|  |  |  |  |
| --- | --- | --- | --- |
| МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  **Кафедра програмних систем і технологій**  Дисципліна  **«**Хмарні технології**»**  **Індивідуальне завдання № 1**  «СТВОРЕННЯ БАЗИ ДАНИХ SQL В СЕРЕДОВИЩІ AZURE» | | | |
| **Виконав:** | Гоша Давід | **Перевірив**: |  |
| Група | ІПЗ-43 | Дата перевірки |  |
| Форма навчання | денна | Оцінка |  |
| Спеціальність | 121 |
| 2022 | | | |

**Мета роботи:** Дослідити процес створення та управління базами даних SQL в хмарному середовищі Azure, використовуючи Azure Portal та Azure CLI. Освоїти створення еластичних пулів та керуючих екземплярів SQL для оптимізації ресурсів та управління навантаженням.

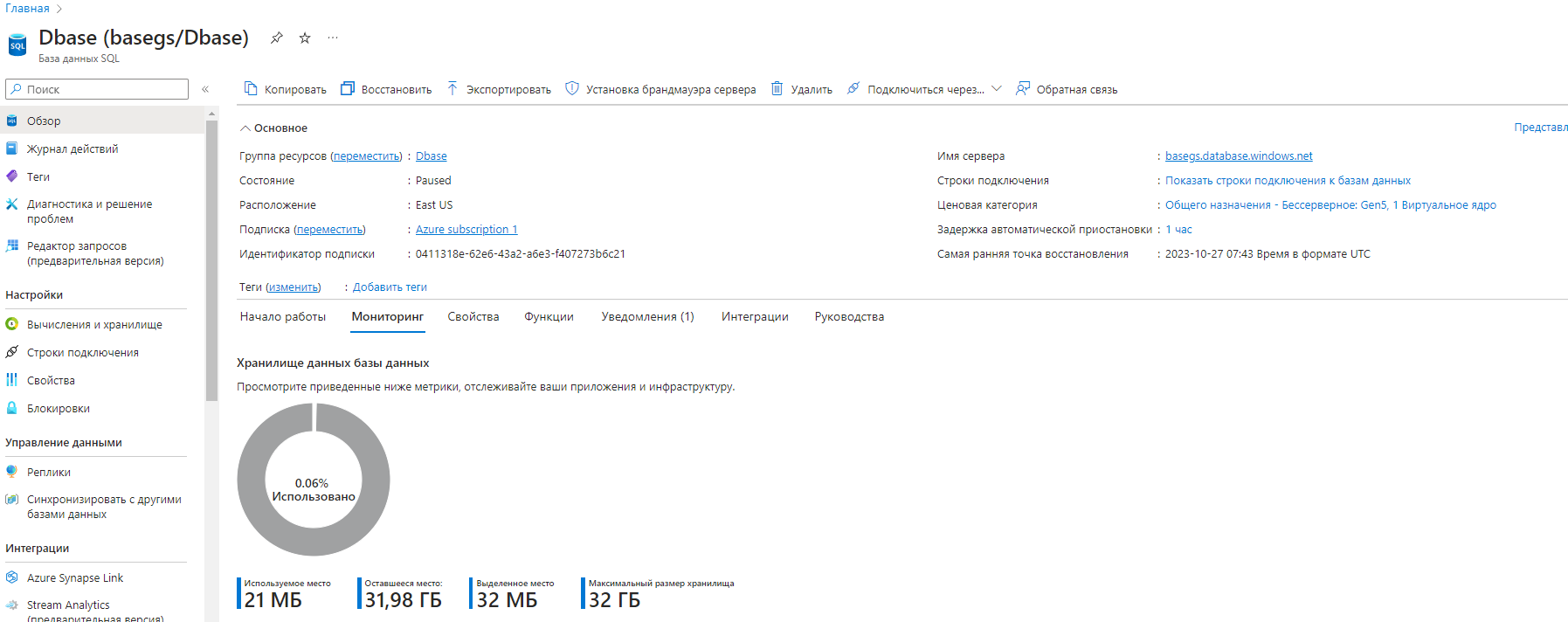
**Завдання роботи:** Розробити компоновану програму з використанням бази даних Azure SQL, створити RESTful API для управління даними в базі за допомогою Flask, SQLAlchemy та розгорнути сервіс на локальному сервері.

