Detección de Mutantes

Prueba técnica - Mercado Libre

Leandro Abraham

El problema

Se considera mutante si se encuentra más de una secuencia de cuatro letras iguales, de forma oblicua, horizontal o vertical en la matriz resultante de el arreglo de Strings.
Por ejemplo:

No-Mutante

A	T	G	C	G	A
C	A	G	T	G	C
T	T	A	T	T	T
A	G	A	C	G	G
G	C	G	T	C	A
T	C	A	C	T	G

Mutante

A	T	G	C	**G**	A
C	**A**	G	T	**G**	C
T	T	**A**	T	**G**	T
A	G	A	**A**	**G**	G
C	**C**	**C**	**C**	T	A
T	C	A	C	T	G

La solución

Se implementó una API REST para detectar si alguien es mutante basándose en su ADN. Esta API recibe como parámetro un array de Strings que representan cada fila de una tabla de (NxN) con la secuencia del ADN. Las letras de los Strings solo pueden ser: (A,T,C,G), las cuales representa cada base nitrogenada del ADN.
El servicio “/mutant/” puede detectar si un humano es mutante enviando la secuencia de ADN mediante un HTTP POST con un Json el cual tenga el siguiente formato:

```
{
"dna":["ATGCGA","CAGTGC","TTATGT","AGAAGG","CCCCTA","TCACTG"]
}
```

En caso de verificar un mutante, devuelve un HTTP 200-OK, en caso contrario un 403-Forbidden
Se puede usar este servicio en el siguiente URL, usando el método POST y pasando el Json correspondiente:
http://ec2-18-231-192-215.sa-east-1.compute.amazonaws.com:8080/mutant/

El servicio “/stats/” devuelve un Json con las estadísticas de las verificaciones de ADN, por ejemplo:
{“count_mutant_dna”: 40, “count_human_dna”: 100: “ratio”: 0.4}
Se puede usar este servicio en el siguiente URL, usando el método GET:
http://ec2-18-231-192-215.sa-east-1.compute.amazonaws.com:8080/stats/

De forma accesoria se implementó un servicio "/clearData/" que permite limpiar la base de datos donde se han registrado los resultados de análisis, el cual se puede acceder en el siguiente URL, usando el método GET:
http://ec2-18-231-192-215.sa-east-1.compute.amazonaws.com:8080/clearData/

