



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Escola Tècnica
Superior d'Enginyeria
Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

**Desarrollo de una aplicación web de análisis y
consultas en una base de datos biológica**

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Vicente Teófilo Marín Gil

Tutor: Dr. Felip Lluís Hurtado Oliver

Curso 2016/2017

Resumen

Este proyecto presenta el desarrollo de una aplicación web responsive, alcanzable desde cualquier punto de acceso a la red, que está enfocada al almacenamiento de muestras genómicas y la facilitación del análisis sobre las mismas basado en el framework Django. Dicho sistema de consulta nace, en colaboración con la empresa SECUGEN, como herramienta de apoyo para el almacenamiento y estudio de la salida del software NextGENe V2.4.

Para conseguir el objetivo mentado, ha sido desarrollada una interfaz gráfica para el envío y el procesamiento de las muestras en formato CSV, así como un sistema de consulta con un potente filtrado para el estudio sobre las muestras que permite trabajar sobre una única o todo el conjunto y localizar, así, en tiempos razonables, mutaciones de conjuntos de cromosomas, conjuntos de posiciones cromosómicas, genes concretos, posiciones cromosómicas referentes a mutaciones concretas y/o rangos de frecuencias de aparición. Además, es posible realizar anotaciones sobre las posiciones cromosómicas resultantes de la búsqueda.

Gracias a una arquitectura basada en la filosofía del patrón de diseño Modelo-Vista-Controlador de Django, se consigue un acoplamiento débil, logrando una estricta separación de las partes de la aplicación. Es por ello que, pese a la utilización en concreto de MariaDB en este proyecto, queda disponible la fácil integración de otras bases de datos como PostgreSQL, MySQL, SQLite u Oracle.

Con el fin de validar la aplicación, se ha puesto a disposición de la empresa la versión beta de la misma para su testeo y corrección de problemas, y obtener así un software útil y funcional.

Palabras clave: sistema de búsqueda, genoma, NextGENe, aplicación web, Django

Resum

Aquest projecte presenta el desenvolupament d'una aplicació web responsive, accessible desde qualsevol punt d'accés a la red, enfocada a l'emmagatzemament de mostres genòmiques i la facilitació de l'anàlisi sobre les mateixes basat en el framework Django. Dita aplicació naix, en col·laboració amb l'empres SECUGEN, com ferramenta de recolzament per l'emmagatzematge i estudi de l'eixida del software NetGENe V2.4.

Per aconseguir l'objectiu esmentat ha sigut desenvolupada una interfície gràfica per l'enviament i processament de les mostres en format CSV, així com un buscador amb un potent filtratge per l'estudi sobre les mostres que permet treballar amb una única o tot el conjunt i localitzar així, en temps raonables, gens concrets, posicions cromosòmiques referents a mutacions concretes i/o rangs de freqüències d'aparició.

Gràcies a una arquitectura basada en la filosofia del patró de disseny Model-Vista-Controlador de Django, s'aconsegueix un acoblament dèbil assolint una estricta separació de les parts de l'aplicació. És per això que, malgrat la utilització en concret de MariaDB en aquest projecte, queda disponible la fàcil integració d'altres bases de dades com PostgreSQL, MySQL, SQLite o Oracle.

Amb la fi de validar l'aplicació s'ha posat a disposició de l'empresa la versió beta de la mateixa, per al testeig i correcció de problemas, obtenint com resultat un software útil i funcional.

Paraules clau: sistema de búsqueda, genoma, NextGENe, aplicación web, Django

Abstract

This paper presents the development of a responsive web application, which is accessible from any network access point. This application is destined to store genomic samples and to facilitate their analysis, based on the Django framework. This search application was born out, in collaboration with the company SECUGEN, as a support tool for the software NextGENe V2.4 to storage and to study its output.

In order to achieve the aforementioned, a graphic interface that facilitates the sending and processing of the samples in a CSV file has been developed. Furthermore, a query system with a powerful filter allows working with a single sample or the whole set of them and localising, in reasonable times, chromosome mutations in organized groups, as well as chromosomal position groups, specific genes, chromosomal positions relating to particular mutations and/or their frequency ranges of occurrence.

Thanks to an architecture based on the Django's Model-View-Controller philosophy design, a weak pattern of loose coupling is obtained, achieving so a strict separation of the different parts of the application. Therefore, despite working only with MariaDB in this project, some other databases, such as PostgreSQL, MySQL, SQLite or Oracle, can be easily integrated, if desired.

In order to validate the application, a beta version of it has been made available to the company to test it and to correct the possible problems. The final result is an useful and functional software.

