Національний технічний університет України

«Київський політехнічний інститут»

ННК «ІПСА»

Кафедра Математичних наук системного аналізу

Київ – 2021

**База даних управління піцерією**

*Курсова робота*  
предмет  
«*Бази даних*»

Виконав:  
Соколенко С.С.  
група КА-96

Керівник:

Афанасьєва І.В.

\_\_\_\_\_\_\_\_\_\_ «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2021 р.

Захищено з оцінкою: \_\_\_\_

Національний технічний університет України “КПІ” ННК “ІПСА”

Кафедра: математичні методи системного аналізу

Предмет: бази даних

Спеціальність Штучний інтелект

Курс III Група КА-96 Семестр 1

ЗАВДАННЯ

на курсову роботу студента

Соколенко Сергій Сергійовича

(прізвище, ім’я, по батькові)

1. Тема роботи:

База даних управління піцерією

2. Строк здачі студентом закінченого проекту (роботи)

3. Вихідні дані до проекту (роботи)

Java, Spring Boot, MySQL

Технічна література

4. Зміст розрахунково – пояснювальної записки (перелік питань, які підлягають розробці)

Розробка алгоритму роботи програми

Архітектура програми

5. Перелік графічного матеріалу ( з точним зазначенням обов’язкових креслень)

6. Дата видачі завдання

**ЗМІСТ**

[ВСТУП 4](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052473)

[РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ 5](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052474)

[1.1. Обґрунтування вибору засобів розробки 5](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052476)

[1.2. Уточнена постановка задачі на розробку програмного забезпечення 6](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052477)

[РОЗДІЛ 2. ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ’ЯЗКУ ЗАДАЧІ ТА РОЗРОБКА АЛГОРИТМУ](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052478) 7

[2.1. Інфологічна модель бази даних](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052480) 7

[РОЗДІЛ 3. ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052481) 9

[3.1. Реалізація зв’язку бази даних із програмним продуктом](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052482) 9

[3.2. Інструкція користувача 1](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052483)0

[ВИСНОВКИ](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052484) 13

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052485) 15

[ДОДАТОК А. ЛІСТИНГ СТВОРЕННЯ USER CONTROLER 1](file:///C:\Users\serge\Downloads\Telegram%20Desktop\Курсовая%20КА-96%20Колесник.docx#_Toc60052486)

# ВСТУП

Основне завдання інформаційного ресурсу – це надання максимально повної та актуальної інформації з певної тематики. На інформаційних сайтах з великою кількістю різного контенту з метою спрощення навігації та пошуку потрібної інформації реалізуються тематичні розділи, пошук по сайту і інші функції.

Метою цієї курсової роботи є поглиблення та закріплення набутих теоретичних знань з дисципліни «Бази даних», а також розробка бази даних для сайту та конкретних відеоігор.

Робота складається зі вступу, трьох розділів та додатку.

Програмне забезпечення було реалізоване за допомогою наступних засобів:

* Мова програмування Java;
* Spring Boot;
* MySQL DBMS;
* MySQLWorkbench.

При виконанні роботи були використано середовище IntelliJ IDEA. Для пошуку джерел інформації було використано мережу Інтернет,   
для малювання DFD-діаграм було використано Figma, а для малювання ERD-діаграм – MySQLWorkbench.

# **РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ**

## **1.1. Обґрунтування вибору засобів розробки**

Для вирішення даної задачі була обрана мова Java. Java є мовою програмування і платформою обчислень, яка була вперше випущена Sun Microsystems в 1995 році. Java відрізняється швидкістю, високим рівнем захисту і надійністю.

В якості основного інструменту розробки було обрано Spring Boot – це корисний проект, метою якого є спрощення створення додатків на основі Spring. Він дозволяє найбільш простим способом створити web-додаток полегшуючи процес розробки дозволяючи користувачу не виконувати постійно монотонні дії, вимагаючи від розробників мінімум зусиль для налаштування і написання коду .

В якості інструменту розробки було використано IDE IntelliJ IDEA. IntelliJ IDEA – це інтегрована середа розробки програмного забезпечення, розроблена компанією JetBrains. Програма має зручний графічний інтерфейс, а також багатий функціонал який полегшує виконання роботи.

