

Документ архітектури системи

**Виконали: студенти групи ПМІ-52
Пиж Андрій, Марта Костецька,
Легка Катерина, Кирил Гальміз**

Зміст

1.1 Архітектура системи. Опис окремих сервісів	3
1.1.1 Config Server	4
1.1.2 Registry Server	5
1.1.3 API Gateway	6
1.1.4 Auth Server	7
1.1.5 Zipkin Server	8
1.1.6 Account Service	9
1.1.7 Apartment Service	10
1.1.8 Notification Service	11
1.1.9 Web Client	12

1.1 Архітектура системи. Опис окремих сервісів

Всю систему можна умовно розподілити на 2 основні частини: набір допоміжних сервісів, таких як Api Gateway, Config Server, Registry Server, Auth Server і Zipkin Server та незалежні самостійні мікросервіси. Всього в системі три незалежні мікросервіси: Account Service, Apartment Service, Notification Service.

При проектуванні системи ми використали патерн - “Декомпозиція за піддоменами”. Тому можна виокремити такі три самостійні сервіси:

- Account Service - сервіс, який відповідає за уся роботу з профілями користувачів.
- Apartment Service - сервіс, що відповідає за логіку пов’язану з об’єктами нерухомості.
- Notification Service - це сервіс, який надсилає сповіщення на електронну пошту користувачам про статус їхніх пропозицій на оренду та інші сповіщення.

Кожен з цих сервісів має окрему базу даних Postgres, згідно з патерном “База даних на сервіс”. Веб-інтерфейс комунікує з бекендом через API Gateway, який в свою чергу перенаправляє запити на сервіси. Auth Server - це реалізація патерну “Сервер авторизації”, він відповідає за уся логіку пов’язану з авторизацією та аутентифікацією.