Keywords: search system, genome, NextGENe, web application, Django

Tabla de contenidos

Tabla de contenidos.....	6
Tabla de figuras	9
1. Introducción.....	11
1.1. Motivación	11
1.2. Objetivos.....	11
1.3. Estructura de la memoria	13
2. Antecedentes.....	13
2.1. Allele*Frequencies	14
2.2. ALFRED.....	14
2.3. FINDbase	15
2.4. NCBI.....	17
2.5. Conclusiones.....	18
3. Análisis	19
<p>De aquí en adelante procedemos a realizar el análisis del proyecto. Se presenta el flujo de trabajo dentro de la aplicación, desde el procesado de las muestras proporcionadas por el software NextGENe V2.4, y el almacenaje de las mismas, hasta la consulta directa sobre la base de datos de mutaciones de posiciones cromosómicas. El objetivo principal de la aplicación es ser usada para analizar y comparar muestras, por lo que las funciones más representativas son las consultas. Hay tres tipos de consulta:</p>	
3.1. Inicio	20
3.1.1. Requisitos y algoritmos	20
3.2. Error en el fichero	21
3.2.1. Requisitos y algoritmos.....	21
3.3. Búsqueda para seleccionar una muestra concreta.....	22
3.3.1. Requisitos y algoritmos.....	22
3.4. Búsqueda sobre una muestra concreta.....	23
3.4.1. Requisitos y algoritmos	23
3.5. Búsqueda genérica	24
3.5.1. Requisitos y algoritmos	25
3.6. Resultados.....	25
3.6.1. Requisitos y algoritmos.....	26
3.7. Rangos	26

3.7.1.	Requisitos y algoritmos	26
4.	Diseño	27
4.1.	Diseño del sistema: DJANGO y el patrón MVC	27
4.2.	Diseño de la capa de negocios y persistencia.	28
4.3.	Diseño de la capa de presentación	30
4.4.	Tecnologías utilizadas	31
4.4.1.	Tecnologías utilizadas en el servidor	31
4.4.1.1.	MariaDB	31
•	El desarrollo de MariaDB es más abierto y activo	31
•	Publicaciones de seguridad más rápidas y transparentes.....	31
•	Más motores de almacenamiento.....	32
•	Mejor rendimiento.....	32
•	La administración de Oracle es incierta	32
•	Compatible y fácil de migrar	32
4.4.2.	Tecnologías utilizadas en la aplicación	33
4.4.2.1.	HTML	33
4.4.2.1.	JavaScript	33
	Lenguaje script por excelencia para el uso en página web, da funcionalidad al código <i>hueco</i> del HTML, en este proyecto se han utilizado algunas de sus librerías más importantes:.....	33
4.4.2.2.	<i>Bootstrap</i>	33
4.4.2.3.	<i>CSS</i>	33
5.	<i>Evaluación del desarrollo</i>	34
5.1.	<i>Varios ejemplos concretos</i>	34
5.1.1.	<i>Buscando sobre una muestra concreta</i>	34
5.1.2.	<i>Una búsqueda genérica</i>	41
5.1.3.	<i>Subiendo un fichero con errores</i>	43
5.2.	<i>Evaluación del sistema:</i>	45
6.	Conclusiones	49
6.1.	<i>Trabajo realizado</i>	49
6.2.	<i>Posibles mejoras y versiones futuras</i>	49
7.	Bibliografía.....	51
8.	Anexos.....	53
	Anexo 1: Especificaciones de la BD	53

ANEXO 2: FORMATO DE LOS CSV DE RANGOS	54
ANEXO 3: FORMATO DE LOS CSV DE MUESTRAS	55
ANEXO 4: TABLA DE TIEMPOS	56
8.1. Glosario	58
8.2. Definición de abreviaturas	59

