

Міністерство освіти і науки України

Національний технічний університет України “Київський політехнічний інститут”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Пояснювальна записка**

до модульної контрольної роботи

“Технології розробки корпоративних застосувань ”.

тему:

“ Сервіс замовлення авіаквитків ”

Виконав: студент 3 курсу

групи ІП-42 Фінчук О. В.

Перевірив: ас. Подрубайло О. О.

Кількість балів:\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2016

Зміст

[Зміст 2](#_Toc467634699)

[АНОТАЦІЯ 3](#_Toc467634700)

[вступ 4](#_Toc467634701)

[загальні положення 5](#_Toc467634702)

[Вимоги до системи 7](#_Toc467634703)

[Вимоги до системи 7](#_Toc467634704)

[Вимоги до функцій системи 7](#_Toc467634705)

[Вимоги до програмного забезпечення 7](#_Toc467634706)

[Вимоги до технічного забезпечення 7](#_Toc467634707)

[Моделі та засоби проектування 8](#_Toc467634708)

[Технології та засоби розробки системи 8](#_Toc467634709)

[Інформаційне забезпечення системи 8](#_Toc467634710)

[Програмне забезпечення 11](#_Toc467634711)

[Опис системи 15](#_Toc467634712)

[Опис структури системи та опис кожної підсистеми 15](#_Toc467634713)

[Опис функціонування системи 15](#_Toc467634714)

[Взаємодію підсистем можна поділити на дві категорії 15](#_Toc467634715)

[ВИСНОВКИ 17](#_Toc467634716)

[ЛІТЕРАТУРА 18](#_Toc467634717)

АНОТАЦІЯ

Дана модульна контрольна робота призначена для здобуття навичок розробки корпоративних застосувань за допомогою засобів JavaEE. В результаті повинна бути розроблена повноцінна система для замовлення авіаквитків. Також повинні бути розроблені модулі, які використовують JPA, JTA, JSF, EJB, JMS, JAAS, MDB.

вступ

У даній модульній контрольній роботі розробляється система для замовлення авіаквитків. Дана тема є актуальною, оскільки на сьогоднішній день не багато систем забезпечують користувачів повним набором функціоналу для зручності замовлень. Система, що розробляється повинна відповідати наступним вимогам:

1. Забезпечувати зручний пошук авіаквитків
2. Зберігати дані користувачів та дані для квитків
3. Забезпечувати швидку та безпомилкову взаємодію клієнта і сервера
4. Підтримувати реєстрацію нових користувачів
5. Регулювати доступ до елементів системи

загальні положення

Характеристика об’єкту автоматизації

Система «Сервіс замовлення авіаквитків» повинна реалізовувати такі бізнес-процеси:

1. Реєстрація та авторизація користувачів.
2. Управління квитками: перегляд, додавання, редагування та видалення.
3. Інтерфейс для купівлі квитків.
   1. Інтерфейс для пошуку квитків за критеріями.
   2. Інтерфейс для вибору квитків на плані літака
   3. Інтерфейс для підтвердження чи скасування квитків
4. Інтерфейс для управління замовленнями (для адміністратора / дизайнера).



Рисунок 1.1 – Use-case діаграм

Постановка задачі

Метою створення системи «Сервіс замовлення авіаквитків» є створення програмного застосунку, який дозволяє додавати адреси, маршрути, рейси та авіаквитки, а також надання користувачеві інтерфейсу для пошуку, вибору та придбання цих квитків. В результаті розробки системи мають бути досягнуті наступні показники:

1. Можливість реєстрації користувачів.
2. Безпечна аутентифікація та авторизація користувачів.
3. Можливість перегляду замовлених квитків.
4. Можливість перегляду, додавання даних для квитків.
5. Можливість пошуку рейсу за містами та датою.
6. Можливість вибору одночасно декілька вільних квитків на плані літака
7. Можливість підтвердження замовлення.
8. Можливість редагування особистих даних.

Вимоги до системи

## Вимоги до системи

Система «Сервіс замовлення авіаквитків» повинна виконувати такі операції:

1. Пошук авіаквитків.
2. Створення замовлення з авіаквитків.
3. Можливість управління даними для авіаквитків.