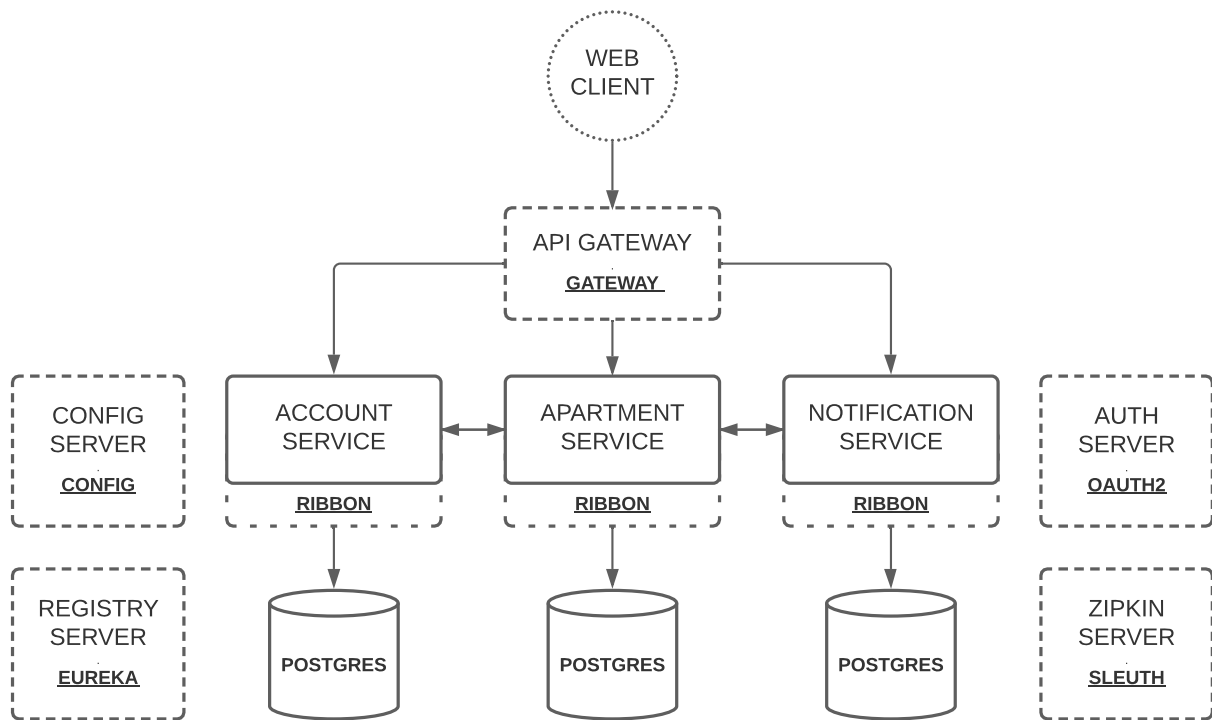


Рисунок 3.1 Загальна архітектура системи

1.1.1 Config Server

Config Server - це централізоване сховище конфігурацій для усіх сервісів. Він реалізований за допомогою фреймворку Spring Cloud Config. Цей фреймворк дозволяє використання репозиторію для зберігання конфігурацій, наприклад - git, subversion. У своїй системі я використав *native profile*, що дозволяє просто зберігати конфігурацію у локальному *classpath*.

При запуску, кожен сервіс робить запит до конфігураційного сервера на відповідні конфігурації. Завдяки Config Server можна змінювати конфігурації динамічно. Отримати доступ до конфігураційного сервера можна лише при наявності правильного паролю та логіну, це зроблено для більшої безпеки. Якщо при ініціалізації сервісу йому не вдається отримати свої конфігурації, то цей сервіс автоматично припиняє свій запуск.

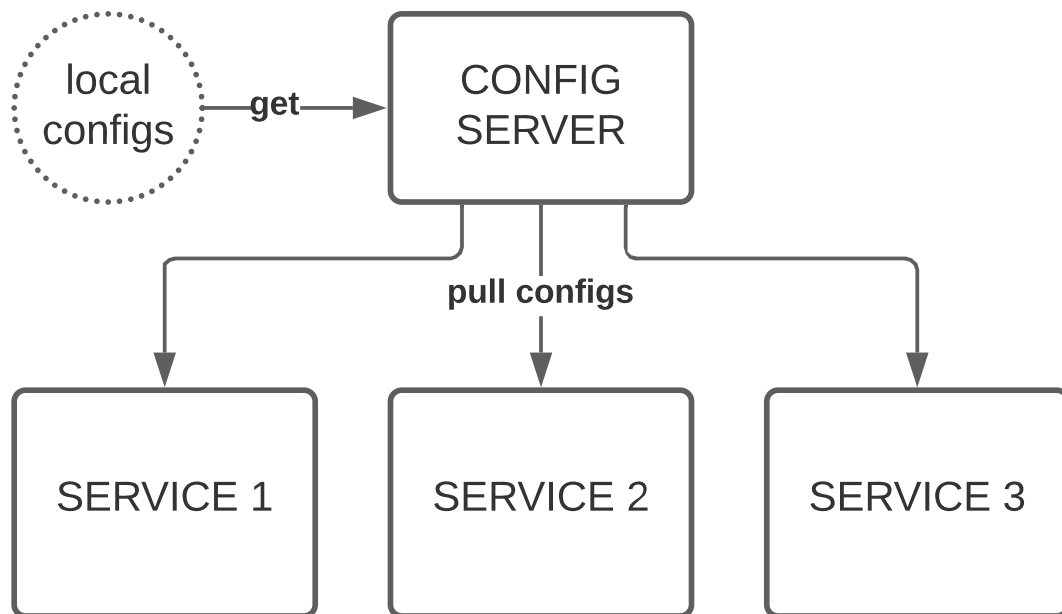


Рисунок 3.2 Принцип роботи конфігураційного сервера

1.1.2 Registry Server

В умовах масштабування система стає дуже динамічною, вона постійно змінюється, оновлює кількість мікросервісів. Мікросервіси можуть додаватися та зупинятися в режимі реального часу. Щоб новий сервіс можна було знайти клієнту або іншому сервісу, була придумана концепція виявлення сервісів. Так, згідно цієї концепції існує такий сервіс як Registry Server. Він займається моніторингом системи та відстежує всі мікросервіси. Кожен сервіс при запуску реєструється у Registry Server, який в свою чергу потім постійно перевіряє, чи цей сервіс все ще є запущеним.