Tabla de figuras

Ilustración 1. Grafo de dependencias de mutaciones entre poblaciones.	16
Ilustración 2 Mapa de mutaciones genéticas por frecuencia y poblaciones.	16
Ilustración 3. Mapa de mutaciones genética, ejemplo de uso	17
Ilustración 4. Descripción del flujo de trabajo	19
Ilustración 5. Flujo de trabajo, <i>inicio</i> al detalle	20
Ilustración 6. Flujo de trabajo, <i>errores</i> al detalle.....	21
Ilustración 7. Flujo de trabajo, <i>buscador</i> al detalle	23
Ilustración 8. Flujo de trabajo, <i>buscador</i> al detalle 2.....	25
Ilustración 9. Flujo de trabajo, <i>resultados</i> al detalle	25
Ilustración 10. Flujo de trabajo, <i>rangos</i> al detalle	26
Ilustración 11. Despachador de URLs Django	29
Ilustración 12. Pantalla inicial.....	34
Ilustración 13. Selección del botón <i>búsqueda independiente</i>	35
Ilustración 14. Menú pantalla emergente	35
Ilustración 15. Buscador, detalle fechas.....	36
Ilustración 16. Buscador de muestras, seleccionando criterio	36
Ilustración 17. Buscador de muestras, criterio añadido.....	37
Ilustración 18. Resultado buscdor de muestras	37
Ilustración 19. Borrando una muestra	38
Ilustración 20. Buscador general de una muestra concreta	38
Ilustración 21. Añadiendo un filtro en el buscador dada una muestra concreta.....	39
Ilustración 22. Filtro añadido en el buscador por fichero	39
Ilustración 23. Resultados respuesta del servidor a una consulta	40
Ilustración 24. Mutación de las posiciones buscadas	40
Ilustración 25. Frecuencia de las mutaciones buscadas.....	40
Ilustración 26. Detalle botón “Búsqueda independiente”	41
Ilustración 27. Buscador genérico detalle añadiendo filtro.....	41
Ilustración 28. Buscador genérico, detalle filtro añadido	42
Ilustración 29. Resultado de búsqueda vacía.....	42

Ilustración 30. Arrastrando una muestra para subirla	43
Ilustración 31. Seleccionando una muestra, desde el explorador de archivos, para subirla	44
Ilustración 32. Botón de validación y registro de la muestra	44
Ilustración 33. Salida de errores al subir muestra corrupta	45
Ilustración 34. Gráfica de tiempo de recálculo de frecuencias (s)	46
Ilustración 35. Gráfico de la media de las frecuencias de la 2 a la 49 en comparación con la media de las medias (s).....	46
Ilustración 36. Gráfica de tiempo de listado de todas las posiciones de las muestras (s)	47
Ilustración 37. Gráfico de la media del tiempo de devolución de todas las entradas en comparativa con la media de las medias obtenidas (s).....	48

1. Introducción

En este capítulo se presentan las motivaciones que dan vida y razón de ser al proyecto así como los objetivos que se han buscado cumplir al final del trabajo y la estructura que seguirá la memoria.

1.1. Motivación

Los grandes avances tecnológicos de los últimos años respecto a la secuenciación genética han permitido una rápida evolución en ámbitos de investigación y uso clínico en múltiples áreas, sin embargo, debido a que esta tecnología provee de grandes cantidades de datos, aparece la necesidad de la creación de herramientas que permitan automatizar el procesamiento, análisis y estudio de las muestras obtenidas en tiempos mínimos.

En un país que ha visto dificultades en los últimos años respecto a I+D, empresas como Secugen ven mermado su potencial debido a la necesidad de herramientas potentes capaces de manejar estas grandes cantidades de datos.

Secugen, que es una empresa especializada en la Secuenciación Automática del ADN y el Análisis Genético, se constituye en 2005 como una spin-off del Centro de Investigaciones Biológicas (CIB). Con más de 800 clientes colabora con entidades como el CSIC u otros centros de investigación, hospitales, universidades, empresas privadas y asociaciones de enfermedades raras de origen genético, dedicándose al estudio y diagnóstico de demencias y Alzheimer, del Síndrome de Dravet, la fibrosis quística y enfermedades genéticas raras entre otras.

Tras la puesta en contacto con Secugen, se acuerda el desarrollo con carácter urgente de una herramienta que substituirá el estudio de frecuencias y análisis manual de las mutaciones mediante excel en pos de una aplicación web que permite un cálculo de los resultados y filtrado automatizado agilizando enormemente el trabajo. A lo largo de los meses que dura el proyecto y tras sucesivas reuniones se va dando forma a la aplicación atendiendo a las necesidades y peticiones de la empresa tanto sobre funcionalidad interna como de la interfaz de la solución.

1.2. Objetivos

Los objetivos que se deciden para el presente proyecto son los siguientes:

- Crear un sistema accesible desde la red así que entre otras cosas sea:
 - *Responsive*, es decir, accesible tanto desde grandes resoluciones como pequeñas pantallas como tablets o móviles.
 - Que tenga en cuenta aspectos de seguridad como la denegación de servicio o CSRF (Cross-site request forgery).

