Benjamin Dalmau Comision A

Parcial Programación 3 - ADN Mutante

URL:https://parcialdnamutantes.onrender.com

1. Introducción

El presente documento detalla el desarrollo de una API REST diseñada para detectar secuencias de ADN mutante. La API permite a los usuarios enviar una secuencia de ADN a través de un endpoint específico y, con base en la lógica implementada, determinar si el ADN corresponde a un humano mutante o no.

El proyecto no solo implica la creación de una API, sino que también abarca la implementación de una base de datos H2 que registra cada ADN verificado. Además, se proporciona un servicio adicional que permite a los usuarios obtener estadísticas sobre las verificaciones de ADN, facilitando el análisis de la relación entre los ADN mutantes y humanos.

2. Descripción del Proyecto

Funcionalidades Principales

1. **Detección de ADN Mutante**:
   - La API recibe una secuencia de ADN en formato JSON a través del endpoint `/mutant/`.
   - Al analizar la secuencia, se determina si contiene las combinaciones de letras que indican la presencia de características mutantes. Si la secuencia es identificada como mutante, la API responde con un código de estado HTTP 200 (OK). En caso contrario, devuelve un código 403 (Forbidden).
2. **Registro de ADN Verificado**:
   - La API incluye la funcionalidad de almacenar cada ADN analizado en una base de datos, asegurando que cada registro sea único. Esto permite realizar un seguimiento de las verificaciones y evita la duplicación de datos.
3. **Estadísticas de Verificación**:
   - Se implementa un endpoint adicional, `/stats`, que proporciona estadísticas sobre las verificaciones de ADN. Este servicio devuelve un JSON con la cantidad de ADN mutante y humano verificado, así como la relación entre ellos. Esto es útil para evaluar el desempeño de la API y para futuras investigaciones.

Tecnologías Utilizadas

- **Lenguaje de Programación**: Java
- **Framework**: Spring Boot
- **Base de Datos**: H2
- **Protocolo de Comunicación**: HTTP/REST
- **Herramientas de Desarrollo**: Gradle para la gestión de dependencias y construcción del proyecto.

3. Endpoints de la API

- Documentación de los endpoints que has creado, incluyendo:
  - **/mutant/**:
    - Método: POST
    - Descripción: Detecta si un ADN es mutante.
    - Parámetros: Ejemplo del JSON que se espera.
    - Respuestas posibles (200 OK, 403 Forbidden).
  - **/stats**:
    - Método: GET
    - Descripción: Devuelve estadísticas de verificaciones de ADN.
    - Respuesta esperada: Formato JSON con `count_mutant_dna`, `count_human_dna`, y `ratio`.

4. Pruebas Automáticas

Se utilizaron pruebas automáticas para garantizar que la API funcione como se esperaba.

1. Estrategia de Pruebas

Se han implementado pruebas unitarias y de integración utilizando el framework de pruebas JUnit junto con Mockito. Esto permite simular el comportamiento de los componentes y verificar su funcionamiento en diferentes escenarios.

Ejemplo de prueba:

```
@Test

   public void testIsMutant_WhenNotMutant_DetectsNotMutant() {

      String[] dna = new String[]{"ATGCGA", "CAGTGC", "TTATGT",
"AGATAG", "CCCTGA", "TCACTG"};

      DnaRequest dnaRequest = new DnaRequest();

      dnaRequest.setDna(dna);


Mockito.when(this.mutantService.esMutante(dna)).thenReturn(false);


Mockito.when(this.dnaRepository.existsByDnaSequence("ATGCGA...")).thenR
eturn(false);
```

```
      ResponseEntity<String> response =
this.mutantController.isMutant(dnaRequest);

      Assertions.assertEquals(HttpStatus.FORBIDDEN,
response.getStatusCode());

      Assertions.assertEquals("Not a mutant", response.getBody());

   }
```

- Se crea un arreglo de cadenas (`String[]`) que representa la secuencia de ADN que se va a probar.
- Se crea un objeto `DnaRequest` y se le asigna el arreglo de ADN.

Simulación de Comportamiento:

- `Mockito.when(this.mutantService.esMutante(dna)).thenReturn(false);`: Aquí se simula que el servicio `mutantService` retornará `false` al invocar el método `esMutante`, lo que significa que la secuencia de ADN no es mutante.
- `Mockito.when(this.dnaRepository.existsByDnaSequence("ATGCGA...")).thenReturn(false);`: Simula que el repositorio de ADN no encuentra la secuencia proporcionada.

Llamada al Controlador:

- `ResponseEntity<String> response = this.mutantController.isMutant(dnaRequest);`: Se realiza la llamada al método del controlador que se está probando.

Verificaciones:

- `Assertions.assertEquals(HttpStatus.FORBIDDEN, response.getStatusCode());`: Verifica que el código de estado de la respuesta sea 403 (Forbidden).
- `Assertions.assertEquals("Not a mutant", response.getBody());`: Verifica que el cuerpo de la respuesta contenga el mensaje "Not a mutant".

9. Conclusiones

. A lo largo del proceso, se han logrado los siguientes puntos:

1. **Detección de ADN Mutante**:
   - Se ha desarrollado un algoritmo eficaz capaz de analizar secuencias de ADN y detectar características mutantes.
2. **Integración de Base de Datos**:

    - La inclusión de una base de datos H2 permite el almacenamiento eficiente de registros de ADN verificados.
3. **Estadísticas y Análisis**:
    - La funcionalidad de estadísticas proporciona información valiosa sobre las verificaciones de ADN, facilitando el monitoreo y la evaluación del rendimiento de la API..
4. **Pruebas y Calidad del Código**:
    - La implementación de pruebas automáticas asegura que la API funcione correctamente y ayuda a mantener un alto estándar de calidad en el código.

Futuras Mejoras

Algunas áreas a considerar para futuras iteraciones incluyen:

- **Optimización del Algoritmo**: Continuar refinando el algoritmo de detección para mejorar su precisión y velocidad.
- **Ampliación de Funcionalidades**: Explorar la posibilidad de agregar nuevos endpoints o características que mejoren el funcionamiento.
- **Mejorar el funcionamiento de host:** Lograr hostear la API me generó demasiados dolores de cabeza hasta que funcionó y creo que es algo que puedo mejorar para el futuro para comprender mejor su uso y mejorar su funcionalidad.

En conclusión, la API de Detección de ADN Mutante ncumple con los requisitos establecidos para la entrega pero es un proyecto que puede ser ampliamente mejorado en todos los aspectos.