

Міністерство освіти і науки України  
Національний технічний університет України „КПІ”  
Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки  
інформації та управління

ЗВІТ

з лабораторної роботи № 2  
дисципліни  
“ТЕХНОЛОГІЇ ПАРАЛЕЛЬНОГО ПРОГРАМУВАННЯ В УМОВАХ  
ВЕЛИКИХ ДАНИХ”  
на тему:

„Паралельні обчислення в моделі з декількома процесами”

Виконали:  
студенти групи ІТ-01мн  
Панасюк Станіслав  
Лесогорський Кирило

Перевірів:  
доц. Жереб К. А.

Київ – 2021

## **Зміст**

|   |   |
|---|---|
| 1. Постановка задачі .....  | 3 |
| 2. Обрані інструменти .....   | 3 |
| 3. Високорівнева архітектура системи .....                          | 4 |
| 4. Опис роботи програмного забезпечення .....                       | 5 |
| 5. Отримані результати .....  | 5 |
| 5.1 Закон Амдала при збільшенні кількості воркерів: .....           | 7 |
| 5.2 Закон Амдала при збільшенні кількості потоків у воркерах: ..... | 7 |
| 5.3 Результати для інших оптимальних варіантів .....                | 7 |
| 6. Висновки .....   | 8 |

## 1. Постановка задачі

Для обраної задачі необхідно реалізувати послідовну (однопоточну) реалізацію, а також мультипоточну реалізацію зі спільною пам'яттю. У якості задачі було обрано побудову системи пошуку схожих зображень. У ядрі системи лежатиме використання D-hash для знаходження хешу зображення. D-hash дозволяє точно та швидко шукати схожі зображення. Він стійкий до скейлінгу зображення, але погано справляються з обрізаними та повернутими під кутом зображеннями. Тому цю техніку аугментовано за допомогою наступного прийому: при завантаженні зображення воно буде аугментовано за допомогою декількох фільтрів, при цьому для кожного фільтру буде згенеровано хеш і збережено у базу даних. При пошуку зображення буде використовуватись оператор XOR для знаходження зображень зі схожими хешами.

## 2. Обрані інструменти

Для виконання перших двох частин лабораторної роботи буде використано стандартні інструменти Java. З самого початку буде використано фреймворк *Spring* для створення веб-інтерфейсу у майбутньому. *Spring Data* буде використано для доступу до бази даних. *Lombok* буде використано для зменшення кількості бойлерплейту. *JUnit* буде використано для тестування. Вбудована бібліотека *AWT* буде використана для роботи з зображеннями. Для роботи з чергою буде використано *RabbitMQ* та відповідно інтеграцію зі *Spring*.

### 3. Високорівнева архітектура системи

На високому рівні система виглядає наступним чином:

