationsApp.css").toString());
    controller = new DonationSceneController(this);
  }

  public Scene getDonationScene() {return donationScene;}

  private Pane createFilterPane() {
    textSearchField = new TextField();
    textSearchField.setId("text-search-field");
    VBox textSearch = new VBox(new Label("Textsuche:"), textSearchField);

    federalStatesDropDown = new ChoiceBox();
    federalStatesDropDown.getItems().add(null);
    federalStatesDropDown.getItems().addAll(FederalState.values());
    VBox federalStateSearch = new VBox(new Label("Bundesland"), federalStatesDropDown);
    federalStateSearch.setId("filter-vBox");

    addressSearchField = new TextField();
    VBox addressSearch = new VBox(new Label("Adresse"), addressSearchField);
    addressSearch.setId("filter-vBox");

    categoryDropDown = new ChoiceBox();
    categoryDropDown.getItems().add(null);
    categoryDropDown.getItems().addAll(Category.values());
    VBox categorySearch = new VBox(new Label("Kategorie"), categoryDropDown);
    categorySearch.setId("filter-vBox");

    TilePane filterSelection = new TilePane(federalStateSearch, addressSearch, categorySearch);
    filterSelection.setId("filter-tile-pane");

    deleteFiltersBtn = new Button("Filter löschen");
    deleteFiltersBtn.setId("standard-button");

    VBox filters = new VBox(textSearch, filterSelection, deleteFiltersBtn);
    filters.setId("filters-pane");
    return filters;
  }

  public HashMap<Button, DemandItem> demandButtonHelperStructure = new HashMap<>();

  public HashMap<Button, DemandItem> getDemandButtonHelperStructure() {
    return demandButtonHelperStructure;
  }

  public VBox createDemandTile(DemandItem d) {
    GridPane resultGrid = new GridPane();
    resultGrid.setId("donations-grid");
    resultGrid.add(new Label("Kategorie:"),0,0);
    String temp = d.getRelatedArticle().getCategory().toString();
    resultGrid.add(new Label(temp),1,0);

    resultGrid.add(new Label("benötigte Menge:"),0,1);

    String amount = d.getAmount().toString();
    Label amountLabel = new Label(amount);
    resultGrid.add(amountLabel,1,1);

    resultGrid.add(new Label("Hilfsgut:"),0,2);
    temp = d.getRelatedArticle().getName();
    resultGrid.add(new Label(temp),1,2);

    resultGrid.add(new Label("Beschreibung:"),0,3);
    temp = d.getRelatedArticle().getDescription();
    resultGrid.add(new Label(temp),1,3);

    resultGrid.add(new Label("Zustand:"),0,4);
    temp = d.getRelatedArticle().getCondition().toString();
    resultGrid.add(new Label(temp),1,4);

    resultGrid.add(new Label("Annahmestelle:"),0,5);
    temp = d.getRelatedOffice().getName();
    resultGrid.add(new Label(temp),1,5);

    resultGrid.add(new Label("Adresse:"),0,6);
    temp = d.getRelatedOffice().getAddress() + " (" + d.getRelatedOffice().getDistrict() + ")";
    resultGrid.add(new Label(temp),1,6);

    addDonationButton = new Button("jetzt spenden");
    addDonationButton.setId("button-donate");
    demandButtonHelperStructure.put(addDonationButton, d);

    VBox demandDetailTile = new VBox(resultGrid, addDonationButton);
    demandDetailTile.setId("demand-tile-vbox");

    return demandDetailTile;
  }

  private Pane createMainPane() {
    Label curUser = new Label("Benutzer: " + dbMock.getInstance().getCurrentUser().getName());

    Button myDonationsButton = new Button();
    myDonationsButton.setId("my-donations-button");
    HBox myDonations = new HBox(curUser, myDonationsButton);
    myDonations.setId("myDonations-pane");

    Pane filters = createFilterPane();
    ScrollPane scrollArea = new ScrollPane();
    scrollArea.setId("donation-scene-scroll-area");
    scrollArea.setContent(filteredResults);

    VBox donationsPane = new VBox(myDonations, new Separator(), filters, new Separator(), scrollArea);
    donationsPane.setId("donations-pane-content");
    return donationsPane;
  }

  public void updateResults() {
    filteredResults.getChildren().clear();
    demandButtonHelperStructure.clear();

    for(int i = 0; i < filteredDemand.size(); i++) {
      VBox demandTile = createDemandTile(filteredDemand.get(i));
      filteredResults.getChildren().add(demandTile);
    }
  }
}


package at.fhooe.swe4.donationsApp.controller;
//File: DonationsSceneController.java

import at.fhooe.swe4.donationsApp.views.DonationsScene;
import at.fhooe.swe4.donationsApp.views.MakeDonationDialog;
import at.fhooe.swe4.model.DemandItem;
import at.fhooe.swe4.model.dbMock;
import at.fhooe.swe4.model.enums.Category;
import at.fhooe.swe4.model.enums.FederalState;
import javafx.collections.ListChangeListener;
import javafx.collections.transformation.FilteredList;
import javafx.event.ActionEvent;
import javafx.scene.control.Button;

import java.util.Set;
import java.util.function.Predicate;

public class DonationSceneController {
  private DonationsScene view;
```

```java
private FilteredList<DemandItem> filteredDemand = new FilteredList<>(dbMock.getInstance().getDemandItems());
private String textFilter;
private String addressFilter;
private Category categoryFilter;
private FederalState federalStateFilter;

public DonationSceneController(DonationsScene view) {
  this.view = view;
  setView();
}

public void addEventListenersToResultList(){
  Set<Button> keySet = view.getDemandButtonHelperStructure().keySet();
  for(Button b : keySet) {
    b.addEventHandler(ActionEvent.ACTION, e -> handleAddDonationEvent(e, view.getDemandButtonHelperStructure().get(b)));
  }
}

private void setView() {
  addEventListenersToResultList();

  //EventListener for completely deleted demandItem
  dbMock.getInstance().getDemandItems().addListener(new ListChangeListener<DemandItem>() {
    @Override
    public void onChanged(Change<? extends DemandItem> c) {
      view.updateResults();
      addEventListenersToResultList();
    }
  });

  //EventListener for changes of amount of deamnd
  for(int i = 0; i < dbMock.getInstance().getDemandItems().size(); i++) {
    dbMock.getInstance().getDemandItems().get(i).getObsAmount().addListener((observable, oldValue, newValue) -> {
      view.updateResults();
      addEventListenersToResultList();
    });
  }

  view.getTextSearchField().textProperty().addListener ((observable, oldValue, newValue)-> {
    textFilter = newValue.toLowerCase();
```

```
        filterDonations();
      });

      view.getAddressSearchField().textProperty().addListener ((observable, oldValue, newValue)-> {
        addressFilter = newValue.toLowerCase();
        filterDonations();
      });

      view.getCategoryDropDown().addEventHandler(ActionEvent.ACTION, (event -> {
        categoryFilter = (Category) view.getCategoryDropDown().getValue();
        filterDonations();
      }));

      view.getFederalStatesDropDown().addEventHandler(ActionEvent.ACTION, e-> {
        federalStateFilter = (FederalState) view.getFederalStatesDropDown().getValue();
        filterDonations();
      });

      view.getDeleteFiltersBtn().addEventHandler(ActionEvent.ACTION, (e) -> handleDeleteFiltersEvent(e));
    }

    private void handleDeleteFiltersEvent(ActionEvent e) {
      view.getTextSearchField().clear();
      view.getAddressSearchField().clear();
      view.getFederalStatesDropDown().setValue(null);
      view.getCategoryDropDown().setValue(null);

      view.getFilteredDemand().setPredicate(null);
      addEventListenersToResultList();
    }

    private void handleAddDonationEvent(ActionEvent e, DemandItem demandItem) {
      MakeDonationDialog dialog = new MakeDonationDialog(view.getWindow(), demandItem, view);
      dialog.showDonationDialog();
    }

    private void filterDonations() {
      Predicate<DemandItem> textPredicate = demandItem -> textFilter == null
                || demandItem.toString().toLowerCase().contains(textFilter)
                || demandItem.getRelatedArticle().getDescription().toLowerCase().contains(textFilter);
      Predicate<DemandItem> addressPredicate = demandItem -> addressFilter == null || addressFilter.equals("")
              || demandItem.getRelatedOffice().getAddress().toLowerCase().contains(addressFilter)
              || demandItem.getRelatedOffice().getDistrict().toLowerCase().contains(addressFilter);

      Predicate<DemandItem> categoryPredicate = demandItem -> categoryFilter == null ||
              demandItem.getRelatedArticle().getCategory().equals(categoryFilter);

      Predicate<DemandItem> federalStatePredicate = demandItem -> federalStateFilter == null ||
              demandItem.getRelatedOffice().getFederalState().equals(federalStateFilter);

      view.getFilteredDemand().setPredicate(textPredicate.and(addressPredicate).
              and(categoryPredicate).and(federalStatePredicate));

      view.updateResults();
      addEventListenersToResultList();
    }

  }
```

```
package at.fhooe.swe4.donationsApp.views;
//File: LoginDialog.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.donationsApp.controller.LoginDialogController;
import at.fhooe.swe4.model.User;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

public class LoginDialog {
  private Stage window;
  private Stage dialogStage;
  private LoginDialogController controller;

  private TextField email;
  private PasswordField pw;
  private Button loginButton;

  public Stage getWindow() {
    return window;
  }

  public TextField getEmail() {
    return email;
  }

  public PasswordField getPw() {
    return pw;
  }

  public Button getLoginButton() {
    return loginButton;
  }

  public LoginDialog(Stage window) {
    dialogStage = new Stage();
    this.window = window;
  }

  private void loginDialog() {
    email = new TextField();
    email.setId("user-input");
    VBox userBox = new VBox(new Label("E-Mail Adresse:"), email);

    pw = new PasswordField();
    pw.setId("password-input");
    VBox pwBox = new VBox(new Label("Passwort:"), pw);

    loginButton = new Button("anmelden");
    loginButton.setId("login-button");