**Оформлення результатів роботи:**

**1. Використання Azure Portal для створення інстансу бази даних SQL:**

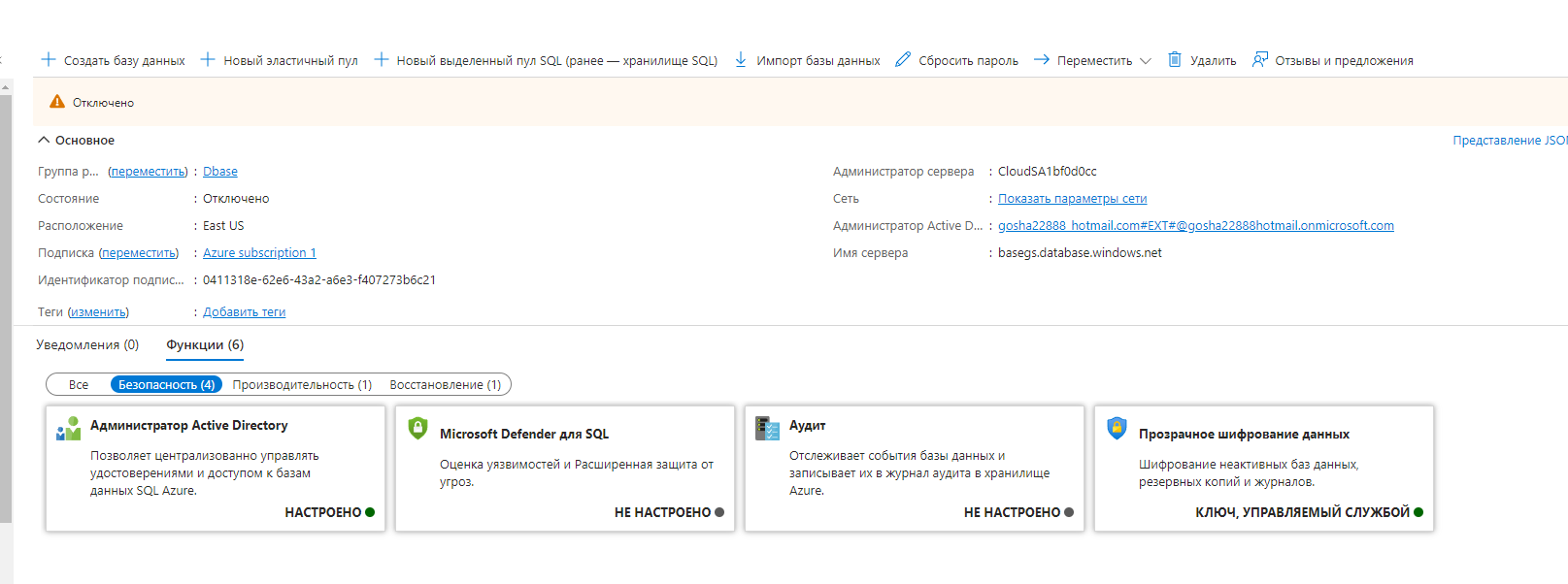
Для початку роботи з базою даних SQL в Azure, я виконав наступні кроки:

* **Авторизація в Azure Portal:** Увійшовши в систему за допомогою моїх облікових даних, я отримав доступ до панелі керування Azure.
* **Створення нової бази даних:** В розділі "Створити ресурс", я знайшов опцію "SQL Database" та ініціював процес створення нової бази даних.
* **Конфігурація інстансу:** Вказав назву бази даних, обрав групу ресурсів, а також створив новий SQL сервер для розміщення бази даних.
* **Визначення параметрів:** Обрав план сервісу та розмір обчислювальних ресурсів, що відповідають потребам мого застосунку.
* **Перевірка та запуск:** Після перевірки введених даних, я підтвердив створення бази даних, яка була успішно розгорнута в хмарі Azure.



**2. Налаштування параметрів безпеки та доступу до бази даних:**

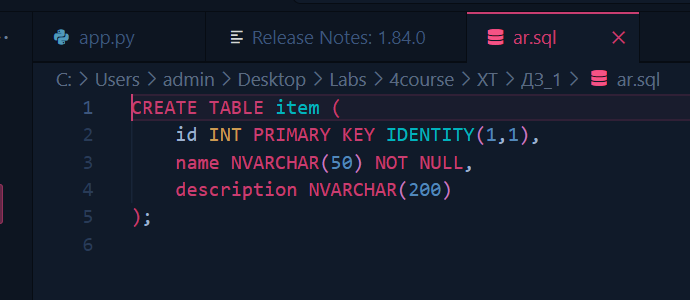
Після створення бази даних я налаштував параметри безпеки:

* **Налаштування правил брандмауера:** Встановив правила брандмауера для забезпечення доступу лише з дозволених IP-адрес.
* **Створення облікових записів:** Створив облікові записи користувачів з відповідними рівнями доступу для управління базою даних.
* **Застосування SSL шифрування:** Активував SSL шифрування для забезпечення безпечної передачі даних.

3. Розробка моделі бази даних та таблиць за допомогою SQLAlchemy ORM:

Використовуючи SQLAlchemy ORM, я створив модель бази даних у Python:

* **Визначення моделі:** Створив клас Item, який відображає таблицю в базі даних з полями id, name та description.
* **Ініціалізація бази даних:** Використав SQLAlchemy(app) для інтеграції з Flask та налаштував з'єднання з базою даних через SQLALCHEMY\_DATABASE\_URI.
* **Міграція схеми:** Застосував міграції для створення таблиць в базі даних Azure SQL.



**4. Створення RESTful API для CRUD операцій над даними:**

Розробив API ендпойнти використовуючи Flask:

* POST /items/: Ендпойнт для створення нового запису в базі даних.
* GET /items/int:item\_id: Ендпойнт для отримання запису за ID.
* PUT /items/int:item\_id: Ендпойнт для оновлення запису за ID.
* DELETE /items/int:item\_id: Ендпойнт для видалення запису за ID.
* GET /items/: Ендпойнт для отримання списку всіх записів.

**5. Тестування API ендпойнтів через Postman:**

Для перевірки роботи API я використав Postman:

* **Відправка запитів:** Надсилав GET, POST, PUT, DELETE запити до розроблених ендпойнтів.
* **Перевірка відповідей:** Аналізував відповіді сервера на коректність та повноту даних.
* **Валідація функціоналу:** Переконався, що всі CRUD операції працюють належним чином.

**Висновки:** В ході виконання проекту було досліджено основні аспекти створення та управління базами даних в Azure. Розроблено програму, яка дозволяє виконувати основні операції з даними через веб-інтерфейс, що робить її доступною для користувачів без необхідності прямого з'єднання з базою даних. Також було встановлено, що використання еластичних пулів дозволяє ефективно масштабувати ресурси та оптимізувати витрати.