

Uniwersytet Przyrodniczo-Humanistyczny

W Siedlcach

Kierunek Informatyka

Imię i nazwisko studenta:

Kacper Trocewicz

Nr indeksu:

88639

Temat Zadania:

2. System obsługi hotelu Program, który umożliwia gromadzenie i przetwarzanie danych o miejscach oferowanych w hotelu i klientach.

- dane przechowywane w plikach (zapis i odczyt, plik/pliki),
- uwzględnij: zwykłego użytkownika, który może tylko przeglądać ofertę hotelu i dokonywać rezerwacji
- klienta hotelu, który może przeglądać ofertę hotelu, dokonywać rezerwacji, przedłużać pobyt i opłacać pobyt
- pracownika hotelu, który wprowadza dane o pokojach, cennik, potwierdza rezerwacje,
- cennik uwzględnia standard pokoju (piętro, położenie na piętrze), liczbę miejsc w pokoju,
- system umożliwia wyszukiwanie pokoi wg. ceny, piętra, liczby miejsc
- system pozwala sporządzić listę wolnych pokoi, listę rezerwacji
- system umożliwia wysyłanie informacji do klientów o ofertach i promocjach.

dr Anny Kołkowicz

Siedlce, 2022

Spis treści

1	Specyfikacja problemu. Przyjęte założenia i ograniczenia:.....	3
2	Opis klas, algorytmów, zmiennych:.....	4
3	Diagram Klas w UML:.....	5
4	Kod Programu:.....	6
5	Przykładowe dane i wyniki(Zrzuty z opisem):.....	37
6	Instrukcja dla użytkownika(jak uruchomić, jakie dane wprowadzać):.....	41

1 Specyfikacja problemu. Przyjęte założenia i ograniczenia:

Ograniczenia:

- Żeby dodać pokój trzeba wpisać jego nazwę.
- W wyszukiwaniu pokoju najniższa cena za noc jest 50 zł a najwyższa nie ma ograniczeń cenowych.
- Po wybraniu użytkownika trzeba zrestartować program żeby zmienić na pracownika.
- Żeby mieć możliwość dostępu do zarządzania rezerwacjami trzeba mieć najpierw zarezerwowany pokój.
- Jak już osoba zarezerwowała pokój na daną datę inna osoba nie może zarezerwować tego samego pokoju na tę samą datę.
- Rezerwacji nie można dokonać na obecny dzień.
- Po zarezerwowaniu przez użytkownika pokoju stajemy się klientem i musimy zrestartować program żeby pokazała się nam możliwość Zarządzania rezerwacją.
- Pracownik Hotelu może dodać maksymalnie 50 pokoi.

Założenia:

- Program ma za zadanie obsługę hotelu, ma w nim znajdować się klient i pracownik. Pracownik ma za zadanie zarządzać rezerwacjami lub dodawać pokoje, ma również uprawnienia do potwierdzania rezerwacji. Klient ma możliwość sprawdzania ofert wolnych pokoi i rezerwacji danego pokoju, lub przedłużenie jego pobytu.

2 Opis klas, algorytmów, zmiennych:

BookingFrame.java-Jest to klasa która tworzy nam okno danej rezerwacji w której możemy przedłużyć pobyt zobaczyć czy rezerwacja jest potwierdzona i czy została opłacona

BookingsFrame.java-Jest to klasa która tworzy nam okno w której jest tabela która pokazuje dane rezerwacji które wpisał klient._

MainFrame.java-jest to klasa która tworzy nam okno główne w którym wybieramy klienta lub pracownika.

NewRoomFrame.java-jest to klasa która tworzy nam okno w którym mamy możliwość dodania nowego pokoju przez pracownika.

OfferListFrame.java- jest to klasa która tworzy okno z tabelą, zawiera ona informacje o dostępności pokoi o danych które klient wpisał w wyszukiwaniu.

OfferSpecificationFrame.java- jest to klasa która tworzy nam okno, w którym mamy możliwość wyszukania danego pokoju jaki chcemy zarezerwować.

Booking.java-Jest to klasa zarządzająca rezerwacjami.

