Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

Пояснительная записка

к курсовому проекту

по курсу «Программирование на языке Java»

на тему «Разработка многомодульного приложения на языке Java»

Выполнил:

Студент группы 21ВВП2

Чабуев Р.П.

Приняли:

к.т.н., доцент Юрова О.В.

к.т.н., доцент Карамышева Н.С.

Пенза 2024

**Содержание**

[Введение 5](#_Toc167697353)

[1 Постановка задачи 6](#_Toc167697354)

[2 Выбор решения 7](#_Toc167697355)

[3 Описание программы 8](#_Toc167697356)

[3.1 Серверная часть 8](#_Toc167697357)

[3.2 Клиентская часть 8](#_Toc167697358)

[4 Описание способа организации пользовательского интерфейса 10](#_Toc167697359)

[5 Описание результатов работы программы 11](#_Toc167697360)

[5.1 Запуск программы 11](#_Toc167697361)

[5.2 Изменение элементов таблицы 12](#_Toc167697362)

[5.3 Обработка исключений 17](#_Toc167697363)

[Заключение 18](#_Toc167697364)

[Список используемой литературы 19](#_Toc167697365)

[Приложение А. Исходный код 20](#_Toc167697366)

[Приложение А.1 - Исходный код сервера 20](#_Toc167697367)

[Приложение А.2 – Исходный код клиента 26](#_Toc167697368)

[Приложение B. UML-диаграммы приложения 63](#_Toc167697369)

[Приложение B.1 – UML-диаграмма вариантов использования 63](#_Toc167697370)

[Приложение B.2 – UML-диаграмма классов сервера 64](#_Toc167697371)

[Приложение B.3 – UML-диаграмма классов клиента 65](#_Toc167697372)

[Приложение B.4 – UML-диаграмма деятельности 66](#_Toc167697373)

[Приложение B.5 – UML-диаграмма развёртывания 67](#_Toc167697374)

[Приложение B.6 – UML-диаграмма последовательности 68](#_Toc167697375)

**Введение**

На сегодняшний момент язык Java является одним из самых распространенных и популярных языков программирования. Java является объектно-ориентированным языком. Он поддерживает полиморфизм, наследование, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. Стоит так же отметить, что приложения написанные на Java не являются платформ зависимыми. Платформ независимость Java опирается на существование виртуальной машины Java (JVM) на каждой поддерживаемой платформе. На разных платформах JVM разные. Но программы на Java могут запуститься на любой из них.

Представленная в данном курсовом проекте программа является простым клиент-серверным приложением, которое позволяет пользователю взаимодействовать с базой данных MySQL.

1. **Постановка задачи**

Основной задачей данного курсового проекта является закрепление знаний и приобретение практических навыков разработки клиент-серверных приложений на языке Java. Для этого необходимо выполнить программную реализацию модульного приложения-базы данных «Hotel» на языке Java в среде разработки NetBeans.

Программа должна работать так, чтобы пользователь мог редактировать базу данных и получать из нее информацию, не храня эти данные у себя на компьютере. В проекте предусмотрен графический интерфейс и используются следующие технологии:

1. Java Collections Framework;
2. Механизм обработки исключительных ситуаций;
3. Java Stream API;
4. Java Multithreading;
5. Сетевое взаимодействие.
6. **Выбор решения**

Для реализации проекта используются:

1. Объектно-ориентированный язык Java;
2. Библиотека Swing;
3. Пакеты, содержащие файлы ресурсов, пользовательский интерфейс;
4. Java Stream API;
5. IDE NetBeans 20;
6. Библиотека MySQL Connector.

Основной задачей курсового проекта являлась реализация взаимодействия клиента и сервера. Для выполнения данной задачи использовались сокеты.

Взаимодействие между клиентом и сервером должно происходить по принципу «Запрос-Ответ».

Интерфейс пользователя реализован с помощью библиотеки JavaSwing и содержит в себе стандартный функционал программы-менеджера отеля:

1. Просмотр элементов каждой таблицы;
2. Добавление элементов каждой таблицы;
3. Редактирование элементов каждой таблицы;
4. Удаление элементов каждой таблицы;
5. **Описание программы**

Программа, состоит из клиентской и серверной части. Сначала запускается сервер, который входит в состояние ожидания запроса клиента. Это можно увидеть в UML-диаграмме деятельности (приложение B.4).

После чего на устройствах запускается клиентская часть, которая по нажатию определенных кнопок, посылает определённые запросы к серверу.

* 1. **Серверная часть**

Server.java – главный файл серверного процесса, в котором находится точка входа в программу, функция Main.

После входа в программу сервер сообщает о своем запуске, открывает сокет и начинает прослушивание клиентов. После подключения клиента он производит вход в систему под определенным пользователем, данные пользователя MySQL указаны внутри кода Server.Java (рис.1), для подключения сервера к базе данных и создаёт отдельную нить для ожидания и исполнения запросов клиента.



Рисунок 1 – Подключение сервера к БД MySQL.

* 1. **Клиентская часть**

Client.java - главный файл пользовательского процесса, с которого начинается вход в приложение. Данный файл подключается к серверу и после чего отдает запросы ему, а также открывается графический интерфейс, с помощью которого он может взаимодействовать с базой данных через сервер. За это отвечают следующие файлы:

1. Postoyalec.java – интерфейс таблицы «Постоялец». С помощью него пользователь может:
   1. Добавить запись в таблицу. Для этого используется специальный метод – AddActionPerformed ();
   2. Изменить запись в таблице. Для этого используется специальный метод – SaveActionPerformed ();
   3. Удалить запись в таблице. Для этого используется специальный метод DeleteActionPerformed ();
   4. Выбрать другую таблицу;
2. Check\_in.java – интерфейс таблицы «Заселение». С помощью него пользователь может:
   1. Добавить запись в таблицу. Для этого используется специальный метод – AddActionPerformed ();
   2. Изменить запись в таблице. Для этого используется специальный метод – SaveActionPerformed ();
   3. Удалить запись в таблице. Для этого используется специальный метод DeleteActionPerformed ();
   4. Выбрать другую таблицу;
3. Reservation.java – интерфейс таблицы «Резервация». С помощью него пользователь может:
   1. Добавить запись в таблицу. Для этого используется специальный метод – AddActionPerformed ();
   2. Изменить запись в таблице. Для этого используется специальный метод – SaveActionPerformed ();
   3. Удалить запись в таблице. Для этого используется специальный метод DeleteActionPerformed ();
   4. Выбрать другую таблицу;
4. Apartament.java – интерфейс таблицы «Апартаменты». С помощью него пользователь может:
   1. Добавить запись в таблицу. Для этого используется специальный метод – AddActionPerformed ();
   2. Изменить запись в таблице. Для этого используется специальный метод – SaveActionPerformed ();
   3. Удалить запись в таблице. Для этого используется специальный метод DeleteActionPerformed ();
   4. Выбрать другую таблицу;
5. **Описание способа организации пользовательского интерфейса**

В качестве среды разработки была выбрана программа NetBeans. Программа предоставляет все средства, необходимые при разработке графического приложения.

Для реализации пользовательского интерфейса была использована библиотека Swing. Эта библиотека содержит более богатый и удобный набор элементов пользовательского интерфейса, обеспечивает одинаковое восприятие конечными пользователями приложений на разных платформах.

Для ввода данных были использованы текстовые поля jTextField.

Кнопки «Добавить», «Изменить», «Удалить» и выбора таблицы были созданы с помощью JButton.

Для вывода таблицы на экран была использована структура JTable.

1. **Описание результатов работы программы**

Среда разработки NetBeans предоставляет все средства, необходимы при разработке и отладке многомодульной программы.

Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных с вводом данных, изменением дизайна клиентской части, алгоритмом программы.

Ниже продемонстрирован результат тестирования функционала программы.

* 1. **Запуск программы**

При запуске программы пользователя приветствует окно Постояльцы, которое изображено на рисунке 2.

