Wypożyczalnia samochodów. Projekt Java.

Autorzy: Sebastian Mikoś, Jan Zapotoczny, Amadeusz Nowak, Łukasz Babiś

1. Opis projektu

Wypożyczalnia samochodów to aplikacja która pozwala klientowi na wypożyczenie jednego z aut po uprzednim zalogowaniu się. W projekcie zastosowano technologię Java FXML (interfejs graficzny) oraz Hibernate (łączenie się z lokalną bazą danych).

2. Opis klas

App.java

Opis

Klasa App jest główną klasą aplikacji JavaFX odpowiedzialną za uruchamianie i inicjalizowanie interfejsu użytkownika. Rozszerza klasę Application, co umożliwia jej konfigurację i wyświetlenie okna aplikacji.

Zawartość Klasy

Deklaracja Klasy

public class App extends Application

Deklaracja klasy App, która rozszerza Application, co oznacza, że jest to główna klasa aplikacji JavaFX.

Pola Klasy

scene
private static Scene scene

Statyczne pole przechowujące bieżącą scenę aplikacji. Scena w JavaFX jest kontenerem dla wszystkich widżetów UI (np. przycisków, etykiet).

Metody Klasy

start(Stage stage)
@Override public void start(Stage stage) throws IOException

Metoda start jest punktem wejścia do aplikacji JavaFX. Ustawia początkową scenę, ikonę i tytuł okna aplikacji. Wywoływana jest automatycznie podczas uruchamiania aplikacji.

Parametry:

- Stage stage: główne okno aplikacji.

Wyjątki:

- IOException: może być rzucane podczas ładowania pliku FXML.

loadFXML(String fxml)
private static Parent loadFXML(String fxml) throws IOException

Prywatna metoda pomocnicza do ładowania pliku FXML i zwracania korzenia hierarchii sceny.

Parametry:

- String fxml: nazwa pliku FXML do załadowania (bez rozszerzenia).

Wyjątki:

- IOException: może być rzucane podczas ładowania pliku FXML.

Zwraca:

- Parent: korzeń hierarchii sceny.

main(String[] args)
public static void main(String[] args)

Główna metoda uruchamiająca aplikację JavaFX. Wywołuje metodę launch, która inicjalizuje aplikację.

Parametry:

- String[] args: argumenty wiersza poleceń.

PrimaryController.java

Opis pliku

Plik PrimaryController.java jest częścią projektu graficznej wypożyczalni samochodów połączonej z bazą danych. Plik ten zawiera kontroler dla aplikacji JavaFX, który zarządza interakcjami

użytkownika z interfejsem użytkownika oraz obsługuje logikę biznesową dotyczącą logowania, rejestracji oraz wypożyczania samochodów.

Struktura Klasy

Pola Klasy

- entityManagerFactory: Fabryka menedżerów encji, służąca do zarządzania sesjami połączeń z bazą danych.
- entityManager: Menedżer encji używany do operacji na bazie danych.
- attemptCount: Licznik nieudanych prób logowania.
- startTime: Czas rozpoczęcia blokady logowania po trzech nieudanych próbach.
- stage, scene, root: Elementy JavaFX używane do zmiany scen.
- carList: Lista dostępnych egzemplarzy samochodów.
- customersList: Lista klientów.

Pola Annotowane (JavaFX)

- myChoiceBox: ChoiceBox do wyboru samochodów.
- rentButton, userLoginButton: Przycisk do wypożyczenia samochodu oraz przycisk logowania.
- lb1, lb2, lb3, lb4, lb5, lb6, rentPrice, rentPriceNotification, incorrect, incorrectRegister, carWasRent, dateReturnCar: Różne etykiety i pola tekstowe wyświetlające informacje oraz komunikaty błędów.
- loginRegistrationField, loginField, nameField, surnameField: Pola tekstowe do wprowadzania danych logowania i rejestracji.
- loginPassword, registrationPassword, registrationRepeatPassword: Pola hasła do logowania i rejestracji.

