```
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Setzt einen ImageButton in ein LinearLayout mit den zuvor festgelegten
Parametern
public static void addViewIBTN (ImageButton imageButton, LinearLayout
linearLayout){
    LinearLayout.LayoutParams params = new
LinearLayout.LayoutParams(400,400);
    params.setMargins(0,20,0,20);
    imageButton.setLayoutParams(params);
    linearLayout.addView(imageButton);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Generiert einen ImageButton mit den Paramertern für das Drawabel, die
Größe und die Hintergrundfarbe
//Gibt dem generierten ImageButton einen OnClickListener mit
//Der ImageButton wird OnClick gelöscht und startet hierbei eine
Löschanimation
public static ImageButton setPicture(String itemname, String itemnamekey,
Context context, DBHelferlein dbHelferlein, Integer color,
AnimationsHelferlein animationsHelferlein, ImageButton muelltonne){
    ImageButton imageButton = new ImageButton(context);

imageButton.setImageBitmap(BitmapFactory.decodeByteArray(dbHelferlein.getDr
awableFromTable(itemname),0,dbHelferlein.getDrawableFromTable(itemname).len
gth));
    imageButton.setScaleType(ImageView.ScaleType.FIT_CENTER);
    imageButton.setBackgroundColor(color);

    imageButton.setOnClickListener(v -> {
        dbHelferlein.deleteIndividuallyfromWarenkorb(itemnamekey,
PreferenceHelferlein.loadUserFromPref(context, KEY_AKTIVERNUTZER));

animationsHelferlein.ownAnimationWithInvisible(imageButton,muelltonne);
    });
     return imageButton;
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Setzt einen Button in ein LinearLayout mit den zuvor festgelegten
Parametern.
public static void addViewBTN (Button button, LinearLayout linearLayout){
    LinearLayout.LayoutParams params = new
LinearLayout.LayoutParams(400,400);
    params.setMargins(0,20,0,20);
    button.setLayoutParams(params);
    linearLayout.addView(button);
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Generiert einen Button mit den Paramtern für den Anzeigetext und die
Hintergrundfarbe
//Gibt dem generierten Button einen OnClickListener mit
//Der Button speichert OnClick seinen Anzeignamen in einer SharedPreference
und öffnet eine Menüauswahl
public static Button generateButtonsAndSetName(String username, Context
context, Integer farbe, DBHelferlein dbHelferlein){
    Button button = new Button(context);
    button.setText(username);
    button.setBackgroundColor(farbe);

    button.setOnClickListener(v -> {
        PreferenceHelferlein.saveUserInPref(context,username,
KEY_AKTIVERNUTZER);
        menueauswahl(context,dbHelferlein);
    });

    return button;
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Gibt dem User die Möglichkeit zu entscheiden was mit dem geklickten
Objekt passieren soll.
//Benutzer wechseln: Setzt den aktivenNutzer neu und lädt zurück auf die
MainActivty
//Benutzer bearbeiten: Der User kann die Daten des ausgewählten Nutzers
bearbeiten
//Benutzer löschen: Der Nutzer wird aus der Datenbank gelöscht
//Abbrehcen: Dialog wird beendet
private static void menueauswahl(Context context, DBHelferlein
dbHelferlein) {
    final CharSequence[] options = {"Benutzer wechsel", "Benutzer
bearbeiten", "Benutzer löschen", "Abbrechen"};
    AlertDialog.Builder builder = new AlertDialog.Builder(context);
    builder.setTitle("Einstellungen für " +
PreferenceHelferlein.loadUserFromPref(context, KEY_AKTIVERNUTZER));
    builder.setItems(options, (dialog, item) -> {
        if (options[item].equals("Benutzer wechsel")) {
            dialog.dismiss();
            Intent refresh = new Intent(context, MainActivity.class);
            context.startActivity(refresh);
        } else if (options[item].equals("Benutzer bearbeiten")) {
            UserOverviewActivity.anlegen_bearbeiten = "Benutzer
aktualisieren";
            dialog.dismiss();
            Intent intentNutzerErstellen = new Intent(context,
UserCreationActivity.class);
            context.startActivity(intentNutzerErstellen);

        } else if (options[item].equals("Benutzer löschen")) {
            wirklichLoeschen(context,dbHelferlein);
            dialog.dismiss();

        } else if (options[item].equals("Abbrechen")) {
            dialog.dismiss();
        }
    });
    builder.show();
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Abfrage ob man einen Nutzer tatsächlich löschen möchte
//JA: Der User wird aus der Datenbank gelöscht
//NEIN: Der Dialog wird beendet und der User wird nicht gelöscht