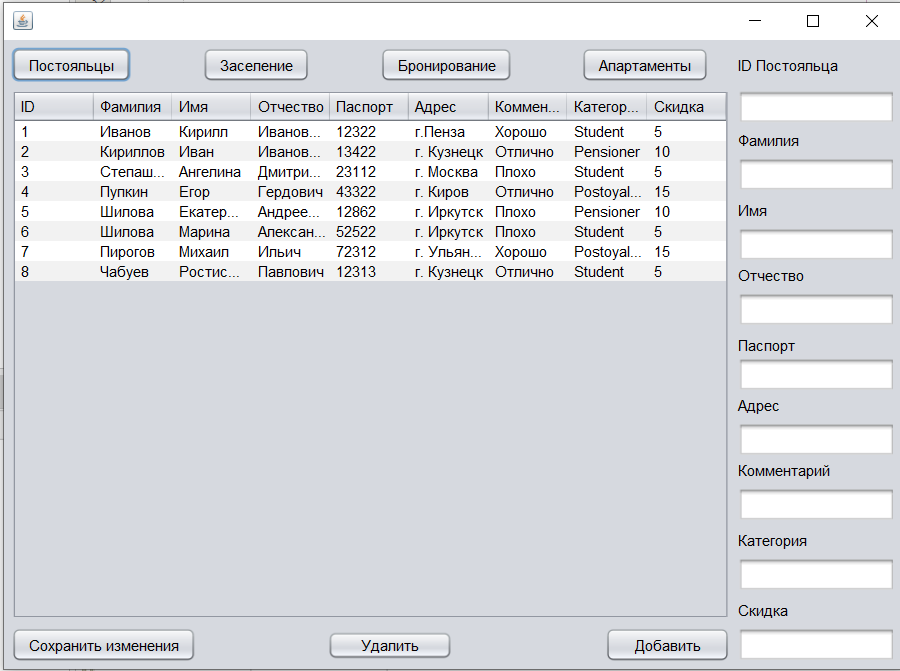


Рисунок 2 – Окно Постояльцы

Если же подключение к базе данных произведено успешно, и сервер корректно отработал запрос, то главная таблица «Постояльцы» заполняется данными, как на рисунке 2.

**Перед пользователем видна сама таблица и кнопки, с помощью которых можно её редактировать. Помимо кнопок для редактирования таблицы, есть кнопки для перехода между таблицами, которые названы в соответствии с названием каждой таблицы.**

* 1. **Изменение элементов таблицы**

Для того, чтобы начать взаимодействовать с элементами таблицы, используются кнопки «Добавить», «Сохранить изменения», «Удалить». Описание взаимодействия начнём с кнопки «Добавить». Справа в окне находятся поля для заполнения данными в таблицу, при полном заполнении нужно нажать на кнопку добавить, тогда произведется добавление пользовательских данных в таблицу, что, собственно, добавится в таблицу MySQL (рис. 3).

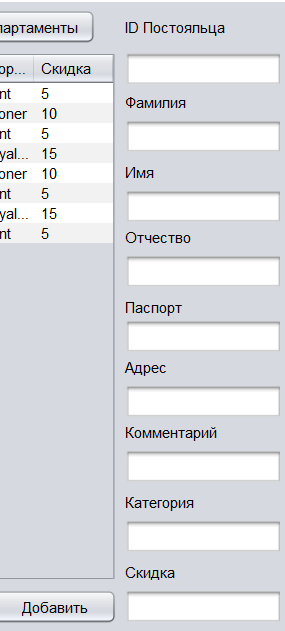


Рисунок 3 – Добавление постояльца

На рисунках 4 и 5 видно, что при добавлении постояльца могут возникнуть исключения, которые предотвращают неправильное формирование записи.

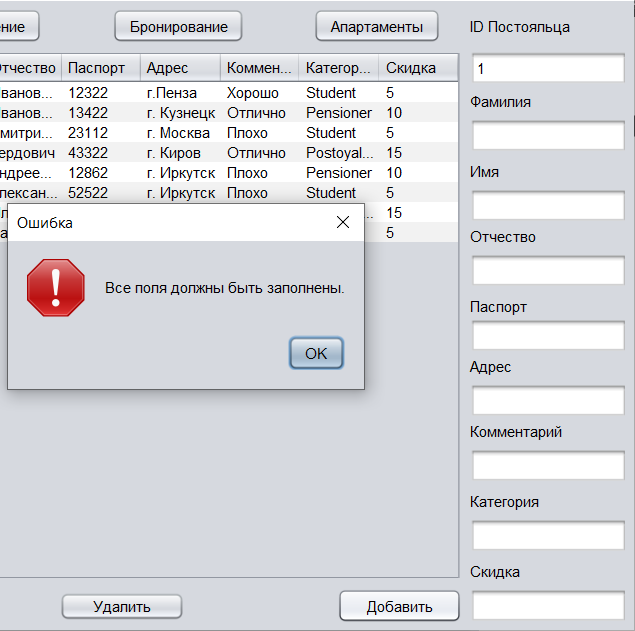


Рисунок 4 – Исключение при пустом поле

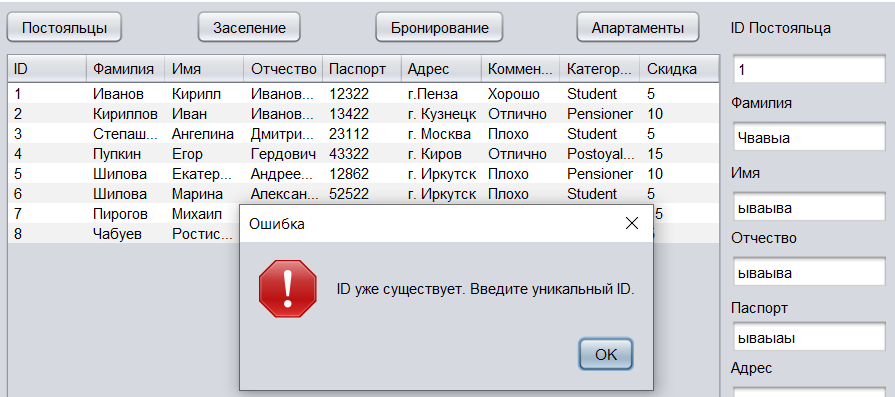


Рисунок 5 – Исключение при неправильном вводе ID Постояльца

Для того, чтобы внести изменения, нужно двойным кликом выбрать интересующую нас запись и заменить информацию, после чего нажать на кнопку «Сохранить изменения» (рис. 6).

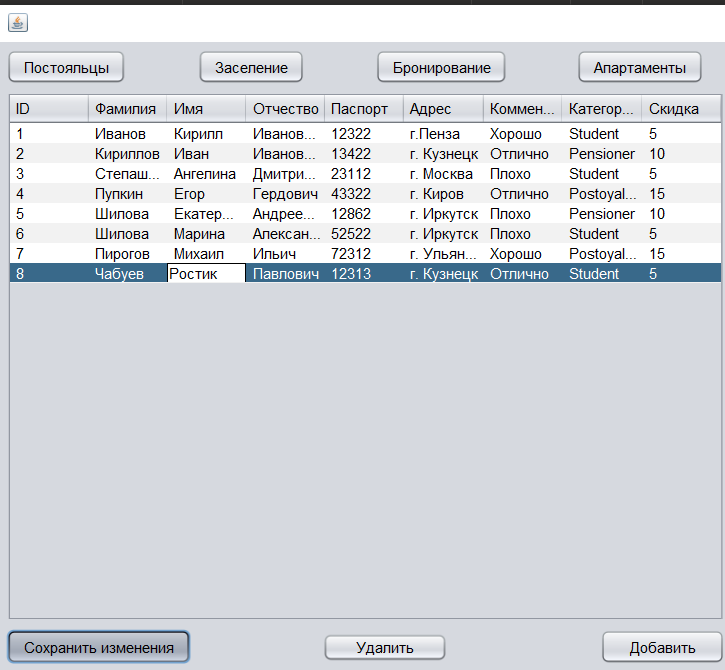


Рисунок 6 – Кнопка изменения записи

Аналогично кнопке «Добавить», у кнопки «Сохранить изменения» есть точно такие же исключения.