Metody Publiczne

- LoginAttempt(ActionEvent event): Obsługuje próbę logowania użytkownika.
- switchToRegister(ActionEvent event): Przełącza scenę do okna rejestracji.
- switchToLogin(ActionEvent event): Przełącza scenę do okna logowania.
- registrationAttempt(ActionEvent event): Obsługuje próbę rejestracji nowego użytkownika.
- checkboxChange(ActionEvent event): Obsługuje zmianę wyboru w ChoiceBox.
- rentCar(ActionEvent event): Dodaje informacje o wypożyczeniu auta do bazy danych.

Metody Prywatne

- loginValidate(String login): Sprawdza poprawność loginu.
- passwordValidate(String password): Sprawdza poprawność hasła.
- nameAndSurnameValidate(String text): Sprawdza poprawność imienia i nazwiska.
- displayScene(String sceneName, ActionEvent event, String sceneTitle): Uniwersalna metoda zmiany sceny.
- setButtonsVisibility(boolean visibilityPrice, boolean visibilityCar): Ustawia widoczność elementów interfejsu.

Metoda initialize
Metoda initialize(URL url, ResourceBundle rb) jest wywoływana po załadowaniu FXML i odpowiada za wczytanie danych do ChoiceBox oraz ustawienie początkowych wartości w interfejsie użytkownika.

Klienci.java

Opis pliku
Plik Klienci.java definiuje encję Klienci, która jest mapowana na tabelę Klienci w bazie danych. Ta klasa jest częścią projektu graficznej wypożyczalni samochodów i reprezentuje klientów w systemie.

Struktura Klasy

Adnotacje Klasy
- @Entity: Informuje JPA, że klasa jest encją i będzie mapowana na tabelę w bazie danych.
- @Table(name = "Klienci"): Określa nazwę tabeli w bazie danych, do której ta encja jest mapowana.

Pola Klasy
- @Id: Adnotacja wskazująca, że pole login jest kluczem głównym w tabeli.
- @Column(name = "login", nullable = false): Adnotacja wskazująca, że pole login jest kolumną w tabeli Klienci i nie może być puste.
- @Column(name = "imie", nullable = false): Adnotacja wskazująca, że pole imie jest kolumną w tabeli Klienci i nie może być puste.
- @Column(name = "nazwisko", nullable = false): Adnotacja wskazująca, że pole nazwisko jest kolumną w tabeli Klienci i nie może być puste.
- @Column(name = "haslo", nullable = false): Adnotacja wskazująca, że pole haslo jest kolumną w tabeli Klienci i nie może być puste.

Pola Prywatne
- private String login: Przechowuje login klienta, który jest jednocześnie kluczem głównym w tabeli.
- private String imie: Przechowuje imię klienta.
- private String nazwisko: Przechowuje nazwisko klienta.
- private String haslo: Przechowuje hasło klienta.

Metody Publiczne

Gettery
- public String getLogin(): Zwraca login klienta.
- public String getImie(): Zwraca imię klienta.
- public String getNazwisko(): Zwraca nazwisko klienta.
- public String getHaslo(): Zwraca hasło klienta.

Settery

- public void setLogin(String login): Ustawia login klienta.
- public void setImie(String imie): Ustawia imię klienta.
- public void setNazwisko(String nazwisko): Ustawia nazwisko klienta.
- public void setHaslo(String haslo): Ustawia hasło klienta.

Egzemplarze.java

Opis Pliku

Plik Egzemplarze.java definiuje encję Egzemplarze, która jest mapowana na tabelę Egzemplarze w bazie danych. Ta klasa jest częścią projektu graficznej wypożyczalni samochodów i reprezentuje dostępne egzemplarze samochodów w systemie.

Struktura Klasy

Adnotacje Klasy

- @Entity: Informuje JPA, że klasa jest encją i będzie mapowana na tabelę w bazie danych.
- @Table(name = "Egzemplarze"): Określa nazwę tabeli w bazie danych, do której ta encja jest mapowana.