    VBox loginPane = new VBox(userBox, pwBox, loginButton);
```

```
    loginPane.setId("loginBox-dialog");

    Scene dialogScene = new Scene(loginPane);
    dialogScene.getStylesheets().add(getClass().getResource("/donationsApp.css").toExternalForm());
    Utilities.sceneSetup(window, dialogStage, dialogScene, "Anmelden");
    controller = new LoginDialogController(this);
  }

  public void closeDialog() {
    dialogStage.close();
  }

  public void showLoginDialog() {
    this.loginDialog();
    dialogStage.show();
  }
}


package at.fhooe.swe4.donationsApp.controller;
//File: LoginDialogController.java

import at.fhooe.swe4.donationsApp.views.DonationsScene;
import at.fhooe.swe4.donationsApp.views.LoginDialog;
import at.fhooe.swe4.model.User;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;

public class LoginDialogController {
  private LoginDialog view;

  public LoginDialogController(LoginDialog view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getLoginButton().addEventHandler(ActionEvent.ACTION, (e) -> handleLoginEvent(e, view.getEmail(), view.getPw()));
  }
  private void handleLoginEvent(ActionEvent e, TextField email, PasswordField pw) {

    User user = dbMock.getInstance().getUserByEMail(email.getText());

    if(user == null) {
      email.setStyle("-fx-border-color: red; -fx-border-width: 1px");
    } else {
      email.setStyle(null);

      if (user.getPassword().equals(pw.getText())) {
        dbMock.getInstance().setCurrentUser(user);
        view.closeDialog();
        DonationsScene donationsScene = new DonationsScene(view.getWindow());
        view.getWindow().setScene(donationsScene.getDonationScene());
      } else {
        pw.setStyle("-fx-border-color: red; -fx-border-width: 1px");
      }
    }
  }

}
```

```
package at.fhooe.swe4.donationsApp.views;
//File: LoginSceneDonations.java

import at.fhooe.swe4.donationsApp.controller.LoginSceneDonationsController;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.layout.Pane;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

public class LoginSceneDonations {

    private Stage window;
```

```java
    private Scene loginSceneDonations;

    private Button registerBtn;
    private Button loginBtn;

    public Stage getWindow() {
        return window;
    }

    public Button getRegisterBtn() {
        return registerBtn;
    }

    public Button getLoginBtn() {
        return loginBtn;
    }

    private LoginSceneDonationsController controller;

    public LoginSceneDonations(Stage window) {
        this.window = window;

        Pane loginPane = new VBox(createLoginPane());
        loginPane.setId("login-pane");

        loginSceneDonations = new Scene(loginPane,  270,550);
        loginSceneDonations.getStylesheets().add(getClass().getResource("/donationsApp.css").toString());
        controller = new LoginSceneDonationsController(this);
    }

    public Scene getLoginScene() {
        return loginSceneDonations;
    }

    private Pane createLoginPane() {
        Label registerLabel = new Label("Registriere dich jetzt");
        registerLabel.setId("register-label");
        registerBtn = new Button("Registrieren");
        registerBtn.setId("register-button");

        VBox register = new VBox(registerLabel, registerBtn);
        register.setId("register-VBox");

        Label loginLabel = new Label("Du hast schon einen Account?");
        loginLabel.setId("login-label");
        loginBtn = new Button("Anmelden");
        loginBtn.setId("login-button");

        VBox login = new VBox(loginLabel, loginBtn);
        login.setId("login-VBox");

        VBox startScreen = new VBox(register, login);
        startScreen.setId("start-screen-VBox");

        return startScreen;
    }
}


package at.fhooe.swe4.donationsApp.controller;
//File: LoginSceneDonationsController.java

import at.fhooe.swe4.donationsApp.views.LoginDialog;
import at.fhooe.swe4.donationsApp.views.LoginSceneDonations;
import at.fhooe.swe4.donationsApp.views.RegisterDialog;
import javafx.event.ActionEvent;

public class LoginSceneDonationsController {
  private LoginSceneDonations view;

  public LoginSceneDonationsController(LoginSceneDonations view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getRegisterBtn().addEventHandler(ActionEvent.ACTION, (actionEvent -> {
      RegisterDialog diag = new RegisterDialog(view.getWindow());
      diag.showRegisterDialog();
    }));

    view.getLoginBtn().addEventHandler(ActionEvent.ACTION, (actionEvent -> {
      LoginDialog diag = new LoginDialog(view.getWindow());
      diag.showLoginDialog();
    }));
  }
}
```

```
package at.fhooe.swe4.donationsApp.views;
//File: MakeDonationDialog.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.donationsApp.controller.MakeDonationDialogController;
import at.fhooe.swe4.model.DemandItem;
import javafx.event.ActionEvent;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;
import javafx.stage.Window;

import java.time.LocalDate;

public class MakeDonationDialog {
  private Window owner;
  private Stage dialogStage;
  private DemandItem selectedDemandItem;
  private DonationsScene ownerScene;
  private Scene dialogScene;
  private VBox makeDonationVBox;
  private MakeDonationDialogController controller;

  private Button donateButton;
  private DatePicker datePicker;
  private TextField amount;

  public DonationsScene getOwnerScene() {
    return ownerScene;
  }
```

```
public DemandItem getSelectedDemandItem() {
  return selectedDemandItem;
}

public DatePicker getDatePicker() {
  return datePicker;
}

public TextField getAmount() {
  return amount;
}

public Button getDonateButton() {
  return donateButton;
}

public MakeDonationDialog(Window owner, DemandItem demandItem, DonationsScene ownerScene) {
  dialogStage = new Stage();
  this.owner = owner;
  this.selectedDemandItem = demandItem;
  this.ownerScene = ownerScene;
}

private void makeDonationDialog() {

  datePicker = new DatePicker();
  datePicker.setId("make-donation-DatePicker");

  //allow only future dates
  datePicker.setDayCellFactory(param -> new DateCell() {
    @Override
    public void updateItem(LocalDate date, boolean empty) {
      super.updateItem(date,empty);
      setDisable(empty || date.compareTo(LocalDate.now()) < 0);
    }
  });

  VBox dateBox = new VBox(new Label("Abgabedatum:"), datePicker);

  amount = new TextField();
```

```
    amount.setId("make-donation-amount-input");

    VBox amountBox = new VBox(new Label("Menge" + " (" + selectedDemandItem.getAmount().toString() + " = max. An-
zahl)"),amount);
    VBox makeDonationInputVBox = new VBox(dateBox, amountBox);
    makeDonationInputVBox.setId("make-donation-input-VBox");

    donateButton = new Button("Spende anmelden");
    donateButton.setId("make-donation-button");

    makeDonationVBox = new VBox(makeDonationInputVBox, donateButton);
    makeDonationVBox.setId("make-donation-dialog");

    dialogScene = new Scene(makeDonationVBox);
    dialogScene.getStylesheets().add(getClass().getResource("/donationsApp.css").toExternalForm());
    Utilities.sceneSetup(owner, dialogStage, dialogScene, "Spende anmelden");
    controller = new MakeDonationDialogController(this);
  }

  public void showToken() {
    Label intro = new Label ("Bitte zeigen Sie diesen Code bei der Abgabe der Spenden vor");
    intro.setId("token-intro-label");
    intro.setPrefWidth(300);
    intro.setWrapText(true);

    Label token = new Label ("#r3kc7");
    token.setId("token-label");

    Button closeBtn = new Button("ist notiert! Fenster schließen");
    closeBtn.setId("tokenClose-btn");

    closeBtn.addEventHandler(ActionEvent.ACTION, e -> {
      dialogStage.close();
    });

    VBox tokenBox = new VBox(intro, token, closeBtn);
    tokenBox.setId("token-VBox");

    makeDonationVBox.getChildren().clear();
    makeDonationVBox.getChildren().add(tokenBox);
  }
  public void showDonationDialog() {
    this.makeDonationDialog();
    dialogStage.show();
  }
}


package at.fhooe.swe4.donationsApp.controller;
//File: MakeDonationDialogController.java

import at.fhooe.swe4.donationsApp.views.MakeDonationDialog;
import at.fhooe.swe4.model.Donation;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;

public class MakeDonationDialogController {
  private MakeDonationDialog view;

  public MakeDonationDialogController(MakeDonationDialog view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getDonateButton().addEventHandler(ActionEvent.ACTION, e -> handleDonateButton(e) );
  }

  private void handleDonateButton(ActionEvent e) {
    if(view.getDatePicker().getValue() == null) {
      view.getDatePicker().setStyle("-fx-border-color: red; -fx-border-width: 1");
    } else {
      view.getDatePicker().setStyle(null);
    }

    if(view.getAmount().getText().equals("")) {
      view.getAmount().setStyle("-fx-border-color: red; -fx-border-width: 1");
    } else {
      view.getDatePicker().setStyle(null);
    }
    if((view.getDatePicker().getValue() != null) && (!view.getAmount().getText().equals(""))) {