Registry Server - це основний компонент для виявлення сервісів. Він зберігає мережеві розташування екземплярів сервісу. Клієнти звертаються до реєстру

коли їм необхідно отримати дані про запущені сервіси, наприклад, щоб відправити на них запит.

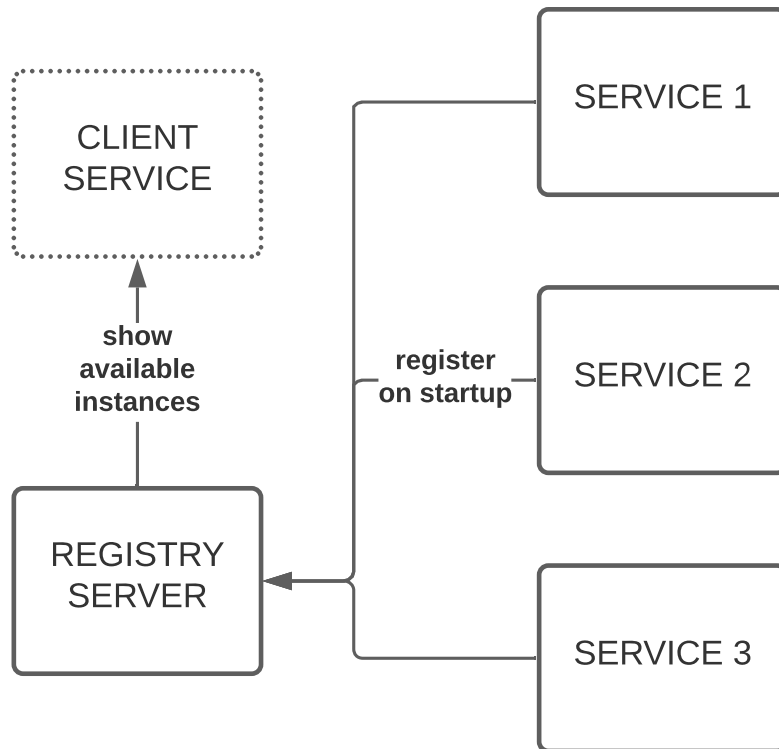


Рисунок 3.3 Принцип роботи Registry Server

1.1.3 API Gateway

API Gateway - це сервіс, який є реалізацією патерну 'API Gateway'. Він слугує "єдиною точкою доступу" до системи для її клієнтів. Тобто усі запити до системи здійснюються, саме через API Gateway, який у свою чергу перенаправляє їх до потрібних сервісів. Завдяки цьому клієнтам не потрібно знати про усі сервіси системи, щоб надіслати на них запит.