Pola Klasy

- @Column(name = "marka"): Adnotacja wskazująca, że pole marka jest kolumną w tabeli Egzemplarze.
- @Column(name = "model", nullable = false): Adnotacja wskazująca, że pole model jest kolumną w tabeli Egzemplarze i nie może być puste.
- @Id: Adnotacja wskazująca, że pole rejestracja jest kluczem głównym w tabeli.
- @Column(name = "rejestracja", nullable = false): Adnotacja wskazująca, że pole rejestracja jest kolumną w tabeli Egzemplarze i nie może być puste.
- @Column(name = "data_produkcji", nullable = false): Adnotacja wskazująca, że pole dataProdukcji jest kolumną w tabeli Egzemplarze i nie może być puste.
- @Column(name = "kolor", nullable = false): Adnotacja wskazująca, że pole kolor jest kolumną w tabeli Egzemplarze i nie może być puste.
- @Column(name = "przebieg", nullable = false): Adnotacja wskazująca, że pole przebieg jest kolumną w tabeli Egzemplarze i nie może być puste.
- @Column(name = "cena", nullable = false): Adnotacja wskazująca, że pole cena jest kolumną w tabeli Egzemplarze i nie może być puste.

Pola Prywatne

- private String marka: Przechowuje markę samochodu.
- private String model: Przechowuje model samochodu.
- private String rejestracja: Przechowuje numer rejestracyjny samochodu, który jest jednocześnie kluczem głównym w tabeli.
- private Date dataProdukcji: Przechowuje datę produkcji samochodu.

- private String kolor: Przechowuje kolor samochodu.
- private Integer przebieg: Przechowuje przebieg samochodu.
- private Double cena: Przechowuje cenę wypożyczenia samochodu.

Metody Publiczne

Gettery
- public String getRejestracja(): Zwraca numer rejestracyjny samochodu.
- public Date getDataProdukcji(): Zwraca datę produkcji samochodu.
- public String getMarka(): Zwraca markę samochodu.
- public String getModel(): Zwraca model samochodu.
- public String getKolor(): Zwraca kolor samochodu.
- public Integer getPrzebieg(): Zwraca przebieg samochodu.
- public Double getCena(): Zwraca cenę wypożyczenia samochodu.

Settery
- public void setRejestracja(String rejestracja): Ustawia numer rejestracyjny samochodu.
- public void setDataProdukcji(Date dataProdukcji): Ustawia datę produkcji samochodu.
- public void setMarka(String marka): Ustawia markę samochodu.
- public void setModel(String model): Ustawia model samochodu.
- public void setKolor(String kolor): Ustawia kolor samochodu.
- public void setPrzebieg(Integer przebieg): Ustawia przebieg samochodu.
- public void setCena(Double cena): Ustawia cenę wypożyczenia samochodu.

Wypozyczenie.java

Opis Pliku
Plik Wypozyczenie.java definiuje encję Wypozyczenie, która jest mapowana na tabelę Wypozyczenia w bazie danych. Ta klasa reprezentuje wypożyczenia samochodów dokonane przez klientów w systemie wypożyczalni samochodów.

Struktura Klasy

Adnotacje Klasy
- @Entity: Informuje JPA, że klasa jest encją i będzie mapowana na tabelę w bazie danych.
- @Table(name = "Wypozyczenia"): Określa nazwę tabeli w bazie danych, do której ta encja jest mapowana.

Pola Klasy
- @Id: Adnotacja wskazująca, że pole id jest kluczem głównym w tabeli.
- @GeneratedValue(strategy = GenerationType.IDENTITY): Określa strategię generowania wartości klucza głównego jako automatyczną inkrementację.
- @Column(name = "id"): Adnotacja wskazująca, że pole id jest kolumną w tabeli Wypozyczenia.

- @ManyToOne(fetch = FetchType.LAZY): Określa wiele-do-jednego relację między Wypozyczenie a Klienci.
- @JoinColumn(name = "login", nullable = false): Określa kolumnę w tabeli Wypozyczenia, która przechowuje klucz obcy do tabeli Klienci.
- @ManyToOne(fetch = FetchType.LAZY): Określa wiele-do-jednego relację między Wypozyczenie a Egzemplarze.
- @JoinColumn(name = "rejestracja", nullable = false): Określa kolumnę w tabeli Wypozyczenia, która przechowuje klucz obcy do tabeli Egzemplarze.
- @Column(name = "data_wypozyczenia", nullable = false): Adnotacja wskazująca, że pole dataWypozyczenia jest kolumną w tabeli Wypozyczenia i nie może być puste.
- @Column(name = "data_zwrotu", nullable = false): Adnotacja wskazująca, że pole dataZwrotu jest kolumną w tabeli Wypozyczenia i nie może być puste.