Для удаления записи используются кнопка «Удалить». Для её использования нужно выбрать записи и нажать её. На рисунках 7 и 8 показано удаление записи.

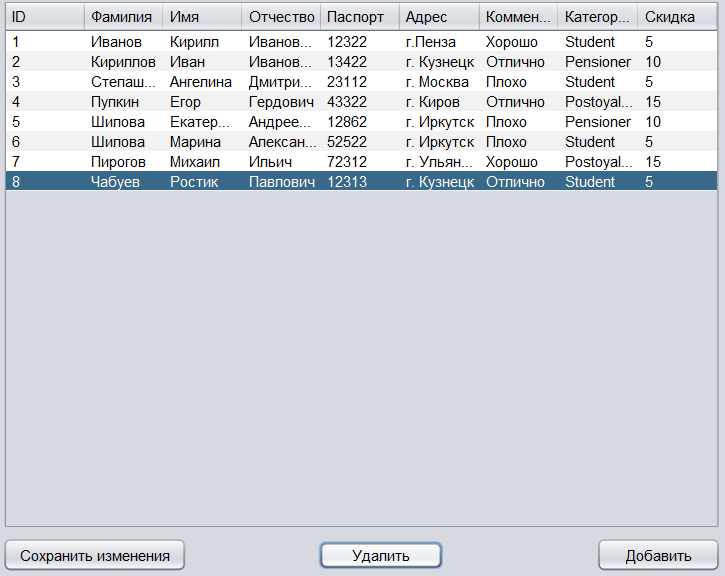


Рисунок 7 – Удаление записи

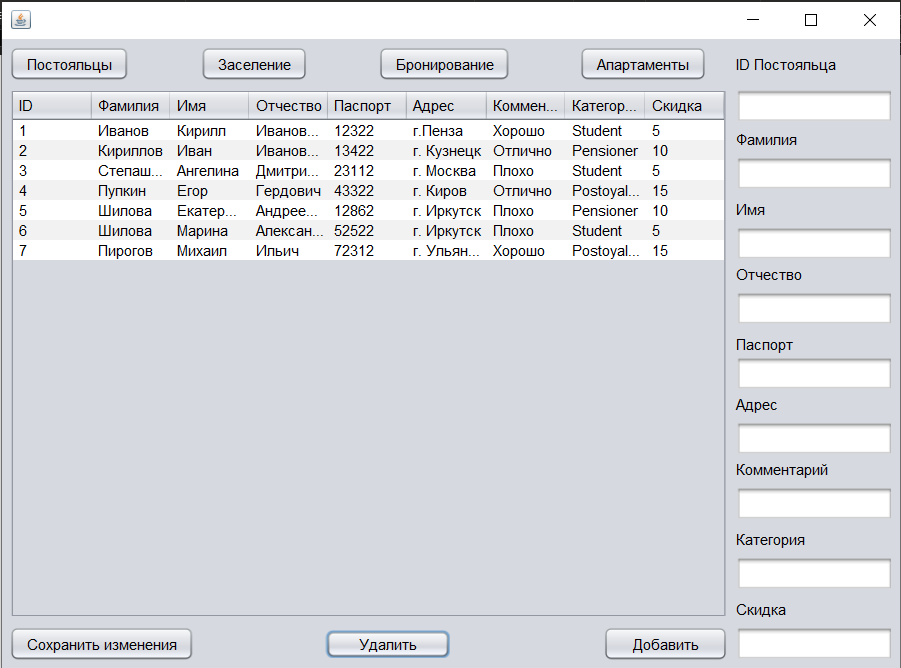


Рисунок 8 – Подтверждение удаления

По любой из 4 кнопок «Постояльцы», «Заселение», «Бронирование», «Апартаменты» можно переходить, и тогда будет открываться интерфейс таблиц (рис. 9)

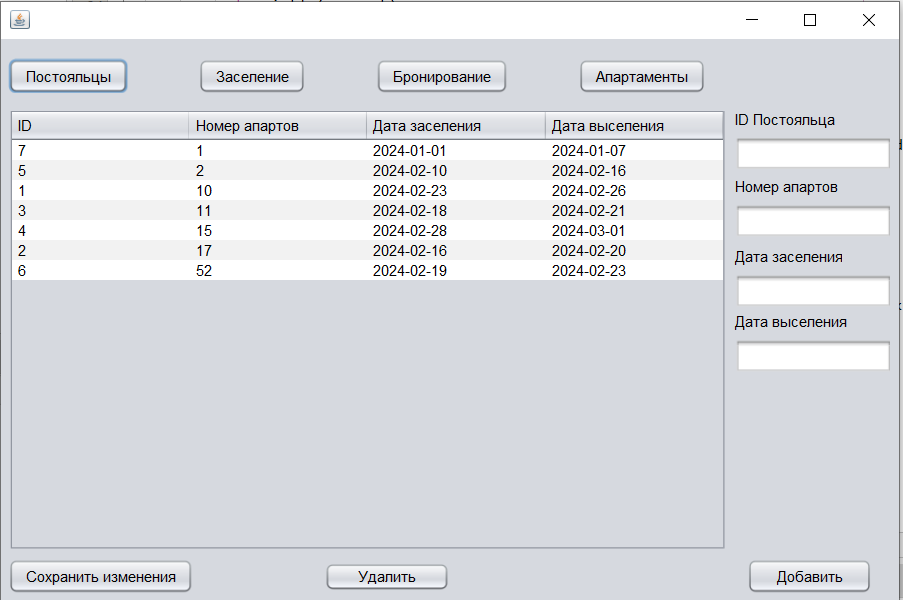


Рисунок 9 – Интерфейс таблицы Заселение

* 1. **Обработка исключений**

Помимо вышесказанных исключений, в других таблицах есть свои обработки. **Например, при добавлении или изменении записи в таблицах "Заселение" и "Резервация" выполняется обработка исключения, если дата выезда меньше даты въезда (рис. 10).**

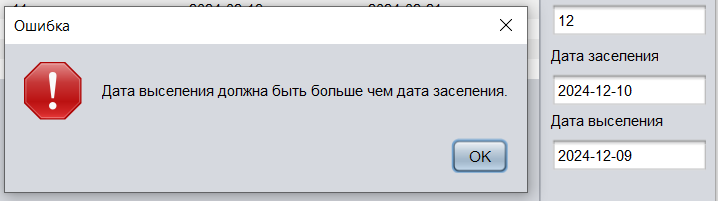


Рисунок 10 – Ошибка корректности ввода даты

**Заключение**

При выполнении данной курсовой работы были получены навыки разработки программ на языке высокого уровня Java. Были освоены навыки разработки клиент серверных приложений на сокетах и графического интерфейса с помощью библиотеки Swing. Изучены возможности среды разработки NetBeans.

В рамках данной работы была разработана программа-менеджер отеля, обладающая базовым функционалом, необходимым любому подобному приложению в наше время.

**Список используемой литературы**

1. Университет Информационных Технологий (ИНТУИТ), 2016.— 285с.
2. Портянкин, Java Swing: Эффектные пользовательские интерфейсы - Издание второе, 2011, 200 с.
3. Package javax.swing // docs.oracle.com : [сайт]. – 2023. – URL: https://docs.oracle.com/javase/8/docs/api/javax/swing/package-summary.html (дата обращения: 01.04.2023).
4. Шилдт, Г. Swing. Руководство для начинающих. – М.: Вильямс, 2007. – 704 с.: ил.
5. Берд, Барри Java для чайников / Барри Берд. - М.: Диалектика / Вильямс, 2013. - 521 c.
6. Дубаков А.А. Сетевое программирование: учебное пособие / А.А. Дубаков – СП: НИУ ИТМО, 2013. – 248 с.