Desarrollo de una aplicación web de análisis y consultas en una base de datos biológica

- Utilizar un software que tenga en cuenta aspectos de seguridad asegurando cierta protección de los ataques comunes, como la denegación de servicio, y garantizando un mínimo de seguridad, como el uso del CSRF (cross-site request forgery).
- Que sea fácilmente modificable y expadible, de acoplamiento débil y diseño de patrón Modelo-Vista-Controlador.
- Diseñar y especificar la base de datos necesaria para trabajar con las muestras del software NextGENe V2.4:
 - Implementar un sistema de subida para las muestras en formato csv que procese el fichero con un sistema tolerante a errores y si los hubiere mostrar por pantalla el conjunto de los mismos y no uno a uno.
 - Permitir el borrado de las muestras almacenadas en la base de datos.
 - Implementar un sistema que permita trabajar con dos versiones de muestras simultáneamente permitiendo combinarlas para maximizar la exactitud de los cálculos.
 - Diseñar un algoritmo para automatizar la configuración y modificación del alcance de las dos versiones a partir de un csv especificado.
 - Crear un sistema estable y eficaz, razonablemente rápido que prevea un aumento en el tiempo del volumen de datos.
- Diseñar un sistema de consulta según las necesidades del cliente:
 - Estudiar y plantear las diferentes pantallas de la solución para que sea una funcional e intuitiva para usuarios no necesariamente familiarizados con el mundo de la informática.
 - Especial importancia al cálculo óptimo de frecuencias de las posiciones genómicas
 - Búsqueda de posiciones cromosómicas según las muestras dadas que permita filtrar según rango de cromosomas, de posiciones, por fechas o ficheros concretos, por mutaciones y/o por frecuencias.
- Validar el programa para convertirlo en una aplicación real y útil:
 - Permitiendo a los profesionales del sector trabajar con el software para apruebar su funcionamiento y comunicar errores encontrados así como propuestas de mejora.
 - Comprobar los tiempos de los procesos más costosos de la aplicación así como comprobar que los costes de espacio son razonables.

1.3. Estructura de la memoria

Esta memoria comienza con los puntos que han motivado su realización y los objetivos que se pretenden cumplir.

Seguido, se detallan los antecedentes como muestra de la realidad de esta tecnología en la actualidad, donde se describen algunos de los sistemas similares más conocidos y se realiza una comparativa respecto al propio proyecto.

En tercer lugar, se trata el análisis de la solución en el que se trata el flujo de trabajo de la aplicación así como los algoritmos utilizados y cómo fluye la información dentro del sistema.

Tras este, se aborda el diseño donde se describe el patrón modelo-vista-controlador utilizado y, como consecuencia, las diferentes capas de la aplicación. Además se detallan los módulos y las especificaciones de los algoritmos utilizados. También contempla qué base de datos se utiliza y por qué así como otras librerías utilizadas en el proyecto.

En quinto lugar se tratan los resultados, con varios ejemplos de uso de la aplicación de las funciones más representativas y la evaluación final del sistema basada en los tiempos.

Finalmente se concluye comentando los resultados del trabajo realizado, proceso y problemas así como las posibles mejoras y versiones futuras.

2. Antecedentes

En un área tecnológica tan en auge y de tanta importancia como es el estudio del lenguaje de programación de la vida, el ADN, nace en 2014 por fin en España el proyecto Spain Mutation Data Base (Spain MDB), una plataforma web para comunicar y consultar mutaciones genéticas germinales, aquellas que predisponen al desarrollo de enfermedades, y que nace de la mano del Instituto de Salud Carlos III en colaboración con la Asociación Española de Genética Humana y la sección de cáncer hereditario de la Sociedad Española de Oncología Médica. Este proyecto, basado en las recomendaciones del Human Variome Project, aconseja crear bases de datos nacionales con el objeto de recolectar toda la variabilidad genética existente en cada país. Pues es el espíritu de esta organización internacional trabajar para que toda la información de las variaciones genéticas y su efecto sobre la salud humana sea recogida, organizada, interpretada y compartida de forma abierta y gratuita (ISCIII, Gabinete de prensa, 20).

Se trata de *“un proyecto abierto a otras sociedades médicas o científicas que quieran contribuir a mejorar la información disponible a nivel nacional sobre las variantes asociadas a las enfermedades de base genética, una plataforma en línea, pública y accesible”* (ISCIII, 2016).

“La idea es que el especialista, tras el estudio genético del ADN de un paciente, si identifica una mutación, además de informar al paciente, la incluya en la base de datos para que otros profesionales tengan la oportunidad de consultarla.