Pola Prywatne

- private Integer id: Przechowuje identyfikator wypożyczenia.
- private Klienci klient: Przechowuje obiekt Klienci reprezentujący klienta dokonującego wypożyczenia.
- private Egzemplarze egzemplarz: Przechowuje obiekt Egzemplarze reprezentujący wypożyczony samochód.
- private LocalDate dataWypozyczenia: Przechowuje datę wypożyczenia samochodu.
- private LocalDate dataZwrotu: Przechowuje datę zwrotu samochodu.

Metody Publiczne

Konstruktory

- public Wypozyczenie(Klienci klient, Egzemplarze egzemplarz, LocalDate dataWypozyczenia, LocalDate dataZwrotu): Konstruktor inicjujący wszystkie pola wypożyczenia.
- public Wypozyczenie(): Domyślny konstruktor bezargumentowy.

Gettery

- public Integer getId(): Zwraca identyfikator wypożyczenia.
- public Klienci getKlient(): Zwraca klienta dokonującego wypożyczenia.
- public Egzemplarze getEgzemplarz(): Zwraca wypożyczony samochód.
- public LocalDate getDataWypozyczenia(): Zwraca datę wypożyczenia.
- public LocalDate getDataZwrotu(): Zwraca datę zwrotu.

Settery

- public void setId(Integer id): Ustawia identyfikator wypożyczenia.
- public void setKlient(Klienci klient): Ustawia klienta dokonującego wypożyczenia.
- public void setEgzemplarz(Egzemplarze egzemplarz): Ustawia wypożyczony samochód.
- public void setDataWypozyczenia(LocalDate dataWypozyczenia): Ustawia datę wypożyczenia.
- public void setDataZwrotu(LocalDate dataZwrotu): Ustawia datę zwrotu.

module-info.java

Opis Pliku

Plik module-info.java definiuje moduł com.mycompany.project_java_3try w projekcie. Moduły wprowadzają enkapsulację na poziomie pakietów i określają zależności między różnymi modułami w aplikacji.

3. Opis bazy danych

Baza danych jest zaprojektowana do zarządzania wypożyczalnią pojazdów, z trzema głównymi tabelami: Egzemplarze (informacje o pojazdach), Klienci (dane klientów) oraz Wypozyczenia (szczegóły wypożyczeń).