**Приложение А. Исходный код**

**Приложение А.1 - Исходный код сервера**

Файл Server.java

package server;

import java.io.\*;

import java.net.\*;

import java.sql.\*;

import java.util.LinkedList;

public class Server {

public static void main(String[] args) throws SQLException, ClassNotFoundException {

try {

ServerSocket serverSocket = new ServerSocket(12345);

System.out.println("Server connected!");

try {

Socket clientSocket = serverSocket.accept();

System.out.println("The client connected to the server!");

BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);

ObjectOutputStream oos = new ObjectOutputStream(clientSocket.getOutputStream());

ObjectInputStream ois = new ObjectInputStream(clientSocket.getInputStream());

while (true){

String request = in.readLine();

System.out.println("Customer request: " + request); // Отладочное сообщение

if ("selectPost".equals(request)) {

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM Postoyalec");

LinkedList<String> answer = new LinkedList<>();

while (rs.next()) {

answer.add(rs.getString("ID\_Postoyalca"));

answer.add(rs.getString("Surname"));

answer.add(rs.getString("Name"));

answer.add(rs.getString("Patronymic"));

answer.add(rs.getString("Passport"));

answer.add(rs.getString("Address"));

answer.add(rs.getString("Comment"));

answer.add(rs.getString("Category"));

answer.add(rs.getString("Discount"));

}

oos.writeObject(answer);

oos.flush();

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

if ("addPost".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

LinkedList<String> answer = new LinkedList<String>();

answer = (LinkedList<String>) ois.readObject();

String sql = "INSERT INTO Postoyalec VALUES("+answer.get(0)+",'"

+answer.get(1)+"','"

+answer.get(2)+"','"

+answer.get(3)+"','"

+answer.get(4)+"','"

+answer.get(5)+"','"

+answer.get(6)+"','"

+answer.get(7)+"','"

+answer.get(8)+"');";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

}

if ("delPost".equals(request)){

try {

String id = in.readLine();

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Postoyalec WHERE ID\_Postoyalca=" + id + ";";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

if("updatePost".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

LinkedList<String> answer = new LinkedList<>();

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Postoyalec;";

stmt.executeUpdate(sql);

answer = (LinkedList<String>) ois.readObject();

for (int i = 0; i < answer.size(); i = i + 9) {

String sqlupd = "INSERT INTO Postoyalec VALUES("+answer.get(i)+",'"

+answer.get(i+1)+"','"

+answer.get(i+2)+"','"

+answer.get(i+3)+"','"

+answer.get(i+4)+"','"

+answer.get(i+5)+"','"

+answer.get(i+6)+"','"

+answer.get(i+7)+"','"

+answer.get(i+8)+"');";

stmt.executeUpdate(sqlupd);

}

stmt.close();

conn.close();

}

if ("selectCheck".equals(request)) {

try {Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM Check\_in");

LinkedList<String> answer = new LinkedList<>();

while (rs.next()) {

answer.add(rs.getString("ID\_PostoyalcaC"));

answer.add(rs.getString("Number\_apartC"));

answer.add(rs.getString("Date Check\_in"));

answer.add(rs.getString("Date Check\_out"));

}

oos.writeObject(answer);

oos.flush();

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

if ("addCheck".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

LinkedList<String> answer = new LinkedList<String>();

answer = (LinkedList<String>) ois.readObject();

String sql = "INSERT INTO Check\_in (ID\_PostoyalcaC, Number\_apartC, `Date Check\_in`,`Date Check\_out`) VALUES("+answer.get(0)+",'"

+answer.get(1)+"','"

+answer.get(2)+"','"

+answer.get(3)+"');";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

}

if ("delCheck".equals(request)){

try {

String id = in.readLine();

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Check\_in WHERE ID\_PostoyalcaC=" + id + ";";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

if("updateCheck".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

LinkedList<String> answer = new LinkedList<>();

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Check\_in;";

stmt.executeUpdate(sql);

answer = (LinkedList<String>) ois.readObject();

for (int i = 0; i < answer.size(); i = i + 4) {

String sqlupd = "INSERT INTO Check\_in VALUES("+answer.get(i)+",'"

+answer.get(i+1)+"','"

+answer.get(i+2)+"','"

+answer.get(i+3)+"');";

stmt.executeUpdate(sqlupd);

}

stmt.close();

conn.close();

}

if ("selectReserv".equals(request)) {

try {Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM Reservation");

LinkedList<String> answer = new LinkedList<>();

while (rs.next()) {

answer.add(rs.getString("ID\_PostoyalcaR"));

answer.add(rs.getString("Number\_apartR"));

answer.add(rs.getString("Booking start date"));

answer.add(rs.getString("Booking finish date"));

}

oos.writeObject(answer);

oos.flush();

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

if ("addReserv".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

LinkedList<String> answer = new LinkedList<String>();

answer = (LinkedList<String>) ois.readObject();

String sql = "INSERT INTO Reservation (ID\_PostoyalcaR, Number\_apartR, `Booking start date`,`Booking finish date`) VALUES("+answer.get(0)+",'"

+answer.get(1)+"','"

+answer.get(2)+"','"

+answer.get(3)+"');";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

}

if ("delReserv".equals(request)){

try {

String id = in.readLine();

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Reservation WHERE ID\_PostoyalcaR=" + id + ";";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

if("updateReserv".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

LinkedList<String> answer = new LinkedList<>();

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Reservation;";

stmt.executeUpdate(sql);

answer = (LinkedList<String>) ois.readObject();

for (int i = 0; i < answer.size(); i = i + 4) {

String sqlupd = "INSERT INTO Reservation VALUES("+answer.get(i)+",'"

+answer.get(i+1)+"','"

+answer.get(i+2)+"','"

+answer.get(i+3)+"');";

stmt.executeUpdate(sqlupd);

}

stmt.close();

conn.close();

}

if ("selectApart".equals(request)) {

try {Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM Apart");

LinkedList<String> answer = new LinkedList<>();

while (rs.next()) {

answer.add(rs.getString("Number\_apart"));

answer.add(rs.getString("Capacity"));

answer.add(rs.getString("Comfort"));

answer.add(rs.getString("Price"));

answer.add(rs.getString("Free"));

}

oos.writeObject(answer);

oos.flush();

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

if ("addApart".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

LinkedList<String> answer = new LinkedList<String>();

answer = (LinkedList<String>) ois.readObject();

String sql = "INSERT INTO Apart (Number\_apart, Capacity, Comfort, Price, Free) VALUES("+answer.get(0)+",'"

+answer.get(1)+"','"

+answer.get(2)+"','"

+answer.get(3)+"','"

+answer.get(4)+"');";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

}

if ("delApart".equals(request)){

try {

String id = in.readLine();

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Apart WHERE Number\_apart=" + id + ";";

stmt.executeUpdate(sql);

stmt.close();

conn.close();

} catch (SQLException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

if("updateApart".equals(request)){

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel", "root", "root");

LinkedList<String> answer = new LinkedList<>();

Statement stmt = conn.createStatement();

String sql = "DELETE FROM Apart;";

stmt.executeUpdate(sql);

answer = (LinkedList<String>) ois.readObject();

for (int i = 0; i < answer.size(); i = i + 5) {

String sqlupd = "INSERT INTO Apart VALUES("+answer.get(i)+",'"

+answer.get(i+1)+"','"

+answer.get(i+2)+"','"

+answer.get(i+3)+"','"

+answer.get(i+4)+"');";

stmt.executeUpdate(sqlupd);

}

stmt.close();

conn.close();

}

if ("END".equals(request)) break;

}

clientSocket.close();

} catch (IOException e) {

e.printStackTrace();

}

serverSocket.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Приложение А.2 – Исходный код клиента**

Файл Client.java

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.io.PrintWriter;

import java.net.Socket;

import java.util.LinkedList;

public final class Client {

public Socket socket;

public PrintWriter out;

public BufferedReader in;

public ObjectInputStream ois;

public ObjectOutputStream oos;

Client() throws IOException, ClassNotFoundException {

socket = new Socket("localhost", 12345);

out = new PrintWriter(socket.getOutputStream(), true); // true для автоочистки

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

ois = new ObjectInputStream(socket.getInputStream());

oos = new ObjectOutputStream(socket.getOutputStream());

