Dokumentácia k projektu OOP

Meno: Filip Paučo

AIS ID: 116270

Zámer Projektu

Pre svoj projekt som si vybral anglický typ aukcie, podľa môjho názoru je to najvhodnejší typ pre môj výber témy. Na svojej aukcií sa výlučne zameriavam na obuv. Značkové alebo zberateľské.

V mojom projekte sa bude vyskytovať, niekoľko typov používateľov s odlišnými právomocami a zámermi. Konkrétnejšie to bude:

1. Administrátor (správca aukcie, bude mať plnú kontrolu nad ponukami.)

2. Predajca (človek, ktorý bude môcť ponúkať veci na predaj, stanoví aj počiatočnú sumu.)

3. Kupujúci (osoba, ktorá bude môcť prihadzovať na položky a kupovať položky. Bude mať len základné právomoci. Tento variant sa rozvetví na dva prípady – fyzická a právnická osoba, podľa možností a premyslenia, určím nejaké podmienky, výhody, odlišnosti medzi týmito dvoma skupinami.)

4.Návštevník (Človek s minimálnymi právomocami len na prezretie ponuky.)

Ďalšia hierarchia v podobe typov tenisiek je uvedená aj nižšie s fotodokumentáciou.

1. **Tenisky**
2. **Športové tenisky**
3. **Topánky (boots)**
4. **Športové tenisky vonkajšie**
5. **Športové tenisky vnútorné**

Ukončenie dražby som zanechal na manuálne zo strany predávajúceho. Na predídenie podvodov som implementoval funkcionality ako napr. ukončiť aukciu môže len predajca, ktorý ju začal, predajca nemôže prihadzovať na svoju aukciu, človek ktorý prihadzuje musí byť prihlásený...

1. O projekte

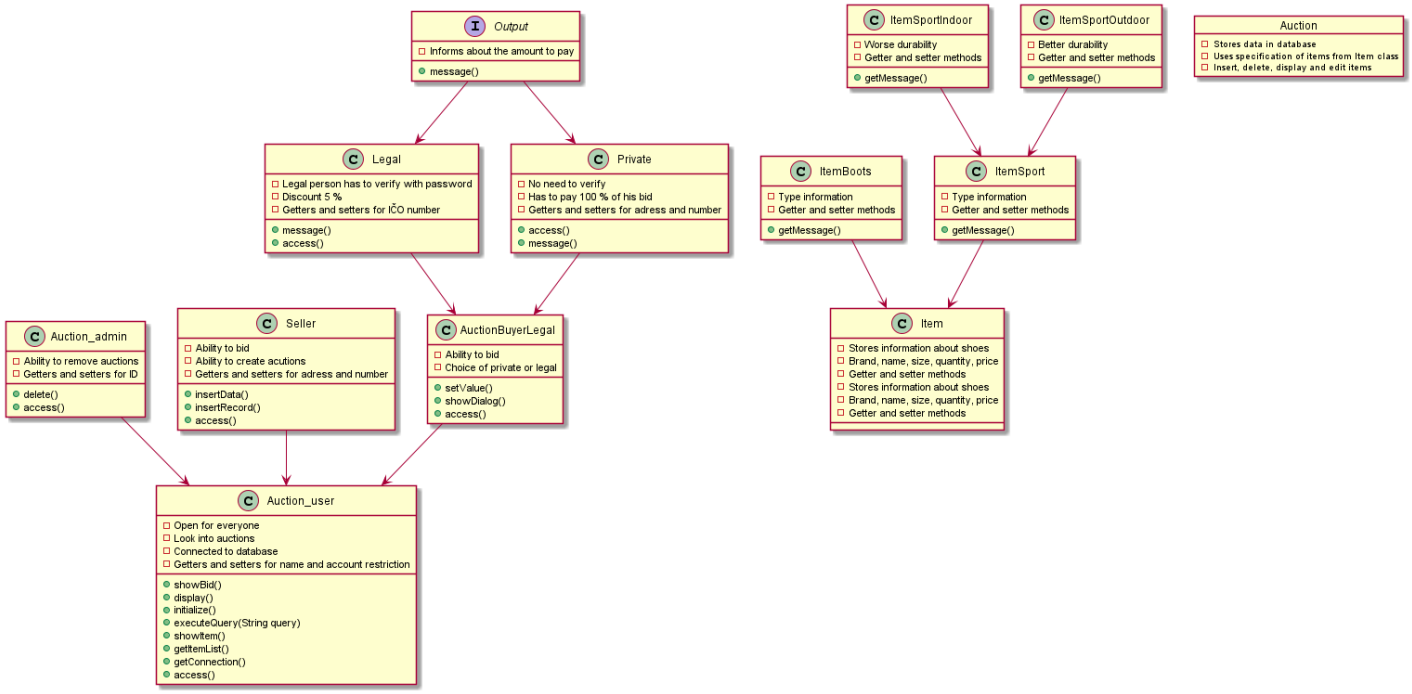
Svoj projekt som zameral na aukciu tenisiek rôznych typov s niekoľkými typmi ôsob alebo používateľov, ktorý môžu aukciu využívať. Samotný projekt je realizovaný pomocou Javy a komponentu JavaFX. Samotné uchovávanie a práca s dátami je realizovaná pomocou JDBC (databázy). V aukcii je mnoho funkcionalít a vychytáviek ako napr. Predajca aukcie nemôže prihadzovať na svoju aukciu, predajca môže ukončiť len svoju aukciu, ktorú spustil. Po ukončení aukcie sa vždy vypíše meno a typ používateľa, ktorý vyhral aukciu. Prihadzovanie na aukcie sa ukladá aj po ukončení aukcie, teda po opätovnom zapnutí programu, bude načítaný správny najvyšší príhoz. Právnická osoba má výhodu od ostatných zľavu 5 % po ukončení aukcie, pokiaľ ju vyhrá. Avšak pre získanie takejto zľavy, sa musí overiť unikátnym kódom, ktorý som vytvoril, aby sa to nezneužívalo. Potom je tam admin, ktorý má úplnú kontrolu a moc nad tým čo sa v aukciách deje, môže ich vymazať, pokiaľ by niečo nebolo správne.