Sin embargo, Alonso apunta a que es difícil de estimar el número de mutaciones que se espera conseguir ‘puesto que es la primera experiencia en España de este tipo de repositorios, todo dependerá de lo motivados que estén los profesionales para introducir mutaciones’.” (EUROPA PRESS, 2014)

A nivel internacional sí existen importantes plataformas web de consulta de bases de datos biológicas por todo el mundo como son el National Center for Biotechnology Information (NCBI) en EEUU, el European Nucleotide Archive (ENA), y el DNA Data Bank of Japan (DDBJ) . Estas almacenan información sobre proteínas, genes, cadenas, mutaciones... sin embargo las principales bases de datos internacionales biológicas no giran en torno a la frecuencia de aparición. (UPV, 2015)

A continuación, se comentan algunas de las bases de datos más importantes que sí trabajan con información sobre la frecuencia del genemoma y se analizan brevemente.

2.1. Allele*Frequencies

Esta base de datos pública contiene información sobre genes inmunes como antígenos de leucocitos huamnos (HLA), receptores de tipo inmunoglobulina de células asesinas (KIR), genes del complejo principal de histocompatibilidad de clase I (MIC y una serie de polimorfismos del gen de la citoquina. Es decir, su web nos habla de genes concretos y no un buscador genérico que albergue un gran espectro en realidad, por lo que, pese a contar con muestras de más de diez millones de individuos y, actualmente, 1486 poblaciones distintas documentadas, centra el estudio en ciertos genes y rangos de posiciones muy concretos.

También anima a colaborar “Si tiene un estudio de población que no ha sido registrado en esta BD, puede enviar sus datos ingresando la información demográfica en la sección de envío. Después de ingresar sus datos demográficos, recibirá una hoja de cálculo para ingresar las frecuencias correspondientes”. (Allele Frequencies, 2015)

2.2. ALFRED

Alfred es un acrónimo que hace referencia a “The ALlele FREquency Database”, la cual cuenta con 724 poblaciones y más de 40 millones de tablas de frecuencia.

En su web, se explica que ALFRED ha sido diseñado para hacer que los datos de frecuencias de alelos sobre poblaciones definidas antropológicamente estén disponibles para la comunidad científica y esta información sobre polimorfismos esté relacionada con las bases de datos de genoma humano. Inicialmente, ALFRED contenía principalmente datos generados en los laboratorios de K.K y J.R Kidd en el Departamento de Genética de Yale, incluyendo extensos datos no publicados. Los datos de la literatura publicada se introducen en ALFRED de forma sistemática, con un enfoque en los polimorfismos estudiados en muchas poblaciones diferentes. Su web también dice así “Los

investigadores que deseen que sus datos se introduzcan en ALFRED deben ponerse en contacto con nosotros.”

En su plataforma se explica que ALFRED es distinta de las bases de datos como DbSNP, que cataloga la variación de secuencia y que ALFRED se centra en las frecuencias de los alelos en diversas poblaciones antropológicamente definidas. Explica que no es un compendio de polimorfismos de ADN humano sino de frecuencias de polimorfismos seleccionados poniendo especial énfasis a aquellos que han sido estudiados en múltiples poblaciones.

Acaba recordando que todos los datos de ALFRED se consideran de dominio público y están disponibles para su uso en investigación y enseñanza. (U.S. National Science Foundation, 2016)

2.3. FINDbase

FINDbase se autodescribe como *“un recuerdo en línea que documenta las frecuencias de variaciones genéticas patógenas que conducen a trastornos hereditarios en diversas poblaciones en todo el mundo. Los datos iniciales provienen de informes publicados previamente, así como de información no publicada aportada por investigadores individuales antes de su publicación.*

Desde 2008, FINDbase ha experimentado una actualización importante y un aumento sustancial de contenido con la documentación de trastornos hereditarios adicionales y un conjunto completamente nuevo de marcadores farmacogenómicos. Esta información está disponible en dos módulos separados, llamados ‘Mutaciones causativas’ y ‘Marcadores farmacogenéticos’.” (Genome Variation Allele Frequencies Database Worldwide, 2016)

Por ejemplo, respecto a las mutaciones causativas, su web explica cómo la actual recopilación de datos en este módulo incluye datos de frecuencias alélicas contrastadas para más de 3800 mutaciones causantes de enfermedades a través de 26 genes fruto de más de 100.000 individuos de 92 poblaciones mundiales. Es decir, pese a la gran cantidad de datos y la internacionalización de los mismos, una vez más se trabaja con genes concretos y posiciones o rangos del genoma concretos. Se habla de genes asociados concretamente a enfermedades, un reducido número de genes cuyo efecto ya ha sido, de un modo u otro, etiquetado.

Muestran herramientas muy gráficas como el grafo de dependencias de mutaciones concretas que compara diversas poblaciones (FINDbase, 2016) o el mapa que permite ver dicha información organizada de otro modo (FINDbase, 2016).