      Integer intAmount = Integer.parseInt(view.getAmount().getText());
      if (intAmount > 0 && intAmount <= view.getSelectedDemandItem().getAmount()) {

        //if full demand is covered remove demand item from demand list
        if(view.getSelectedDemandItem().getAmount() - intAmount == 0) {
          dbMock.getInstance().deleteDemand(view.getSelectedDemandItem());
        } else { //if not full demand is covered just reduce needed amount
          dbMock.getInstance().reduceDemand(view.getSelectedDemandItem(), intAmount);
        }

        dbMock.getInstance().addDonation(new Donation(view.getSelectedDemandItem(), view.getDatePicker().getValue(),
                dbMock.getInstance().getCurrentUser(), intAmount));

        //update displayed list of owner window
        view.showToken();

      } else {
        view.getAmount().setStyle("-fx-border-color: red; -fx-border-width: 1");
      }
    }
  }
}



package at.fhooe.swe4.donationsApp.views;
//File: RegisterDialog.java

import at.fhooe.swe4.Utilities;
import at.fhooe.swe4.donationsApp.controller.RegisterDialogController;
import at.fhooe.swe4.model.User;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
```

```java
import javafx.scene.control.TextField;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

import java.util.regex.Pattern;

public class RegisterDialog {
    private Stage window;
    private Stage dialogStage;
    private RegisterDialogController controller;

    private TextField name;
    private TextField email;
    private PasswordField pw;
    private PasswordField pwConfirm;
    private Button registerButton;

    public Stage getWindow() {
        return window;
    }

    public TextField getName() {
        return name;
    }

    public TextField getEmail() {
        return email;
    }

    public PasswordField getPw() {
        return pw;
    }

    public PasswordField getPwConfirm() {
        return pwConfirm;
    }

    public Button getRegisterButton() {
        return registerButton;
    }
    public RegisterDialog(Stage window) {
        dialogStage = new Stage();
        this.window = window;
    }

    private void registerDialog() {
        name = new TextField();
        name.setId("name-input");
        VBox nameBox = new VBox(new Label("Name:"), name);

        email = new TextField();
        email.setId("user-input");
        VBox userBox = new VBox(new Label("E-Mail Adresse:"), email);

        pw = new PasswordField();
        pw.setId("password-input");
        VBox pwBox = new VBox(new Label("Passwort:"), pw);

        pwConfirm = new PasswordField();
        pwConfirm.setId("password-input");
        VBox pwConfirmBox = new VBox(new Label("Passwort bestätigen:"), pwConfirm);

        registerButton = new Button("Jetzt registieren");
        registerButton.setId("register-button");

        VBox registerPane = new VBox(nameBox, userBox, pwBox, pwConfirmBox, registerButton);
        registerPane.setId("registerBox-dialog");

        Scene dialogScene = new Scene(registerPane);
        dialogScene.getStylesheets().add(getClass().getResource("/donationsApp.css").toExternalForm());
        Utilities.sceneSetup(window, dialogStage, dialogScene, "Registrieren");
        controller = new RegisterDialogController(this);
    }

    public void closeDialog(){
        dialogStage.close();
    }

    public void showRegisterDialog() {
        this.registerDialog();
        dialogStage.show();
    }
}


package at.fhooe.swe4.donationsApp.controller;
//File: RegisterDialogController.java

import at.fhooe.swe4.donationsApp.views.DonationsScene;
import at.fhooe.swe4.donationsApp.views.RegisterDialog;
import at.fhooe.swe4.model.User;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;

import java.util.regex.Pattern;

public class RegisterDialogController {

  private RegisterDialog view;

  public RegisterDialogController(RegisterDialog view) {
    this.view = view;
    setView();
  }

  private void setView() {
    view.getRegisterButton().addEventHandler(ActionEvent.ACTION, (e) -> handleRegisterEvent(e, view.getName(), view.ge-
tEmail(),
            view.getPw(), view.getPwConfirm()));
  }

  private void handleRegisterEvent(ActionEvent e, TextField name, TextField email, PasswordField pw, PasswordField pwCon-
firm) {
    boolean checkName = checkName();
    boolean checkEmail = checkEMail();
    boolean checkPW = checkPW();

    if(!checkName) {
      view.getName().setStyle("-fx-border-color: red; -fx-border-width: 1px");
    } else {
      view.getName().setStyle(null);
    }

    if(!checkEmail) {
      view.getEmail().setStyle("-fx-border-color: red; -fx-border-width: 1px");
    } else {
      view.getEmail().setStyle(null);
    }

    if(!checkPW) {
      view.getPw().setStyle("-fx-border-color: red; -fx-border-width: 1px");
      view.getPwConfirm().setStyle("-fx-border-color: red; -fx-border-width: 1px");
    } else {
      view.getPw().setStyle(null);
      view.getPwConfirm().setStyle(null);
    }

    if(checkName && checkEmail && checkPW) {
      email.setStyle(null);
      pw.setStyle(null);
      pwConfirm.setStyle(null);
      User newUser = new User(name.getText(), pw.getText(), email.getText());
      dbMock.getInstance().addUser(email.getText(), newUser);
      dbMock.getInstance().setCurrentUser(newUser);

      view.closeDialog();
      DonationsScene donationsScene = new DonationsScene(view.getWindow());
      view.getWindow().setScene(donationsScene.getDonationScene());
    }
  }

  private boolean checkEMail() {
    String regexPattern = "^(?=.{1,64}@)[A-Za-z0-9_-]+(\\.[A-Za-z0-9_-]+)*@"
            + "[^-][A-Za-z0-9-]+(\\.[A-Za-z0-9-]+)*(\\.[A-Za-z]{2,})$";

    return Pattern.compile(regexPattern).matcher(view.getEmail().getText()).matches();
  }

  private boolean checkName() {
    TextField name = view.getName();
    return  name != null && !name.getText().equals("");
  }

  private boolean checkPW() {
    return view.getPw().getText().length()>4 &&
            view.getPw().getText().equals(view.getPw().getText());
  }
}
```

2.4. Utilities

Code der in mehreren Views benötigt wurde.

```
package at.fhooe.swe4;
//File: Utilities.java

import at.fhooe.swe4.model.ReceivingOffice;
import at.fhooe.swe4.administration.views.ArticleScene;
import at.fhooe.swe4.administration.views.DemandScene;
import at.fhooe.swe4.administration.views.OfficesScene;
import at.fhooe.swe4.model.dbMock;
import javafx.event.ActionEvent;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;
import javafx.stage.Window;

public class Utilities {

  public static Button createTextButton(String id, String caption) {
    Button button = new Button(caption);
    button.setId(id);
    button.setPadding(new Insets(5));
    button.setPrefSize(50,40);
    button.setMinSize(50,40);
    return button;
```

```java
}

public static MenuBar createMenuBar(Stage window) {
  MenuItem manageDemand = new MenuItem("Bedarf verwalten");
  MenuItem manageReceivingOffices = new MenuItem("Annahmestellen verwalten");
  MenuItem manageArticles = new MenuItem("Hilfsgüter verwalten");
  Menu adminMenu = new Menu("Verwaltung");
  adminMenu.getItems().addAll(manageDemand, manageReceivingOffices, manageArticles);

  manageArticles.addEventHandler(ActionEvent.ACTION, (e) -> handleManageArticlesEvent(e, window));
  manageDemand.addEventHandler(ActionEvent.ACTION, (e) -> handleManageDemandEvent(e, window));
  manageReceivingOffices.addEventHandler(ActionEvent.ACTION, (e) -> handleManageOfficesEvent(e, window));

  MenuBar menuBar = new MenuBar();
  menuBar.getMenus().add(adminMenu);
  return menuBar;
}

private static void handleManageOfficesEvent(ActionEvent e, Stage window) {
  double x = window.getWidth();
  double y = window.getHeight();
  window.setWidth(x);
  window.setHeight(y);
  OfficesScene officesScene = new OfficesScene(window);
  window.setScene(officesScene.getOfficesScene());
}

private static void handleManageDemandEvent(ActionEvent e, Stage window) {
  double x = window.getWidth();
  double y = window.getHeight();
  window.setWidth(x);
  window.setHeight(y);
  DemandScene demandScene = new DemandScene(window);
  window.setScene(demandScene.getMainScene());
}

private static void handleManageArticlesEvent(ActionEvent e, Stage window) {
  double x = window.getWidth();
  double y = window.getHeight();

  window.setWidth(x);
```

```
        window.setHeight(y);

        ArticleScene articleScene = new ArticleScene(window);
        window.setScene(articleScene.getManageArticleScene());
    }

    public static void sceneSetup(Window owner, Stage stage, Scene scene, String stageTitle) {
        stage.setScene(scene);
        stage.setTitle(stageTitle);
        stage.setMinWidth(300);
        stage.setMinHeight(300);
        stage.initModality(Modality.WINDOW_MODAL);
        stage.initStyle(StageStyle.UTILITY);
        stage.initOwner(owner);
    }

    public static TableView<ReceivingOffice> createOfficesTable() {
        TableView<ReceivingOffice> officesTable = new TableView<>();
        officesTable.setId("offices-table");
        officesTable.setItems(dbMock.getInstance().getOffices());
        officesTable.setColumnResizePolicy(TableView.CONSTRAINED_RESIZE_POLICY);