Dodatočne som pridal možnosť vytvorenia účtu a možnosť prihlásenia ako akýkoľvek používateľ so zaregistrovanými údajmi.

Na GitHube mám niekoľko verzií, v podstate v každej som priebežne opravoval nejaké chyby, pridával funkcie a upravoval svoj projekt.

2. Hlavné kritériá

2.1 Hierarchia Tried



Admin, kupujúci a predávajúci dedia vlastnosti prezerania a otvárania aukcií. Kupujúci získa naviac možnosť kupovať a v ďalšej špecifikácií dedenia, cenu, právnická osoba má zľavu 5 %, inak sa platí 100 % sumy dražby.

V druhej hierarchií je predmet tenisky. Ktorý sa podobne rozvrstvuje na topánky, športové tenisky a tie na vonkajšie a vnútorné. Rozdiel medzi vonkajšími a vnútornými je v ich odolnosti. Pri vnútorných to je menej – 0.75, naopak pri vonkajších to je až 1.5.

2.2 Polymorfizmom vo forme prekonávania (overriding)

Funkcia upravuje výpis podľa toho či je užívateľ prihlásený ako fyzická osoba alebo klasický kupujúci (obr.1, trieda kupujúceho) alebo ako špecifický kupujúci – právnická osoba. Kedy má zľavu 5 % (obr. 2 trieda právnickej osoby). Na obrázku č. 3 je znázornené využitie tohto výpisu pri samotnom ukončení aukcie.