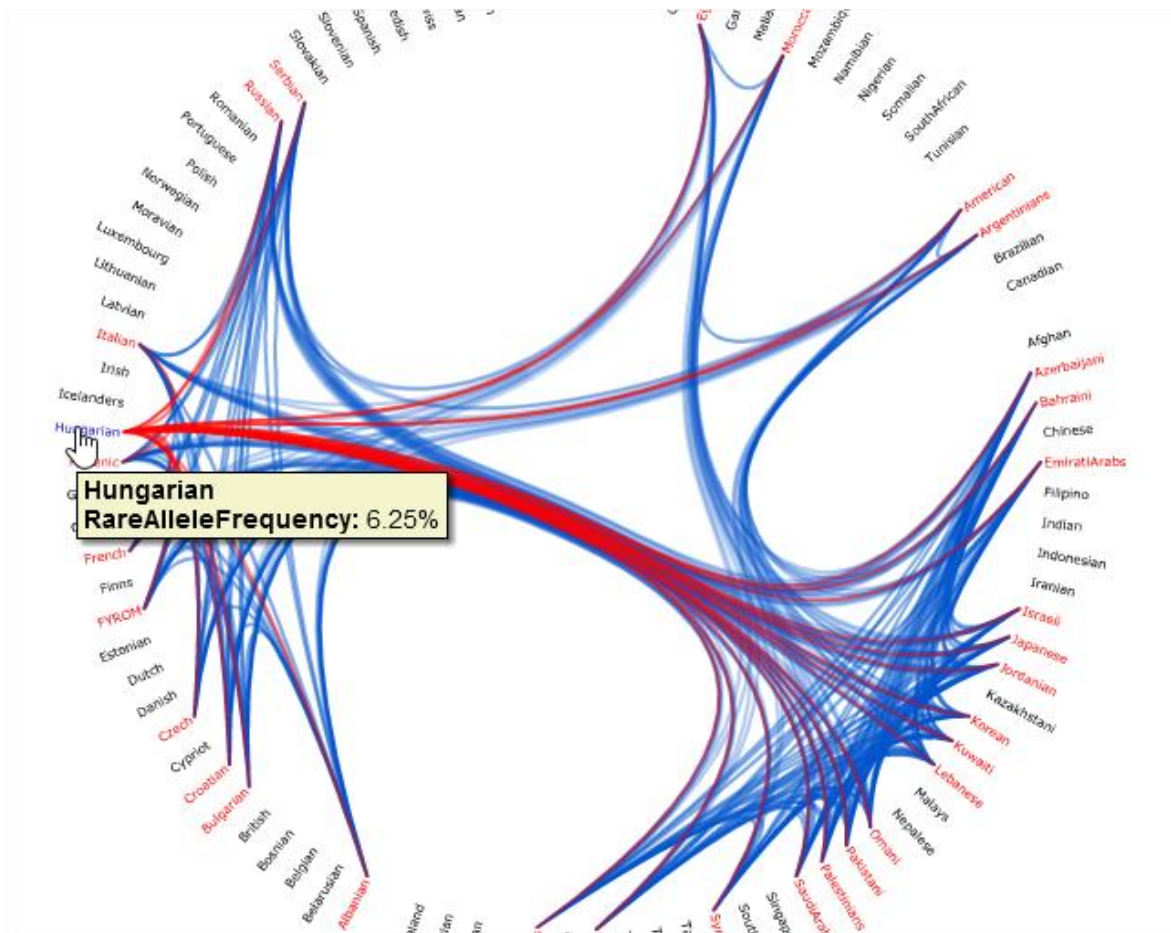


Ilustración 1. Grafo de dependencias de mutaciones entre poblaciones.

