Microservicios con Spring Boot

Propuesta de un escolástico para matriculación

Proyecto en GitHUB:

Justificación

Microservicios

La arquitectura de microservicios permite tener una gran escalabilidad y flexibilidad sin olvidar la robustez que los sistemas de este tipo requieren

Herramientas

-Eureka

-Zuul

-Hystrix

-Zipkin

-RabbitMQ

Bases de Datos

-MySQL

-PostgreSQL

Interfaz y servidor externo

-Angular

-Github

Zipkin!

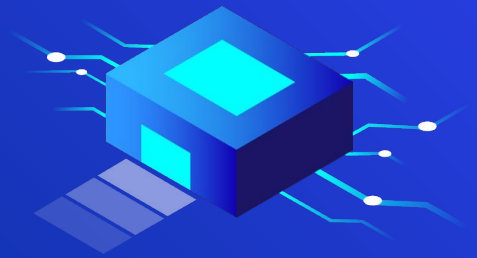
Es una herramienta que recolecta las transacciones creadas por Sleuth en la ejecución de los microservicios e información de los tiempos de respuesta de las invocaciones que han intervenido en una transacción. Ofrece las dos funcionalidades la recolección de datos y la obtención de los mismos

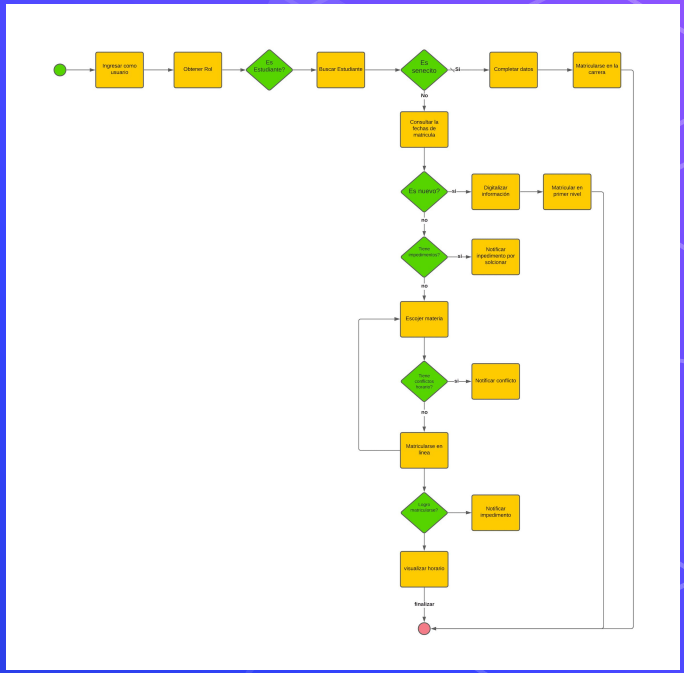
RabbitMQ!

RabbitMQ es un software de encolado de mensajes llamado broker de mensajería o gestor de colas. Dicho de forma simple, es un software donde se pueden definir colas, las aplicaciones se pueden conectar a dichas colas y transferir/leer mensajes en ellas.

Problema a resolver

- Ingreso por usuario
- Búsqueda de materias por su nrc
- Visualización de horario
- Gestión de matriculación