obr. 1

```
public  String message(int i){
    return "Auction ended with "+i+" €";
}
```

```
public class Legal extends AuctionBuyerPrivate {


    @Override
    public String message(int value) { return "Auction ended with "+value*0.95+" €"; }


}
```

obr. 2

```
public void endAuction(ActionEvent actionEvent) throws IOException, InterruptedException, RuntimeException {
    int k = Integer.parseInt(actual.getText());
    if(legal.isSelected()) {
        setError.setText("Auction successfully ended with " + k * 0.95 + " €");
        Legal l = new Legal();
        l.message(k);
        System.out.println(l.message(k));
        Output message = (num) -> "Auction ended with "+num*0.95+" €";
        System.out.println(message);
    }
```

obr. 3

2.2.1 Polymorfizmus v druhej hierarchií (predmetovej)

```java
public void getMessage() { System.out.println("We are classic type of shoes"); }
```

(Item)

```java
@Override
public void getMessage() { System.out.println("We are not classic type of shoes, We are boots !"); }
```

(ItemBoots)

```java
@Override
public void getMessage(){
    System.out.println("We are not classic type of shoes, We are boots with lower durability !");
    System.out.println(getdurablity());
}
```

(ItemSportIndoor)

```java
@Override
public double info(String material){
    double a = 1;
    if(material.equals("Leather")){
        System.out.println("Not suitable");
        a = 0;
        return a;
    }
    else if (material.equals("Knit")){
        System.out.println("Very useful");
        a = 2;
        return a;
    }
    else{
        System.out.println("Average usability");
        return a;
    }

}
```

```java
@Override
public double info(String material){
    double a = 1;
    if(material.equals("Flyknit")){
        System.out.println("Best performance type");
        a = 0;
        return a;
    }
    else if (material.equals("Lightknit")){
        System.out.println("Great performance type");
        a = 2;
        return a;
    }
    else{
        System.out.println("Average performance type");
        return a;
    }

}
```

2.3 Agregácia

Pre využitie a ukážku agregácie som využil objekt Size (veľkosť topánky), ktorá uchováva len jednu premennú a to je veľkosť.
Obrázok 6 – Objekt (tenisiek).
Obrázok 7 – Ukladá veľkosť
Obrázok 8 – využitie v praxi (napr. Načítavanie z aukcie)

```java
public class Item implements Serializable {
    private String brand;
    private String name;
    Size size;
    private double price;
    private String quantity;
    private int id;

    public Item(int id,String brand, String name, Size size,String quantity,double price) {
        this.brand = brand;
        this.name = name;
        this.size = size;
        this.quantity = quantity;
        this.price = price;
        this.id = id;
    }
```

Obr. 6

```java
public class Size {
    private int size;

    public Size(int size) { this.size = size; }

    public int getSize() { return size; }

    public void setSize(int size) { this.size = size; }
}
```

Obr.7

```java
public void display(){
    Size size = new Size(Integer.parseInt(sizeText.getText()));
    Item itemm = new Item(brandText.getText(),nameText.getText(),size,quantityText.getValue(),Double.parseDouble(priceText.getText()));

    itemm.setBrand(brandText.getText());
    itemm.setName(nameText.getText());
    itemm.setSize(size);
    itemm.setQuantity(quantityText.getValue());
    itemm.setPrice(Double.parseDouble(priceText.getText()));
```

Obr.8

2.3.1 Druhá agregácia

```java
public class BuyerPData extends UserData{
    private Adress adress;
    private String cislo;

    public BuyerPData(String meno, Adress adress, String cislo) {
        super(meno);
        this.adress = adress;
        this.cislo = cislo;
    }
```

```java
public class Adress {
    private String mesto;
    private int byt;
    private int psc;

    public Adress(String mesto, int byt, int psc) {
        this.mesto = mesto;
        this.byt = byt;
        this.psc = psc;
    }
```

```java
public void Submit(ActionEvent actionEvent) {
    try {
        Adress adress = new Adress(mesto.getText(),Integer.parseInt(byt.getText()),Integer.parseInt(psc.getText()));
        BuyerLData s = new BuyerLData(meno.getText(),adress,cislo.getText(),ico.getText());
        System.out.println("Buyer-Legal data has been added !");
```