private static void wirklichLoeschen(Context context, DBHelferlein
dbHelferlein) {
    final CharSequence[] options = {"Ja", "Nein"};
    AlertDialog.Builder builder = new AlertDialog.Builder(context);
    builder.setTitle("Wollen Sie den Nutzer wirklich löschen?");
    builder.setItems(options, (dialog, item) -> {
        if (options[item].equals("Ja")) {
            dialog.dismiss();

dbHelferlein.deletefromUserdaten(PreferenceHelferlein.loadUserFromPref(cont
ext, KEY_AKTIVERNUTZER));
            dialog.dismiss();
            PreferenceHelferlein.saveUserInPref(context,"Benutzer wählen
oder erstellen", KEY_AKTIVERNUTZER);
            Intent refresh = new Intent(context,
UserOverviewActivity.class);
            context.startActivity(refresh);
        } else if (options[item].equals("Nein")) {
            dialog.dismiss();
            Intent intent = new Intent(context,UserOverviewActivity.class);
            context.startActivity(intent);
        }
    });
    builder.show();
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein
class: AddAndSetHelferlein

//Gibt eine Audi wieder wenn ein Prdoukt zum Warenkorb hinzugefügt werden
soll, der Warenkorb jedoch bereits die erlaubte Menge beinhaltet
public static void playAudioFlagVoll(Context context) {
    MediaPlayer mediaPlayernichtsmehr = MediaPlayer.create(context,
R.raw.nichtsmehr);
    mediaPlayernichtsmehr.start();
    Toast.makeText(context, "Es dürfen keine Produkte mehr hinzugefügt
werden", Toast.LENGTH_SHORT).show();
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Beim erstmaligen öffnen der App wird die Datenbank Einkaufsdatenbank.db
angelegt
public DBHelferlein(Context context) {
    super(context, "Einkaufsdatenbank.db", null, 1);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Beim erstmaligen öffnen der App werden die Tabellen Sortiment und
Userdaten angelegt
@SuppressLint("SQLiteString")
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE IF NOT EXISTS Sortiment(id_key INTEGER primary
key, bildname STRING, bild BLOB, flag STRING)");
    db.execSQL("CREATE TABLE IF NOT EXISTS Userdaten(id_key INTEGER primary
key, username STRING, flaggruen INTEGER, flagblau INETGER, flagrot
INTEGER)");
    startInsertIntoSortiment(db);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Sollten die Tabellen Sortiment und Userdaten bereits exisitieren werden
diese nicht erneut erstellt
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE if EXISTS " + sortiment);
    db.execSQL("DROP TABLE if EXISTS " + userdaten);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Initilisierung des Inhalts der Tabelle Sortiment
public void startInsertIntoSortiment(SQLiteDatabase database){

    ContentValues contentValues = new ContentValues();

    contentValues.put("id_key", 1); contentValues.put("bildname",
"roterApfel"); contentValues.put("bild",""); contentValues.put("flag",
flaggruen);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 2); contentValues.put("bildname",
"gruenerApfel"); contentValues.put("bild", ""); contentValues.put("flag",
flaggruen);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 3); contentValues.put("bildname",
"Salatgurke"); contentValues.put("bild", ""); contentValues.put("flag",
flaggruen);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 4); contentValues.put("bildname",
"Hartkaese"); contentValues.put("bild", ""); contentValues.put("flag",
flagrot);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 5); contentValues.put("bildname",
"Streichkaese"); contentValues.put("bild", ""); contentValues.put("flag",
flagrot);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 6); contentValues.put("bildname",
"Kaeseaufschnitt"); contentValues.put("bild", "");
contentValues.put("flag", flagrot);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 7); contentValues.put("bildname",
"sechserPackungEier"); contentValues.put("bild", "");
contentValues.put("flag", flagblau);
    database.insert(sortiment, null, contentValues);

    contentValues.put("id_key", 8); contentValues.put("bildname",
"zehnerPackungEier"); contentValues.put("bild","");
contentValues.put("flag", flagblau);
    database.insert(sortiment, null, contentValues);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Geniert eine Warenkorbtabelle nach dem Schemata WarenkorbUsername
@SuppressLint("SQLiteString")
public void createWarenkorbOnClick(String name){
    SQLiteDatabase db = this.getWritableDatabase();
    db.execSQL("CREATE TABLE IF NOT EXISTS Warenkorb"+name+"(btnID INTEGER,