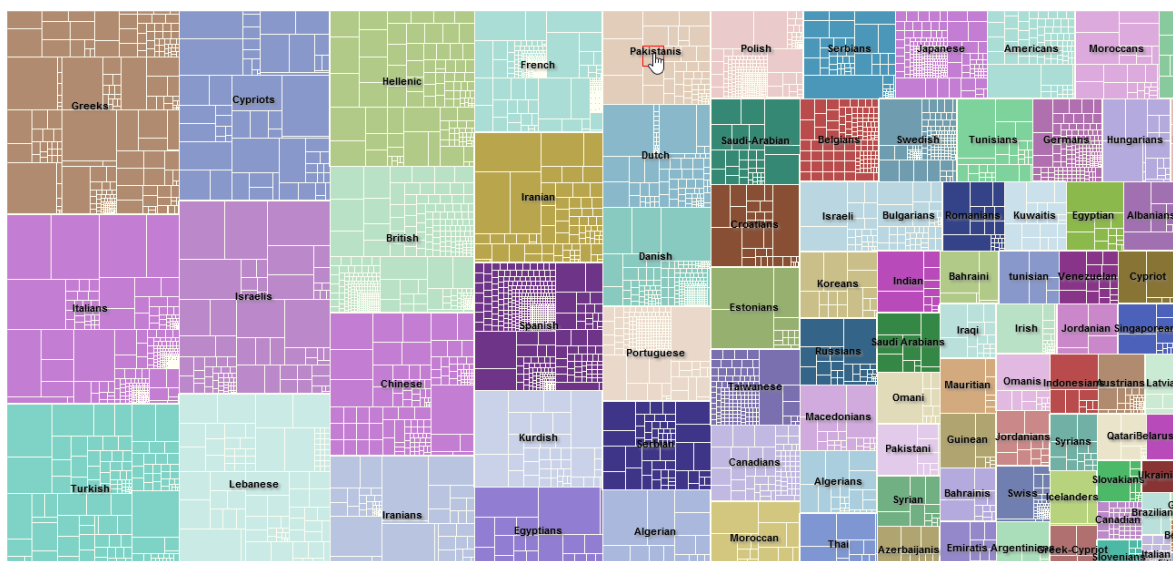


Ilustración 2 Mapa de mutaciones genéticas por frecuencia y poblaciones.

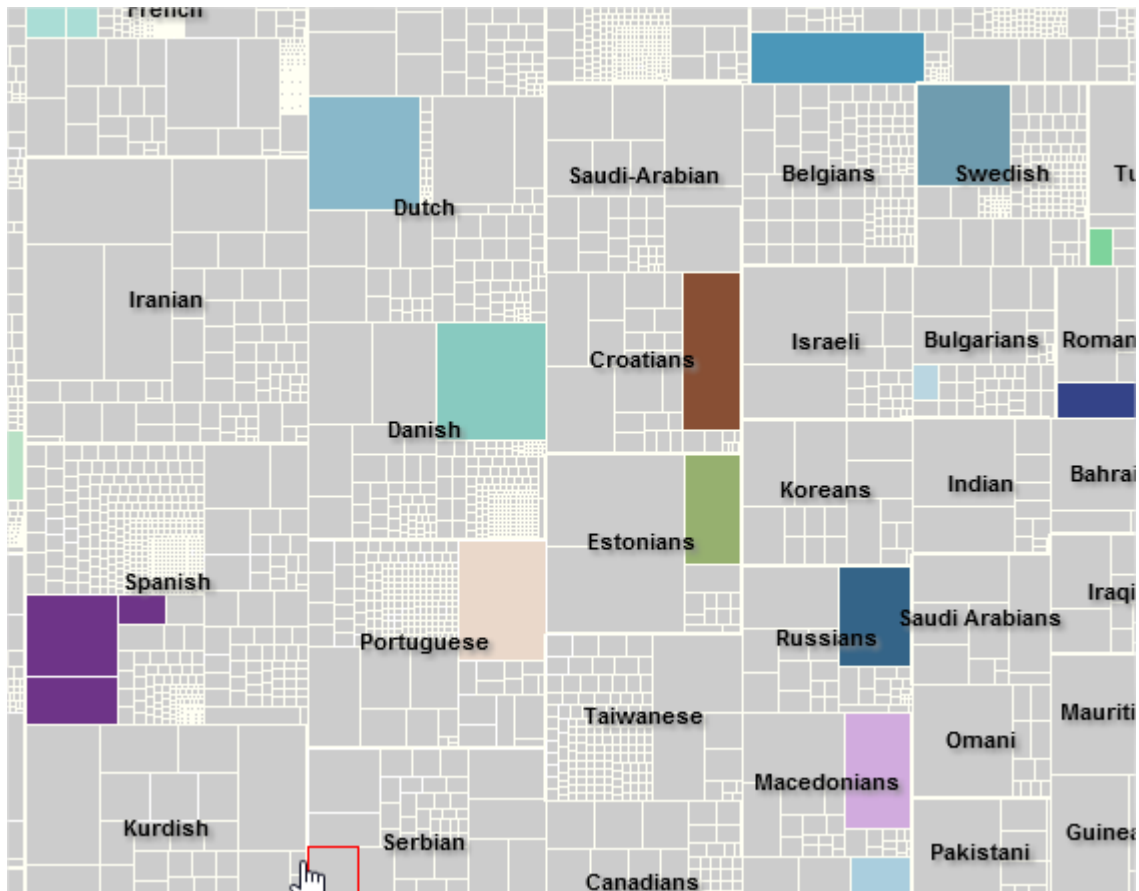


Ilustración 3. Mapa de mutaciones genética, ejemplo de uso

En este mapa observamos los genes comunes entre poblaciones, coloreados, clicando sobre de una población cualquiera. Se trata de una herramienta muy gráfica que ayuda a hacer un análisis por poblaciones mucho más visual e intuitivo. Por otro lado, esta aplicación web obliga a instalar Silverlight para trabajar con su base de datos y hacer consultas. Sin duda características interesantes a la hora de desarrollar un proyecto sobre la misma temática pero no exactamente lo que se busca.