Оскільки на сайті відображається велика кількість інформації, то треба використовувати відповідний інструмент – базу даних. За СУБД була обрана MySQL – реляційна система управління базами даних, адже саме її ми проходили протягом курсу з баз даних.

Обрані засоби розробки підходять для реалізації усіх задач, що передбачені завданням курсової роботи.

## **1.2. Уточнена постановка задачі на розробку програмного забезпечення**

Розроблюване програмне забезпечення передбачає реалізацію ігрових баз даних. На підставі аналізу інформаційних джерел, поставленої задачі та програм-аналогів, можна сформувати уточнену постановку задачі при розробці:

1. Обґрунтувати вибір інструментів розробки;
2. Створити відповідні таблиці у базі даних, які необхідні для роботи із сайтом.
3. Розробити бекенд частину програми, підключити віддалену базу даних та налагодити отримання та додавання туди даних;
4. Розробити зручний інтерфейс користувача та зв’язати між собою бекенд та фронтенд;
5. Дослідити якість роботи програми на контрольних прикладах, перевірити коректність отриманих даних;
6. Створити опис програми;
7. Визначити перспективи удосконалення розробки.

# **РОЗДІЛ 2.** **ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ’ЯЗКУ ЗАДАЧІ ТА** **РОЗРОБКА АЛГОРИТМУ**

## **2.1. Інфологічна модель бази даних**

У програмному додатку використовується одна база даних - coursedb. В ній зберігаються такі таблиці:

* users – інформація про користувачів (юзернейм, та додаткова інформація);
* user\_role – інформація о ролях користувачів.
* product\_size – інформація о різних розмірностях товару
* product\_group – інформація о різних видах продуктів
* product – інформація про назву товару та групи до якої він належить
* product\_price – інформація о ціні яка залежить від комбінації розміру та продукту
* order – загальна інформація про замовлення (користувач який зробив замовлення, статус замовлення та інше)
* status – інформація про те у якому стані замовлення на даних момент

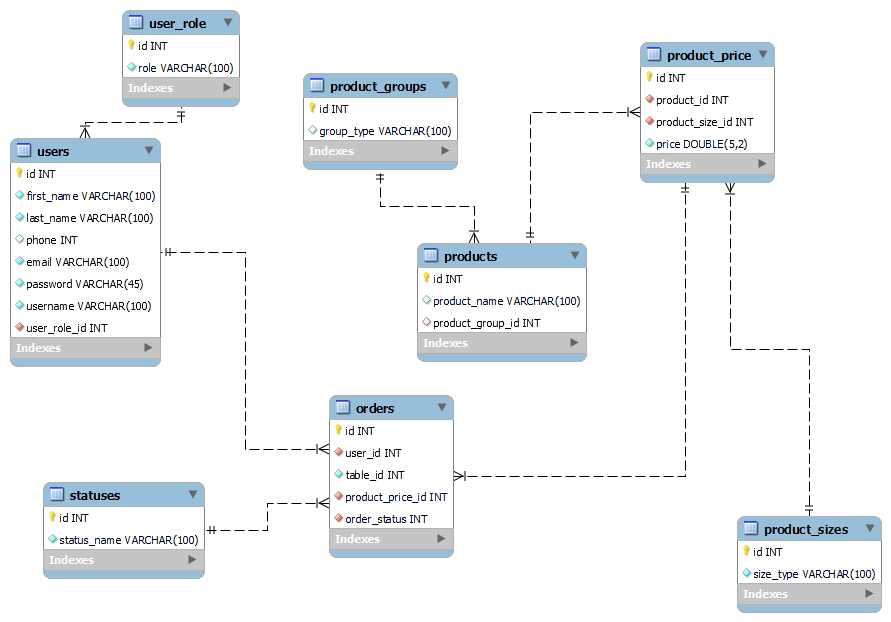
Для мого проекту інфологічна модель виглядає наступним чином (рис. 1): 

Рис. 1 ERD Diagram

# **РОЗДІЛ 3.** **ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ**

## **3.1. Реалізація зв’язку бази даних із програмним продуктом**

Для зв’язування бази даних із бізнес-частиною програмного додатку було використано Hibernate, а для створення репозиторіїв – Spring Data JPA. Spring Data JPA – це частина великого сімейства Spring Data, яка дозволяє легко створювати репозиторії на основі JPA. Цей модуль стосується розширеної підтримки рівнів доступу до даних на основі JPA. Це спрощує створення програм, що використовують бази даних.