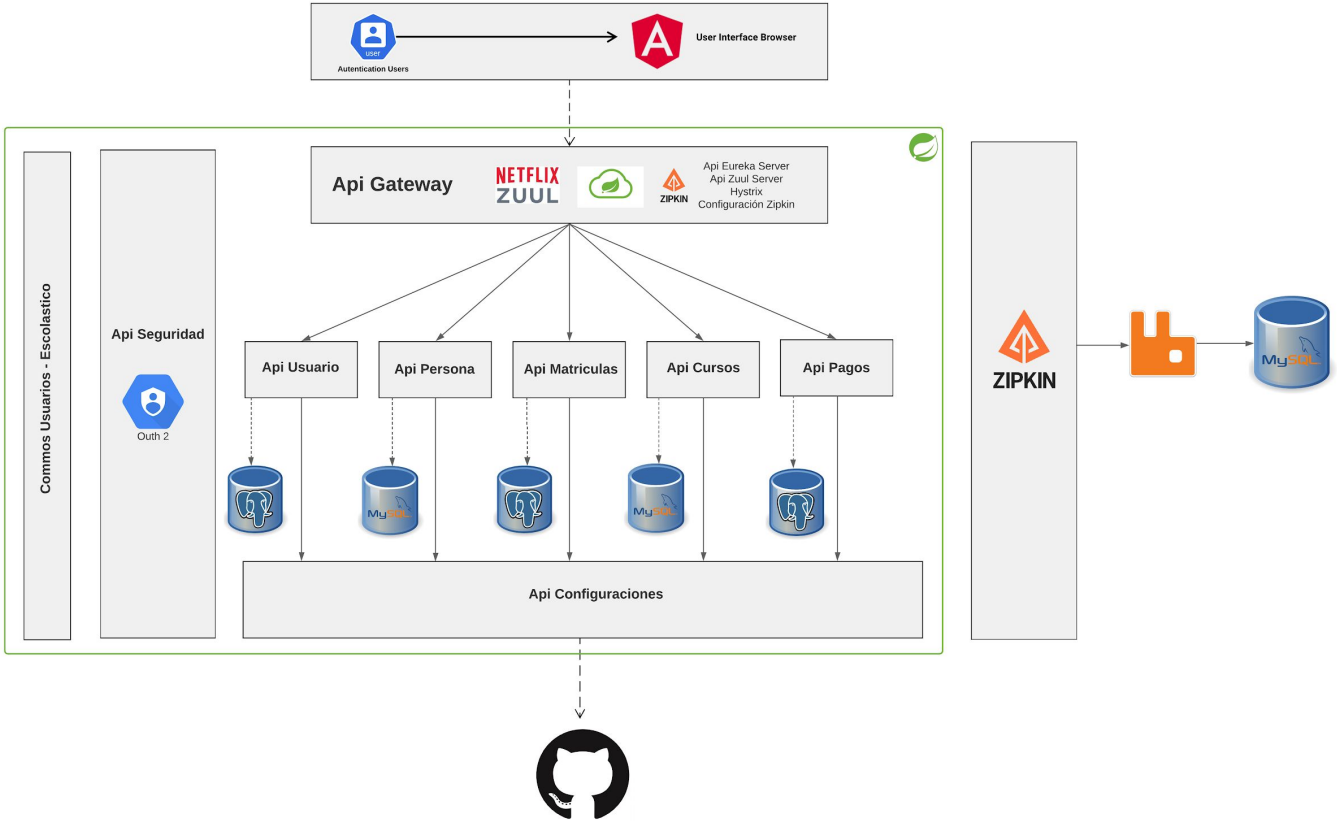




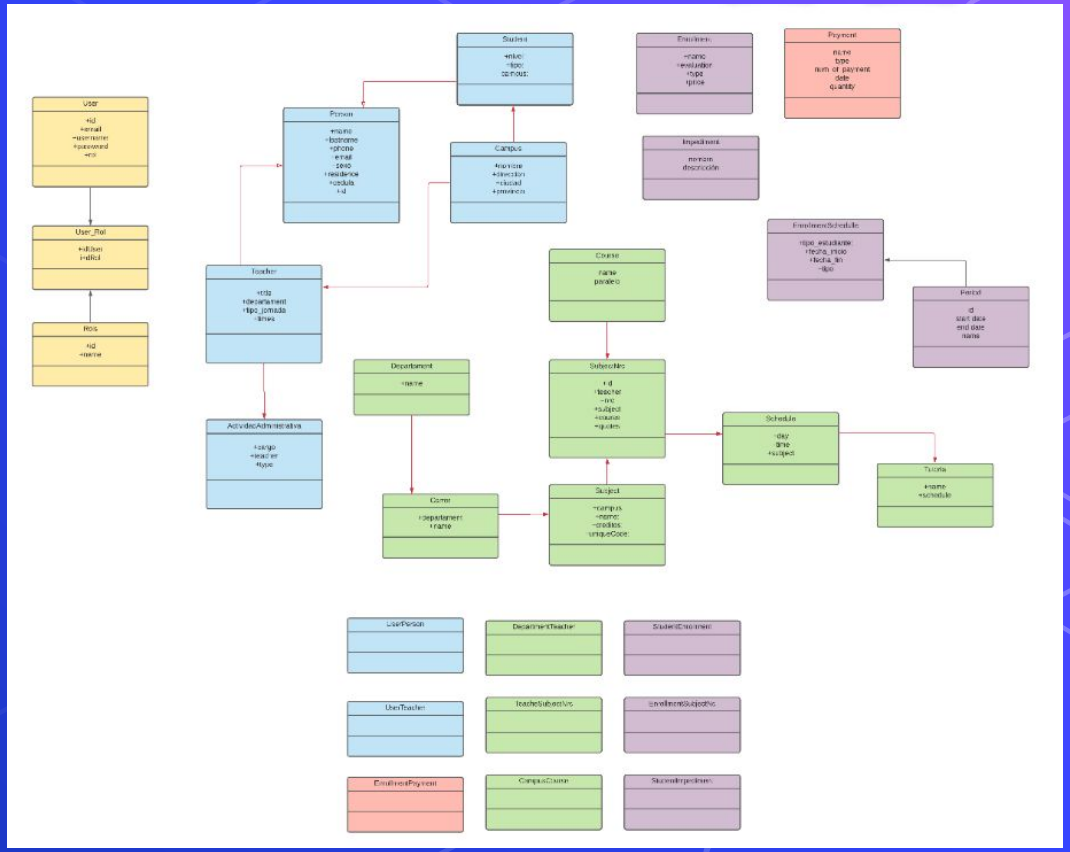
Arquitectura!

Arquitectura de microservicios para la resolución del problema

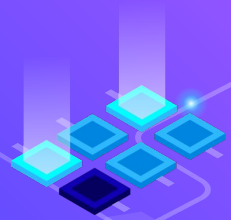
Secuencia de pasos


user
Autentication Users
User Interface Browser
Api Gateway
NETFLIX
ZUUL
ZIPKIN
Api Eureka Server
Api Zuul Server
Hystrix
Configuración Zipkin
Commos Usuarios - Escolastico
Api Seguridad
Outh 2
Api Usuario
Api Persona
Api Matriculas
Api Cursos
Api Pagos
Api Configuraciones
ZIPKIN
MySQL

1. Diseño de la base de datos



2. Estructura de los microservicios



3. Configuraciones

-Eureka Server

-Zuul Server

-Config Server

-OAuth2 Service

-Api Gateway (Enrollments Service)

-User, course, person, payments services

4. Zuul - Eureka - Config

```
spring.application.name= servicio-zuul-server
server.port= 8090
eureka.client.serviceUrl.defaultZone= http://localhost:8761/eureka/
zuul.routes.courses.service-id=service-courses
zuul.routes.courses.path=/api/courses/**

zuul.routes.payments.service-id=service-payments
zuul.routes.payments.path=/api/payments/**

zuul.routes.persons.service-id=service-persons
zuul.routes.persons.path=/api/persons/**