Client.java-Jest to klasa która zarządza klientem, dziedziczy ona login, name, contactnumber, email. Ta Klasa pozwala nam dostać się do takich opcji jak „Wyświetl ofertę”, „Zarządzaj rezerwacjami” ma uprawnienia również takie żeby opłacić rezerwacje jak i zrobić rezerwacje.

Employee.java- Jest to klasa która zarządza pracownikiem, dziedziczy ona login, name, contactnumber, email. Ta Klasa pozwala nam dostać się do takich opcji jak „Dodaj pokój”, „Zarządzaj rezerwacjami” ma uprawnienia również takie żeby potwierdzić rezerwacje klienta.

Hotel.java-Jest to klasa która posiada w sobie tablice dynamiczną pokoi, służy ona do przeszukiwania list dzięki którym można znaleźć wolny pokój.

Room.java-Jest to klasa która posiada metody do segregacji, po wpisanych danych przez klienta hotelu.

User.java-Jest to klasa abstrakcyjna w której mamy zmienne login, name, contactnumber.

BookingService.java-Jest to klasa która odczytuje i zapisuje rezerwacje klientów jak i również przedłużania pobytu.

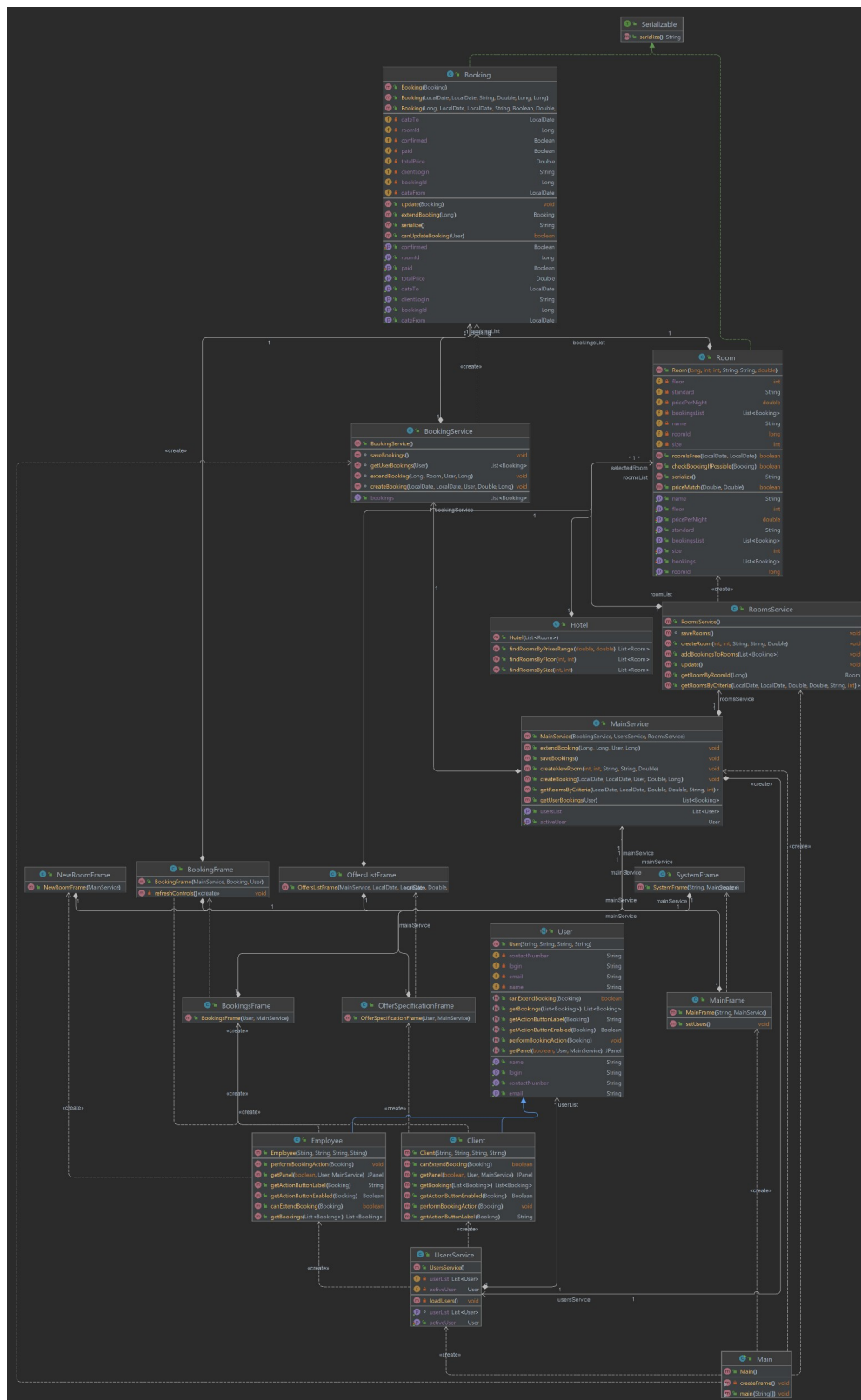
MainService.java-Jest to klasa która posiada tablice dynamiczną, używana jest do zapisywania i tworzenia pokoi i również ich wyszukiwania.