## Вимоги до функцій системи

Система «Сервіс замовлення авіаквитків» повинна забезпечувати такі функції:

1. Реєстрація та авторизація користувачів.
2. Управління даними: перегляд, додавання адрес, маршрутів, рейсів, та авіаквитків(для адміністратора).
3. Інтерфейс для пошуку рейсів за критеріями.
4. Інтерфейс для вибору квитків на плані літака.
5. Інтерфейс для купівлі авіаквитків.
6. Інтерфейс для перегляду куплених квитків
7. Інтерфейс для управління особистою інформацією

## Вимоги до програмного забезпечення

Для роботи системи необхідне таке програмне забезпечення:

1. Java 8 SE, Java EE 7.
2. MySql.
3. WildFly 10.0.

## Вимоги до технічного забезпечення

Для роботи системи необхідне таке технічне забезпечення:

1. Сервер зі встановленим програмним забезпеченням, що описано вище.
2. Браузер на стороні клієнта.

**Опис видів забезпечення системи**

## Моделі та засоби проектування

На етапі проектування системи «Сервіс замовлення авіаквитків» побудовані такі моделі:

1. Use case model.
2. ER-diagram.
3. Діаграма фізичної моделі даних.

Для побудови діаграм на етапі проектування використовувались такі CASE-засоби:

1. Enterprise Architect.
2. Intellij IDEA.
3. Workbench MySQL.

## Технології та засоби розробки системи

При розробці системи «Сервіс замовлення авіаквитків» були використані такі технології:

* Java Servlet API
* Java Persistence API
* Java Server Faces
* Enterprise Java Bean
* Java Authentication and Authorization Service
* AJAX
* Java Message Service
* Message Driver Bean
* Log4j
* Bootstrap

При розробці системи «Сервіс замовлення авіаквитків» були використані такі засоби:

1. Intellij IDEA.
2. MySQL

## Інформаційне забезпечення системи

При організації інформаційного забезпечення системи «Сервіс замовлення авіаквитків» були використана база даних MySQL. Для розробки її схеми була використана технологія JPA з Hibernate реалізацією. На діаграмі нижче наведено ER-модель бази даних.