zuul.routes.enrollments.service-id=service-enrollments
zuul.routes.enrollments.path=/api/enrollments/**

zuul.routes.users.service-id=service-users
zuul.routes.users.path=/api/users/**

zuul.routes.security.service-id=service-oauth
zuul.routes.security.path=/api/security/**
zuul.routes.security.sensitive-headers=Cookie,Set-Cookie

hystrix.command.default.execution.isolation.thread.timeoutInMilliseconds: 30000
ribbon.ConnectTimeout: 1000
ribbon.ReadTimeout: 10000

spring.sleuth.sampler.probability=1.0
```

Zuul

Eureka

```
spring.application.name=service-eureka-server
server.port=8761

eureka.client.fetch-registry=false
eureka.client.register-with-eureka=false

```

Config

```
spring.application.name= service-config-server
server.port= 8888


spring.cloud.config.server.git.uri = https://github.com/DavidVique1998/service-escolastico-config.git
spring.cloud.config.server.git.username=
spring.cloud.config.server.git.password=
```

5. Microservicios

Enrollment

```
spring.application.name= service-enrollments
server.port= 9092
eureka.client.service-url.defaultZone= http://localhost:8761/eureka
spring.sleuth.sampler.probability=1.0
spring.zipkin.base-url= http://localhost:9411/
```