        TableColumn<ReceivingOffice, Integer> idCol = new TableColumn<>("ID");
        TableColumn<ReceivingOffice, String> nameCol = new TableColumn<>("Name");
        TableColumn<ReceivingOffice, String> fedStateCol = new TableColumn<>("Bundesland");
        TableColumn<ReceivingOffice, String> distCol = new TableColumn<>("Bezirk");
        TableColumn<ReceivingOffice, String> addCol = new TableColumn<>("Adresse");
        TableColumn<ReceivingOffice, String> statCol = new TableColumn<>("Status");

        idCol.setCellValueFactory(new PropertyValueFactory<>("id"));
        nameCol.setCellValueFactory(new PropertyValueFactory<>("name"));
        fedStateCol.setCellValueFactory(new PropertyValueFactory<>("federalState"));
        distCol.setCellValueFactory(new PropertyValueFactory<>("district"));
        addCol.setCellValueFactory(new PropertyValueFactory<>("address"));
        statCol.setCellValueFactory(new PropertyValueFactory<>("status"));

        idCol.setMinWidth(40);
        nameCol.setMinWidth(150);
        fedStateCol.setMinWidth(80);
        distCol.setMinWidth(80);
        addCol.setMinWidth(150);
        statCol.setMinWidth(80);

        officesTable.getColumns().add(idCol);
        officesTable.getColumns().add(nameCol);
        officesTable.getColumns().add(addCol);
        officesTable.getColumns().add(fedStateCol);
        officesTable.getColumns().add(distCol);
        officesTable.getColumns().add(statCol);

        return officesTable;
      }

    }
```

2.5. Design

```
/*File administration.css*/

#demand-buttons-pane, #article-buttons-pane, #offices-buttons-pane{
    -fx-padding: 10;
    -fx-spacing: 10;
}

#standard-button{
    -fx-min-width: 100;
}

#demand-table, #offices-table, #article-table, #donations-table{
    -fx-max-width: 1500;
    -fx-max-height: 400;
    -fx-border-color: grey;
}

#headline{
    -fx-font-weight: bold;
    -fx-font-size: 20;
    -fx-padding: 10;
}

#activeOfficeControl-hbox{
    -fx-max-width: 700;
    -fx-padding: 10;
    -fx-spacing: 20;
    -fx-hgap: 20;
}

#article-input-grid, #demand-input-grid, #offices-input-grid{
    -fx-padding: 10;
    -fx-hgap: 20;
    -fx-vgap: 20;
}

#demand-screen-start-pane, #articles-scree-start-pane, #offices-screen-start-pane {
    -fx-alignment: center;
    -fx-spacing: 20;
}

#demand-pane-content, #articles-pane-content, #offices-pane-content{
    -fx-padding: 10;
}

#add-demand-input-grid{
    -fx-padding: 10;
    -fx-hgap: 20;
    -fx-vgap: 20;
}

#demand-amount-selection-pane{
    -fx-hgap: 20;
    -fx-vgap: 20;
    -fx-spacing: 10;
}


#button-login {
    -fx-padding: 5;
    -fx-min-height: 35;
    -fx-min-width: 100;
    -fx-font-weight: bold;
    -fx-text-fill: #FFD600;
    -fx-background-color: #005BBC;
    -fx-spacing: 10;
}

#button-login:hover {
    -fx-background-color: #1B7EE8;
}

#login-pane{
    -fx-padding: 10;
}

#login-screen-start-pane {
    -fx-alignment: center;
    -fx-spacing: 20;
}

#filters-pane{
    -fx-padding: 10;
    -fx-spacing: 10;
    -fx-hgap: 20;
    -fx-vgap: 20;
}

#filter-tile-pane{
    -fx-hgap: 40;
    -fx-vgap: 20;
}

#manage-demand-input-amount{
    -fx-max-width: 150;
}



/*donationsApp.css*/

#my-donations-button{
    -fx-graphic: url('myDonations-button.png');
    -fx-background-color: none;
    -fx-border-width: 0;
```

```
}

#my-donations-button:hover {
    -fx-graphic: url('myDonations-button_hovered.png');
    -fx-background-color: none;
    -fx-border-width: 0;
}

#myDonations-pane{
    -fx-alignment: top-right;
    -fx-padding: 5
}

#filters-pane {
    -fx-spacing: 5;
    -fx-padding: 10;
}

#filter-tile-pane{
    -fx-padding: 5;
    -fx-spacing: 5;
    -fx-hgap: 20;
    -fx-vgap: 5;
}

#standard-button{
    -fx-min-width: 100;
}

#demand-tile-vbox{
    -fx-border-radius: 2;
    -fx-border-color: #005BBC;
    -fx-border-width: 2;
    -fx-padding: 10;
    -fx-vgap: 5;
}

#text-search-field{
    -fx-max-width: 200;
}
```

```css
#button-donate, #register-button, #login-button,
#make-donation-button, #tokenClose-btn{
    -fx-padding: 5;
    -fx-min-height: 35;
    -fx-min-width: 100;
    -fx-font-weight: bold;
    -fx-text-fill: #FFD600;
    -fx-background-color: #005BBC;
    -fx-spacing: 10;
}

#button-donate:hover, #register-button:hover, #login-button:hover,
#make-donation-button:hover, #tokenClose-btn:hover{
    -fx-background-color: #4293EA;
}

#donation-scene-scroll-area{
    -fx-padding: 10;
    -fx-spacing: 10;
    -fx-max-width: 440;
}

#donations-grid{
    -fx-padding: 5;
    -fx-spacing: 5;
    -fx-hgap: 20;
    -fx-vgap: 5;
}

#demand-tile-vbox{
    -fx-alignment: center;
    -fx-min-width: 400;
    -fx-max-width: 601;
    -fx-hgap: 100;
}

#filtered-results-vbox{
    -fx-spacing: 10;
    -fx-alignment: center;
}
```

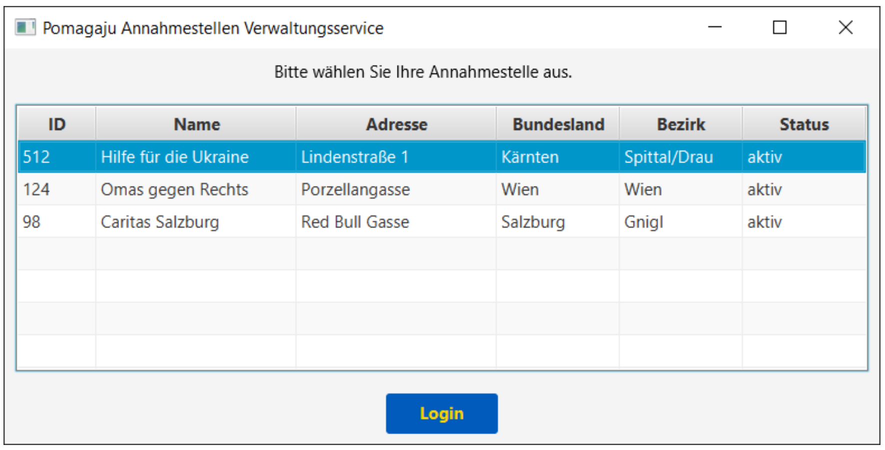
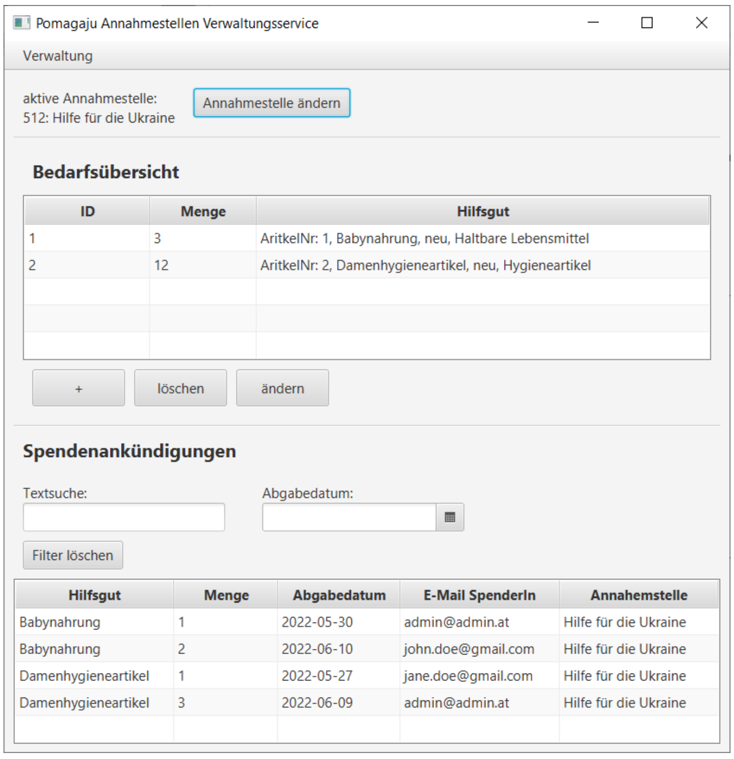
```
#register-label, #login-label{
    -fx-font-size: 18px;
    -fx-font-weight: bold;
}

#login-VBox, #register-VBox{
    -fx-alignment: center;
    -fx-padding: 20;
    -fx-spacing: 10;
}