RoomsService.java-Jest to klasa która odczytuje i zapisuje wszystkie pokoje i ich dane.

UsersService.java-Jest to klasa która odczytuje użytkowników z pliku tekstowego i ustala czy jest pracownikiem czy klientem.

3 Diagram Klas w UML:

W E-mailu wysyłam zdjęcie diagramu aby było ono bardziej czytelne.



4 Kod Programu(Parę przykładowych klas reszta wysłana E-mail):

Booking.Java:

```
package Project.model;

import java.time.LocalDate;

public class Booking implements Serializable {
    static Long bookings = 0L;

    private Long bookingId;
    private Long roomId;
    private LocalDate dateFrom;
    private LocalDate dateTo;
    private String clientLogin;
    private Boolean confirmed;
    private Double totalPrice;
    private Boolean paid;

    public Booking(LocalDate dateFrom, LocalDate dateTo, String
clientLogin, Double totalPrice, Long roomId, Long bookingId) {
        this.dateFrom = dateFrom;
        this.dateTo = dateTo;
        this.clientLogin = clientLogin;
        this.totalPrice = totalPrice;
        this.roomId = roomId;
        this.confirmed = false;
        this.paid = false;
        this.bookingId = bookingId;
    }

    public Booking(Booking other){
        this.dateFrom = other.dateFrom;
        this.dateTo = other.dateTo;
        this.clientLogin = other.clientLogin;
        this.confirmed = other.confirmed;
        this.totalPrice = other.totalPrice;
        this.paid = other.paid;
        this.bookingId = other.bookingId;
        this.roomId = other.roomId;
    }

    public Booking(Long bookingId, LocalDate dateFrom, LocalDate
dateTo, String clientLogin, Boolean confirmed, Double totalPrice,
Boolean paid, Long roomId) {
        this.dateFrom = dateFrom;
        this.dateTo = dateTo;
        this.clientLogin = clientLogin;
        this.confirmed = confirmed;
        this.totalPrice = totalPrice;
```

```

        this.paid = paid;
        this.bookingId = bookingId;
        this.roomId = roomId;
        if(bookingId >= bookings)
            bookings = bookingId + 1;
    }

    public Long getRoomId() {
        return roomId;
    }

    public boolean canUpdateBooking(User updatingUser){

        return updatingUser.getLogin().equals(getClientLogin());
    }

    public LocalDate getDateFrom() {
        return dateFrom;
    }

    public LocalDate getDateTo() {
        return dateTo;
    }

    public String getClientLogin() {
        return clientLogin;
    }

    public Boolean getConfirmed() {
        return confirmed;
    }

    public Double getTotalPrice() {
        return totalPrice;
    }

    public Boolean getPaid() {
        return paid;
    }

    public Long getBookingId() {
        return bookingId;
    }

    public Booking extendBooking(Long days){
        Booking tmpBooking = new Booking(this);
        tmpBooking.dateTo = dateTo.plusDays(days);
        return tmpBooking;
    }

    public void setPaid(Boolean paid) {
        this.paid = paid;
    }