Courses

```
spring.application.name=service-courses
server.port= ${PORT:0}

eureka.instance.instance-id= ${spring.application.name}:${spring.application.instance_id:${random.value}}

eureka.client.service-url.defaultZone= http://localhost:8761/eureka

spring.sleuth.sampler.probability=1.0
```

Person

```
spring.application.name=service-persons
server.port= ${PORT:0}

eureka.instance.instance-id= ${spring.application.name}:${spring.application.instance_id:${random.value}}

eureka.client.service-url.defaultZone= http://localhost:8761/eureka

spring.sleuth.sampler.probability=1.0
```

Payments

```
spring.application.name=service-payments
server.port= ${PORT:0}

eureka.instance.instance-id= ${spring.application.name}:${spring.application.instance_id:${random.value}}

eureka.client.service-url.defaultZone= http://localhost:8761/eureka

spring.sleuth.sampler.probability=1.0
```

User

```
spring.application.name= service-users
server.port= ${PORT:0}

eureka.instance.instance-id= ${spring.application.name}:${spring.application.instance_id:${random.value}}

eureka.client.service-url.defaultZone= http://localhost:8761/eureka
logging.level.org.hibernate.SQL=debug


spring.sleuth.sampler.probability=1.0
```

OAuth

```
spring.application.name=service-oauth
server.port=9101
eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka/

spring.sleuth.sampler.probability=1.0
```

6. Acceso a configuraciones

Enrollment

```
spring.application.name= service-enrollments
spring.profiles.active = dev
spring.cloud.config.uri= http://localhost:8888
```

Courses

```
spring.application.name= service-courses
spring.profiles.active = dev
spring.cloud.config.uri= http://localhost:8888
```

Person

```
spring.application.name= service-persons
spring.profiles.active = dev
spring.cloud.config.uri= http://localhost:8888
```

Payments

```
spring.application.name= service-payments
spring.profiles.active = dev
spring.cloud.config.uri= http://localhost:8888
```

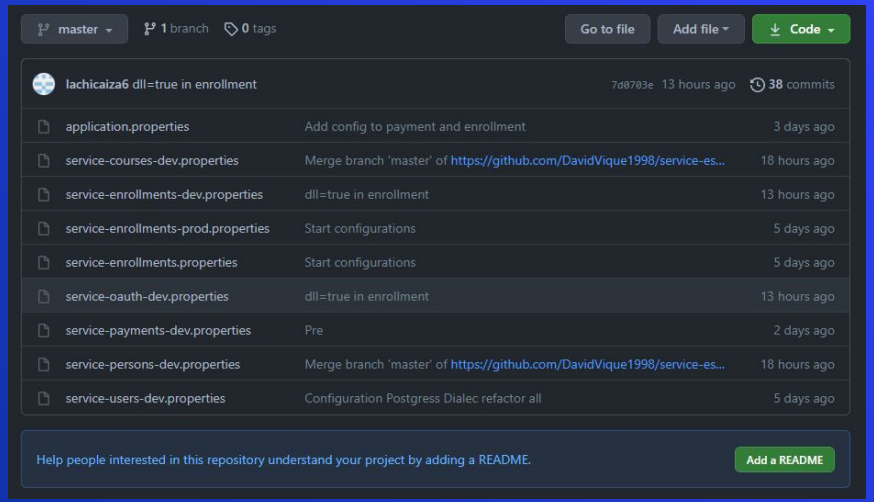
User

```
spring.application.name= service-users
spring.profiles.active = dev
spring.cloud.config.uri= http://localhost:8888
```