#registerBox-dialog, #loginBox-dialog, #make-donation-dialog {
    -fx-vgap: 15;
    -fx-padding: 10;
    -fx-hgap: 15;
    -fx-spacing: 20;
}

#make-donation-input-VBox, #token-VBox{
    -fx-spacing: 20;
}

#token-label{
    -fx-font-size: 20;
    -fx-font-weight: bold;
}

#token-label, #token-intro-label{
    -fx-text-alignment: center;
}

#token-VBox{
    -fx-alignment: center;
}
```

3. Tests

Für das GUI ist es schwierig Unit-Tests zu schreiben. Ich habe daher Use-Cases definiert und dokumentiere diese mit Screenshots. Dadurch soll die Funktionalität der App gezeigt werden.

3.1. Verwaltungsanwendung:

Use Case 1: Bedarfsmeldung erstellen:

	aktive Annahmestelle wird ausgewählt
	Infos für ausgewählte Annahmestelle werden angezeigt. User klickt auf + Button, um Bedarfsmeldung hinzuzufügen

Pomagaju Annahmestellen Verwaltungsservice

Bitte wählen Sie Ihre Annahmestelle aus.

ID	Name	Adresse	Bundesland	Bezirk	Status
512	Hilfe für die Ukraine	Lindenstraße 1	Kärnten	Spittal/Drau	aktiv
124	Omas gegen Rechts	Porzellangasse	Wien	Wien	aktiv
98	Caritas Salzburg	Red Bull Gasse	Salzburg	Gnigl	aktiv

Pomagaju Annahmestellen Verwaltungsservice

Verwaltung

aktive Annahmestelle:
512: Hilfe für die Ukraine

Bedarfsübersicht

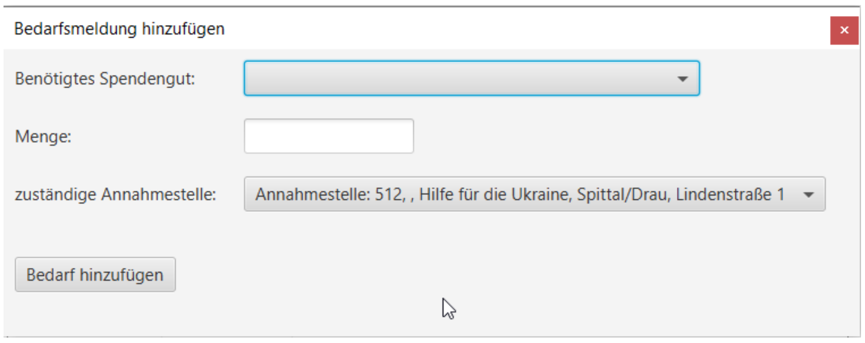
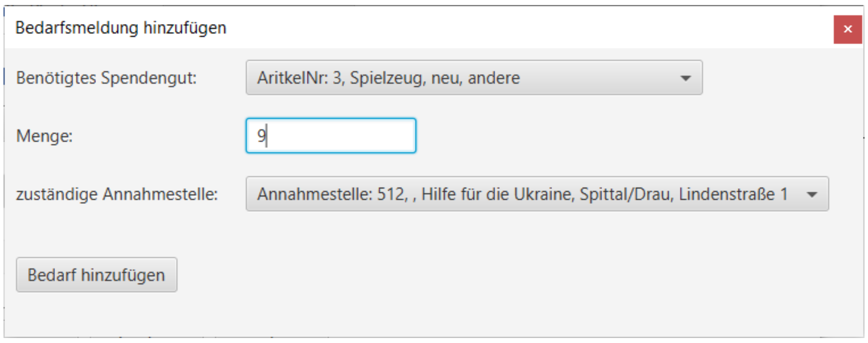
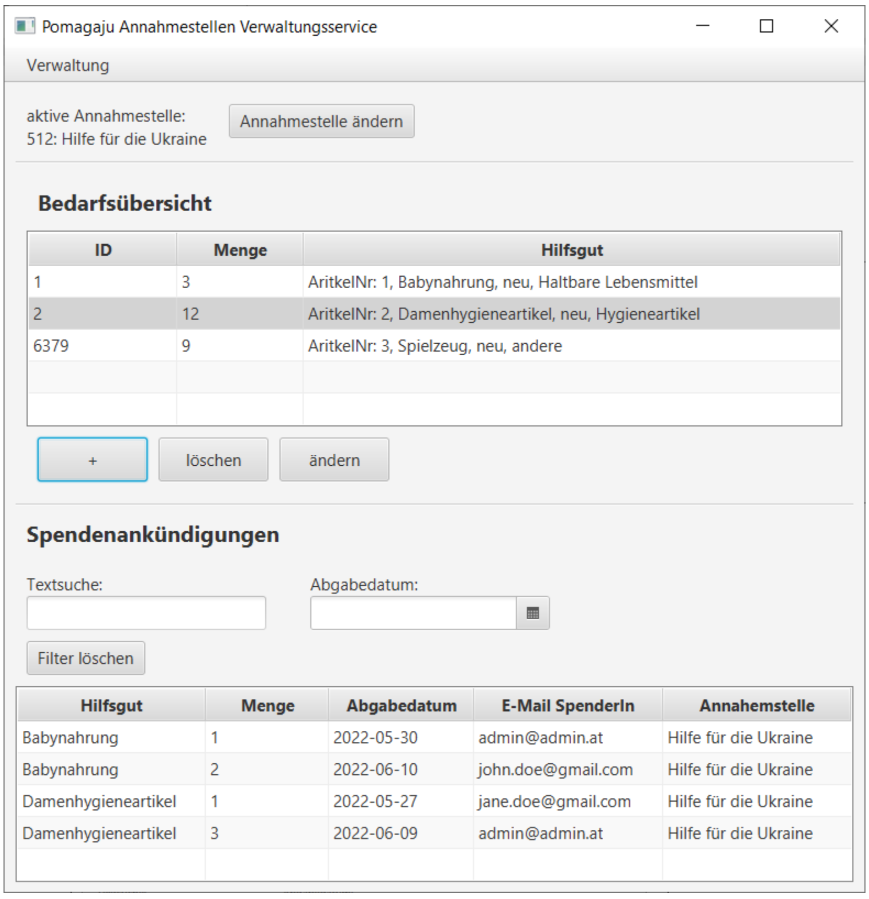
ID	Menge	Hilfsgut
1	3	AritkelNr: 1, Babynahrung, neu, Haltbare Lebensmittel
2	12	AritkelNr: 2, Damenhygieneartikel, neu, Hygieneartikel

Spendenankündigungen

Textsuche:

Abgabedatum:

Hilfsgut	Menge	Abgabedatum	E-Mail SpenderIn	Annahemstelle
Babynahrung	1	2022-05-30	admin@admin.at	Hilfe für die Ukraine
Babynahrung	2	2022-06-10	john.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	1	2022-05-27	jane.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	3	2022-06-09	admin@admin.at	Hilfe für die Ukraine

	Maske öffnet sich. Die aktive Annahmestelle ist vorausgewählt.
	User vervollständigt die Maske
	Der neu angelegte Bedarf wird nun angezeigt

Bedarfsmeldung hinzufügen

Benötigtes Spendengut:	
Menge:	
zuständige Annahmestelle:	Annahmestelle: 512, , Hilfe für die Ukraine, Spittal/Drau, Lindenstraße 1

Bedarfsmeldung hinzufügen

Benötigtes Spendengut:	AritkelNr: 3, Spielzeug, neu, andere
Menge:	9
zuständige Annahmestelle:	Annahmestelle: 512, , Hilfe für die Ukraine, Spittal/Drau, Lindenstraße 1