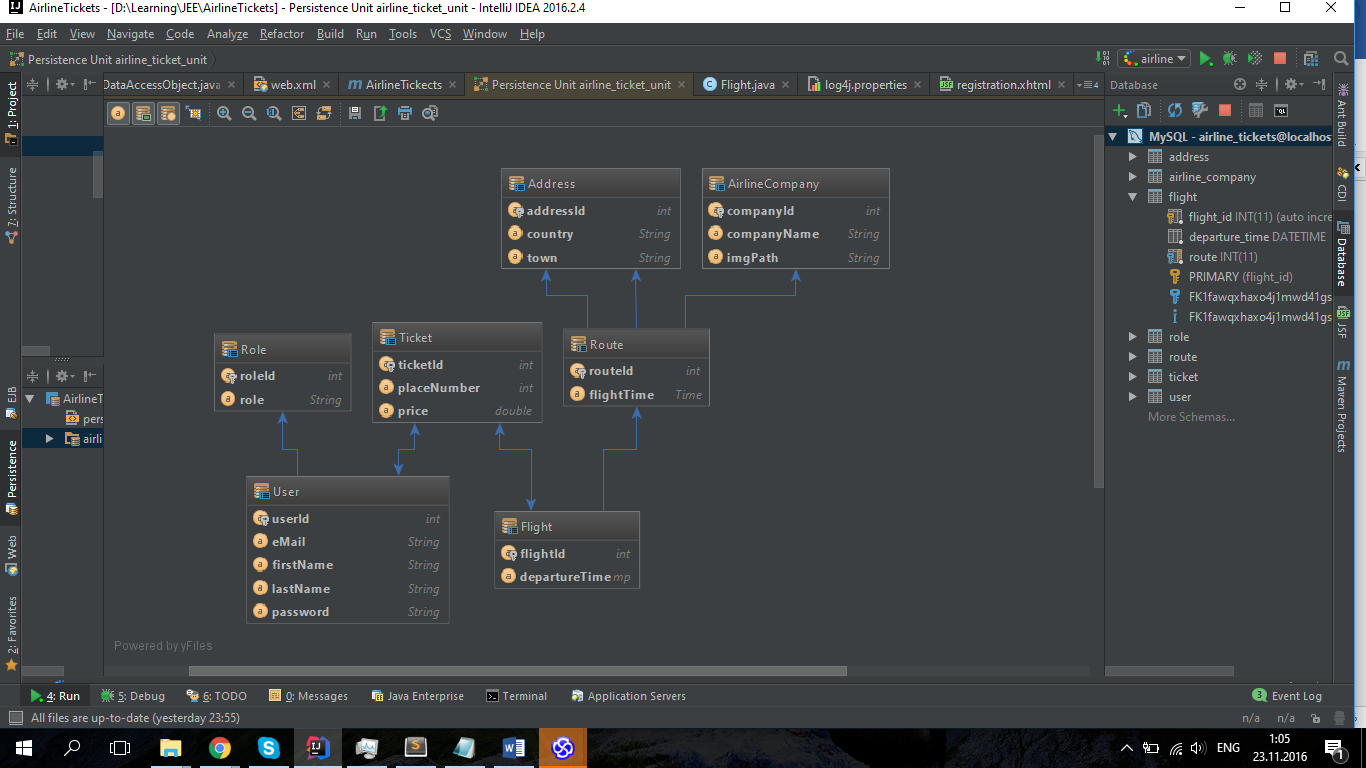


Рисунок 1.2 – ER-модель бази даних

Відповідно, база даних, що була створена технологією JPA, може бути представлена наступною фізичною моделлю.

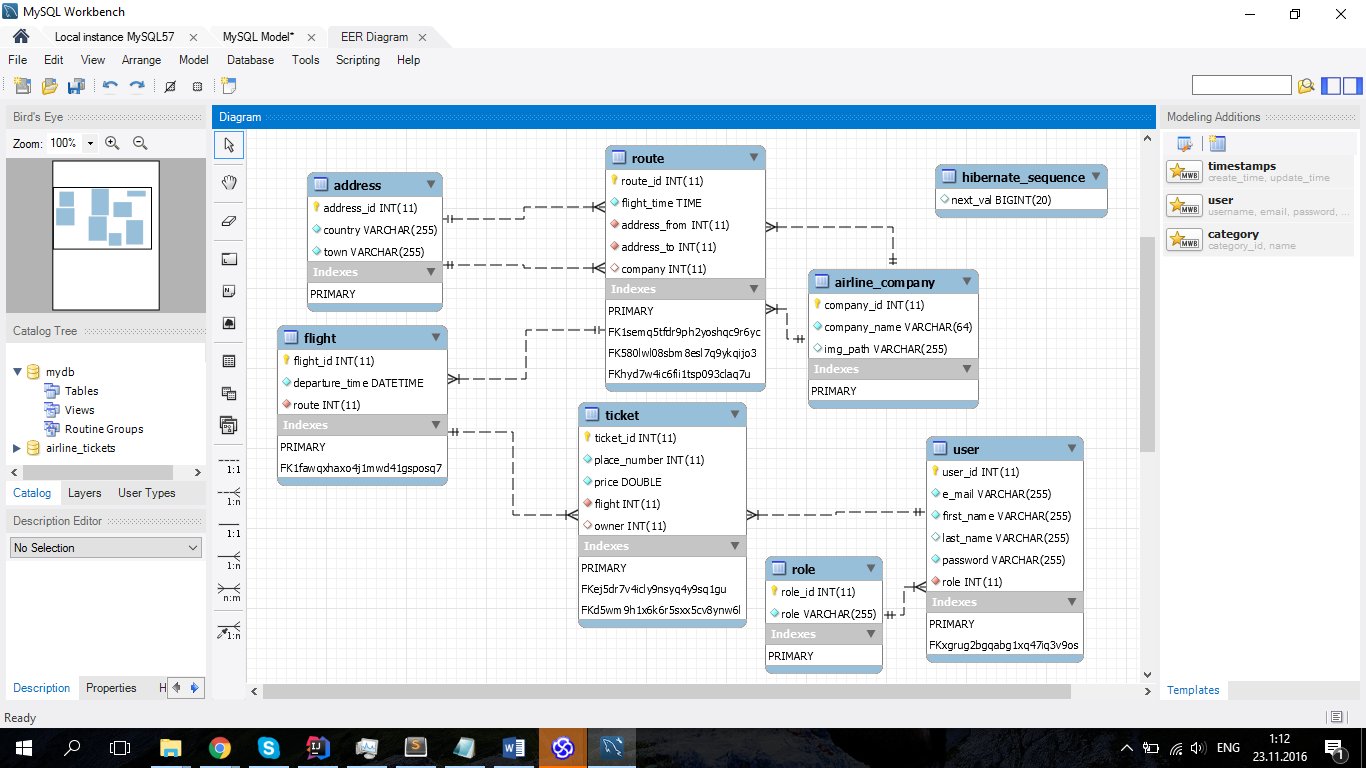


Рисунок 1.3. Фізична модель бази даних

Взаємодія між клієнтом та сервером організована за допомогою протоколу HTTP. Взаємодія компонентів серверної частини реалізована з використанням технологій JSF, EJB, JMS та MDB.