}

public LinkedList<String> refresh(String request) throws IOException, ClassNotFoundException, InterruptedException {

out.println(request);

out.flush(); // Добавлено для уверенности в отправке

Thread.sleep(500);

LinkedList<String> answer = new LinkedList<>();

answer = (LinkedList<String>) ois.readObject();

return answer;

}

public void add(String request, LinkedList<String> answer) throws IOException, InterruptedException{

out.println(request);

out.flush();

Thread.sleep(500);

oos.writeObject(answer);

oos.flush();

}

public void update(String request, LinkedList<String> answer) throws InterruptedException, IOException{

out.println(request);

out.flush();

Thread.sleep(500);

oos.writeObject(answer);

oos.flush();

}

public void del(String request, String answer) throws IOException, InterruptedException{

out.println(request);

out.flush();

Thread.sleep(500);

out.println(answer);

out.flush();

}

public void end(){

out.println("END");

out.flush();

}

public static void main(String[] args) throws ClassNotFoundException {

try {

Client client = new Client();

} catch (IOException e) {

e.printStackTrace();

}

}

}

Файл Postoyalec.java

import javax.swing.table.DefaultTableModel;

import java.io.IOException;

import java.util.LinkedList;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

/\*\*

\*

\* @author RCh

\*/

public class Postoyalec extends javax.swing.JFrame {

Client c;

public Postoyalec(Client c){

this.c = c;

initComponents();

}

// Конструктор класса

public Postoyalec() throws IOException, ClassNotFoundException {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

Post = new javax.swing.JButton();

Check\_in = new javax.swing.JButton();

Reservation = new javax.swing.JButton();

Apart = new javax.swing.JButton();

Save = new javax.swing.JButton();

Delete = new javax.swing.JButton();

Add = new javax.swing.JButton();

jScrollPane2 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

ID = new javax.swing.JTextField();

Surname = new javax.swing.JTextField();

Name = new javax.swing.JTextField();

Patronymic = new javax.swing.JTextField();

Passport = new javax.swing.JTextField();

Address = new javax.swing.JTextField();

Comment = new javax.swing.JTextField();

Category = new javax.swing.JTextField();

Discount = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel6 = new javax.swing.JLabel();

jLabel7 = new javax.swing.JLabel();

jLabel8 = new javax.swing.JLabel();

jLabel9 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent evt) {

formWindowClosing(evt);

}

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

Post.setText("Постояльцы");

Post.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

PostActionPerformed(evt);

}

});

Check\_in.setText("Заселение");

Check\_in.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

Check\_inActionPerformed(evt);

}

});

Reservation.setText("Бронирование");

Reservation.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ReservationActionPerformed(evt);

}

});

Apart.setText("Апартаменты");

Apart.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ApartActionPerformed(evt);

}

});

Save.setText("Сохранить изменения");

Save.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SaveActionPerformed(evt);

}

});

Delete.setText("Удалить");

Delete.setPreferredSize(new java.awt.Dimension(84, 23));

Delete.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

DeleteActionPerformed(evt);

}

});

Add.setText("Добавить");

Add.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

AddActionPerformed(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"ID", "Фамилия", "Имя", "Отчество", "Паспорт", "Адрес", "Комментарий", "Категория", "Скидка"

}

) {

boolean[] canEdit = new boolean [] {

false, true, true, true, true, true, true, true, true

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jScrollPane2.setViewportView(jTable1);

if (jTable1.getColumnModel().getColumnCount() > 0) {

jTable1.getColumnModel().getColumn(0).setResizable(false);

jTable1.getColumnModel().getColumn(1).setResizable(false);

jTable1.getColumnModel().getColumn(2).setResizable(false);

jTable1.getColumnModel().getColumn(3).setResizable(false);

jTable1.getColumnModel().getColumn(4).setResizable(false);

jTable1.getColumnModel().getColumn(5).setResizable(false);

jTable1.getColumnModel().getColumn(6).setResizable(false);

jTable1.getColumnModel().getColumn(7).setResizable(false);

jTable1.getColumnModel().getColumn(8).setResizable(false);

}

ID.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

IDActionPerformed(evt);

}

});

jLabel1.setText("ID Постояльца");

jLabel3.setText("Фамилия");

jLabel4.setText("Имя");

jLabel5.setText("Отчество");

jLabel2.setText("Паспорт");

jLabel6.setText("Адрес");

jLabel7.setText("Комментарий");

jLabel8.setText("Категория");

jLabel9.setText("Скидка");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(Post)

.addGap(57, 57, 57)

.addComponent(Check\_in)

.addGap(56, 56, 56)

.addComponent(Reservation)

.addGap(55, 55, 55)

.addComponent(Apart))

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 575, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addComponent(Save)

.addGap(105, 105, 105)

.addComponent(Delete, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Add, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(ID)

.addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Surname)

.addComponent(Name)

.addComponent(Patronymic)

.addComponent(Passport)

.addComponent(Comment)

.addComponent(Category)

.addComponent(Discount)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 37, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED\_SIZE, 61, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED\_SIZE, 37, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 62, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel7)

.addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED\_SIZE, 80, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(0, 46, Short.MAX\_VALUE))

.addComponent(Address))

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(0, 0, Short.MAX\_VALUE))))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(6, 6, 6)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(Apart)

.addComponent(Reservation)

.addComponent(Check\_in)

.addComponent(Post))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(ID, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(4, 4, 4)

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Surname, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Name, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Patronymic, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel2)

.addGap(2, 2, 2)

.addComponent(Passport, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Address, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(jLabel7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Comment, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Category, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel9))

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Discount, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(Save)

.addComponent(Delete, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(Add))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void ReservationActionPerformed(java.awt.event.ActionEvent evt) {

new Reservation(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void Check\_inActionPerformed(java.awt.event.ActionEvent evt) {

new Check\_in(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void ApartActionPerformed(java.awt.event.ActionEvent evt) {

new Apart(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {

try {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

LinkedList<String> answer = new LinkedList<>();

int row = jTable1.getSelectedRow();

String s = (String) model.getValueAt(row, 0);

String request = "delPost";

c.del(request,s);

request = "selectPost";

answer = c.refresh(request);

refreshPost(answer);

} catch (IOException | InterruptedException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void formWindowClosing(java.awt.event.WindowEvent evt) {

c.end();

}

void refreshPost(LinkedList<String> answer){

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

int rows = model.getRowCount();

while (rows != 0) {

model.removeRow(0);

rows--;

}

Object[] row;

for (int i = 0; i < answer.size(); i = i+9){

row = new Object[9];

row[0] = answer.get(i);

row[1] = answer.get(i+1);

row[2] = answer.get(i+2);

row[3] = answer.get(i+3);

row[4] = answer.get(i+4);

row[5] = answer.get(i+5);

row[6] = answer.get(i+6);

row[7] = answer.get(i+7);

row[8] = answer.get(i+8);

model.addRow(row);

}

}

private void PostActionPerformed(java.awt.event.ActionEvent evt) {

}

private void IDActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void SaveActionPerformed(java.awt.event.ActionEvent evt) {

try {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

LinkedList<String> answer = new LinkedList<>();

int rows = jTable1.getRowCount();

for (int i = 0; i < rows; i++) {

for (int j = 0; j < 9; j++) {

answer.add((String) model.getValueAt(i, j));

}

}

String request = "updatePost";

c.update(request, answer);

request = "selectPost";

answer = c.refresh(request);

refreshPost(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void AddActionPerformed(java.awt.event.ActionEvent evt) {

String newID = ID.getText();

String surname = Surname.getText();

String name = Name.getText();

String patronymic = Patronymic.getText();

String passport = Passport.getText();

String address = Address.getText();

String comment = Comment.getText();

String category = Category.getText();

String discountStr = Discount.getText();

// Проверка на пустые поля

if (newID.isEmpty() || surname.isEmpty() || name.isEmpty() || patronymic.isEmpty() || passport.isEmpty() || address.isEmpty() || comment.isEmpty() || category.isEmpty() || discountStr.isEmpty()) {

javax.swing.JOptionPane.showMessageDialog(this, "Все поля должны быть заполнены.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

boolean idExists = false;