Приклад реалізації:

@Repository  
public interface UserRepo extends JpaRepository<User, Long> {  
}

Цей код означає, що доступ до бази даних буде відбуватись через інтерфейс UserRepo.

Оскільки модуль JPA підтримує визначення запиту вручну як String або від імені методу [8], написання типових запитів до бази даних відбувається за допомогою методів репозиторію.

@Service  
public class UserService {  
 private final UserRepo userRepo;  
 private final UserRoleRepo userRoleRepo;  
  
 public List<User> getUsers() {  
 return userRepo.findAll();  
 }  
  
 public User saveUser(UserDto newUser) {  
 User user=User.*builder*()  
 .firstName(newUser.getFirstName())  
 .lastName(newUser.getLastName())  
 .phone(newUser.getPhone())  
 .email(newUser.getEmail())  
 .password(newUser.getPassword())  
 .username(newUser.getUsername())  
 .userRole(userRoleRepo.findById(newUser.getUserRoleId()).orElseThrow(UserRoleNotFoundException::new))  
 .build();  
 return userRepo.save(user);  
 }

public String deleteUserById(Long id) {  
 if (userRepo.findById(id).isPresent()) {  
 userRepo.deleteById(id);  
 return "User was successfully deleted";  
 }  
 throw new UserNotFoundException();  
}

## 

## **3.2. Інструкція користувача**

Оскільки не було реалізовано фронтенд частину проекту для перевірки роботи проекту скористаємося програмою Postman.

Тестування проведемо з таблицею Product Price.