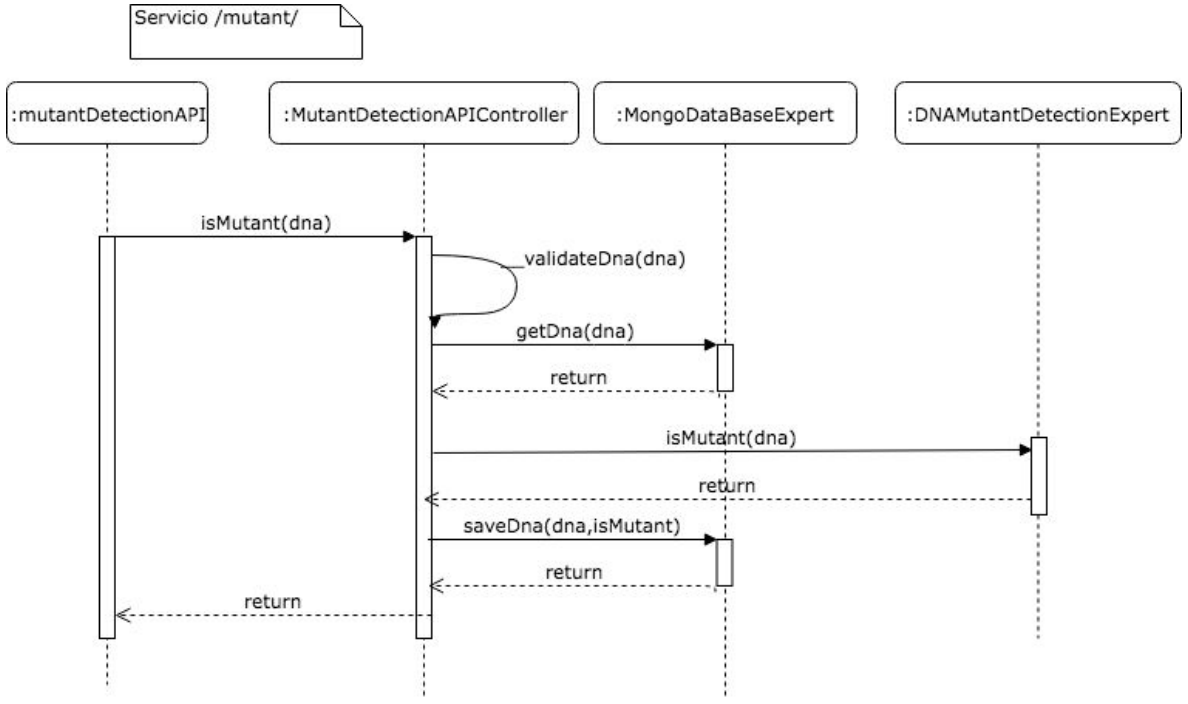
Arquitectura de la solución

La API mencionada se implementó como una aplicación Flask en lenguaje Python y como motor de base de datos se consideró una base de datos no relacional MongoDB. El código fuente de esta API se encuentra en un repositorio público en github y la misma está alojada y accesible en una instancia EC2 Ubuntu server accesible a través del siguiente link:

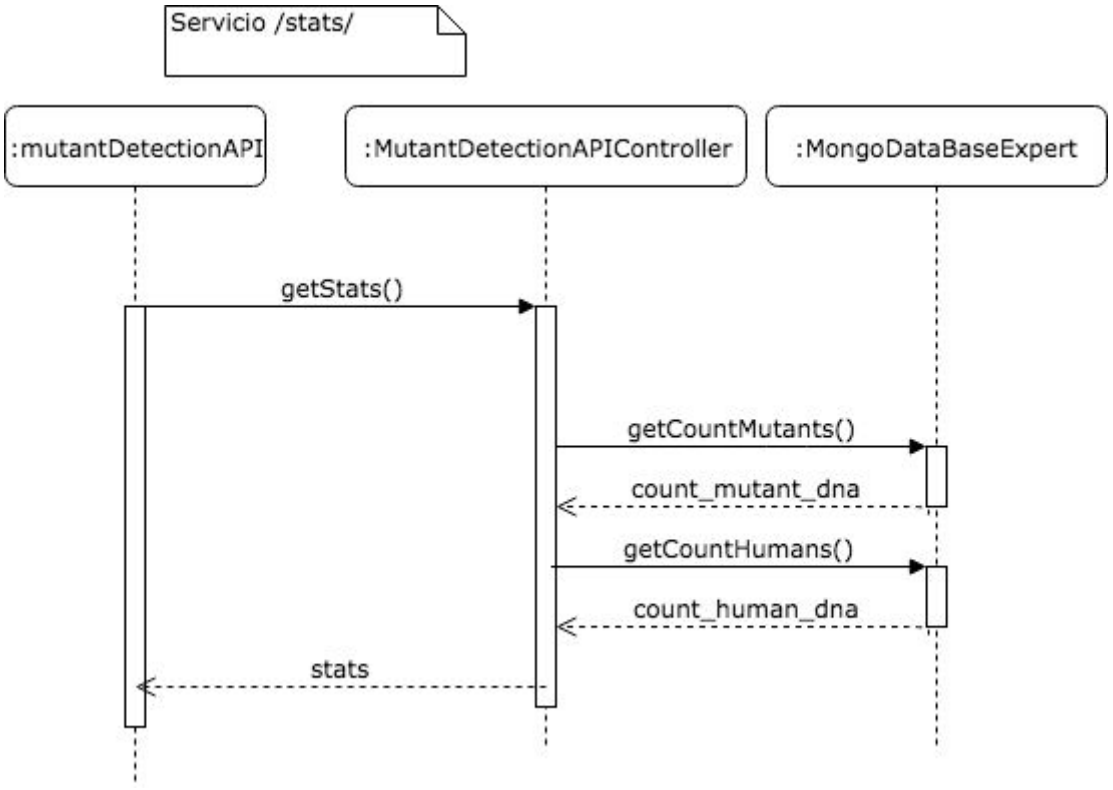
http://ec2-18-231-192-215.sa-east-1.compute.amazonaws.com:8080

La aplicación Flask es el punto de entrada o interfaz de API la cual está implementada en el archivo ***mutantDetectionAPI.py.*** Desde esta aplicación se pasan mensajes a un objeto de tipo ***MutantDetectionAPIController*** el cual es el encargado de manejar el paso de mensajes entre la interfaz y los expertos: ***DNAMutantDetectionExpert***, encargado de analizar un ADN y decidir si corresponde a un mutante o no a través de su método *isMutant*; y el ***MongoDataBaseExpert***, encargado de escribir y leer de la base de datos MongoDB.