```

```

    }

    public void setConfirmed(Boolean confirmed) {
        this.confirmed = confirmed;
    }

    public void update(Booking booking) {
        this.dateTo = booking.getDateTo();
        this.confirmed = false;
        this.paid = false;
    }

    @Override
    public String serialize() {
        return "" + bookingId + "," + clientLogin + "," + roomId +
        "," + dateFrom.getDayOfMonth() + "," + dateFrom.getMonthValue() + "," +
        dateFrom.getYear()
            + "," + dateTo.getDayOfMonth() + "," +
        dateTo.getMonthValue() + "," + dateTo.getYear() + "," + totalPrice +
        "," + confirmed + "," + paid;
    }
}

```

Client.Java:

```

package Project.model;

import Project.Main;
import Project.frames.BookingsFrame;
import Project.frames.MainFrame;
import Project.frames.OfferSpecificationFrame;
import Project.service.MainService;

import javax.swing.*;
import java.util.List;
import java.util.stream.Collectors;

public class Client extends User {

    public Client(String login, String name, String contactNumber,
String email) {
        super(login, name, contactNumber, email);
    }

    @Override
    public JPanel getPanel(boolean hasReservations, User activeUser,
MainService mainService) {
        JPanel panel = new JPanel();

        JButton offerButton = new JButton("Wyświetl ofertę");
        offerButton.addActionListener(
            e -> {

```



```

        JFrame offerFrame = new
OfferSpecificationFrame(activeUser, mainService);
        offerFrame.setVisible(true);
    }

);

panel.add(offerButton);
if(hasReservations) {
    JButton bookingsManagement = new JButton("Zarządzaj
rezerwacjami");
    bookingsManagement.addActionListener(
        e -> {
            JFrame clientFrame = new
BookingsFrame(activeUser, mainService);
            clientFrame.setVisible(true);
        }
    );
    panel.add(bookingsManagement);
}
return panel;
}

@Override
public boolean canExtendBooking(Booking booking) {
    return booking.getClientLogin().equals(getLogin());
}

@Override
public String getActionButtonLabel(Booking booking) {
    if(booking.getClientLogin().equals(getLogin()))
        return "Opłacić";
    else return "Zarezerwuj";
}

@Override
public Boolean getActionButtonEnabled(Booking booking) {
    if(booking.getClientLogin().equals(getLogin()))
        return !booking.getPaid();
    else return true;
}

@Override
public List<Booking> getBookings(List<Booking> bookings) {
    return bookings.stream().filter(it ->
it.getClientLogin().equals(getLogin())).collect(Collectors
.toList());
}

@Override
public void performBookingAction(Booking booking) {
    booking.setPaid(true);
}

```

```
}
```

Employee.Java:

```
package Project.model;

import Project.frames.BookingsFrame;
import Project.frames.NewRoomFrame;
import Project.service.MainService;

import javax.swing.*.*;
import java.util.List;

public class Employee extends User {

    public Employee(String login, String name, String contactNumber,
String email) {
        super(login, name, contactNumber, email);
    }

    @Override
    public JPanel getPanel(boolean hasReservations, User activeUser,
MainService mainService) {
        JPanel panel = new JPanel();
        JButton addRoomButton = new JButton("Dodaj pokój");
        panel.add(addRoomButton);
        addRoomButton.addActionListener(
            e -> {
                JFrame newRoomFrame = new
NewRoomFrame(mainService);
                newRoomFrame.setVisible(true);
            }
        );
        JButton manageBookingsButton = new JButton("Zarządzaj
rezerwacjami");
        panel.add(manageBookingsButton);
        manageBookingsButton.addActionListener(
            e -> {
                JFrame newRoomFrame = new
BookingsFrame(activeUser, mainService);
                newRoomFrame.setVisible(true);
            }
        );
        panel.add(manageBookingsButton);
        return panel;
    }

    @Override
    public boolean canExtendBooking(Booking booking) {
        return false;
    }
}
```

```

@Override
public String getActionButtonLabel(Booking booking) {
    return "Potwierdź";
}

@Override
public Boolean getActionButtonEnabled(Booking booking) {
    return !booking.getConfirmed();
}

@Override
public List<Booking> getBookings(List<Booking> bookings) {
    return bookings;
}

@Override
public void performBookingAction(Booking booking) {
    booking.setConfirmed(true);
}
}