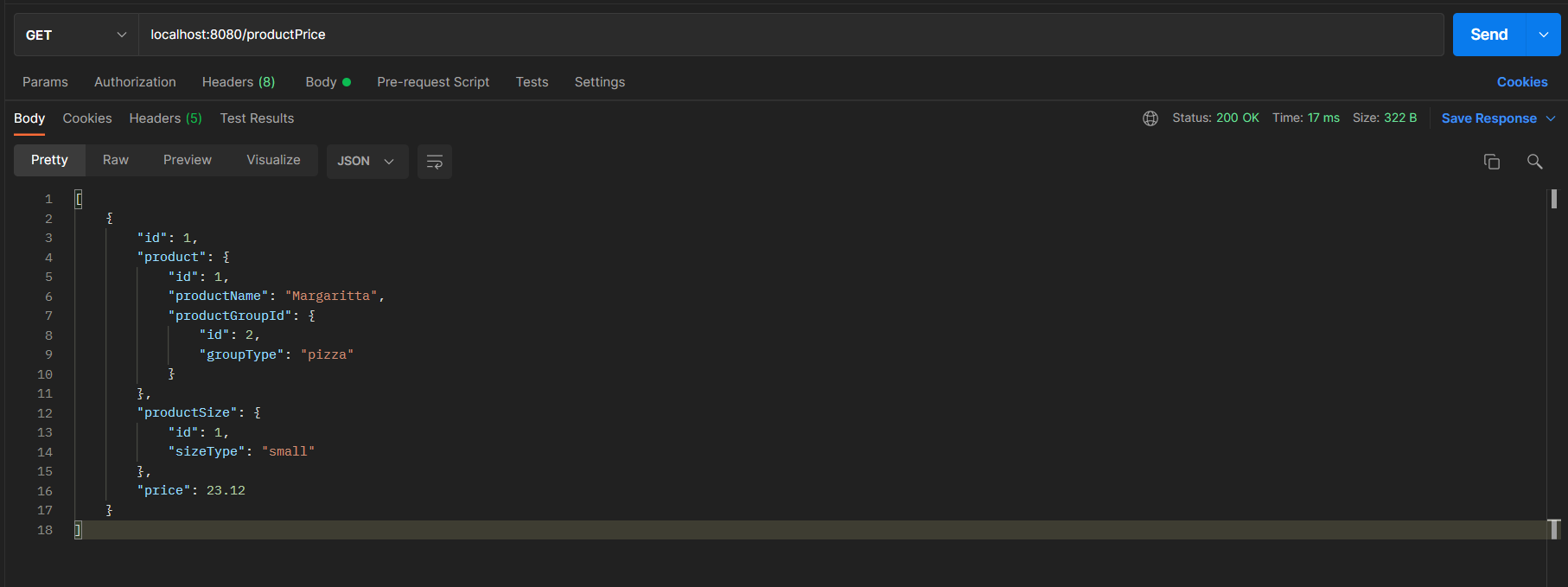


Рис.2 Get запрос

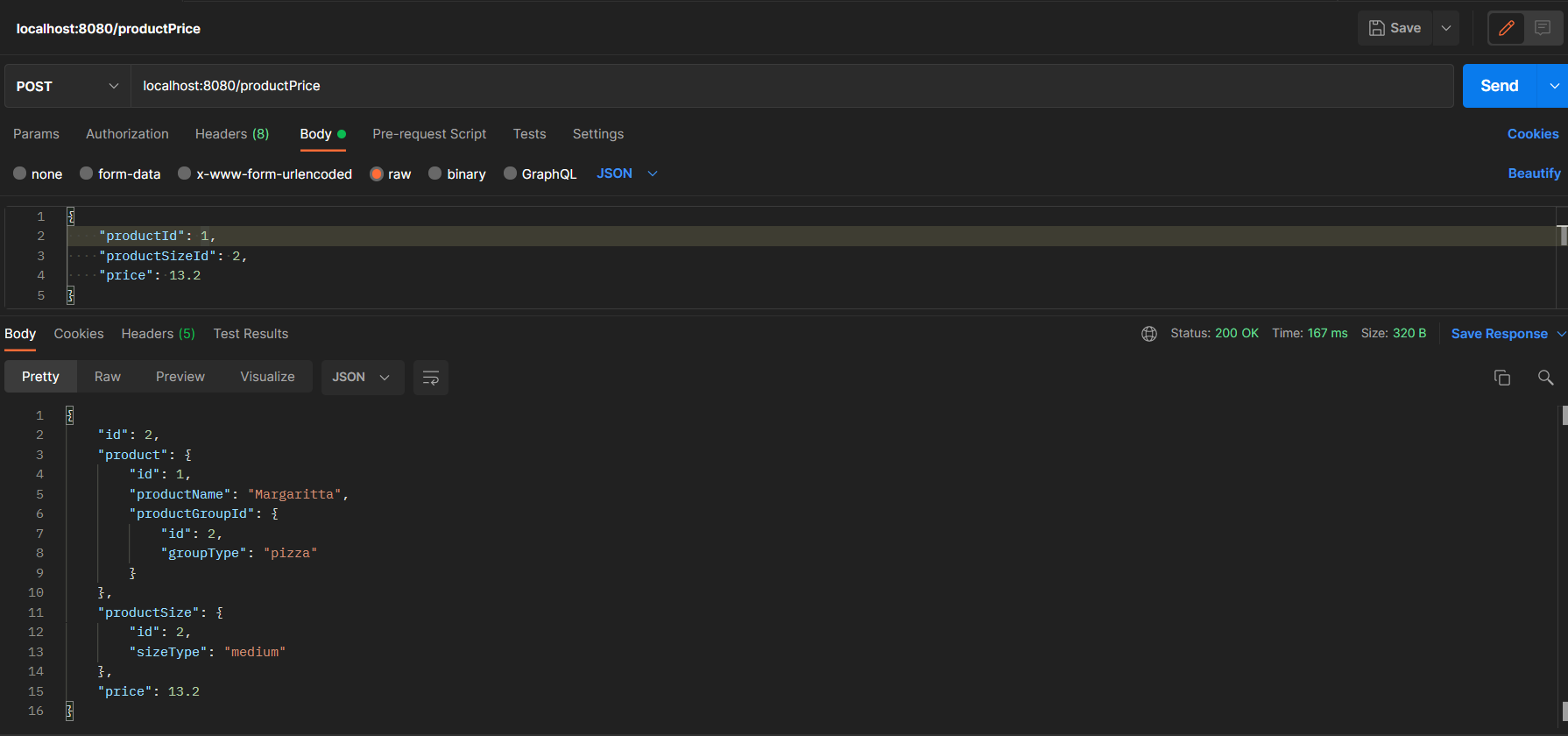


Рис. 3 Post Запрос

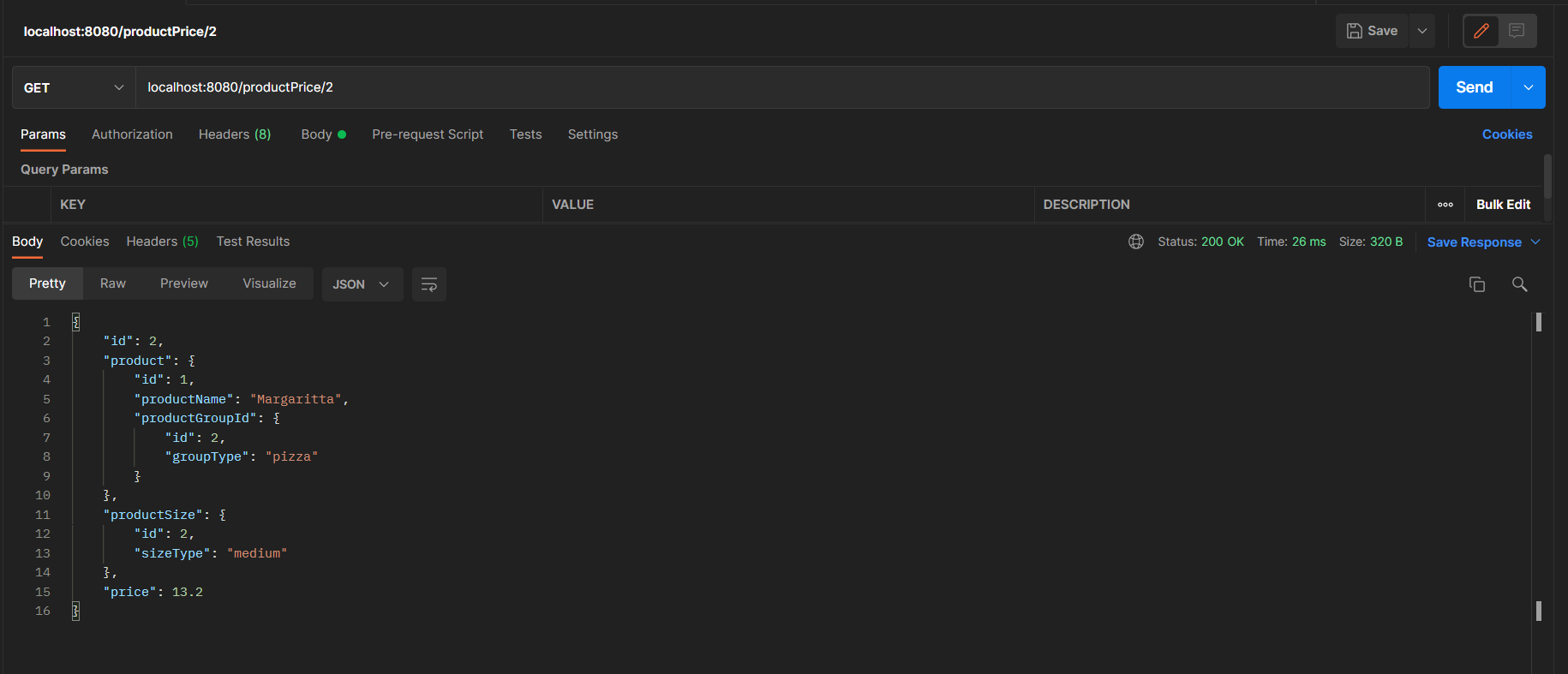


Рис. 4 Get Запрос по Id

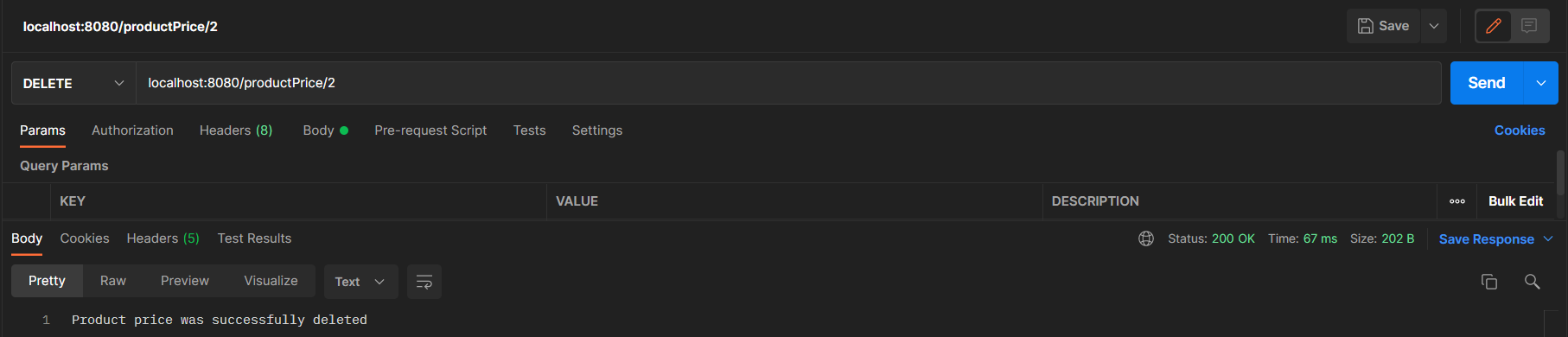


Рис.5 Delate Запрос по Id

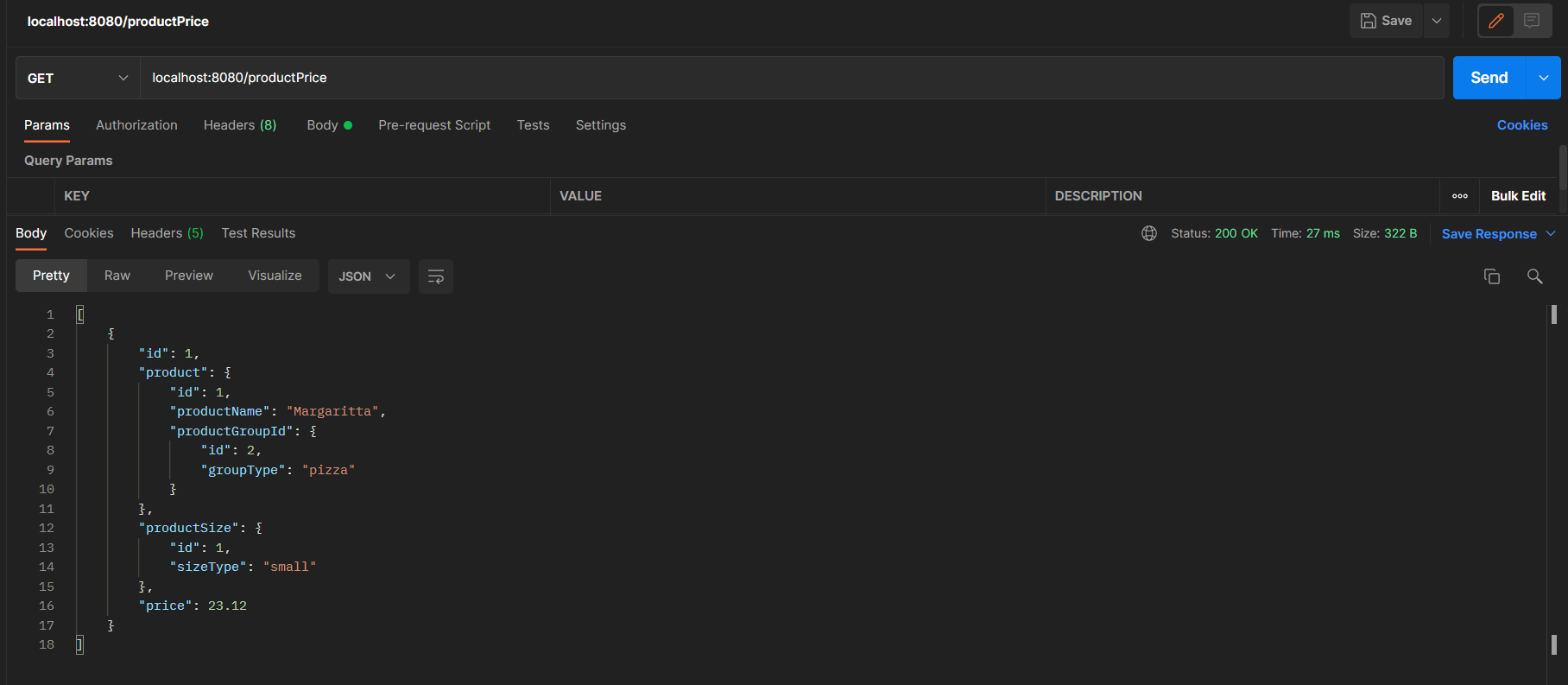


Рис 6. Таблица product price после команды Delate

# **ВИСНОВКИ**

В результаті виконання курсової роботи було розроблено базу даних для керування даними піцерії. Також наповнено базу даних, знайдено спосіб відображення даних, отриманих з бази та зміни цих даних.

У процесі виконання цієї роботи я поліпшив свої навички із розробки веб-додатків із використанням такого технологічного стеку:

* Мова програмування Java;
* Spring Boot;
* MySQL DBMS;