2.4 Interface (1. obr – itemy, 2. obr. Výpis typu osoby, ktorý vyhrala aukciu)

```java
public interface PriceOutput {
    default void say(double price){
        if(price > 500)
            System.out.println("These shoes were really expensive !");
        else if(price > 150)
            System.out.println("These shoes were quite expensive !");
        else
            System.out.println("These shoes were at quite fine price!");
    }
}
```

```java
public interface Output {
     String say(String seller);

}
```

3. Ďalšie kritériá

3.1 Singleton

```
class Singleton {
    private static Singleton single_instance = null;
    public String s;
    private Singleton()
    {

        s = " Has has been added to seller's data ! ";
    }

    public static Singleton Singleton()
    {
        if (single_instance == null) {
            single_instance = new Singleton();
        }
        return single_instance;
    }
}
```

```
Singleton x = Singleton.Singleton();
Singleton y = Singleton.Singleton();
Singleton v = Singleton.Singleton();
Singleton w = Singleton.Singleton();
Singleton z = Singleton.Singleton();
y.s = (y.s).toLowerCase();
System.out.println("Name - "+meno.getText()+ x.s);
System.out.println("City - "+mesto.getText()+y.s);
System.out.println("House number - "+byt.getText()+ v.s);
System.out.println("ZIP - "+psc.getText()+w.s);
x.s = (x.s).toUpperCase();
System.out.println("Phone number - "+cislo.getText()+z.s);
```

3.2 Iterátor (návrhový vzor)

Tento návrhový vzor funguje ako klasický iterátor, čiže využitie pri cykloch.

```
public interface Iterator {
    boolean hasNext();
    Object next();
}
```

```
public interface Container {
    public Iterator createIterator();
}
```

3.3 Vlastný exception

```
public class NumberException extends Exception{
    public NumberException (String str)
    {
        super(str);
    }
}
```

```
private boolean validateSize(int size) throws NumberException{
    boolean i;
    if(size >= 50){
        throw new NumberException("Size too big !");
        // System.out.println("Hello world");
    }
    else
        return false;
}
```

3.4 RTTI

```
String a = selectedItem.getType();
Object selectedlitem = check(a);
```

```
System.out.println("Class of Object is : " + selectedlitem.getClass());
```

3.5 Lambda výrazy

```
Output message = (String seller) -> {return seller+", Thanks for using this auction !";};
System.out.println(message.say(seller.getText()));
```

(Output je interface)

3.6 Implementácia default metódy

```
Details ds = new Details();
ds.say(Double.parseDouble(actual.getText()));
```

```java
public interface PriceOutput {
    default void say(double price){
        if(price > 500)
            System.out.println("These shoes were really expensive !");
        else if(price > 150)
            System.out.println("These shoes were quite expensive !");
        else
            System.out.println("These shoes were at quite fine price!");
    }
}
```

3.7 Serializácia

```java
    Item itemm = new Item(typeText.getValue(), brandText.getText(), nameText.getText(), size, quantityText.getValue());
    //ItemSport s = new ItemSport("Sport");
    itemm.setType(typeText.getValue());
    itemm.setBrand(brandText.getText());
    itemm.setName(nameText.getText());
    itemm.setSizo(Integer.parseInt(sizeText.getText()));
    itemm.setQuantity(quantityText.getValue());
    itemm.setPrice(Double.parseDouble(priceText.getText()));
    itemm.setSeller(sellerChoice.getValue());

    try {
        FileOutputStream fileOut = new FileOutputStream( name: "ItemList.ser");
        ObjectOutputStream out = new ObjectOutputStream(fileOut);
        out.writeObject(itemm);
        out.close();
        fileOut.close();
       // System.out.println("Serialized data is saved in ItemList.ser");
    } catch (IOException i) {
        i.printStackTrace();
    }
}
```