```

User.Java:

```

package Project.model;

import Project.service.MainService;

import javax.swing.*.*;
import java.util.List;

public abstract class User {
    private String login;
    private String name;
    private String contactNumber;
    private String email;

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

    public String getLogin() {
        return login;
    }

    public String getName() {
        return name;
    }
}

```

```

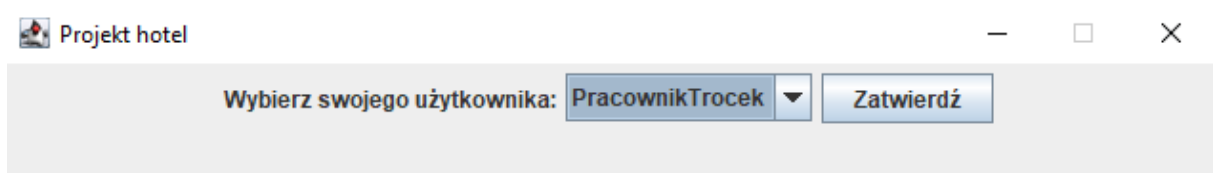
public String getContactNumber() {
    return contactNumber;
}

public String getEmail() {
    return email;
}

public abstract JPanel getPanel(boolean hasReservations, User
activeUser, MainService mainService);
public abstract boolean canExtendBooking(Booking booking);
public abstract String getActionButtonLabel(Booking booking);
public abstract Boolean getActionButtonEnabled(Booking booking);
public abstract List<Booking> getBookings(List<Booking>
bookings);
public abstract void performBookingAction(Booking booking);
}

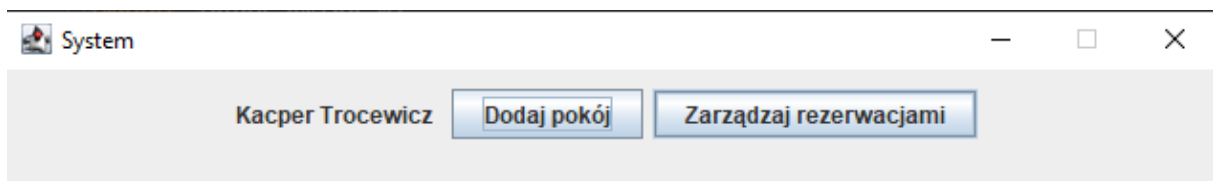
```

5 Przykładowe dane i wyniki(Zrzuty z opisem):



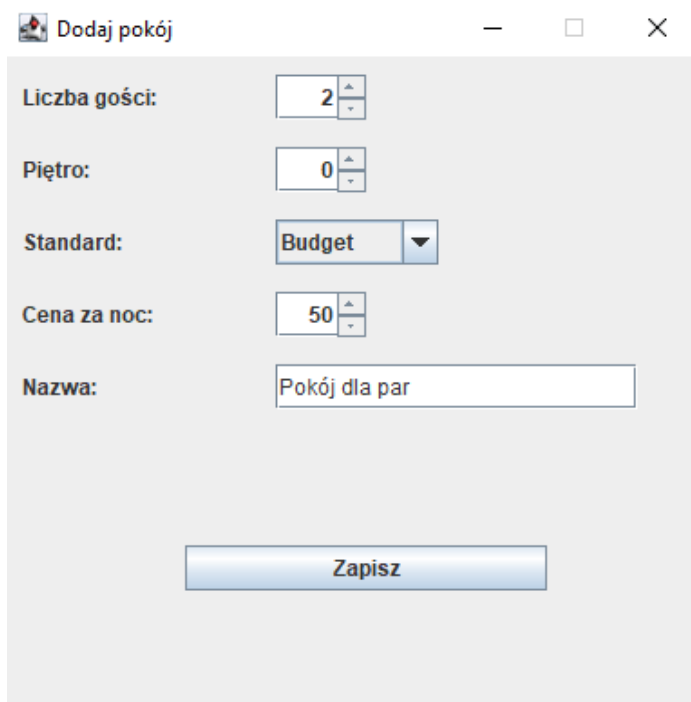
Rysunek 1.Menu Główne.