bildwert INTEGER, itemnamekey STRING primary key, itemname STRING)");
    close();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Wird immer dann aufgerufen, wenn ein Produkt zum Warenkorb hinzugefügt
wird
//Setzt die erhaltenen Werte und schreibt diese in den WarenkorbUsername
public long insertIntoWarenkorb(ImageButton ibtn, Integer bildInteger,
String itemnamekey, String itemname, String username) {

    SQLiteDatabase database = this.getWritableDatabase();

    Cursor cursor = database.rawQuery("SELECT * FROM Warenkorb" +username,
null);

    Integer btnid = ibtn.getId();

    ContentValues contentValues = new ContentValues();
    contentValues.put("btnID", btnid);
    contentValues.put("bildwert", bildInteger);
    contentValues.put("itemnamekey", itemnamekey);
    contentValues.put("itemname", itemname);

    cursor.close();

    return database.insert(warenkorb+username, null, contentValues);

}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Löscht den gesamten Warenkorbinhalt eines Users
public void deleteCompletefromWarenkorb(String name) {
    SQLiteDatabase database = this.getWritableDatabase();
    database.delete(this.warenkorb+name, null, null);
    database.close();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Löscht einzelne Elemnete aus dem WarenkorbUsername
public void deleteIndividuallyfromWarenkorb(String itemname_local, String
username) {
    SQLiteDatabase database = this.getWritableDatabase();
    database.delete(this.warenkorb+username, "itemnamekey =?", new
String[]{itemname_local});
    database.close();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Generiert eine Liste mit allen zum aktuelln Zeitpunkt im
WarenkorbUsername Tabelle befindlichen Items und gibt diese zurück
public ArrayList<String> createArrayListOfWarenkorbItems(String username) {
    SQLiteDatabase database = this.getReadableDatabase();
    Cursor cursor = database.rawQuery("SELECT itemnamekey FROM Warenkorb"
+username, null);
    cursor.moveToFirst();

    ArrayList<String> arrayOfWarenkorbItemsNAME = new ArrayList<>();

    if(cursor.getCount() == 0){
        arrayOfWarenkorbItemsNAME.clear();
    }

    while(!cursor.isAfterLast()){

arrayOfWarenkorbItemsNAME.add(cursor.getString(cursor.getColumnIndexOrThrow
("itemnamekey")));
        cursor.moveToNext();
    }

    Collections.sort(arrayOfWarenkorbItemsNAME);

    cursor.close();
    return arrayOfWarenkorbItemsNAME;
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Befüllt die Tabelle Userdaten mit den erhaltenen Werten
public long insertIntoUserdaten(String username, Integer flaggruen, Integer
flagblau, Integer flagrot) {

    SQLiteDatabase database = this.getWritableDatabase();

    Cursor cursor = database.rawQuery("SELECT * FROM " + userdaten, null);

    if (countUserdaten >= 1) {
        countUserdaten = cursor.getCount() + 1;
    }

    ContentValues contentValues = new ContentValues();
    contentValues.put("id_key", countUserdaten);
    contentValues.put("username", username);
    contentValues.put("flaggruen", flaggruen);
    contentValues.put("flagrot", flagrot);
    contentValues.put("flagblau", flagblau);

    countUserdaten++;
    cursor.close();

    return database.insert(userdaten, null, contentValues);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Generiert eine Liste mit allen in der Userdaten Tabelle zum aktuellen
Zeitpunkt befindlichen Usern und gibt diese zurück
public ArrayList<String> createArrayListOfUserdaten() {

    SQLiteDatabase database = this.getReadableDatabase();

    Cursor cursor = database.rawQuery("SELECT username FROM Userdaten",
null);
    cursor.moveToFirst();

    ArrayList<String> arrayOfUsers = new ArrayList<>();

    if(cursor.getCount() == 0){
        arrayOfUsers.clear();
    }

    while(!cursor.isAfterLast()){

arrayOfUsers.add(cursor.getString(cursor.getColumnIndexOrThrow("username"))
);
        cursor.moveToNext();
    }

    Collections.sort(arrayOfUsers);

    cursor.close();
    return arrayOfUsers;
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Löscht einen User aus der Userdaten Tablle und seinen dazugehörigen
WarenkorbUsername
public void deletefromUserdaten(String username) {
    SQLiteDatabase database = this.getWritableDatabase();
    database.execSQL("DROP TABLE if EXISTS " + warenkorb+username);
    database.delete(this.userdaten, "username = ?", new
String[]{username});
    database.close();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Holt sich zum ausgewählten Prdouktnamen den dazugehörigen Flag