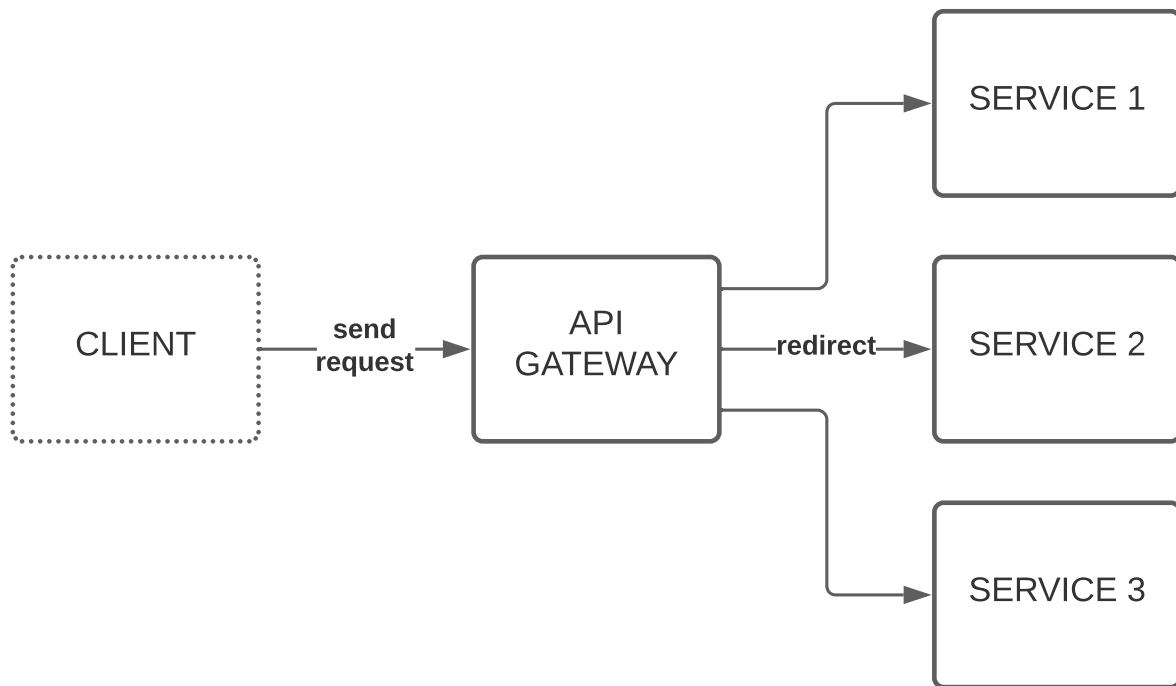


Рисунок 3.4 Принцип роботи API Gateway

1.1.4 Auth Server

Сервіс Auth Server є реалізацією патерну “сервер авторизації”. Уся авторизація та аутентифікація у системі здійснюється через цей сервіс. У своїй системі я використовую протокол авторизації OAuth 2. Цей протокол - це схема авторизації, яка дозволяє надати клієнту обмежений доступ до захищених ресурсів без необхідності передавати логін і пароль з кожним запитом. Також я використав відкритий стандарт JWT (Json Web Token) для створення токенів доступу, що базуються на JSON форматі.

Щоб отримати доступ до захищених ресурсів, користувачу потрібно відправити запит на Auth Server з правильним логіном та паролем лише перший раз, у відповідь користувач отримає токен, у якому закодована інформація про нього. У подальшому, користувачу варто лише прикріплювати до запитів цей токен, щоб отримати доступ до ресурсів. Коли інші сервіси будуть отримувати запити на захищені ресурси, вони відправлятимуть токен з запиту до Auth Server

для перевірки його валідності. Можливість перевіряти токен та отримувати інформацію про користувача можуть лише мікросервіси системи.