El diagrama de secuencia del “caso de uso” o servicio /mutant/ para un camino básico donde el *dna* a analizar es correcto pero no se ha analizado antes, se presenta a continuación



Además, el diagrama de secuencia del “caso de uso” o servicio /stats/ se presenta a continuación



Tests automáticos

Dentro de la carpeta raíz del proyecto existe un archivo ***tests.py*** que es un script python que ejecuta una serie de tests para validar el funcionamiento de la API. Los resultados de la ejecución de ese script están además volcados al archivo ***testsResults.txt*** y ese mismo contenido se presenta a continuación también en este documento:

Testeando clearData

Respuesta
{"count_human_dna":0,"count_mutant_dna":0,"ratio":0}
 status code = 200
Pasó el test

Testeando /mutant/ con un mutante:
{"dna":["TTGCTG","CTGTAC","TTTACG","AAATGG","CCCTTG","TCACTG"]}
Respuesta
{"dna":["TTGCTG","CTGTAC","TTTACG","AAATGG","CCCTTG","TCACTG"],"isMutant":true}
 status code = 200
Pasó el test

Testeando /mutant/ con un humano:
{"dna":["TAGCTG","CTGTAC","TTTACA","AAATGG","CCCTTG","TCACTG"]}
Respuesta
{"dna":["TAGCTG","CTGTAC","TTTACA","AAATGG","CCCTTG","TCACTG"],"isMutant":false
}
status code = 403
Pasó el test

Testeando /mutant/ con otro mutante:
{"dna":["TAGCTG","CTGTAC","TTTACA","TAATGG","TCCTTG","TCACTG"]}
Respuesta
{"dna":["TAGCTG","CTGTAC","TTTACA","TAATGG","TCCTTG","TCACTG"],"isMutant":true}
status code = 200
Pasó el test

Testeando /mutant/ con el mismo mutante:
{"dna":["TAGCTG","CTGTAC","TTTACA","TAATGG","TCCTTG","TCACTG"]}
Respuesta
{"dna":["TAGCTG","CTGTAC","TTTACA","TAATGG","TCCTTG","TCACTG"],"isMutant":true}
status code = 200
Pasó el test

Testeando /stats/
Respuesta
{"count_human_dna":1,"count_mutant_dna":2,"ratio":2.0}
status code = 200
Pasó el test

Testeando /mutant/ con otro humano:
{"dna":["TAGCTG","CTGTAC","TCTACA","AGATGG","CCCTTG","TCACTA"]}
Respuesta
{"dna":["TAGCTG","CTGTAC","TCTACA","AGATGG","CCCTTG","TCACTA"],"isMutant":false
}
status code = 403
Pasó el test

Testeando /mutant/ con otro humano:
{"dna":["GAGCTG","CTGTAG","ACTACA","AGATCG","CCCTTG","TCACTA"]}
Respuesta
{"dna":["GAGCTG","CTGTAG","ACTACA","AGATCG","CCCTTG","TCACTA"],"isMutant":fals
e}
status code = 403
Pasó el test

Testeando /mutant/ con otro humano:
{"dna":["GAGCTG","CTGTAG","ACTCCA","AGACCA","CCATTT","TCACGA"]}
Respuesta
{"dna":["GAGCTG","CTGTAG","ACTCCA","AGACCA","CCATTT","TCACGA"],"isMutant":fals
e}
status code = 403
Pasó el test

Testeando /stats/
Respuesta
{"count_human_dna":4,"count_mutant_dna":2,"ratio":0.5}
status code = 200
Pasó el test

Testeando /mutant/ con un dna que no es matriz cuadrada:
{"dna":["GAGCTG","CTGTA","ACTCCA","AGA","CCATTT","TCACGA"]}
Respuesta
{"message":"DNA No valido para analizar","status":"Error"}
status code = 403
Pasó el test