* MySQLWorkbench.

У курсовій роботі було досліджено роботу реляційних баз даних, а також їх взаємодію із користувачами через певні користувацькі інтерфейси, наприклад, веб-ресурс.

У першому розділі було розглянуто інструментів розробки. Також було сформульовано уточнену постановку даної задачі, яка включає в собі пукти, описані вище.

У другому розділі було створено діаграму потоків даних та інфологічну модель бази даних.

У третьому розділі було виконано опис розробленої програми. Також було створено інструкцію користувача.

Отже, програма задовольняє вимогам поставленого завдання. В подальшому вона може бути вдосконалена поліпшенням інтерфейсу та додаванням більшої кількості баз даних для зручнішої реалізації необхідних нам методів.

**ДОДАТОК А: Лістінг створення бази даних**

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `coursedb` DEFAULT CHARACTER SET utf8 ;

USE `coursedb` ;

-- -----------------------------------------------------

-- Table `coursedb`.`product\_groups`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`product\_groups` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`product\_groups` (

`id` INT NOT NULL AUTO\_INCREMENT,

`group\_type` VARCHAR(100) CHARACTER SET 'ascii' NULL DEFAULT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `group\_type\_UNIQUE` (`group\_type` ASC) VISIBLE)

ENGINE = InnoDB

AUTO\_INCREMENT = 5

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`products`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`products` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`products` (

`id` INT NOT NULL AUTO\_INCREMENT,

`product\_name` VARCHAR(100) NULL DEFAULT NULL,

`product\_group\_id` INT NULL DEFAULT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `product\_name\_UNIQUE` (`product\_name` ASC) VISIBLE,

UNIQUE INDEX `product\_group\_id\_UNIQUE` (`product\_group\_id` ASC) VISIBLE,

CONSTRAINT `products`

FOREIGN KEY (`product\_group\_id`)

REFERENCES `coursedb`.`product\_groups` (`id`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB

AUTO\_INCREMENT = 2

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`product\_sizes`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`product\_sizes` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`product\_sizes` (