Na tym screen'ie widoczna jest możliwość wybrania użytkownika lub pracownika.



Rysunek 2.Pracownik Hotelu.

Po wybraniu pracownika wyświetlają się jego uprawnienia – jak widoczne jest na screen'ie istnieje możliwość dodania pokoju i zarządzania rezerwacjami.



Dodaj pokój

Liczba gości: 2

Piętro: 0

Standard: Budget

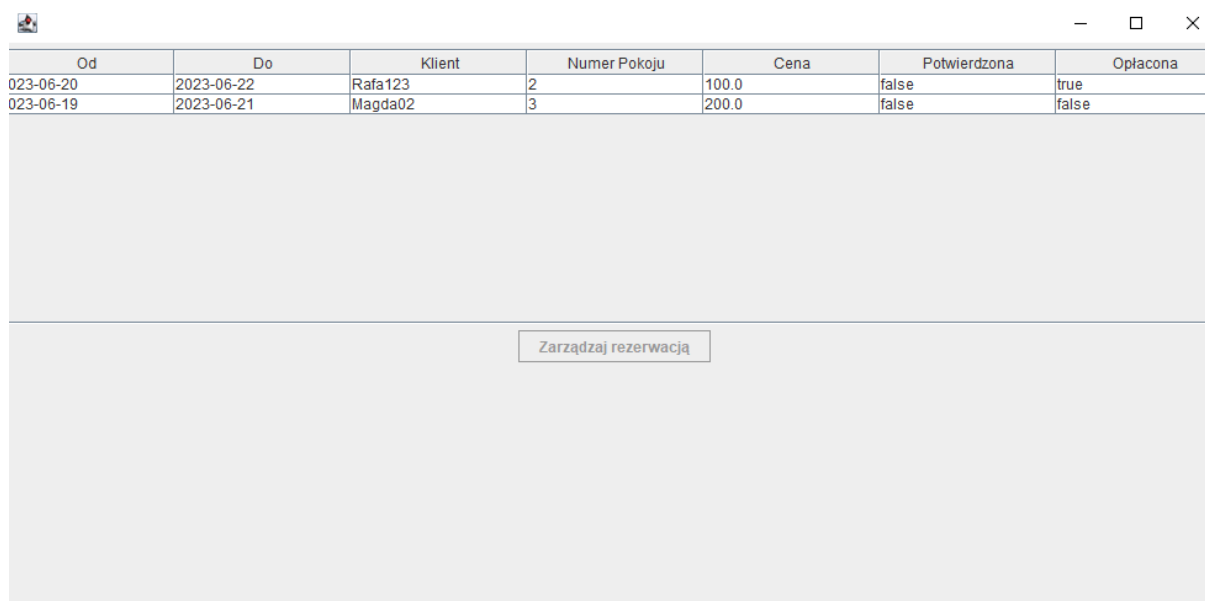
Cena za noc: 50

Nazwa: Pokój dla par

Zapisz

Rysunek 3. Dodawanie Pokoi w hotelu przez pracownika.

Po wybraniu opcji „Dodaj pokój” wyświetli się okno oraz opcja wyboru rodzaju pokoju, który chcemy dodać do oferty.



Od	Do	Klient	Numer Pokoju	Cena	Potwierdzona	Opłacona
023-06-20	2023-06-22	Rafa123	2	100.0	false	true
023-06-19	2023-06-21	Magda02	3	200.0	false	false

Zarządzaj rezerwacją

Rysunek 4. Spis Rezerwacji przez klientów u pracownika.