OAuth

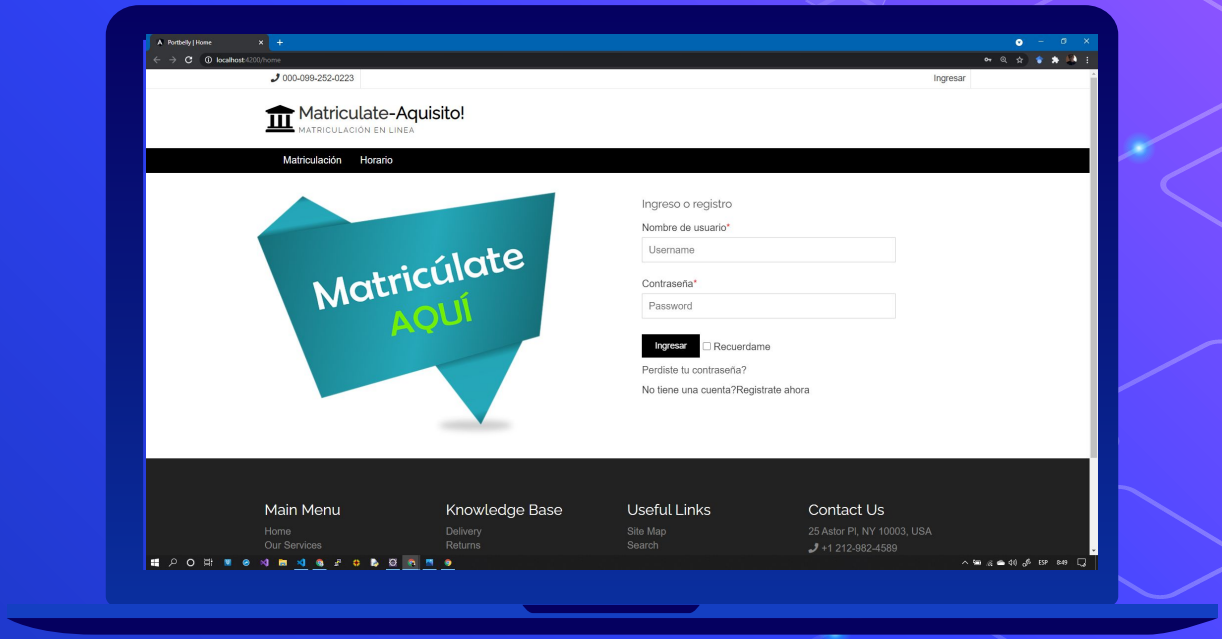
```
spring.application.name= service-oauth
spring.cloud.config.uri=http://localhost:8888
```