public String fetchSortiment(String produktname) {
   SQLiteDatabase database = this.getWritableDatabase();
   Cursor cursor = database.query(sortiment, new String[]{"flag" },
"bildname =?", new String[]{produktname }, null, null, null);
    if (cursor != null) {
        cursor.moveToFirst();
    }
    return cursor.getString(0);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Gibt die Anzahl der erlaubten Falgs zu einer bestimmten Flagfarbe als
Integer zurück
public Integer getFlaganzahl(String aktuellerUser,String flag) {
    SQLiteDatabase database = this.getWritableDatabase();
    Cursor cursor = database.query(userdaten, new String[]{flag}, "username
=?", new String[]{aktuellerUser }, null, null, null);
    if (cursor != null) {
        cursor.moveToFirst();
    }
    return cursor.getInt(0);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Gibt die Anzahl der erlaubten Falgs zu einer bestimmten Flagfarbe als
String zurück
public String getFlaganzahlString(String aktuellerUser,String flag) {
    SQLiteDatabase database = this.getWritableDatabase();
    Cursor cursor = database.query(userdaten, new String[]{flag}, "username
=?", new String[]{aktuellerUser }, null, null, null);
    if (cursor != null) {
        cursor.moveToFirst();
    }
    return cursor.getString(0);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Gibt zu einem bestimmten Produkt die Produktart wieder
public String getWarenkorbItemname (String itemnamekey, String
aktuellerUser) {
    SQLiteDatabase database = this.getWritableDatabase();
    Cursor cursor = database.query(warenkorb + aktuellerUser, new
String[]{"itemname"}, "itemnamekey =?", new String[]{itemnamekey}, null,
null, null);
    if (cursor != null) {
        cursor.moveToFirst();
    }
    return cursor.getString(0);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Gibt die aktuelle Anzahl eines bestimmten Falgs im WarenkorbUser wieder
public Integer flagCountEinkaufswagen(String flagart, String aktivernutzer)
{
    int count = 0;
    int x = 0;

    ArrayList<String> arrayListOfWarenkorbitems =
this.createArrayListOfWarenkorbItems(aktivernutzer);

    while (x < arrayListOfWarenkorbitems.size()) {
        String itemname =
getWarenkorbItemname(arrayListOfWarenkorbitems.get(x),aktivernutzer);
        String itemflag = fetchSortiment(itemname);

        if (flagart.equals(itemflag)) {
            count++;
            x++;
        } else {
            x++;
        }
    }
    return count;
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Vergleicht Flaganzahl des Users mit Falganzahl im WarenkorbUser und
entscheidet ob das Produkt dem WarenkorbUser hinzugefügt werden darf
public Boolean darfHinzugefuegtWerden(String produktname, String
aktiverNutzer){
    Integer countEinkaufswagen =
flagCountEinkaufswagen(fetchSortiment(produktname),aktiverNutzer);
    Integer personenmaximum = getFlaganzahl(aktiverNutzer,
fetchSortiment(produktname));
    if(countEinkaufswagen < personenmaximum){
        return true;
    } else return false;
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Holt sich das Image aus der Tabelle Sortiment
public byte[] getDrawableFromTable(String bildname){
    SQLiteDatabase database = this.getWritableDatabase();
    Cursor cursor = database.query(sortiment, new
String[]{"bild"},"bildname =?",new String[]{bildname},null,null,null);
    if (cursor != null) {
        cursor.moveToFirst();
    }
    assert cursor != null;
    return cursor.getBlob(0);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Setzt das Image in der Tabelle Sortiment
public void setDrawableFromGallery(String bildname, byte[] bildID){

    SQLiteDatabase database = this.getWritableDatabase();

    ContentValues contentValues = new ContentValues();
    contentValues.put("bild", bildID);

    database.update(sortiment,contentValues,"bildname =?",new
String[]{bildname});

}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Prüft ob die Userdaten Tabelle leer ist
public boolean isEmpty(){
    SQLiteDatabase database = this.getReadableDatabase();

    long NoOfRows = DatabaseUtils.queryNumEntries(database,userdaten);

    if (NoOfRows == 0){
        return true;
    } else {
        return false;
    }
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Konvertiert ein R.Drawabel.Image zu einem byte[]
public byte[] drawableToByteArray (Context context, Integer i) {
    InputStream inputStream = context.getResources().openRawResource(i);