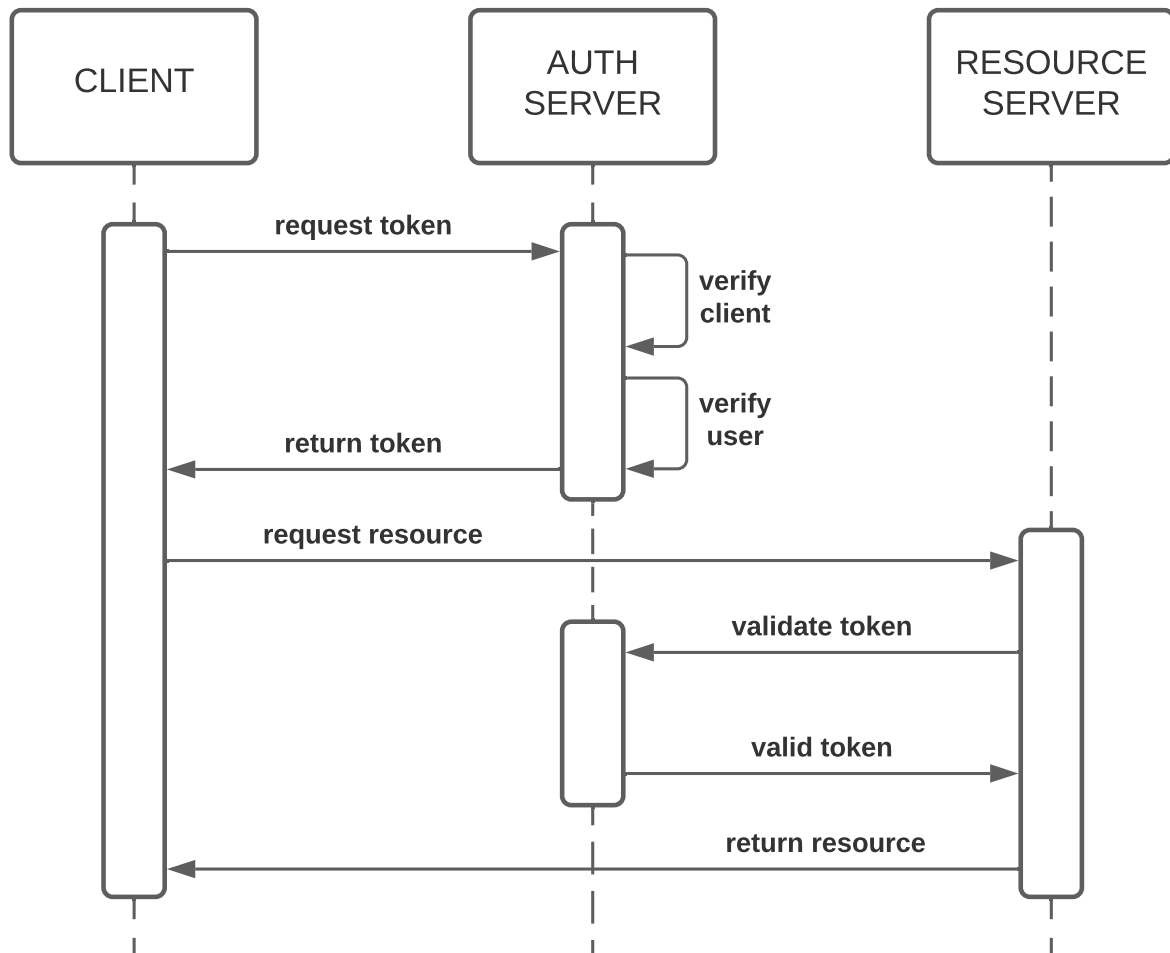


Рисунок 3.5 Принцип роботи Auth Server

1.1.5 Zipkin Server

Ще одною складовою системи є мікросервіс - Zipkin Server, який є реалізацією патерну “розподілене трасування”. В умовах розподіленої системи, яка містить велику кількість мікросервісів може бути важко налагоджувати її та відслідковувати запити між сервісами. Патерн “розподілене трасування запитів” це метод, який використовується для моніторингу взаємодії між мікросервісами. Zipkin Server дозволяє відслідковувати усі запити в системі і використовується для швидшого виявлення помилок в системі та виявлення причин низької

продуктивності. Це було реалізовано за допомогою готових рішення - OpenZipkin та Spring Sleuth і налаштував їх під свої потреби.