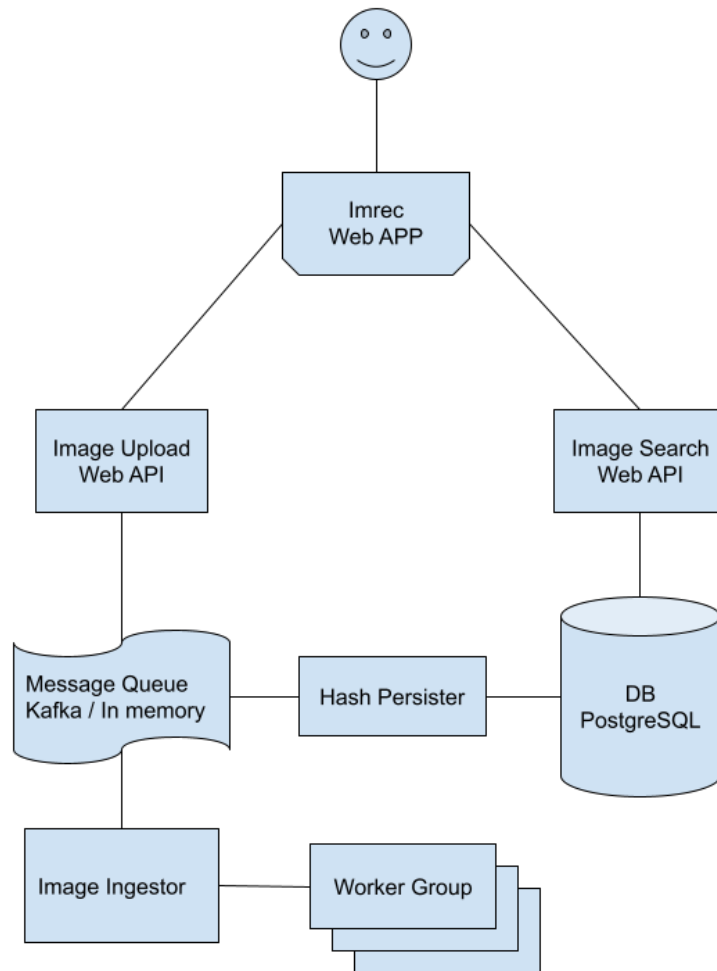


Рис. 3.1 – Високорівнева архітектура

1. **Imrec Web APP** – браузерний додаток, який надає інтерфейс для користувача. Підтримується додавання нового зображення і пошук у переліку уснуючих.

2. **Image Upload Web API** – веб сервер для завантаження зображень. Виконує валідацію запиту, зберігає файл у персистентне сховище і передає його на подальшу обробку у Image Ingestor через Kafka або InMemory чергу.

3. **Image Ingestor** – оркестратор процесу обробки зображень. Отримує повідомлення з черги і передає їх на обробку воркерам. Після обчислення хешу, передає інформацію назад у чергу

4. **Worker** – обчислює необхідні хеші для зображення.

5. **Hash Persister** – зберігає отриманні хеші у персистентне сховище.

6. **Image Search Web API** – веб сервер, який відповідає за пошук серед вже існуючих зображень.

Така архітектура має низьку зчепність і дуже модульна. Слід зазначити, що кожен воркер буде обчислювати хеш для зображення з фільтрами, це дозволяє знизити затримку при обратній збірці результатів при розподіленні. Також це дозволить винести усю роботу, яка потребує багато обчислювальних ресурсів на окремі машини, дозволяючи інджестору працювати у однопоточному режимі при умові використання асинхронних інтерфейсів вводу-виводу. Також такий підхід дозволить знизити навантаження на мережу, коли декілька воркерів завантажують досить великі фотографії (більше 10 мегабайт) з файлового сховища.

У порівнянні з попередньою лабораторною роботою тепер інджестор та воркер і комунікують за допомогою черги повідомлень. У якості черги було використано RabbitMQ. Також обов'язки інджестора та веб АПІ було поєднано, оскільки тоді робота виходить занадто гранулярною. Для простоти(спрощення у рамках лабораторної роботи) Hash Persister є частиною WebAPI.