Po wybraniu opcji „Zarządzaj rezerwacjami” istnieje możliwość wyboru jednej, konkretnej rezerwacji ze wszystkich aktualnych rezerwacji.

Rezerwacja nr 2

Od	2023-06-20
Do	2023-06-22
Użytkownik	Rafa123
Cena	100.0 zł
Oplacono	<input checked="" type="checkbox"/>
Zatwierdzono	<input type="checkbox"/>

Potwierdź

Powrót

Rysunek 5. Zarządzanie rezerwacją przez Pracownika.

Po naciśnięciu opcji „Zarządzaj rezerwacją” wyświetli się opcja potwierdzenia rezerwacji dokonanej rezerwacji.

System

Rafał Szydłowski Wyświetl ofertę

Rysunek 6. Menu Użytkownika.

Po wybraniu użytkownika wyświetli się okienko „Wyświetl ofertę”.

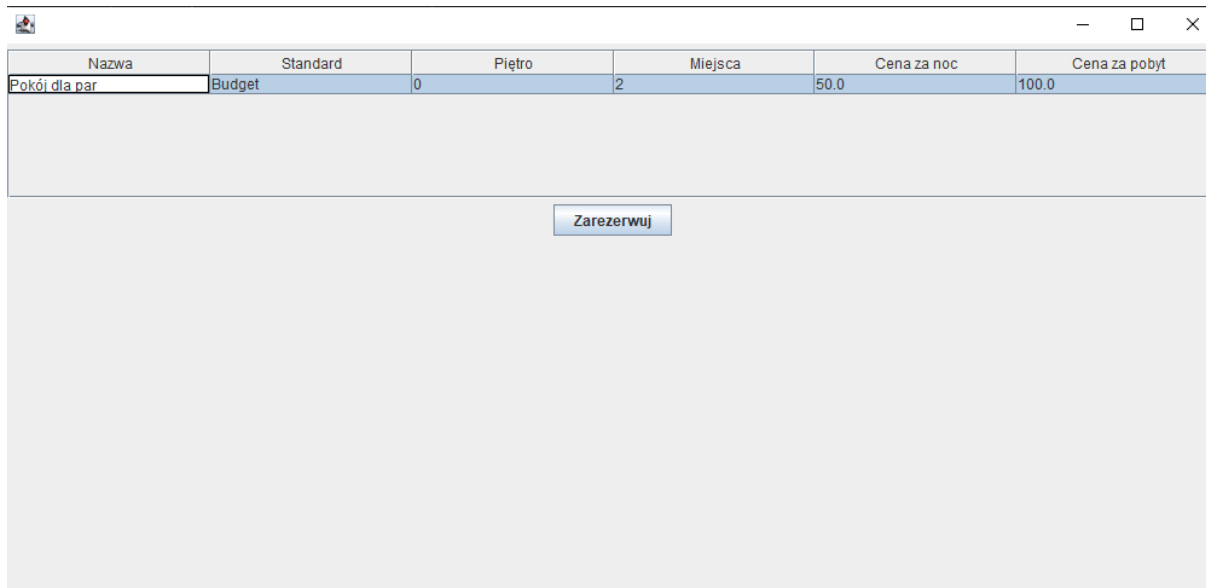
Specyfik...

Data od:	20.06.2023
Data do:	22.06.2023
Cena od:	50
Cena do:	75
Standard:	Budget
Goście:	1

Wyszukaj

Rysunek 7. Parametry wolnych pokoi do wyszukania.

Po wybraniu opcji „Wyświetl ofertę” pojawi się okienko z możliwością wybrania daty rezerwacji, od jakiej ceny do jakiej możliwy do wyboru jest pakiet i ile gości z nami będzie.



Nazwa	Standard	Piętro	Miejsca	Cena za noc	Cena za pobyt
Pokój dla par	Budget	0	2	50.0	100.0

Zarezerwuj

Rysunek 8. Wszystkie wolne pokoje według parametrów.

Po naciśnięciu wyszukaj wyświetli się tabela ze wszystkimi możliwymi pokojami i możliwością zarezerwowania go.