    Bitmap bitmap = BitmapFactory.decodeStream(inputStream);
    bitmap = getResizedBitmap(bitmap,400);
    return getBytes(bitmap);
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Konvertiert eine Bitmap in ein byte[]
public static byte[] getBytes(Bitmap bitmap) {
    ByteArrayOutputStream stream = new ByteArrayOutputStream();
    bitmap.compress(Bitmap.CompressFormat.PNG, 0, stream);
    return stream.toByteArray();
}
```

```java
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: DBHelferlein

//Verändert die Größe einer Bitmap
public static Bitmap getResizedBitmap(Bitmap image, int maxSize) {
    int width = image.getWidth();
    int height = image.getHeight();

    float bitmapRatio = (float)width / (float) height;
    if (bitmapRatio > 1) {
        width = maxSize;
        height = (int) (width / bitmapRatio);
    } else {
        height = maxSize;
        width = (int) (height * bitmapRatio);
    }
    return Bitmap.createScaledBitmap(image, width, height, true);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: AnimationsHelferlein

//Button bleibt nach der Animation
public void ownAnimation(ImageButton item, ImageButton warenkorb){

    //ANIMATIONSSET
    AnimationSet animationSet = new AnimationSet(false);

    //KLEINER WERDEN
    ScaleAnimation scaleAnimation = new ScaleAnimation(1, (float) 0.15, 1,
(float) 0.15);
    scaleAnimation.setDuration(400);
    animationSet.addAnimation(scaleAnimation);

    //NACH RECHTS FAHREN
    TranslateAnimation translateToRightAnimation = new
TranslateAnimation(0, warenkorb.getX()-item.getX(), 0, 0);
    translateToRightAnimation.setDuration(400);
    translateToRightAnimation.setStartOffset(400);
    animationSet.addAnimation(translateToRightAnimation);

    //NACH OBEN FAHREN
    TranslateAnimation translateToTopAnimation = new TranslateAnimation(0,
0,0,warenkorb.getY()-item.getY()-50);
    translateToTopAnimation.setDuration(400);
    translateToTopAnimation.setStartOffset(800);
    animationSet.addAnimation(translateToTopAnimation);

    //ANIMATIONSSET CALL
    item.startAnimation(animationSet);

    animationSet.setAnimationListener(new Animation.AnimationListener() {
        @Override
        public void onAnimationStart(Animation animation) {
            item.setVisibility(ImageButton.VISIBLE);

        }

        @Override
        public void onAnimationEnd(Animation animation) {

        }

        @Override
        public void onAnimationRepeat(Animation animation) {

        }
    });

}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: AnimationsHelferlein

//Button bleibt nach der Animation
public void ownAnimationWithInvisible(ImageButton item, ImageButton
warenkorb){

    //ANIMATIONSSET
    AnimationSet animationSet2 = new AnimationSet(false);

    //KLEINER WERDEN
    ScaleAnimation scaleAnimation = new ScaleAnimation(1, (float) 0.15, 1,
(float) 0.15);
    scaleAnimation.setDuration(400);
    animationSet2.addAnimation(scaleAnimation);

    //NACH RECHTS FAHREN
    TranslateAnimation translateToRightAnimation = new
TranslateAnimation(0, warenkorb.getX()-item.getX(), 0, 0);
    translateToRightAnimation.setDuration(400);
    translateToRightAnimation.setStartOffset(400);
    animationSet2.addAnimation(translateToRightAnimation);

    //NACH OBEN FAHREN
    TranslateAnimation translateToTopAnimation = new TranslateAnimation(0,
0,0,warenkorb.getY()-item.getY()-50);
    translateToTopAnimation.setDuration(400);
    translateToTopAnimation.setStartOffset(800);
    animationSet2.addAnimation(translateToTopAnimation);

    //ANIMATIONSSET CALL
    item.startAnimation(animationSet2);

    animationSet2.setAnimationListener(new Animation.AnimationListener() {
        @Override
        public void onAnimationStart(Animation animation) {
            item.setVisibility(ImageButton.VISIBLE);

        }

        @Override
        public void onAnimationEnd(Animation animation) {
            item.setVisibility(ImageButton.GONE);
        }

        @Override
        public void onAnimationRepeat(Animation animation) {

        }
    });

}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: PreferenceHelferlein


//Speichter eine Integervariable in einer SharedPrefrence
public static void saveTotalInPref(Context context, int total, String key){
    SharedPreferences sharedPreferences =
context.getSharedPreferences(MY_PREFERENCE_NAME, Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putInt(key, total);