// Проверка на существующий ID

for (int i = 0; i < model.getRowCount(); i++) {

if (model.getValueAt(i, 0).equals(newID)) {

idExists = true;

break;

}

}

if (idExists) {

javax.swing.JOptionPane.showMessageDialog(this, "ID уже существует. Введите уникальный ID.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

// Проверка Discount

int discount;

try {

discount = Integer.parseInt(discountStr);

if (discount != 5 && discount != 10 && discount != 15) {

throw new IllegalArgumentException("Недопустимое значение скидки.");

}

} catch (NumberFormatException e) {

javax.swing.JOptionPane.showMessageDialog(this, "Скидка должна быть числом. Допустимые значения: 5, 10, 15.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

} catch (IllegalArgumentException e) {

javax.swing.JOptionPane.showMessageDialog(this, "Допустимые значения скидки: 5, 10, 15.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

// Проверка Category

if (!category.equals("Student") && !category.equals("Pensioner") && !category.equals("Postoyalec")) {

javax.swing.JOptionPane.showMessageDialog(this, "Допустимые категории: Student, Pensioner, Postoyalec.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

LinkedList<String> answer = new LinkedList<>();

answer.add(ID.getText());

answer.add(Surname.getText());

answer.add(Name.getText());

answer.add(Patronymic.getText());

answer.add(Passport.getText());

answer.add(Address.getText());

answer.add(Comment.getText());

answer.add(Category.getText());

answer.add(Discount.getText());

try {

String request = "addPost";

c.add(request, answer);

request = "selectPost";

answer = c.refresh(request);

refreshPost(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

try {

if (c == null) this.c = new Client();

//Thread.sleep(1000);

String request = "selectPost";

LinkedList<String> answer = c.refresh(request);

refreshPost(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*\*

\* @param args the command line arguments

\*/

public void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException | InstantiationException | IllegalAccessException | javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Check\_in.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

try {

new Postoyalec().setVisible(true);

} catch (IOException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton Add;

private javax.swing.JTextField Address;

private javax.swing.JButton Apart;

private javax.swing.JTextField Category;

private javax.swing.JButton Check\_in;

private javax.swing.JTextField Comment;

private javax.swing.JButton Delete;

private javax.swing.JTextField Discount;

private javax.swing.JTextField ID;

private javax.swing.JTextField Name;

private javax.swing.JTextField Passport;

private javax.swing.JTextField Patronymic;

private javax.swing.JButton Post;

private javax.swing.JButton Reservation;

private javax.swing.JButton Save;

private javax.swing.JTextField Surname;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

private javax.swing.JLabel jLabel7;

private javax.swing.JLabel jLabel8;

private javax.swing.JLabel jLabel9;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTable jTable1;

// End of variables declaration

}

Файл CheckIn.java

import javax.swing.table.DefaultTableModel;

import java.io.IOException;

import java.util.LinkedList;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

/\*\*

\*

\* @author RCh

\*/

public class Check\_in extends javax.swing.JFrame {

Client c;

public Check\_in(Client c){

this.c = c;

initComponents();

}

// Конструктор класса

public Check\_in() throws IOException, ClassNotFoundException {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

Post = new javax.swing.JButton();

Check\_in = new javax.swing.JButton();

Reservation = new javax.swing.JButton();

Apart = new javax.swing.JButton();

Save = new javax.swing.JButton();

Delete = new javax.swing.JButton();

Add = new javax.swing.JButton();

jScrollPane2 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

ID = new javax.swing.JTextField();

Napart = new javax.swing.JTextField();

Checkin = new javax.swing.JTextField();

Checkout = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent evt) {

formWindowClosing(evt);

}

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

Post.setText("Постояльцы");

Post.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

PostActionPerformed(evt);

}

});

Check\_in.setText("Заселение");

Check\_in.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

Check\_inActionPerformed(evt);

}

});

Reservation.setText("Бронирование");

Reservation.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ReservationActionPerformed(evt);

}

});

Apart.setText("Апартаменты");

Apart.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ApartActionPerformed(evt);

}

});

Save.setText("Сохранить изменения");

Save.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SaveActionPerformed(evt);

}

});

Delete.setText("Удалить");

Delete.setPreferredSize(new java.awt.Dimension(84, 23));

Delete.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

DeleteActionPerformed(evt);

}

});

Add.setText("Добавить");

Add.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

AddActionPerformed(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"ID", "Номер апартов", "Дата заселения", "Дата выселения"

}

) {

boolean[] canEdit = new boolean [] {

false, true, true, true

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jScrollPane2.setViewportView(jTable1);

if (jTable1.getColumnModel().getColumnCount() > 0) {

jTable1.getColumnModel().getColumn(0).setResizable(false);

jTable1.getColumnModel().getColumn(1).setResizable(false);

jTable1.getColumnModel().getColumn(2).setResizable(false);

jTable1.getColumnModel().getColumn(3).setResizable(false);

}

ID.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

IDActionPerformed(evt);

}

});

Napart.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

NapartActionPerformed(evt);

}

});

Checkin.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

CheckinActionPerformed(evt);

}

});

jLabel1.setText("ID Постояльца");

jLabel3.setText("Номер апартов");

jLabel4.setText("Дата заселения");

jLabel5.setText("Дата выселения");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(Post)

.addGap(56, 56, 56)

.addComponent(Check\_in)

.addGap(56, 56, 56)

.addComponent(Reservation)

.addGap(56, 56, 56)

.addComponent(Apart))

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 575, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addComponent(Save)

.addGap(105, 105, 105)

.addComponent(Delete, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 16, Short.MAX\_VALUE)

.addComponent(Add, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(22, 22, 22))

.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(ID)

.addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Napart)

.addComponent(Checkin)

.addComponent(Checkout)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel5)

.addComponent(jLabel4))

.addGap(0, 0, Short.MAX\_VALUE)))

.addContainerGap())))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(16, 16, 16)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Post)

.addComponent(Check\_in)

.addComponent(Reservation)

.addComponent(Apart))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 354, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Save)

.addComponent(Delete, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(Add)))

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(ID, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(4, 4, 4)

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Napart, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Checkin, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Checkout, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void ReservationActionPerformed(java.awt.event.ActionEvent evt) {

new Reservation(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void Check\_inActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void ApartActionPerformed(java.awt.event.ActionEvent evt) {

new Apart(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {

try {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

LinkedList<String> answer = new LinkedList<>();

int row = jTable1.getSelectedRow();

String s = (String) model.getValueAt(row, 0);

String request = "delCheck";

c.del(request, s);

request = "selectCheck";

answer = c.refresh(request);

refreshCheck(answer);

} catch (IOException | InterruptedException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void formWindowClosing(java.awt.event.WindowEvent evt) {

c.end();

}

void refreshCheck(LinkedList<String> answer){

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

int rows = model.getRowCount();

while (rows != 0) {

model.removeRow(0);

rows--;

}

Object[] row;

for (int i = 0; i < answer.size(); i = i+4){

row = new Object[4];

row[0] = answer.get(i);

row[1] = answer.get(i+1);

row[2] = answer.get(i+2);

row[3] = answer.get(i+3);

model.addRow(row);

}

}

private void PostActionPerformed(java.awt.event.ActionEvent evt) {

new Postoyalec(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void IDActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void SaveActionPerformed(java.awt.event.ActionEvent evt) {

try {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

LinkedList<String> answer = new LinkedList<>();

int rows = jTable1.getRowCount();

for (int i = 0; i < rows; i++) {

for (int j = 0; j < 4; j++) {

answer.add((String) model.getValueAt(i, j));

}

}

String request = "updateCheck";

c.update(request, answer);

request = "selectCheck";

answer = c.refresh(request);

refreshCheck(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void AddActionPerformed(java.awt.event.ActionEvent evt) {

String newID = ID.getText();