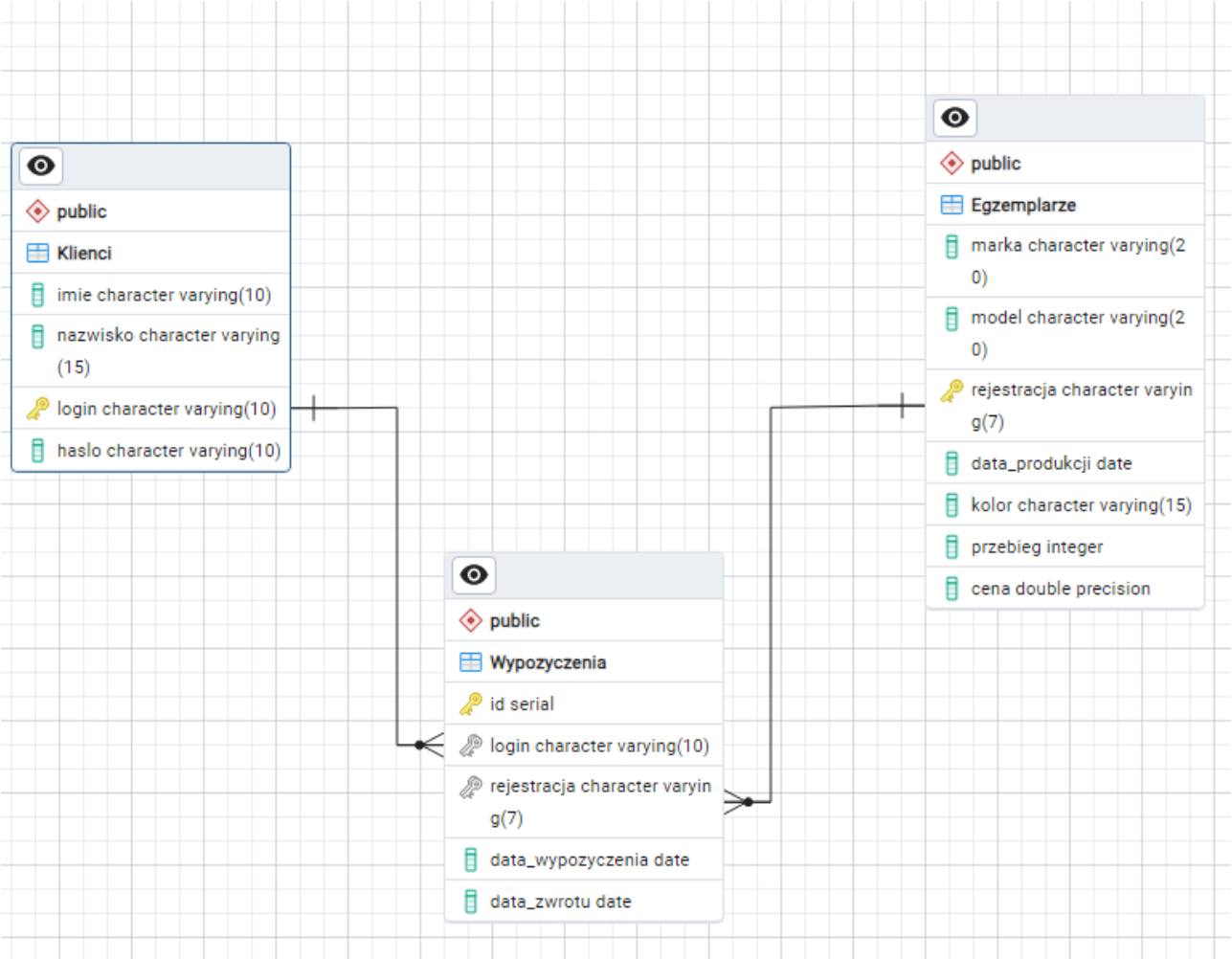
Tabela Klienci otrzymuje wpisy jeżeli zostanie dodany nowy użytkownik (klient wypożyczalni). Kluczem głównym jest login klienta.

Tabela Egzemplarze przechowuje informacje o konkretnym samochodzie który ma daną markę,rejestrację (identyfikator), model, datę produkcji pojazdu, kolor pojazdu, przebieg, oraz cena którą trzeba zapłacić za wypożyczenie konkretnego auta na 30 dni. Tabela

Wypożyczenia przechowuje dane jeżeli ktoś wypożyczy któryś z dostępnych egzemplarzy. Tabela zawiera: login (identyfikator klienta), rejestracja (identyfikator egzemplarza, pewnego auta), datę wypożyczenia, oraz datę zwrotu.

Połączenie do bazy danych: Połączenie oraz operowanie na bazie jest realizowane przez javax.persistence (javax.persistence-api) [2.2], org.hibernate (hibernate-core) [5.6.15.Final], org.hibernate (hibernate-entitymanager) [5.6.15.Final], oraz sterownik do połączenia z bazą danych: org.postgresql (postgresql) [42.2.20]

Informację o tym oraz konfigurację dependencies można znaleźć w pliku 'pom.xml' oraz 'nazwa_projektu\src\main\java\module-info.java' . Natomiast konfigurację połączenia z bazą danych można znaleźć w pliku: 'nazwa_projektu/src/main/resources/META-INF/persistence.xml'


public
Klienci
imie character varying(10)
nazwisko character varying (15)
login character varying(10)
haslo character varying(10)
public
Egzemplarze
marka character varying(20)
model character varying(20)
rejestracja character varying(7)
data_produkcji date
kolor character varying(15)
przebieg integer
cena double precision
public
Wypozyczenia
id serial
login character varying(10)
rejestracja character varying(7)
data_wypozyczenia date
data_zwrotu date