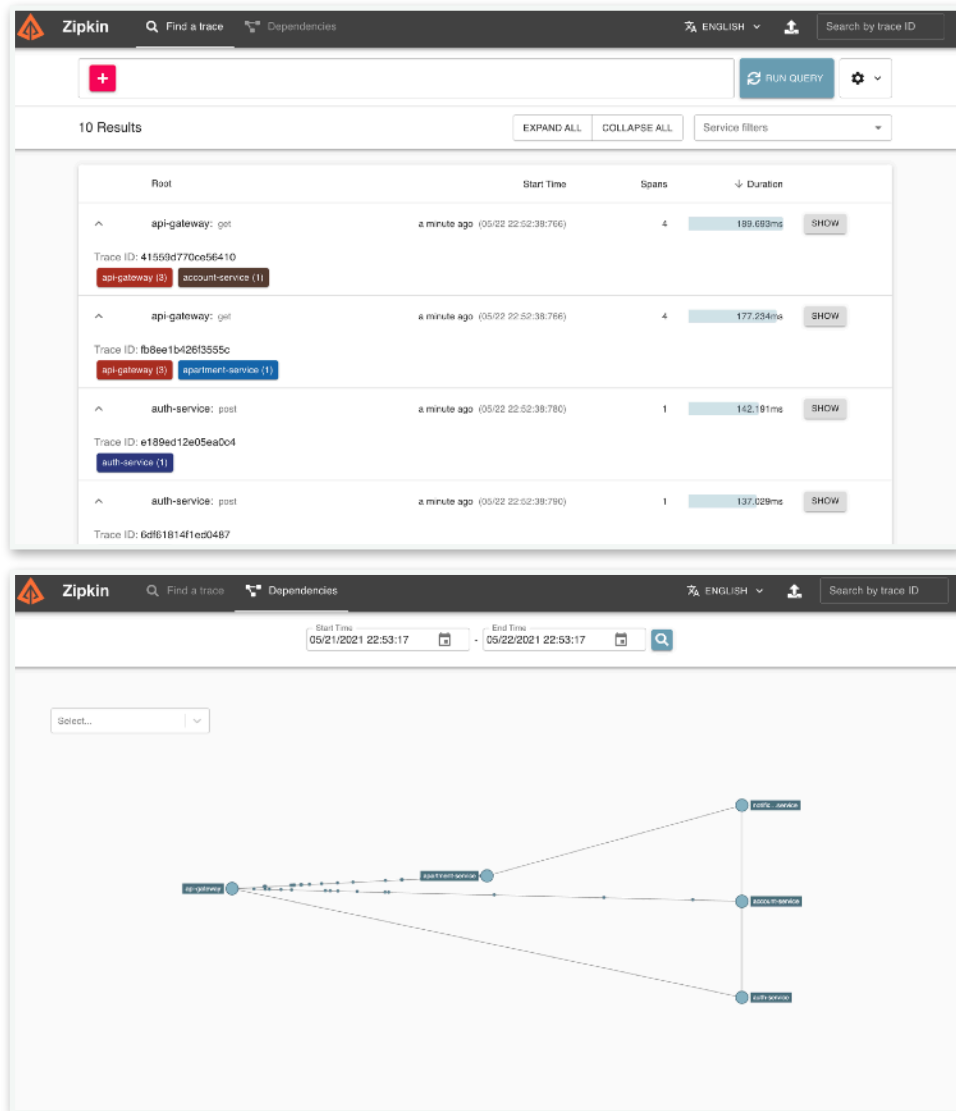


Рисунок 3.6 Відслідковування запитів за допомогою Zipkin Server

1.1.6 Account Service

Account Service - це мікросервіс, який відповідає за усю логіку пов'язану з акаунтами користувачів. Цей сервіс реалізує таку функціональність як створення акаунту, отримання інформації про акаунт та його редагування. Внутрішня архітектура цього сервісу - це класична тришарова архітектура:

- Repository - через цей шар відбувається уся взаємодія з базою даних;
- Service - у цьому шарі реалізовується бізнес логіка;

- Controllers - представляє собою контроллери, які обробляють запити від користувачів;

Account Service має доступ до бази даних Postgres, яка приватна для нього. Взаємодія з іншими сервісами відбувається по протоколу HTTP через фреймворк Spring Cloud OpenFeign.