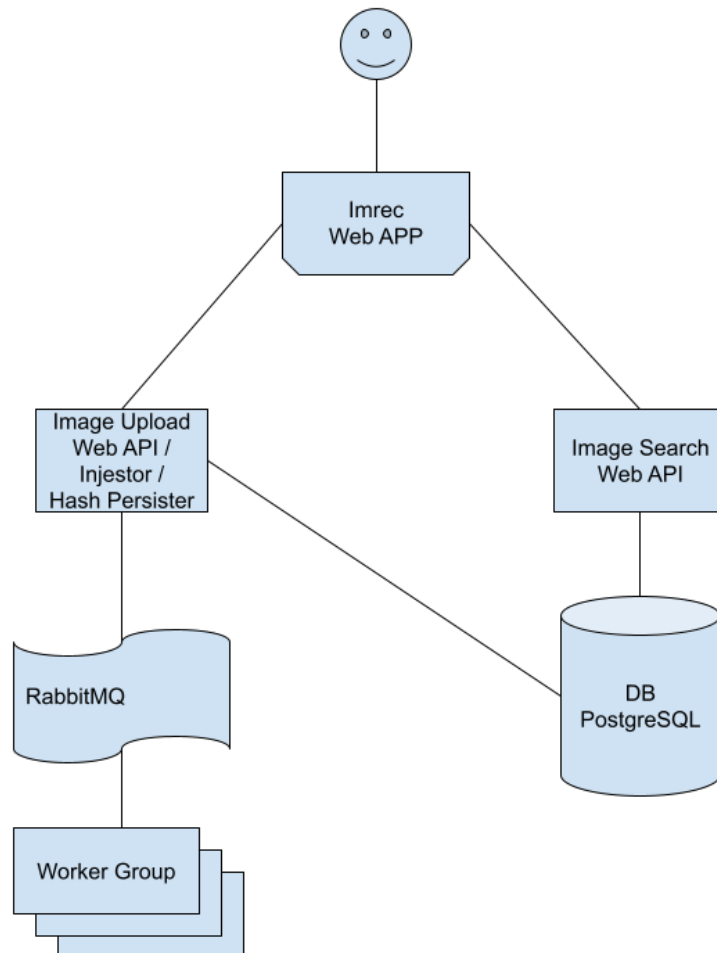


Рис. 3.2 – Оновлена схема архітектури

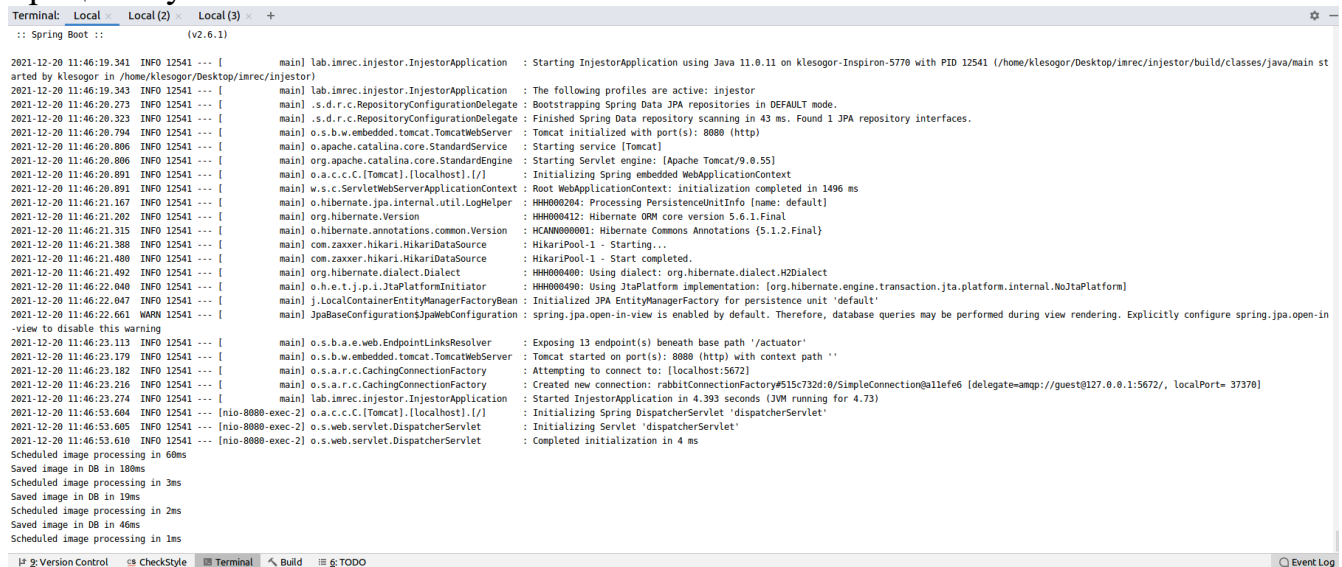
## 4. Опис роботи програмного забезпечення

Багато коду базується на результатах попередніх робіт, тому ми розглянемо лише зміни, які додалися у зв'язку з багатопроцесною реалізацією.