Pomagaju Annahmestellen Verwaltungsservice

aktive Annahmestelle:
512: Hilfe für die Ukraine

Bedarfsübersicht

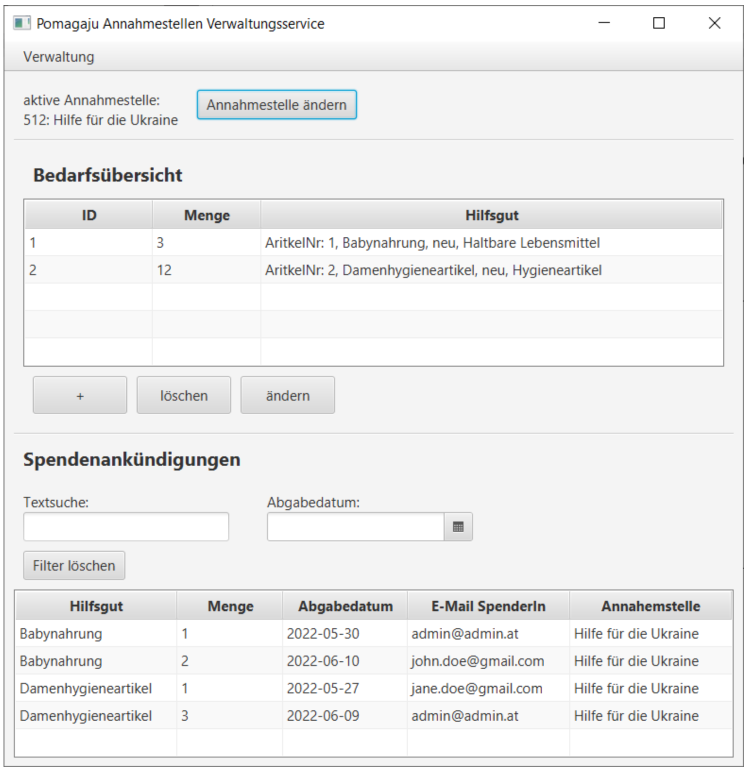
ID	Menge	Hilfsgut
1	3	AritkelNr: 1, Babynahrung, neu, Haltbare Lebensmittel
2	12	AritkelNr: 2, Damenhygieneartikel, neu, Hygieneartikel
6379	9	AritkelNr: 3, Spielzeug, neu, andere

Spendenankündigungen

Textsuche:
Abgabedatum:

Hilfsgut	Menge	Abgabedatum	E-Mail SpenderIn	Annahemstelle
Babynahrung	1	2022-05-30	admin@admin.at	Hilfe für die Ukraine
Babynahrung	2	2022-06-10	john.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	1	2022-05-27	jane.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	3	2022-06-09	admin@admin.at	Hilfe für die Ukraine

Use Case 2: Löschen einer Annahmestelle:



Pomagaju Annahmestellen Verwaltungsservice

Verwaltung

aktive Annahmestelle:
512: Hilfe für die Ukraine

Annahmestelle ändern

Bedarfsübersicht

ID	Menge	Hilfsgut
1	3	AritkelNr: 1, Babynahrung, neu, Haltbare Lebensmittel
2	12	AritkelNr: 2, Damenhygieneartikel, neu, Hygieneartikel

+ löschen ändern

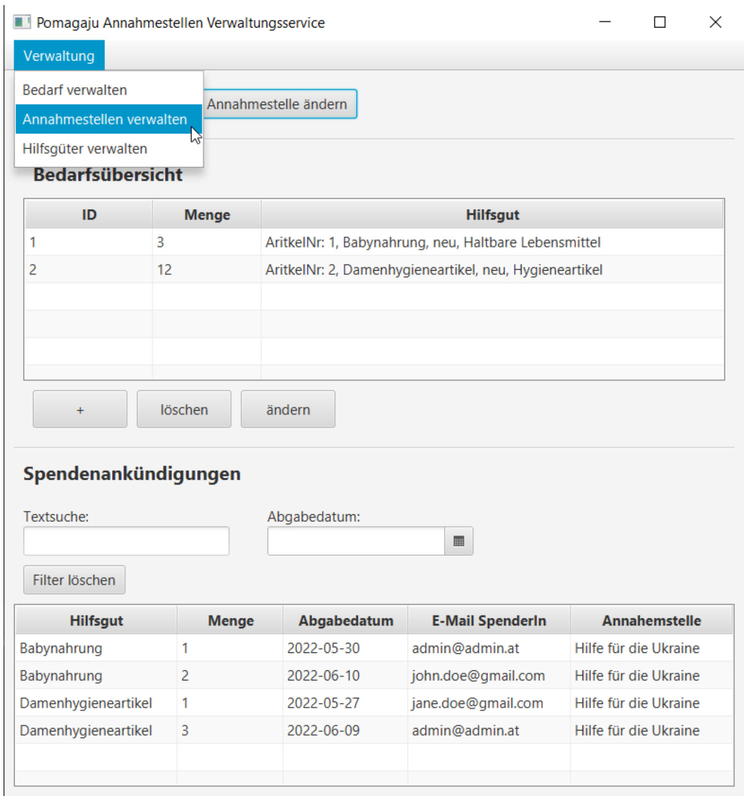
Spendenankündigungen

Textsuche:
Abgabedatum:

Filter löschen

Hilfsgut	Menge	Abgabedatum	E-Mail SpenderIn	Annahemstelle
Babynahrung	1	2022-05-30	admin@admin.at	Hilfe für die Ukraine
Babynahrung	2	2022-06-10	john.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	1	2022-05-27	jane.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	3	2022-06-09	admin@admin.at	Hilfe für die Ukraine

User hat bereits eine Annahmestelle ausgewählt



Pomagaju Annahmestellen Verwaltungsservice

Verwaltung

Annahmestelle ändern

Bedarfsübersicht

ID	Menge	Hilfsgut
1	3	AritkelNr: 1, Babynahrung, neu, Haltbare Lebensmittel
2	12	AritkelNr: 2, Damenhygieneartikel, neu, Hygieneartikel

+ löschen ändern

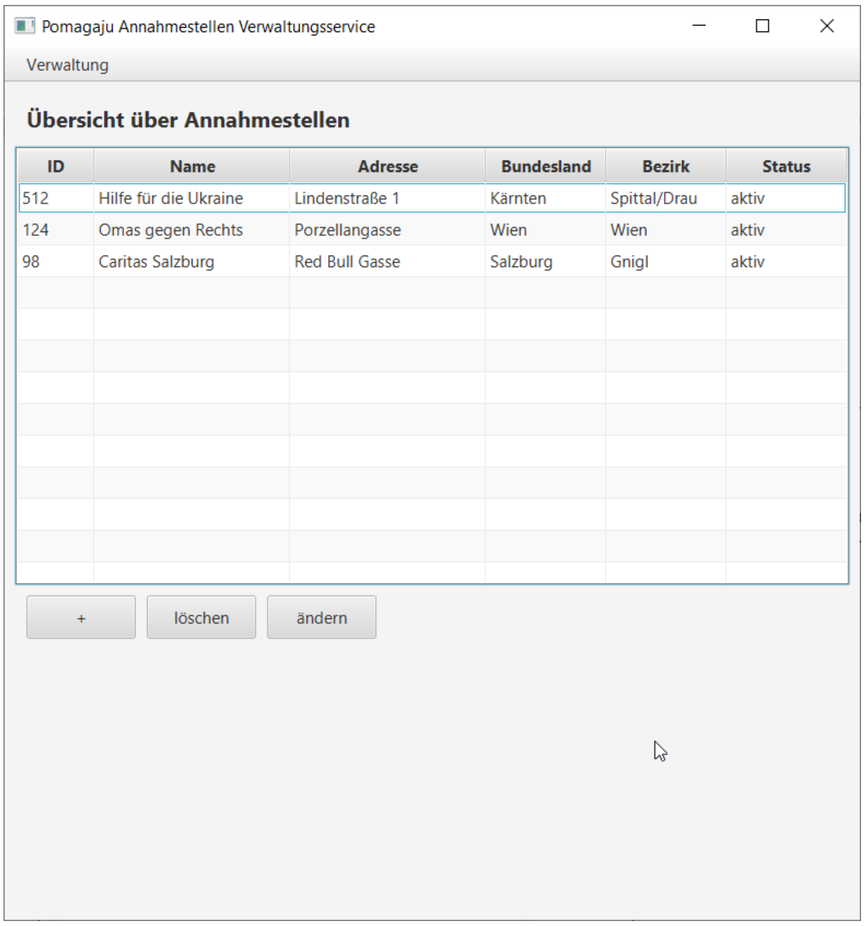
Spendenankündigungen

Textsuche:
Abgabedatum:

Filter löschen

Hilfsgut	Menge	Abgabedatum	E-Mail SpenderIn	Annahemstelle
Babynahrung	1	2022-05-30	admin@admin.at	Hilfe für die Ukraine
Babynahrung	2	2022-06-10	john.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	1	2022-05-27	jane.doe@gmail.com	Hilfe für die Ukraine
Damenhygieneartikel	3	2022-06-09	admin@admin.at	Hilfe für die Ukraine

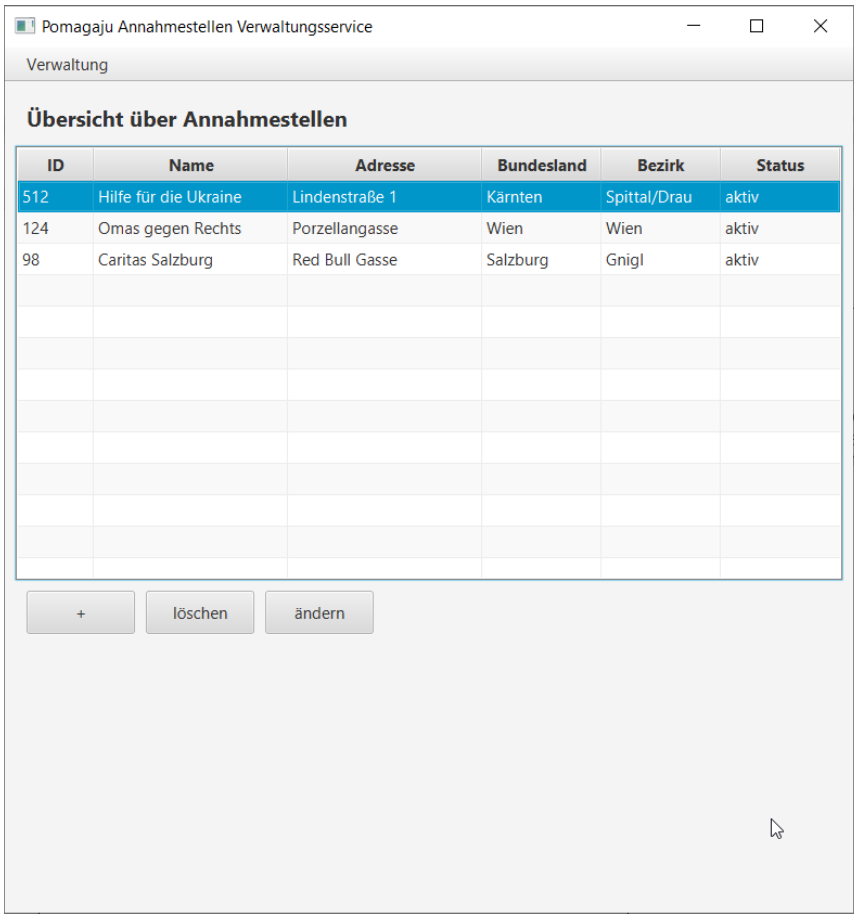