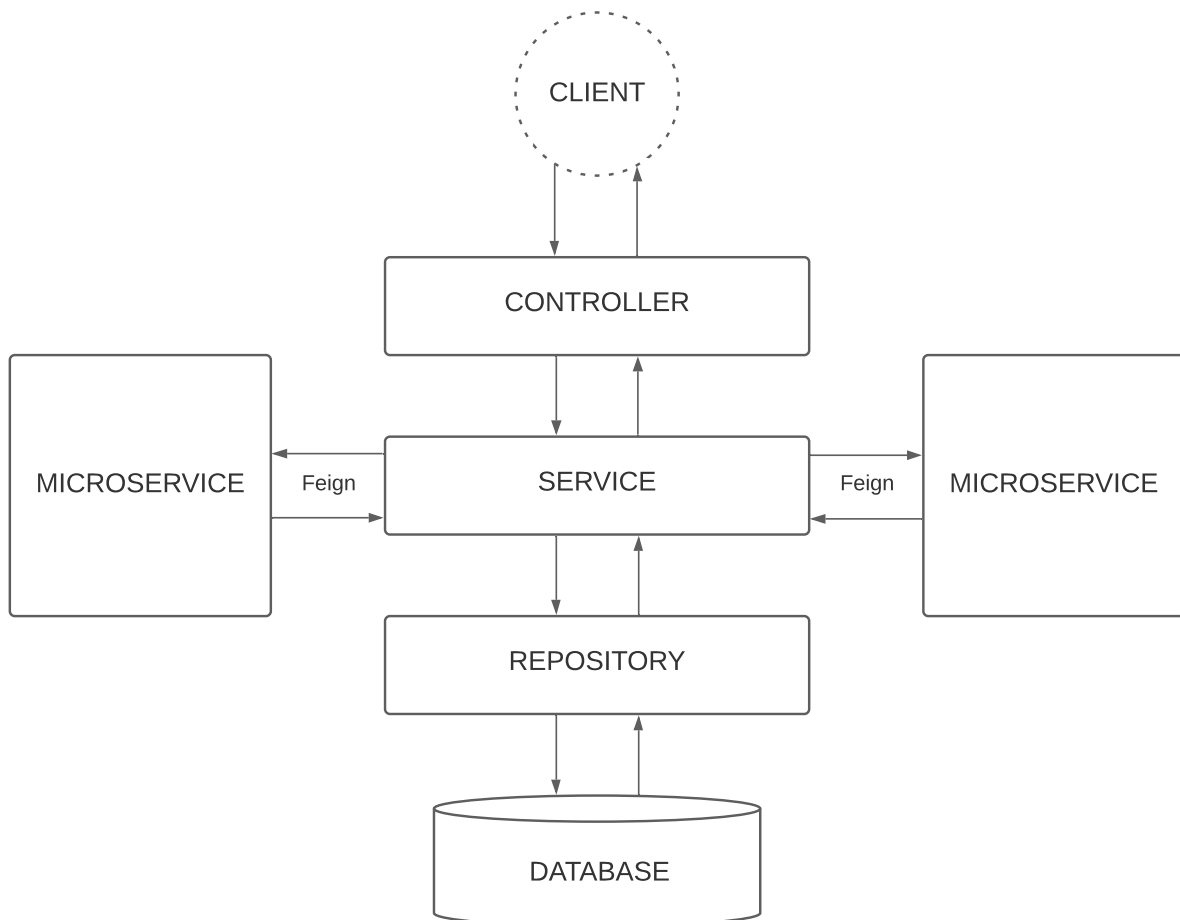


Рисунок 3.7 Внутрішня архітектура сервісу

1.1.7 Apartment Service

Apartment Service - це компонент системи, що реалізує таку функціональність:

- створення оголошення про оренду нерухомості, отримання списку усіх об'єктів виставлених на оренду, оновлення даних про нерухомість;
- створення заявок на оренду нерухомості, редагування, видалення відхилення, або схвалення цих заявок.

Загалом, внутрішня архітектура цього сервісу дуже схожа з архітектурою Account Service.