По-перше, з'явився контролер та сервіс для завантаження зображень та його додавання у чергу обробки. У цьому сервісі відбувається валідація зображень, збереження у файлове сховище та делегація роботи до воркерів. У якості основи воркеру було обрано однопоточну реалізацію з попередньої роботи, щоб запобігти неефективному використанню ресурсів при синхронізації. Час обробки одного зображення є достатнім і не вимагає багатопоточної реалізації. Паралелізм може бути доданий на рівні лістєнеру черги. Також було додано необхідну для зберігання результатів роботи інфраструктуру: репозиторій, модель зображення та хешу, лістєнер черги результатів.

## 5. Отримані результати

В результаті виконання роботи було отримано наступні результати з процесінгу:



```
Terminal: Local Local (2) Local (3) +
:: Spring Boot ::
(v2.6.1)

2021-12-20 11:46:19.341 INFO 12541 --- [main] lab.imrec.injestor.InjestorApplication : Starting InjestorApplication using Java 11.0.11 on Klesogor-Inspiron-5770 with PID 12541 (/home/klesogor/Desktop/imrec/injestor/build/classes/java/main st
arted by klesogor in /home/klesogor/Desktop/imrec/injestor)
2021-12-20 11:46:19.343 INFO 12541 --- [main] lab.imrec.injestor.InjestorApplication : The following profiles are active: injestor
2021-12-20 11:46:20.273 INFO 12541 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2021-12-20 11:46:20.323 INFO 12541 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 43 ms. Found 1 JPA repository interfaces.
2021-12-20 11:46:20.794 INFO 12541 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2021-12-20 11:46:20.806 INFO 12541 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2021-12-20 11:46:20.806 INFO 12541 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.55]
2021-12-20 11:46:20.891 INFO 12541 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2021-12-20 11:46:20.891 INFO 12541 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1496 ms
2021-12-20 11:46:21.167 INFO 12541 --- [main] o.hibernate.jpa.internal.util.LogHelper : HH0000204: Processing PersistenceUnitInfo [name: default]
2021-12-20 11:46:21.202 INFO 12541 --- [main] org.hibernate.Version : HH0000412: Hibernate ORM core version 5.6.1.Final
2021-12-20 11:46:21.315 INFO 12541 --- [main] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
2021-12-20 11:46:21.308 INFO 12541 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2021-12-20 11:46:21.480 INFO 12541 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2021-12-20 11:46:21.492 INFO 12541 --- [main] org.hibernate.dialect.Dialect : HH0000408: Using dialect: org.hibernate.dialect.H2Dialect
2021-12-20 11:46:22.040 INFO 12541 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HH0000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2021-12-20 11:46:22.047 INFO 12541 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2021-12-20 11:46:22.661 WARN 12541 --- [main] JpaBaseConfigurationsJpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2021-12-20 11:46:23.113 INFO 12541 --- [main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 13 endpoint(s) beneath base path '/actuator'
2021-12-20 11:46:23.179 INFO 12541 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2021-12-20 11:46:23.182 INFO 12541 --- [main] o.s.a.r.c.CachingConnectionFactory : Attempting to connect to: [localhost:5672]
2021-12-20 11:46:23.216 INFO 12541 --- [main] o.s.a.r.c.CachingConnectionFactory : Created new connection: rabbitConnectionFactory#515c732d:0/SimpleConnection@allefe6 [delegate=amp://guest@127.0.0.1:5672/, localPort= 37370]
2021-12-20 11:46:23.274 INFO 12541 --- [main] lab.imrec.injestor.InjestorApplication : Started InjestorApplication in 4.393 seconds (JVM running for 4.73)
2021-12-20 11:46:53.604 INFO 12541 --- [nio-8080-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2021-12-20 11:46:53.605 INFO 12541 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2021-12-20 11:46:53.610 INFO 12541 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Completed initialization in 4 ms
Scheduled image processing in 60ms
Saved image in DB in 100ms
Scheduled image processing in 3ms
Saved image in DB in 19ms
Scheduled image processing in 2ms
Saved image in DB in 46ms
Scheduled image processing in 1ms
```

Рис. 5.1 – Результат процесінгу зображення у черзі та збереження у БД