## Програмне забезпечення

Архітектура системи «Сервіс замовлення авіаквитків» має такі частини:

1. Converters
2. Entities
3. Jpa\_dao
4. Managing\_beans
5. webapp

Далі наведено опис кожної з частин:

**Entities**

В цій частині описані сутності (Entity**)** у вигляді JPA Entity-класів. Всі вони мають структуру POJO. Далі наведно перелік всіх сутностей:

1. Address.java
2. AirlineCompany.java
3. Flight.java
4. Role.java
5. Route.java
6. Ticket.java
7. User.java

**Jpa\_dao**

В цій частині описані класи, які забезпечують CRUD-операції для JPA Entity-класів. Всі ці класи реалізують методи абстрактного класу JPADataAccessObject.java:

1. ArrayList<T> getObjectsByCondition(String condition) - повертає список об’єктів, які відповідають заданій умові.
2. ArrayList<T> getAllObjects() - повертає список усіх об’єктів.
3. T getObjectById(int id) – повертає об’єкт із заданим ідентифікатором.
4. void addObject(T obj) – додає новий об’єкт.
5. void updateObject (T obj) – обновляє даний об’єкт.
6. void deleteObject (T obj) – видаляє даний об’єкт.
7. void deleteObjectById(int id) – видаляє об’єкт із заданим ідентифікатором.
8. void deleteAll() – видаляє всі об’єкти.

Всі ці операції реалізовані в generic класі JPADataAccessObject.java. Кожному Entity відповідає клас з jpa\_dao, який конкретизує generic клас JPADataAccessObject.java:

1. JPADAOAddress.java
2. JPADAOAirlineCompany.java
3. JPADAOFlight.java
4. JPADAORole.java
5. JPADAORoute.java
6. JPADAOTicket.java
7. JPADAOUser.java

**Managing Bean**

В цій частині описані класи, які забезпечують логіку системи. Вони використовують класи з jpa\_dao та реалізують весь функціонал, необхідний для роботи системи.

1. AddressBean – клас для доступу до аеропортів. Основні методи:
   1. void updateCurrAdresses()– обновляє список із бази даних.
   2. Collection<Address> getAddresses() – повертає всі поточні адреси.
   3. void addNewAdress() – додає нову адресу у базу даних та обновляє список
2. AirlineCompanyBean – клас для доступу до авіакомпаній. Основні методи:
   1. void updateCurrCompanies ()– обновляє список із бази даних.
   2. Collection<Company> getCompanies() – повертає всі поточні компанії.
   3. void addNewCompany () – додає нову авіакомпанію у базу даних та обновляє список
3. AuthorizationBean – клас, що забезпечує авторизаціюреєстрацію. та вихід з облікового запису користувача.
   1. void logout() – виконує операцію виходу користувача із системи.
   2. void logIn(String username) – виконує операцію входу
   3. String signUp() – вносить нового користувача у базу
   4. public static String getEncodedPassword(String password) – повертає захешований за алгоритмом SHA-256 пароль
4. FlightBean – клас для доступу до авіарейсів. Основні методи:
   1. void updateCurrFlights ()– обновляє список із бази даних.
   2. Collection<Flight> getFlights() – повертає всі поточні авіарейси.
   3. void addNewFlight () – додає новий авіарейс у базу даних генерує три класи квитків, заносить їх у базу та обновляє список
   4. int getOpenTickets(Flight flight) – повертає кількість вільних місць для рейсу
5. RouteBean – клас для доступу до маршрутів. Основні методи:
   1. void updateCurrRoutes ()– обновляє список із бази даних.
   2. Collection<Route> getRoutes() – повертає всі поточні маршрути.
   3. void addNewRoute () – додає новий маршрут у базу даних
6. OrderListUpdater – клас для асинхронного оновлення списку замовлень.
   1. public void onMessage(Message message) – метод для асинхронного оновлення замовлення по коду, переданому в message.
7. TicketBean – клас для роботи із замовленням авіаквитків. Реалізує пошук авіаквитків та вибір їх на плані літака. Забезбечує Основні методи:
   1. void swapCurrTicket(Ticket t) – міняє стан вибраного квитка, додає/видаляє його у список вибраних квитків
   2. boolean isChosen(Ticket t) – перевіряє стан квитка
   3. String updateCurrFlights() обновляє список знайдених квитків на основі заданих параметрів
   4. ArrayList<Ticket> getTicketsLuxFirst()
   5. ArrayList<Ticket> getTicketsLuxSecond()
   6. ArrayList<Ticket> getTicketsFirst()
   7. ArrayList<Ticket> getTicketsSecond()
   8. ArrayList<Ticket> getTicketsThird() – методи для вибору квитків за класом
   9. String saveCurrFlight(Flight c)- метод зберігає вибраний рейс для подальшої роботи з ним
   10. Double totalPrice() – сумарна вартість вибраних квитків.
   11. String buyTickets() – реалізує покупку вибраних квитків для користувача та зберігає дані у базі, тепер ці квитки недоступні для інших.
8. UserBean – клас для доступу до користувача.
   1. public void updateUser() – знаходить поточного користувача у базі
   2. public void saveUser() – зберігає нові дані користувача
   3. public String edit() – надає можливість редагувати дані користувача