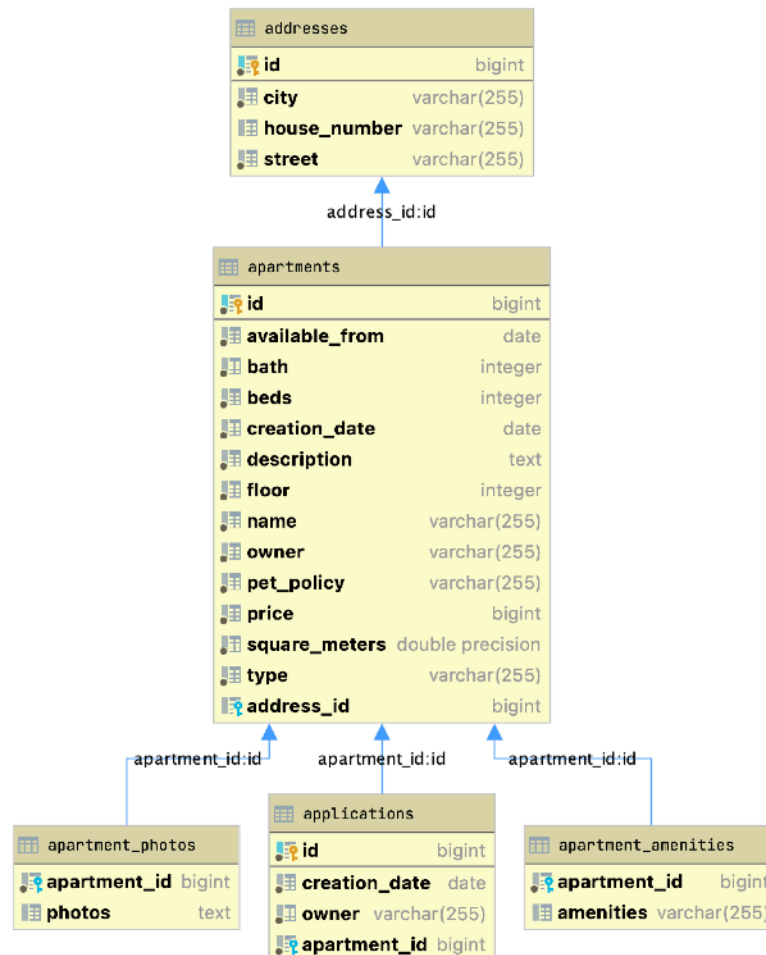


Рисунок 3.8 Схема бази даних Apartment Service

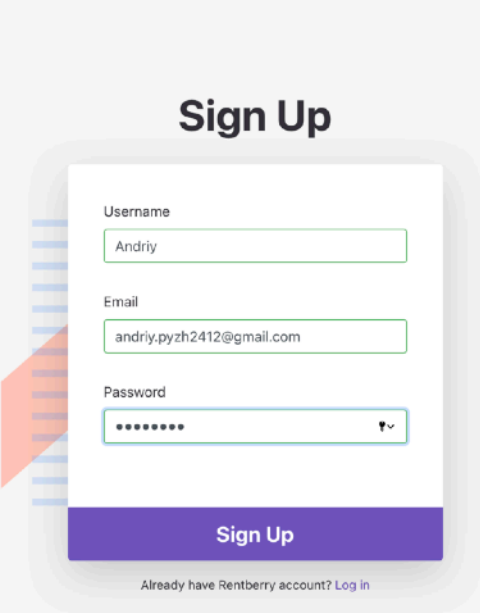
1.1.8 Notification Service

Notification Service - це сервіс, який відповідає за сповіщення користувачів через електронну пошту. При реєстрації нового користувача, створення нової заявки на оренду нерухомості, при її схваленні чи відхиленні Account Service та Apartment Service відправляють потрібні запити на Notification Service, а той в свою чергу відправляє потрібні листи на електронні пошти користувачів.

Відправлення листів реалізовано через фреймворк Spring Mail. Також ми використали Google Mail Server - сервер для відправлення повідомлень з електронної пошти - "rentler.a.p@gmail.com" з якої відправляються повідомлення з Notification Service.

1.1.9 Web Client

Для демонстрації функціональності моєї системи ми створили веб-інтерфейс - Web Client. Для його створення ми використаємо бібліотеку для створення юзер інтерфейсів - ReactJS. Web Client комунікує з системою через тільки через API Gateway. Нижче продемонстрована сторінка реєстрації користувачів.



Sign Up

Username

Email

Password

Sign Up

[Already have Rentberry account? Log in](#)

R

Rentler

RentBuy

Where do you want to leave?

Properties

List a Property

Apartment

80 Shevchenka, Lviv

My Apartment

♡ Save

price

\$10,000

highest offer: \$10,100

Available From: May 31


Apply


Bed: 2


Bath: 2

Sq Ft: 55

No Pets







Apartment Description