`id` INT NOT NULL AUTO\_INCREMENT,

`size\_type` VARCHAR(100) NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `size\_type\_UNIQUE` (`size\_type` ASC) VISIBLE)

ENGINE = InnoDB

AUTO\_INCREMENT = 3

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`product\_price`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`product\_price` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`product\_price` (

`id` INT NOT NULL AUTO\_INCREMENT,

`product\_id` INT NOT NULL,

`product\_size\_id` INT NOT NULL,

`price` DOUBLE(5,2) NOT NULL,

PRIMARY KEY (`id`),

INDEX `product\_price\_idx` (`product\_id` ASC) VISIBLE,

INDEX `product\_price\_fk\_idx` (`product\_size\_id` ASC) VISIBLE,

CONSTRAINT `product\_price`

FOREIGN KEY (`product\_id`)

REFERENCES `coursedb`.`products` (`id`),

CONSTRAINT `product\_price\_fk`

FOREIGN KEY (`product\_size\_id`)

REFERENCES `coursedb`.`product\_sizes` (`id`))

ENGINE = InnoDB

AUTO\_INCREMENT = 3

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`user\_role`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`user\_role` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`user\_role` (

`id` INT NOT NULL AUTO\_INCREMENT,

`role` VARCHAR(100) NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `role\_UNIQUE` (`role` ASC) VISIBLE)

ENGINE = InnoDB

AUTO\_INCREMENT = 5

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`users`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`users` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`users` (

`id` INT NOT NULL AUTO\_INCREMENT,

`first\_name` VARCHAR(100) NOT NULL,

`last\_name` VARCHAR(100) NOT NULL,

`phone` INT NULL DEFAULT NULL,

`email` VARCHAR(100) NOT NULL,

`password` VARCHAR(45) NOT NULL,

`username` VARCHAR(100) NOT NULL,

`user\_role\_id` INT NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `email\_UNIQUE` (`email` ASC) VISIBLE,

UNIQUE INDEX `users\_username\_uindex` (`username` ASC) VISIBLE,

UNIQUE INDEX `phone\_UNIQUE` (`phone` ASC) VISIBLE,

INDEX `users\_fk` (`user\_role\_id` ASC) VISIBLE,

CONSTRAINT `users\_fk`

FOREIGN KEY (`user\_role\_id`)

REFERENCES `coursedb`.`user\_role` (`id`))

ENGINE = InnoDB

AUTO\_INCREMENT = 2

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`statuses`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`statuses` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`statuses` (

`id` INT NOT NULL AUTO\_INCREMENT,

`status\_name` VARCHAR(100) NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `statuses\_status\_name\_uindex` (`status\_name` ASC) VISIBLE)

ENGINE = InnoDB

AUTO\_INCREMENT = 7

DEFAULT CHARACTER SET = utf8mb3;

-- -----------------------------------------------------

-- Table `coursedb`.`orders`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `coursedb`.`orders` ;

CREATE TABLE IF NOT EXISTS `coursedb`.`orders` (

`id` INT NOT NULL AUTO\_INCREMENT,

`user\_id` INT NOT NULL,

`table\_id` INT NOT NULL,

`product\_price\_id` INT NOT NULL,

`order\_status` INT NOT NULL,

PRIMARY KEY (`id`),

INDEX `orders\_idx` (`product\_price\_id` ASC) VISIBLE,

INDEX `orders\_fk\_idx` (`user\_id` ASC) VISIBLE,

INDEX `orders\_status\_fk` (`order\_status` ASC) VISIBLE,

CONSTRAINT `orders`

FOREIGN KEY (`product\_price\_id`)

REFERENCES `coursedb`.`product\_price` (`id`),

CONSTRAINT `orders\_fk`

FOREIGN KEY (`user\_id`)

REFERENCES `coursedb`.`users` (`id`),

CONSTRAINT `orders\_status\_fk`

FOREIGN KEY (`order\_status`)

REFERENCES `coursedb`.`statuses` (`id`))

ENGINE = InnoDB

AUTO\_INCREMENT = 2

DEFAULT CHARACTER SET = utf8mb3;