UserAPI - Prueba Técnica Especialista en Integración BCI

API REST desarrollada en Java 17 con Spring Boot para el registro de usuarios, validación de correo y contraseña mediante expresiones regulares configurables, generación de token JWT, persistencia con JPA y documentación Swagger UI.

# Tecnologías utilizadas

* Java 17
* Spring Boot 3
* Spring Data JPA (H2 in-memory)
* Spring Security con JWT
* Lombok
* Maven
* Swagger (Springdoc OpenAPI 2.x)
* JUnit 5 + Mockito

# Requisitos cumplidos

|  |  |
| --- | --- |
| Requisito | Estado |
| Java 8+ | OK |
| Spring Boot | OK |
| JPA + H2 | OK |
| Maven | OK |
| Endpoint /api/register | OK |
| Validación de email por regex configurable | OK |
| Validación de password por regex configurable | OK |
| Token JWT generado y guardado | OK |
| Swagger UI disponible | OK |
| Pruebas unitarias (UserService) | OK |
| README explicando la solución | OK |

# Validaciones

- Email debe tener formato válido según expresión regular en application.properties  
- Contraseña debe cumplir:  
 - Mínimo 8 caracteres  
 - Al menos una mayúscula, un número y un carácter especial  
- Email debe ser único en la base de datos  
- Todos los errores retornan JSON del tipo:

{  
 "mensaje": "Texto descriptivo del error"  
}

# Ejecución del proyecto

1. Clonar repositorio y compilar:

git clone <repo>  
cd userapi  
mvn clean install

2. Levantar la aplicación:

mvn spring-boot:run

# Swagger UI

Disponible en: http://localhost:8080/swagger-ui.html

Ejemplo de request:

{  
 "name": "Juan Rodriguez",  
 "email": "juan@rodriguez.org",  
 "password": "Hunter2024#",  
 "phones": [  
 {  
 "number": "1234567",  
 "citycode": "1",  
 "contrycode": "57"  
 }  
 ]  
}

# Test unitarios

Ubicados en src/test/java/com/bci/userapi/service/UserServiceTest.java

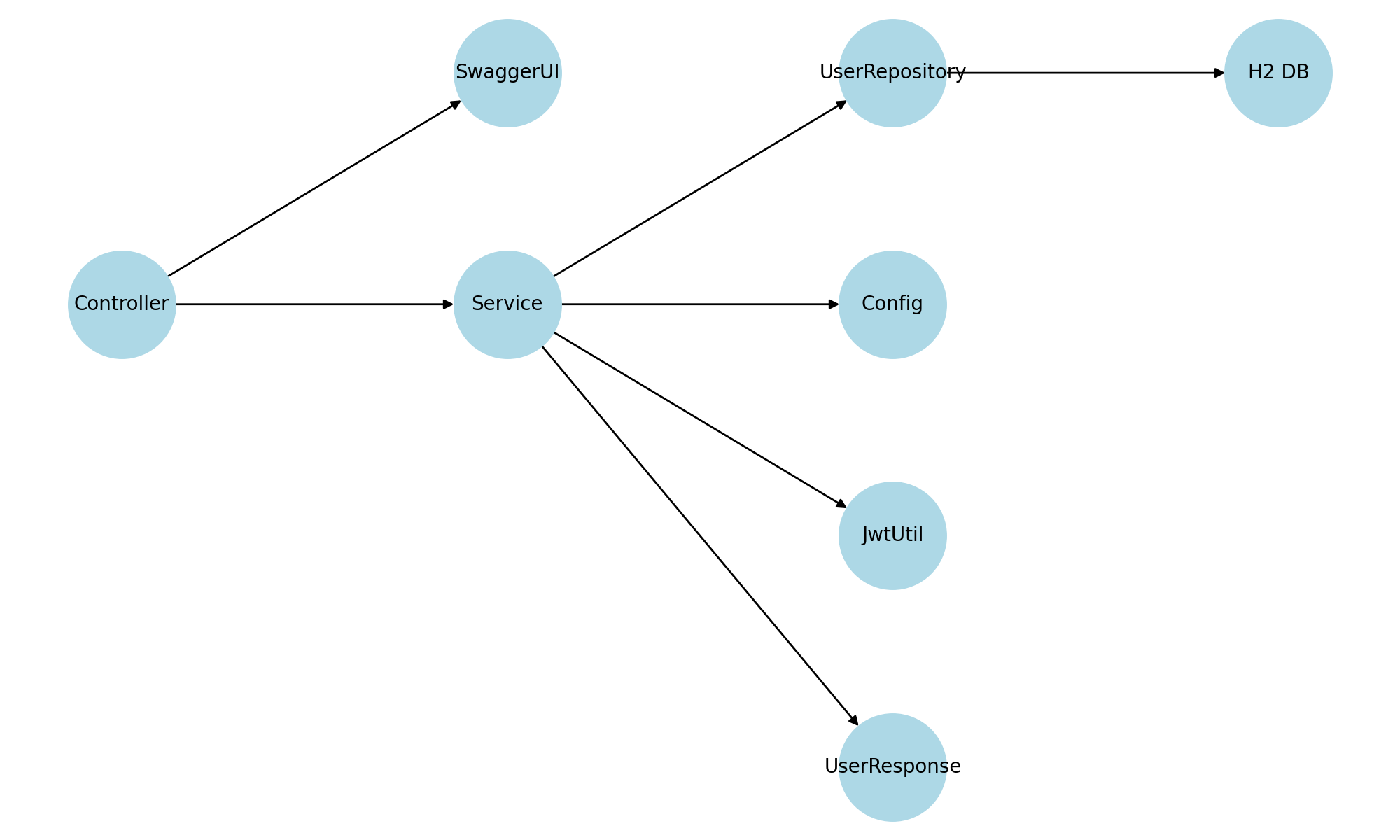
- Registro exitoso  
- Correo duplicado  
- Contraseña inválida  
- Email inválido

Ejecutar con: mvn test

# Diagrama de la solución

Este diagrama ilustra la interacción entre las capas del sistema:

* El Controller recibe la petición y la pasa al Service
* El Service interactúa con UserRepository, JwtUtil, y los objetos de configuración
* Finalmente se genera un UserResponse y Swagger permite probar la API visualmente



# Notas de seguridad

- /api/register es público  
- Cualquier otro endpoint debe ir protegido por JWT  
- Swagger, consola H2 y docs también están excluidos del filtro JWT

# Autor

Desarrollado por Héctor Miguel Vergara Canevaro como parte de la prueba técnica para BCI.

Listo para subir a GitHub.