**Converters**

В дані частині описані класи, які конвертують об’єкти для передачі через запити

1. AddressConvert
2. CompanyConvert
3. DateConvert
4. DateTimeConvert
5. RouteConvert
6. TimeConvert

**webapp**

В цій частині описані веб-форми, які забезпечують відображення елементів системи. Всі вони є JSF-формами, деякі використовують технологію AJAX.

Опис системи

## Опис структури системи та опис кожної підсистеми

Система «Сервіс замовлення авіаквитків» складається з таких підсистем:

1. Реєстрація та авторизація користувачів – забезпечує реєстрацію, вхід та вихід із системи.
2. Управління даними для авіаквитків – створення, перегляд данихя авіаквитків.
3. Замовлення авіаквитків – пошук рейсів, вибір місць, замовлення обраних квитків.

## Опис функціонування системи

1. Реєстрація та авторизація користувачів
   1. Реєстрація користувачів.
   2. Аутентифікація користувачів.
   3. Авторизація користувачів.
   4. Вихід із системи.
2. Управління даними для квитків
   1. Перегляд даних.
   2. Створення нових даних на основі поточних.
3. Замовлення авіаквитків
   1. пошук рейсів.
   2. вибір місць.
   3. замовлення обраних квитків.

Взаємодію підсистем можна поділити на дві категорії.

1. Взаємодія між підсистемою «Реєстрація та авторизація користувачів» та іншими підсистемами. Реалізується через JSF форми та механізм JaaS: користувач звертається до підсистеми, потрапляє до підсистеми «Реєстрація та авторизація користувачів», виконує вхід або реєстрацію. Подібним чином реалізовано і вихід з системи.
2. Взаємодія між підсистемою «Управління даними для квитків» та підсистемою «Замовлення авіаквитків» реалізована за допомогою ін’єкцій EJB які зберігають свої стани, та змінюються динамічно у підсистемах.

ВИСНОВКИ

Після процесу проектування класів та визначення взаємодії між ними, а також програмної реалізації нашої системи, ми можемо стверджувати, що розроблена система відповідає всім вищезазначеним вимогам та може у повній мірі задовольнити функціонал, що був описаний у третьому розділі даної роботи. Під час виконання модульної контрольної роботи було засвоєно такі технології:

1. Java Servlet API
2. Java Persistence API
3. Java Server Faces
4. Enterprise Java Bean
5. Java Authentication and Authorization Service
6. AJAX
7. Java Message Service
8. Message Driver Bean
9. Log4j
10. Bootstrap

Ціль роботи була досягнута.

ЛІТЕРАТУРА

1. JSF 2.0 Tutorials. - http://www.coreservlets.com/JSF-Tutorial/jsf2/
2. JSF for nonbelievers: The JSF application lifecycle. – http://www.ibm.com/developerworks/library/j-jsf2
3. Mike Keith, Merrick Schincariol. Pro JPA 2: Mastering the JavaTM Persistence API. – APress: 2009. – 538 p.
4. Bruce W. Perry. Java Servlet & JSP Cookbook. - O'Reilly. - 2004. - 746p.
5. Bob McCune. Exploring the Java Persistence API. -http://www.intertech.com/resource/usergroup/Exploring%20the%20JPA.pdf
6. J2EE BluePrints. Designing Enterprise Applications with the J2EE Platform.. http://www.adobe.com/support/jrun/documentation/pdf/j2ee\_ent\_app\_design
7. Deby Panda, Reza Rahman, Derek Lane. EJB 3 in Action. – Manning: 2007.-677p.