Repositorio en GitHub

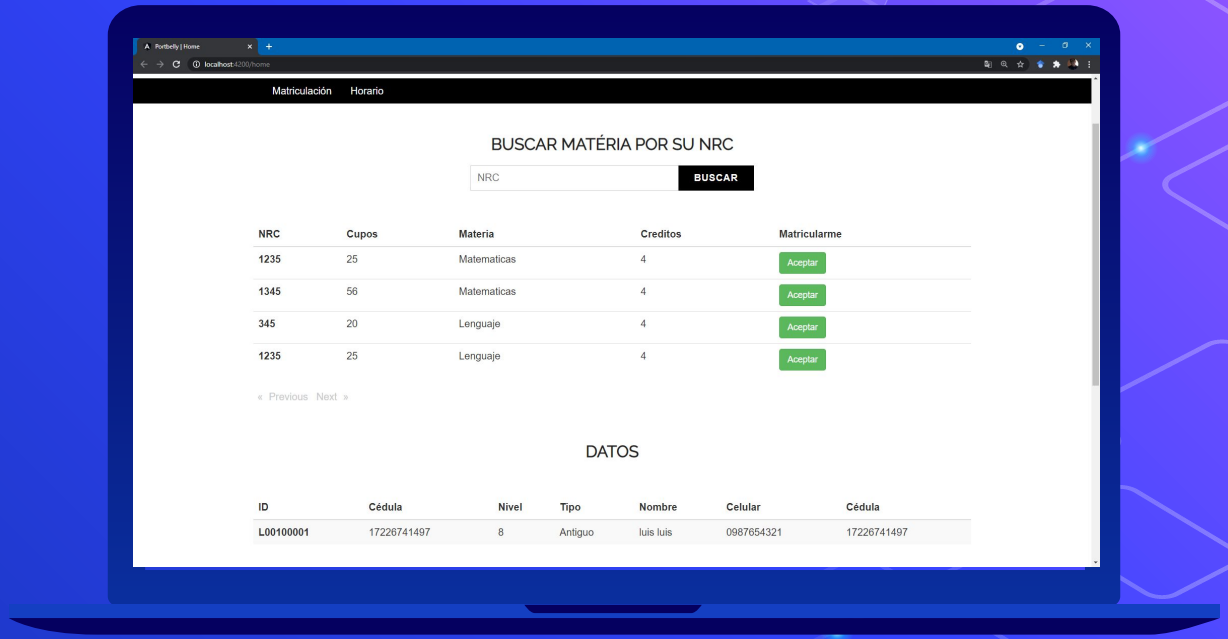


Interfaces de usuario

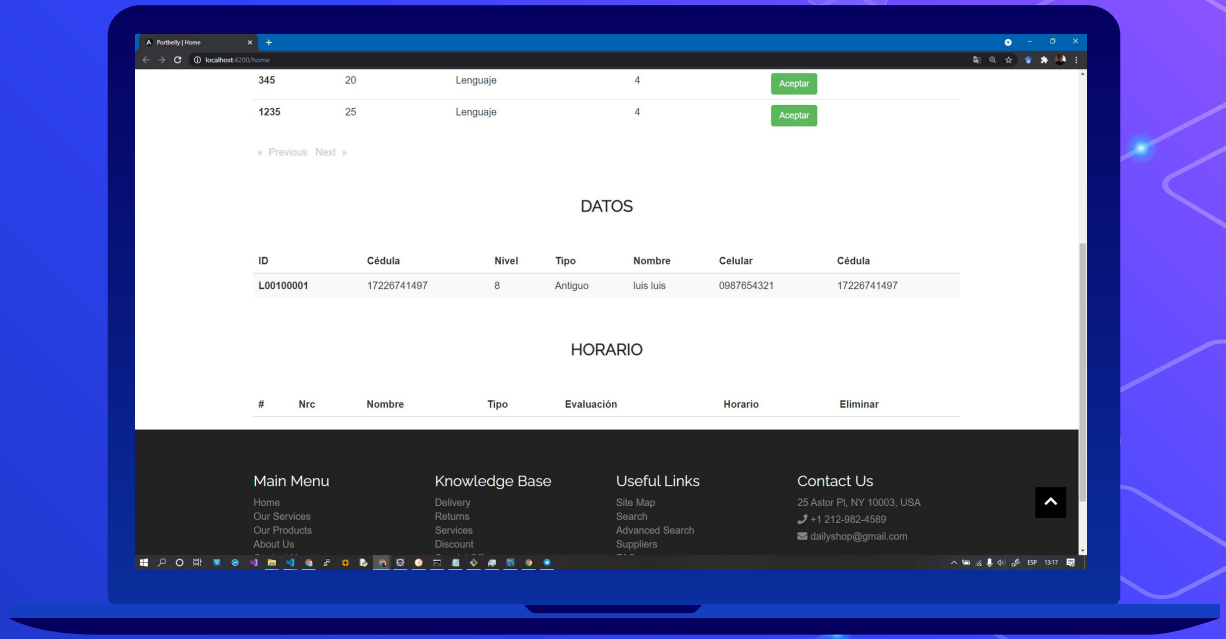
Inicio de sesión



Seleccionar materia nrc



Información Estudiante



Gracias!

Alguna pregunta?

Puedes encontrarnos en:

@davique (github)

@lachicaiza6 (github)

Bibliografía

Nebel, A. (2019). Arquitectura de microservicios para plataformas de integración.

Sucasas, V., Mantas, G., Radwan, A., & Rodriguez, J. (2016, May). An OAuth2-based protocol with strong user privacy preservation for smart city mobile e-Health apps. In 2016 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.

Siriwardena, P. (2020). Edge Security with an API Gateway. In Advanced API Security (pp. 103-127). Apress, Berkeley, CA.

Molchanov, H., & Zhmaiev, A. (2018). Circuit breaker in systems based on microservices architecture. Advanced Information Systems, 2(4), 74-77.

Hoffmann, T. R. (1992). EUREKA: A hybrid system for assembly line balancing. Management Science, 38(1), 39-47.

PostgreSQL, B. (1996). PostgreSQL. Web resource: http://www. PostgreSQL. org/about.

MySQL, A. B. (2001). MySQL.

Zipkin (2015) from https://zipkin.io/