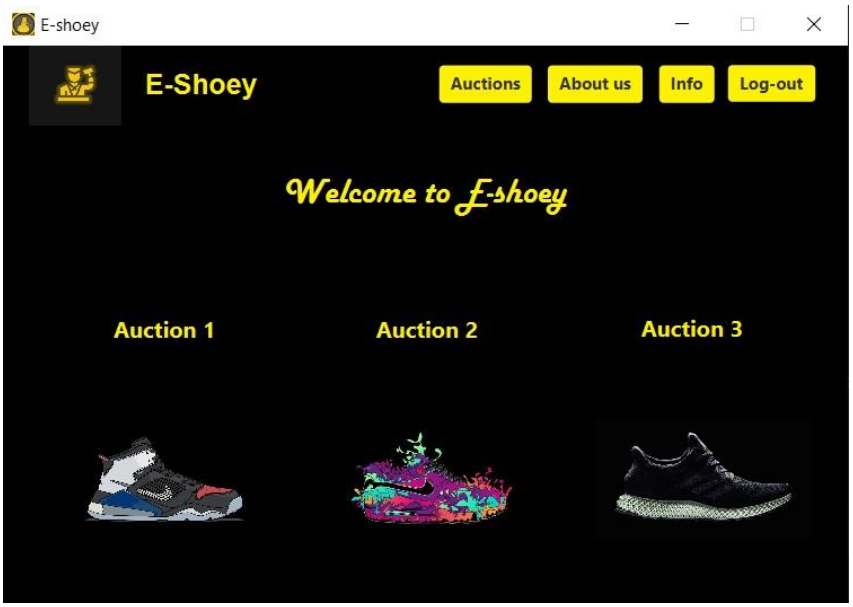
3.8 Použitie GUI

```
public void start(Stage stage) throws IOException {
    stg = stage;
    stage.setResizable(false);
    Parent root = FXMLLoader.load(getClass().getResource( name: "Login.fxml"));
    //FXMLLoader fxmlLoader = new FXMLLoader(HelloApplication.class.getResource("Login.fxml"));
    stage.setTitle("E-shoey");
    Image image = new Image(getClass().getResourceAsStream( name: "9769813.jpg"));
    stage.getIcons().add(image);
    stage.setScene(new Scene(root,  v: 600,  v1: 400));
    stage.show();
}

public void changeScene(String fxml) throws IOException{
    Parent pane = FXMLLoader.load(getClass().getResource(fxml));
    stg.getScene().setRoot(pane);
}
```

```
        FXMLLoader loader = new FXMLLoader();
        loader.setLocation(getClass().getResource( name: "Details.fxml"));
        Parent tableviewParent = loader.load();
        Scene tableViewScene = new Scene(tableviewParent);

        Details controller = loader.getController();
        controller.initData(tableView.getSelectionModel().getSelectedItem());
        controller.inData(sellerChoice.getValue(),typeText.getValue());
        Stage window = (Stage) ((Node) actionEvent.getSource()).getScene().getWindow();
        window.setScene(tableViewScene);
        window.show();
    }
```



3.9 Manuálne robené handlery v GUI

Bola potrebná implementácia interface – EventHandler<ActionEvent>

```
public void register(ActionEvent event) throws IOException{
    b = new Button();
    b.setText("Register");
    b.setOnAction(this);
    HBox sp = new HBox();
    //sp.getChildren().addAll(b);
    btn = new TextField();
    bto = new TextField();
    btn.setMaxSize( v: 80, v1: 30);
    btn.setPromptText("Name");
    bto.setPromptText("Password");
    bto.setMaxSize( v: 80, v1: 30);
    sp.getChildren().addAll(btn,bto,b);
    sp.setAlignment(Pos.CENTER);
    sp.setSpacing(25);
    Scene s = new Scene(sp, v: 600, v1: 400);
    HelloApplication m = new HelloApplication();
    m.changeScene2(s);

}
```

```
@Override
public void handle(ActionEvent actionEvent) {
    if(actionEvent.getSource()== b && !btn.getText().isBlank() && !bto.getText().isBlank()){
        try {
            PrintWriter out = new PrintWriter( fileName: "filename.txt");
            out.println(btn.getText());
            out.println(bto.getText());
            System.out.println("Data added !");
            out.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
    else
        System.out.println("Empty data !");
}
}
```