System

Rafał Szydłowski

Wyświetl ofertę

Zarządzaj rezerwacjami

Rysunek 9. Menu Klienta.

Dopiero po zarezerwowaniu pokoju wyświetla się opcja „Zarządzaj rezerwacjami”.

Od	Do	Klient	Numer Pokoju	Cena	Potwierdzona	Opłacona
2023-06-20	2023-06-22	Rafa123	2	100.0	false	false

Zarządzaj rezerwacją

Rysunek 10..Spis rezerwacji danego klienta.

Po kliknięciu Zarządzaj rezerwacjami pojawi się tabela z pokojami, które klient zarezerwował czy je opłacił czy pracownik potwierdził cena i data.

Rezerwacja nr 2

Od	2023-06-20		
Do	2023-06-22	<input type="checkbox"/> Przedłużyć pobyt?	<input type="text" value="0"/>
Użytkownik	Rafa123		
Cena	100.0 zł		
Opłacono	<input type="checkbox"/>		
Zatwierdzono	<input type="checkbox"/>		

Przedłuż pobyt

Opłać

Powrót

Rysunek 11.Zarządzanie rezerwacją przez klienta.

Po kliknięciu Zarządzaj rezerwacją wyświetlają się wszystkie dane, które klient wybrał i ma możliwość opłacenia rezerwacji, jak i przedłużenia pobytu.

6 Instrukcja dla użytkownika(jak uruchomić, jakie dane wprowadzać):

Po skompilowaniu programu wyświetli się okienko, w którym istnieje możliwość wyboru użytkownika lub pracownika. Po wybraniu jednej z dwóch opcji zależne od wyboru pokazuje się okienko, np. u Pracownika pokażą się dwie opcje: Dodaj Pokój, Zarządzaj Rezerwacjami. Po wybraniu opcji Dodaj pokój pokazuje się kolejne okienko w którym są informacje o pokoju, tj. Liczba gości, Piętro, Standard, Cena za noc, Nazwa. Jednak gdy chcemy wybrać opcję Zarządzaj rezerwacjami pokazuje się nam tabela ze wszystkimi rezerwacjami jakie klienci dokonali podana jest Data Od, Do, Nazwisko Klienta, Cena, Potwierdzona, Opłacona, po naciśnięciu jednej rezerwacji możemy nią zarządzać i jako pracownik mamy możliwość potwierdzenia rezerwacji i przycisk powrotu, który pozwala powrócić do tabelki z klientami. Po Wybraniu użytkownika pokazuje się okienko, w którym są do wyboru opcje Wyświetl ofertę lub Zarządzaj rezerwacjami. Po Wybraniu Wyświetl ofertę pokaże się okienko, w którym należy wybrać Data od, Data do, Cena od, Cena do, Standard, Goście, po dostosowaniu opcji klikamy przycisk wyszukaj, który przeszuka wszystkie pokoje i pokaże te dostępne na wybrane dane. Mamy możliwość wybrania pokoi, które są dostępne i następnie musimy nacisnąć przycisk Zarezerwuj, który zamknie okno i zarezerwuje dany pokój. Po wybraniu drugiej opcji Zarządzaj rezerwacjami ukaże się tabela, wskazująca które pokoje zarezerwowaliśmy i wszystkie dane które wybraliśmy. Po wybraniu pokoju możemy przedłużyć pobyt lub opłacić rezerwację.

Rysunek 1.Menu Główne.....	37
Rysunek 2.Pracownik Hotelu.....	37
Rysunek 3.Dodawanie Pokoi w hotelu przez pracownika.....	37
Rysunek 4.Spis Rezerwacji przez klientów u pracownika.....	38
Rysunek 5.Zarządzanie rezerwacją przez Pracownika.....	38
Rysunek 6.Menu Użytkownika.....	39
Rysunek 7.Parametry wolnych pokoi do wyszukania.....	39
Rysunek 8..Wszystkie wolne pokoje według parametrów.....	39
Rysunek 9.Menu Klienta.....	40
Rysunek 10..Spis rezerwacji danego klienta.....	40
Rysunek 11.Zarządzanie rezerwacją przez klienta.....	40