    editor.apply();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: PreferenceHelferlein

//Lädt eine Integer aus einer SharedPreference
public static int loadTotalFromPref (Context context, String key){
    SharedPreferences pref =
context.getSharedPreferences(MY_PREFERENCE_NAME, Context.MODE_PRIVATE);
    return pref.getInt(key,0);
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: PreferenceHelferlein

//Prüft, ob die App das erste mal auf einem Gerät gestartet wurde
public static boolean firstAppStart(Context context, String key) {
    SharedPreferences sharedPreferences =
context.getSharedPreferences(MY_PREFERENCE_NAME, Context.MODE_PRIVATE);
    if (sharedPreferences.getBoolean(key, true)) {
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putBoolean(key, false);
        editor.apply();
        return true;
    } else {
        return false;
    }
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: PreferenceHelferlein

//Lädt einen String aus einer SharedPreference
public static String loadUserFromPref(Context context, String key){
    SharedPreferences sharedPreferences =
context.getSharedPreferences(MY_PREFERENCE_NAME, Context.MODE_PRIVATE);
    return sharedPreferences.getString(key, "");
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Helferlein;
class: PreferenceHelferlein

//Speichter eine Stringvariable in einer SharedPrefrence
public static void saveUserInPref(Context context, String user, String
key){
    SharedPreferences sharedPreferences =
context.getSharedPreferences(MY_PREFERENCE_NAME, Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putString(key, user);
    editor.apply();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Menuebereich;
class: IbtnChangerActivity

//Öffnet ein Dialog wo der User auswählen kann
//Foto aufnehmen: ein Bild mit der Kamera aufnehmen
//Gallerie: ein Bild aus der Gallerie des Smartphones wählen
//Abbrechen: Dialog beenden
private void selectImage(String string) {
    final CharSequence[] options = { "Foto aufnehmen", "Bild aus Gallerie
auswählen","Abbrechen" };
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Bild ändern");
    builder.setItems(options, (dialog, item) -> {
        if (options[item].equals("Foto aufnehmen"))
        {
            Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
            File file = new File(Environment.getExternalStorageDirectory(),
"temp.jpg");
            intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(file));
            startActivityForResult(intent, 1);
            agreement(string);
        }
        else if (options[item].equals("Bild aus Gallerie auswählen"))
        {
            Intent intent = new   Intent(Intent.ACTION_PICK,
MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
            intent.setType("image/*");
            startActivityForResult(intent, 2);
            agreement(string);
        }
        else if (options[item].equals("Abbrechen")) {
            dialog.dismiss();
        }
    });
    builder.show();
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Menuebereich;
class: IbtnChangerActivity

//Das in selectImage() gewählte Bild auf einer Bitmap speichern
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == RESULT_OK) {
        if (requestCode == 1) {
            //Hier müsste man das Kamerabild verarbeiten
        } else if (requestCode == 2) {
            try {
                InputStream inputStream =
getContentResolver().openInputStream(data.getData());
                Bitmap bitmap = BitmapFactory.decodeStream(inputStream);
                bitmap = getResizedBitmap(bitmap,400);
                byteArray = getBytes(bitmap);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
package de.epprojekt.ep_sjpp_prototyp.Menuebereich;
class: IbtnChangerActivity

/Öffnet ein Dialogfenster
//Ja: gewähltes Bild aus selectImage() speichern
//Nein: gewähltes Bild aus selectImage() nicht speichern
private void agreement(String string) {
    final CharSequence[] options = { "Ja", "Nein" };
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Neues Bild speichern?");
    builder.setItems(options, (dialog, item) -> {
        if (options[item].equals("Ja"))
        {
            hilfMirDaddyDB.setDrawableFromGallery(string, byteArray);
            Intent intent = new Intent(this, IbtnChangerActivity.class);
            startActivity(intent);
        }
        else if (options[item].equals("Nein")) {
            dialog.dismiss();
        }
    });
    builder.show();
}
```