Über die Menüleiste kann der User zur Maske für die Verwaltung der Annahmestellen navigieren.



Übersicht über Annahmestellen

ID	Name	Adresse	Bundesland	Bezirk	Status
512	Hilfe für die Ukraine	Lindenstraße 1	Kärnten	Spittal/Drau	aktiv
124	Omas gegen Rechts	Porzellangasse	Wien	Wien	aktiv
98	Caritas Salzburg	Red Bull Gasse	Salzburg	Gnigl	aktiv

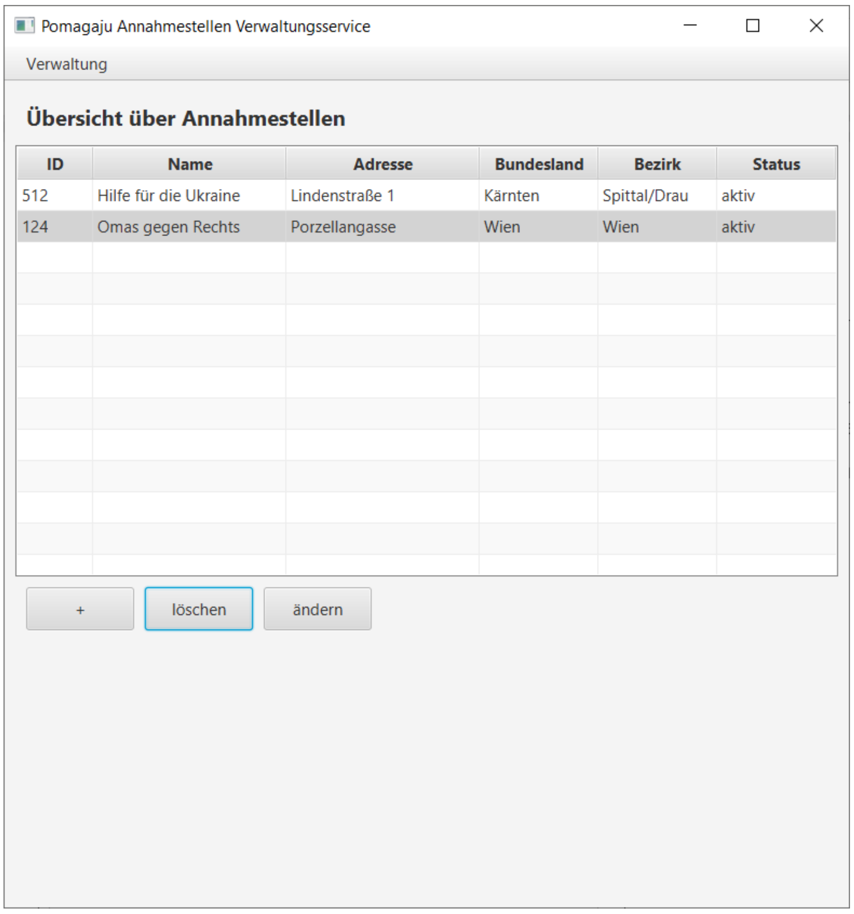
User erhält hier die Übersicht über die Annahmestellen. Über die Buttons unterhalb können diese verwaltet werden.



Übersicht über Annahmestellen

ID	Name	Adresse	Bundesland	Bezirk	Status
512	Hilfe für die Ukraine	Lindenstraße 1	Kärnten	Spittal/Drau	aktiv
124	Omas gegen Rechts	Porzellangasse	Wien	Wien	aktiv
98	Caritas Salzburg	Red Bull Gasse	Salzburg	Gnigl	aktiv

Die Annahmestelle "Hilfe für die Ukraine" kann nicht gelöscht werden, weil es für diese noch Bedarfsmeldungen gibt.

	Die Annahmestelle "Caritas Salzburg" konnte gelöscht werden, weil dafür keine Bedarfsmeldungen vorliegen.

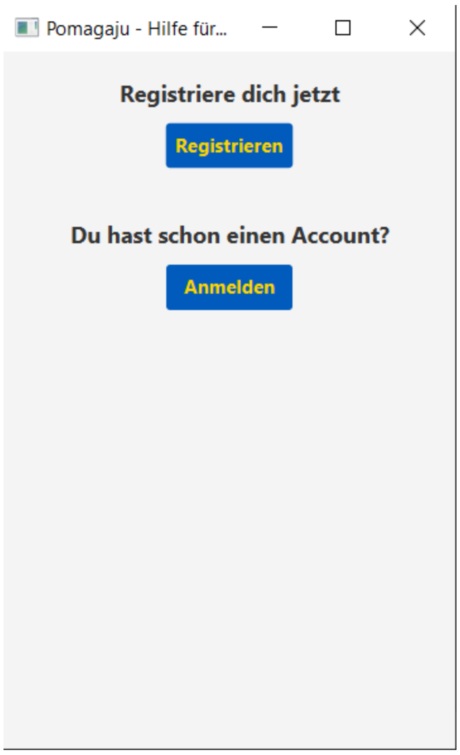
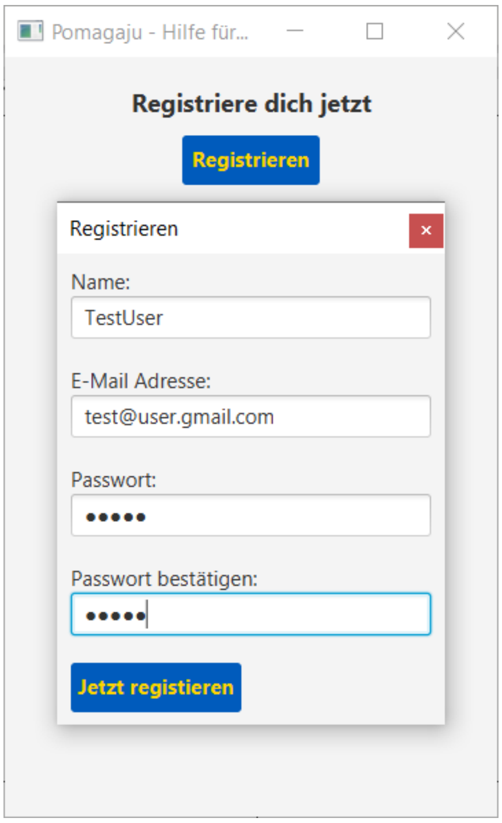
Pomagaju Annahmestellen Verwaltungsservice
Verwaltung
Übersicht über Annahmestellen
+
löschen
ändern

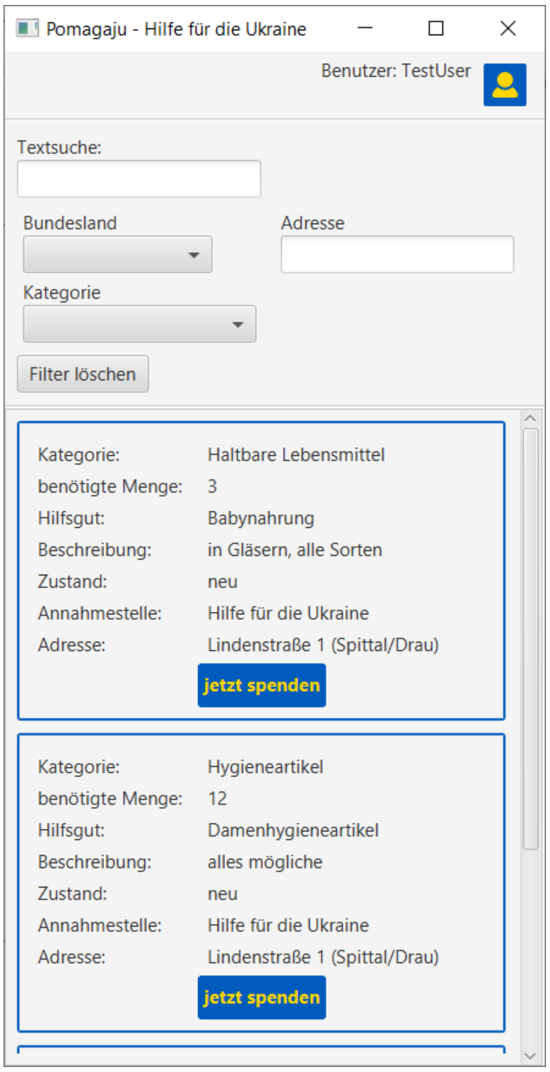
ID	Name	Adresse	Bundesland	Bezirk	Status
512	Hilfe für die Ukraine	Lindenstraße 1	Kärnten	Spittal/Drau	aktiv
124	Omas gegen Rechts	Porzellangasse	Wien	Wien	aktiv

3.2. Verwaltungsanwendung:

Use Case 1:

- Einen User Registrieren
- Bedarfsmeldungen im Bundesland Wien suchen
- Für eine Meldung eine Spendenanmeldung abgeben.

 Pomagaju - Hilfe für... Registriere dich jetzt Registrieren Du hast schon einen Account? Anmelden	Ein neuer User kann sich über "Registrieren" registrieren.
 Pomagaju - Hilfe für... Registriere dich jetzt Registrieren Registrieren Jetzt registrieren Name: TestUser E-Mail Adresse: test@user.gmail.com Passwort: ••••• Passwort bestätigen: •••••	Der User muss seinen Namen, Email und Passwort eingeben. Das Passwort muss übereinstimmen und mindestens 5 Zeichen enthalten. Die E-Mail-Adresse muss zumindest grob der Struktur einer richtigen E-Mail entsprechen (Regex-Überprüfung).

	Der neu registrierte User sieht die Bedarfsmeldungen aller Annahmestellen.

Pomagaju - Hilfe für die Ukraine

Benutzer: TestUser

Textsuche:

Bundesland

Adresse

Kategorie

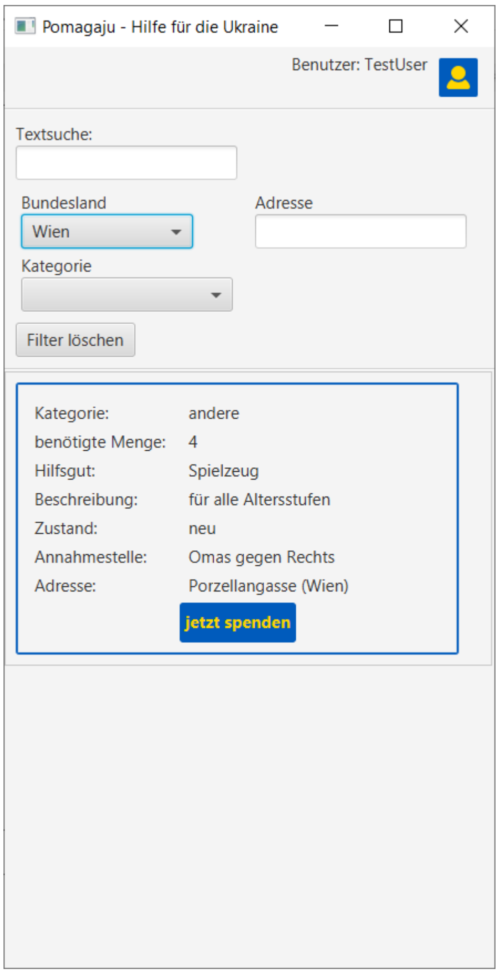