String napart = Napart.getText();

String startDateStr = Checkin.getText();

String endDateStr = Checkout.getText();

// Проверка на пустые поля

if (newID.isEmpty() || napart.isEmpty() || startDateStr.isEmpty() || endDateStr.isEmpty()) {

javax.swing.JOptionPane.showMessageDialog(this, "Все поля должны быть заполнены.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

boolean idExists = false;

// Проверка на существующий ID

for (int i = 0; i < model.getRowCount(); i++) {

if (model.getValueAt(i, 0).equals(newID)) {

idExists = true;

break;

}

}

if (idExists) {

javax.swing.JOptionPane.showMessageDialog(this, "ID уже существует. Введите уникальный ID.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

// Проверка дат

try {

java.text.SimpleDateFormat dateFormat = new java.text.SimpleDateFormat("yyyy-MM-dd");

java.util.Date startDate = dateFormat.parse(startDateStr);

java.util.Date endDate = dateFormat.parse(endDateStr);

if (endDate.before(startDate)) {

javax.swing.JOptionPane.showMessageDialog(this, "Дата выселения должна быть больше чем дата заселения.", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

} catch (java.text.ParseException e) {

javax.swing.JOptionPane.showMessageDialog(this, "Неверный формат даты. Используйте формат yyyy-MM-dd. Обязательно выберите месяца от 01 до 12", "Ошибка", javax.swing.JOptionPane.ERROR\_MESSAGE);

return;

}

LinkedList<String> answer = new LinkedList<>();

answer.add(ID.getText());

answer.add(Napart.getText());

answer.add(Checkin.getText());

answer.add(Checkout.getText());

try {

String request = "addCheck";

c.add(request, answer);

request = "selectCheck";

answer = c.refresh(request);

refreshCheck(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

try {

String request = "selectCheck";

LinkedList<String> answer = c.refresh(request);

refreshCheck(answer);

} catch (IOException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

} catch (InterruptedException ex) {

Logger.getLogger(Check\_in.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void NapartActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void CheckinActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

/\*\*

\* @param args the command line arguments

\*/

public void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Check\_in.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Check\_in.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Check\_in.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Check\_in.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

try {

new Postoyalec().setVisible(true);

} catch (IOException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton Add;

private javax.swing.JButton Apart;

private javax.swing.JButton Check\_in;

private javax.swing.JTextField Checkin;

private javax.swing.JTextField Checkout;

private javax.swing.JButton Delete;

private javax.swing.JTextField ID;

private javax.swing.JTextField Napart;

private javax.swing.JButton Post;

private javax.swing.JButton Reservation;

private javax.swing.JButton Save;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTable jTable1;

// End of variables declaration

}

Файл Apartament.java

import javax.swing.table.DefaultTableModel;

import java.io.IOException;

import java.util.LinkedList;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JOptionPane;

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

/\*\*

\*

\* @author RCh

\*/

public class Apart extends javax.swing.JFrame {

Client c;

public Apart(Client c){

this.c = c;

initComponents();

}

// Конструктор класса

public Apart() throws IOException, ClassNotFoundException {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

Post = new javax.swing.JButton();

Check\_in = new javax.swing.JButton();

Reservation = new javax.swing.JButton();

Apart = new javax.swing.JButton();

Save = new javax.swing.JButton();

Delete = new javax.swing.JButton();

Add = new javax.swing.JButton();

jScrollPane2 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

NumAp = new javax.swing.JTextField();

Capacity = new javax.swing.JTextField();

Comfort = new javax.swing.JTextField();

Price = new javax.swing.JTextField();

Free = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent evt) {

formWindowClosing(evt);

}

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

Post.setText("Постояльцы");

Post.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

PostActionPerformed(evt);

}

});

Check\_in.setText("Заселение");

Check\_in.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

Check\_inActionPerformed(evt);

}

});

Reservation.setText("Бронирование");

Reservation.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ReservationActionPerformed(evt);

}

});

Apart.setText("Апартаменты");

Apart.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ApartActionPerformed(evt);

}

});

Save.setText("Сохранить изменения");

Save.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SaveActionPerformed(evt);

}

});

Delete.setText("Удалить");

Delete.setPreferredSize(new java.awt.Dimension(84, 23));

Delete.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

DeleteActionPerformed(evt);

}

});

Add.setText("Добавить");

Add.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

AddActionPerformed(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Номер апарт", "Вместимость", "Комфорт", "Стоимость", "Свободно"

}

) {

boolean[] canEdit = new boolean [] {

false, true, true, true, true

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jScrollPane2.setViewportView(jTable1);

if (jTable1.getColumnModel().getColumnCount() > 0) {

jTable1.getColumnModel().getColumn(0).setResizable(false);

jTable1.getColumnModel().getColumn(1).setResizable(false);

jTable1.getColumnModel().getColumn(2).setResizable(false);

jTable1.getColumnModel().getColumn(3).setResizable(false);

jTable1.getColumnModel().getColumn(4).setResizable(false);

}

NumAp.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

NumApActionPerformed(evt);

}

});

jLabel1.setText("Номер апарт");

jLabel3.setText("Вместимость");

jLabel4.setText("Комфорт");

jLabel5.setText("Стоимость");

jLabel2.setText("Свободно");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(Post)

.addGap(56, 56, 56)

.addComponent(Check\_in)

.addGap(56, 56, 56)

.addComponent(Reservation)

.addGap(56, 56, 56)

.addComponent(Apart))

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 575, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addComponent(Save)

.addGap(105, 105, 105)

.addComponent(Delete, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(NumAp)

.addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Capacity)

.addComponent(Comfort)

.addComponent(Price)

.addComponent(Free)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED\_SIZE, 61, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 62, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel4))

.addGap(0, 64, Short.MAX\_VALUE)))

.addContainerGap())))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Add, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(19, 19, 19))))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(16, 16, 16)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Post)

.addComponent(Check\_in)

.addComponent(Reservation)

.addComponent(Apart))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 354, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(NumAp, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(4, 4, 4)

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Capacity, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Comfort, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Price, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel2)

.addGap(2, 2, 2)

.addComponent(Free, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Save)

.addComponent(Delete, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(Add))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void ReservationActionPerformed(java.awt.event.ActionEvent evt) {

new Reservation(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void Check\_inActionPerformed(java.awt.event.ActionEvent evt) {

new Check\_in(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void ApartActionPerformed(java.awt.event.ActionEvent evt) {

}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {

try {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

LinkedList<String> answer = new LinkedList<>();

int row = jTable1.getSelectedRow();

String s = (String) model.getValueAt(row, 0);

String request = "delApart";

c.del(request, s);

request = "selectApart";

answer = c.refresh(request);

refreshApart(answer);

} catch (IOException | InterruptedException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void formWindowClosing(java.awt.event.WindowEvent evt) {

c.end();

}

void refreshApart(LinkedList<String> answer){

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

int rows = model.getRowCount();

while (rows != 0) {

model.removeRow(0);

rows--;

}

Object[] row;

for (int i = 0; i < answer.size(); i = i+5){

row = new Object[5];

row[0] = answer.get(i);

row[1] = answer.get(i+1);

row[2] = answer.get(i+2);

row[3] = answer.get(i+3);

row[4] = answer.get(i+4);

model.addRow(row);

}

}

private void PostActionPerformed(java.awt.event.ActionEvent evt) {

new Postoyalec(c).setVisible(true);

this.setVisible(false);

this.dispose();

}

private void NumApActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void SaveActionPerformed(java.awt.event.ActionEvent evt) {

try {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

LinkedList<String> answer = new LinkedList<>();

int rows = jTable1.getRowCount();

for (int i = 0; i < rows; i++) {

for (int j = 0; j < 5; j++) {

answer.add((String) model.getValueAt(i, j));

}

}

String request = "updateApart";

c.update(request, answer);

request = "selectApart";

answer = c.refresh(request);

refreshApart(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void AddActionPerformed(java.awt.event.ActionEvent evt) {

String numAp = NumAp.getText();

String capacity = Capacity.getText();