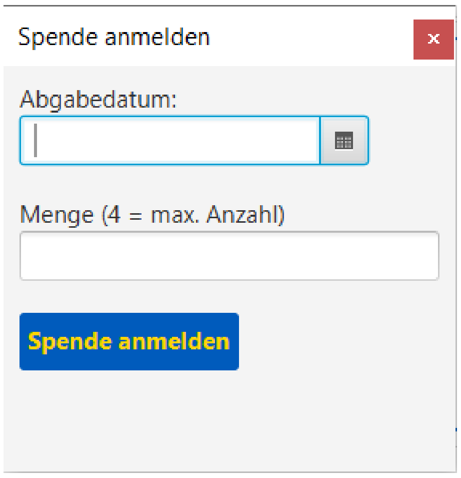
Filter löschen

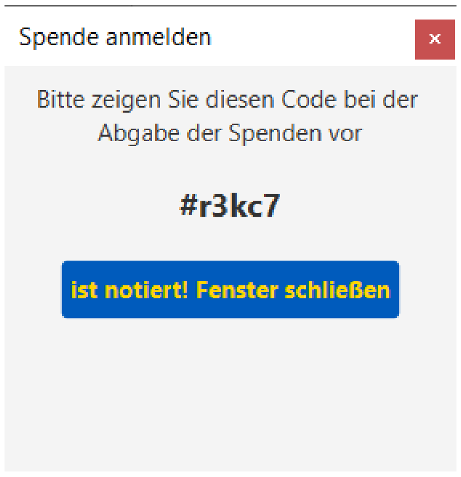
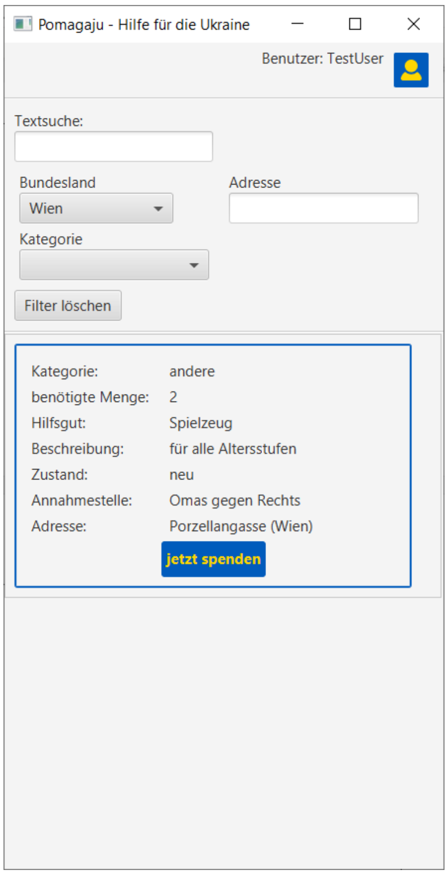
Kategorie:	Haltbare Lebensmittel
benötigte Menge:	3
Hilfsgut:	Babynahrung
Beschreibung:	in Gläsern, alle Sorten
Zustand:	neu
Annahmestelle:	Hilfe für die Ukraine
Adresse:	Lindenstraße 1 (Spittal/Drau)

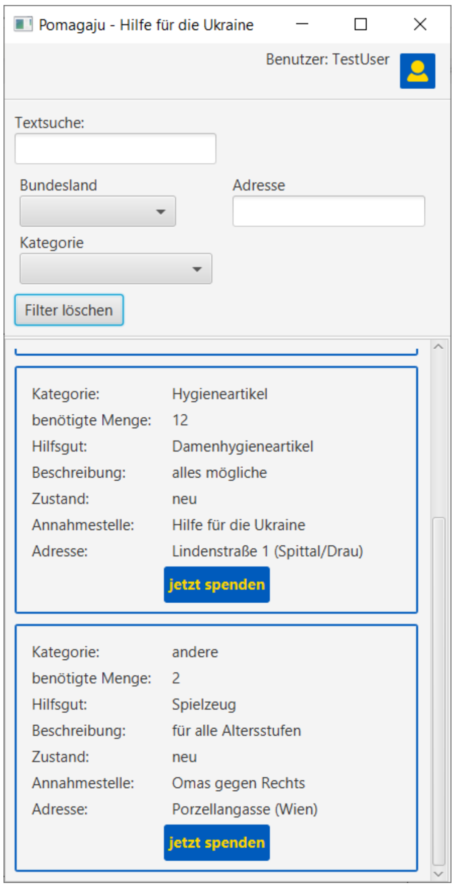
jetzt spenden

Kategorie:	Hygieneartikel
benötigte Menge:	12
Hilfsgut:	Damenhygieneartikel
Beschreibung:	alles mögliche
Zustand:	neu
Annahmestelle:	Hilfe für die Ukraine
Adresse:	Lindenstraße 1 (Spittal/Drau)

jetzt spenden

 Pomagaju - Hilfe für die Ukraine Benutzer: TestUser Textsuche: Bundesland: Wien Adresse Kategorie Filter löschen Kategorie: andere benötigte Menge: 4 Hilfsgut: Spielzeug Beschreibung: für alle Altersstufen Zustand: neu Annahmestelle: Omas gegen Rechts Adresse: Porzellangasse (Wien) jetzt spenden	Über die Filter kann eine Bedarfsmeldung gesucht werden.
 Spende anmelden Abgabedatum: Menge (4 = max. Anzahl) Spende anmelden	Für die angezeigten Meldungen können Spendenanmeldungen abgegeben werden. Die Menge kann maximal 4 betragen (weil max. 4 Stück benötigt werden).

 Spende anmelden Bitte zeigen Sie diesen Code bei der Abgabe der Spenden vor **#r3kc7**	Dem User wird nun ein Token angezeigt (aktuell immer der selbe).
 Kategorie: andere; benötigte Menge: 2; Hilfsgut: Spielzeug; Beschreibung: für alle Altersstufen; Zustand: neu; Annahmestelle: Omas gegen Rechts; Adresse: Porzellangasse (Wien)	Der Bedarf ist nun um 2 reduziert (was der Menge der vorhin abgegebenen Spendenanmeldung ent-spricht).

 Pomagaju - Hilfe für die Ukraine Benutzer: TestUser Textsuche: Bundesland Adresse Kategorie Filter löschen Kategorie: Hygieneartikel benötigte Menge: 12 Hilfsgut: Damenhygieneartikel Beschreibung: alles mögliche Zustand: neu Annahmestelle: Hilfe für die Ukraine Adresse: Lindenstraße 1 (Spittal/Drau) jetzt spenden Kategorie: andere benötigte Menge: 2 Hilfsgut: Spielzeug Beschreibung: für alle Altersstufen Zustand: neu Annahmestelle: Omas gegen Rechts Adresse: Porzellangasse (Wien) jetzt spenden	Wenn der Filter gelöscht wird, werden wieder alle Bedarfsmeldungen angezeigt.

Natürlich könnten noch weitere Use-Cases demonstriert werden aber bei 101 Seiten und 35 Stunden Aufwand kann man es auch mal gut sein lassen 😉