String comfort = Comfort.getText();

String price = Price.getText();

String freeValue = Free.getText();

// Проверка на пустые поля

if (numAp.isEmpty() || capacity.isEmpty() || comfort.isEmpty() || price.isEmpty() || freeValue.isEmpty()) {

JOptionPane.showMessageDialog(this, "Все поля должны быть заполнены.", "Ошибка", JOptionPane.ERROR\_MESSAGE);

return;

}

if (isNumApExists(numAp)) {

JOptionPane.showMessageDialog(this, "Апартамент с таким номером уже существует", "Предупреждение", JOptionPane.WARNING\_MESSAGE);

return; // Прерывание операции добавления

}

if (!("0".equals(freeValue) || "1".equals(freeValue))) {

JOptionPane.showMessageDialog(this, "Значение поля 'Свободно' может быть только 0 или 1, где 0 - нет, 1 - да", "Предупреждение", JOptionPane.WARNING\_MESSAGE);

return; // Прерывание операции добавления

}

// Проверка значения поля "Comfort"

if (!("lux".equals(comfort) || "polulux".equals(comfort) || "econom".equals(comfort))) {

JOptionPane.showMessageDialog(this, "Допустимые значения поля 'Comfort': lux, polulux, econom.", "Ошибка", JOptionPane.ERROR\_MESSAGE);

return; // Прерывание операции добавления

}

LinkedList<String> answer = new LinkedList<>();

answer.add(numAp);

answer.add(Capacity.getText());

answer.add(Comfort.getText());

answer.add(Price.getText());

answer.add(Free.getText());

try {

String request = "addApart";

c.add(request, answer);

request = "selectApart";

answer = c.refresh(request);

refreshApart(answer);

} catch (IOException | ClassNotFoundException | InterruptedException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

private boolean isNumApExists(String numAp) {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

for (int row = 0; row < model.getRowCount(); row++) {

if (numAp.equals(model.getValueAt(row, 0))) {

return true; // Найден существующий номер апартамента

}

}

return false; // Номер апартамента не найден

}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

try {

String request = "selectApart";

LinkedList<String> answer = c.refresh(request);

refreshApart(answer);

} catch (IOException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

} catch (InterruptedException ex) {

Logger.getLogger(Check\_in.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*\*

\* @param args the command line arguments

\*/

public void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException | InstantiationException | IllegalAccessException | javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Check\_in.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

try {

new Postoyalec().setVisible(true);

} catch (IOException | ClassNotFoundException ex) {

Logger.getLogger(Postoyalec.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton Add;

private javax.swing.JButton Apart;

private javax.swing.JTextField Capacity;

private javax.swing.JButton Check\_in;

private javax.swing.JTextField Comfort;

private javax.swing.JButton Delete;

private javax.swing.JTextField Free;

private javax.swing.JTextField NumAp;

private javax.swing.JButton Post;

private javax.swing.JTextField Price;

private javax.swing.JButton Reservation;

private javax.swing.JButton Save;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTable jTable1;

}

**Приложение B. UML-диаграммы приложения**

**Приложение B.1 – UML-диаграмма вариантов использования**



Рисунок 12 – UML-диаграмма вариантов использования

**Приложение B.2 – UML-диаграмма классов сервера**

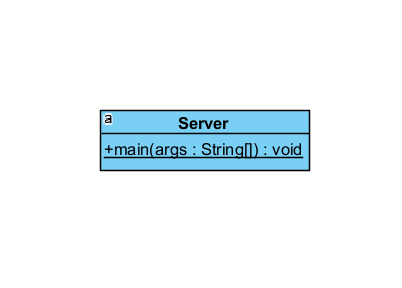


Рисунок 13 – UML-диаграмма классов сервера

**Приложение B.3 – UML-диаграмма классов клиента**

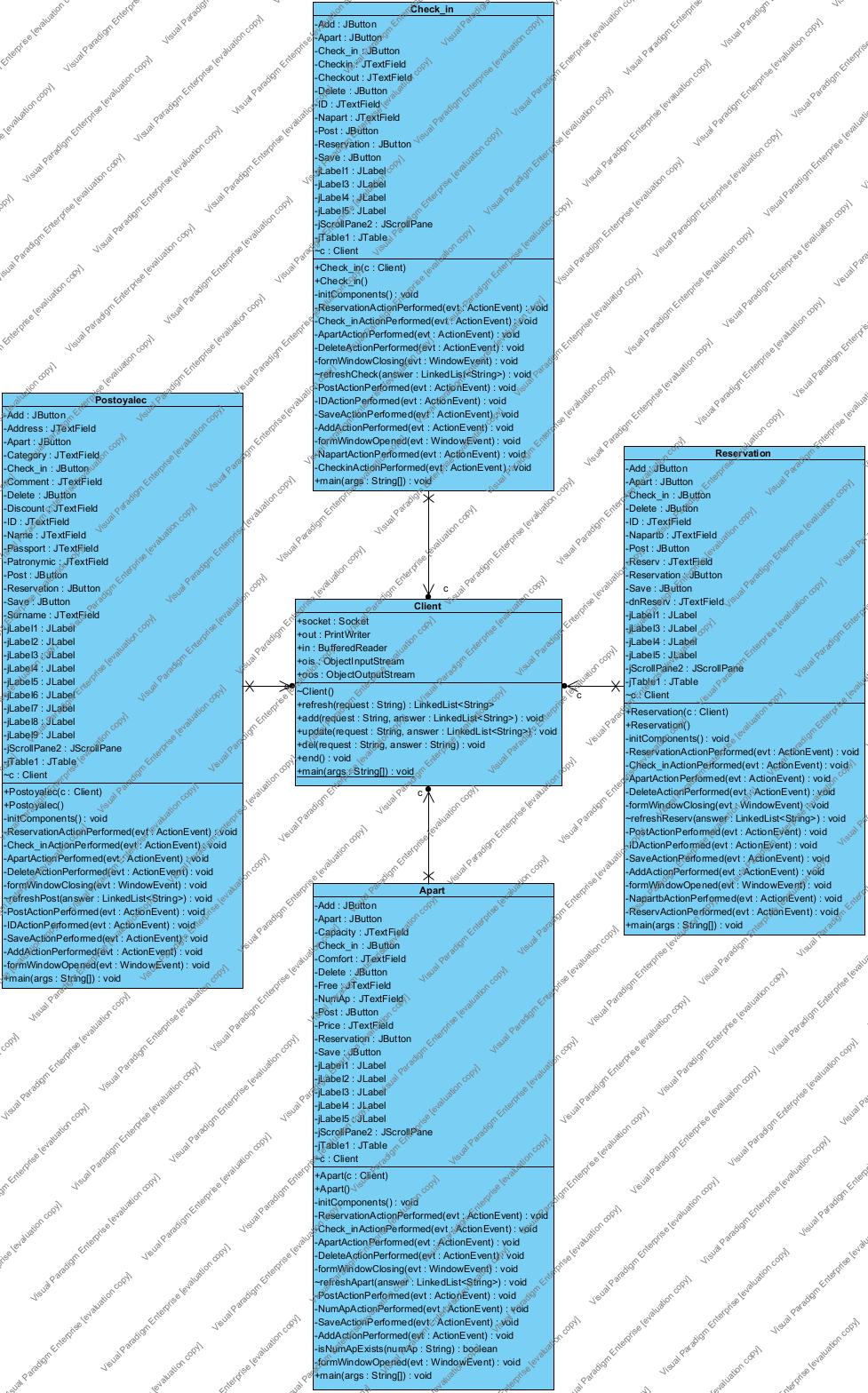


Рисунок 14 – UML-диаграмма классов клиента

**Приложение B.4 – UML-диаграмма деятельности**



Рисунок 15 – UML-диаграмма деятельности

**Приложение B.5 – UML-диаграмма развёртывания**



Рисунок 16 – UML-диаграмма развёртывания

**Приложение B.6 – UML-диаграмма последовательности**



Рисунок 17 – UML-диаграмма последовательности