2.4. NCBI

El importante NCBI, National Center for Biotechnology Information también habla de frecuencias respecto a sus registros. Efectivamente, al acceder a un RS concreto puede buscarse la referencia a su frecuencia según su población, aunque, por desgracia, también es bastante frecuente que aparezca un mensaje como este **“There is no frequency submission for ss167704337”** o un mensaje como este. **“No snp in above gene model(gene_id=827854) when searching by:snp having frequency**

data.” (es decir, no hay envíos sobre la frecuencia para “ss167704337” o “No se encuentran snp en el model de gen X (en ese caso asociado al cáncer) que tengan información sobre su frecuencia”), por lo que tampoco resulta ser una fuente 100% fiable de la frecuencia de las posiciones genómicas en general. (dbSNP Short Genetic Variations NCBI, 2016)

2.5. Conclusiones

Tal y como puede verse respecto a las bases de datos más importantes que tienen en cuenta la frecuencia de aparición de las mutaciones, o son de genes y demás marcadores concretos o especialmente seleccionados, o se relacionan a una enfermedad concretamente. Incluso casos en que se registran pero no hay datos sobre su frecuencia pese a contemplar el recuadro y la búsqueda sobre la misma.

Es decir, o son bases de datos muy concretas o son bases de datos más genéricas pero sin una disponibilidad total de los datos que nos interesan. De cualquier modo, si en algo coinciden todas estas plataformas es que están hechas mayormente por la colaboración de la comunidad científica y que todas ellas animan a colaborar con las mismas y a compartir las investigaciones y el saber.

Es, pues, muy difícil disponer de una aplicación donde subir con normalidad muestras propias y hacer un análisis sobre ellas, pues, pese al protocolo con el que generalmente cuentan para compartir información y que sea aceptada por las plataformas, estas tratan con genes etiquetados y no con genes por descubrir, no con genes raros, pertenecientes a enfermedades raras o no catalogados, o en el caso del NCBI, llega a incluso no contemplar siempre la información referente a frecuencia.

Esta aplicación pensada para el trabajo con muestras que destacan por hacer referencias a pacientes con enfermedades genéticas hereditarias raras y muy raras en su mayoría, difícilmente trabajará con las opciones más potentes que se encuentran relacionadas con el campo.

Es por ello que esta aplicación supone una herramienta muy interesante para el análisis, estudio, descubrimiento y catalogación de nuevos genes. Una herramienta que permite avanzar en los estudios y la investigación para que pueda colaborarse con las páginas web mentadas anteriormente, una herramienta necesaria e imprescindible para el trabajo si se quiere poder colaborar y/o trabajar con estas importantes bases de datos o, simplemente, avanzar en las investigaciones propias, más a sabiendas que muchas veces este trabajo ha sido realizado a mano mediante hojas de Excel y no de una forma automatizada en plena era de la información.

3. Análisis

De aquí en adelante procedemos a realizar el análisis del proyecto. Se presenta el flujo de trabajo dentro de la aplicación, desde el procesado de las muestras proporcionadas por el software NextGENe V2.4, y el almacenaje de las mismas, hasta la consulta directa sobre la base de datos de mutaciones de posiciones cromosómicas. El objetivo principal de la aplicación es ser usada para analizar y comparar muestras, por lo que las funciones más representativas son las consultas. Hay tres tipos de consulta:

- Búsquedas para seleccionar una muestra.
- Búsquedas genéricas de posiciones cromosómicas.
- Búsquedas de posiciones cromosómicas sobre muestras concretas.

Así mismo, también existe la posibilidad de borrar muestras ya subidas así como subir un fichero que determine o actualice los rangos pertenecientes a cada versión partiendo de la base de que pueden aparecer nuevas versiones de muestras en el futuro.

A continuación se comentan los diferentes estados con que cuenta la aplicación, cómo se llega a ellas, qué funcionalidades ofrece cada una y qué algoritmos y requisitos exige.

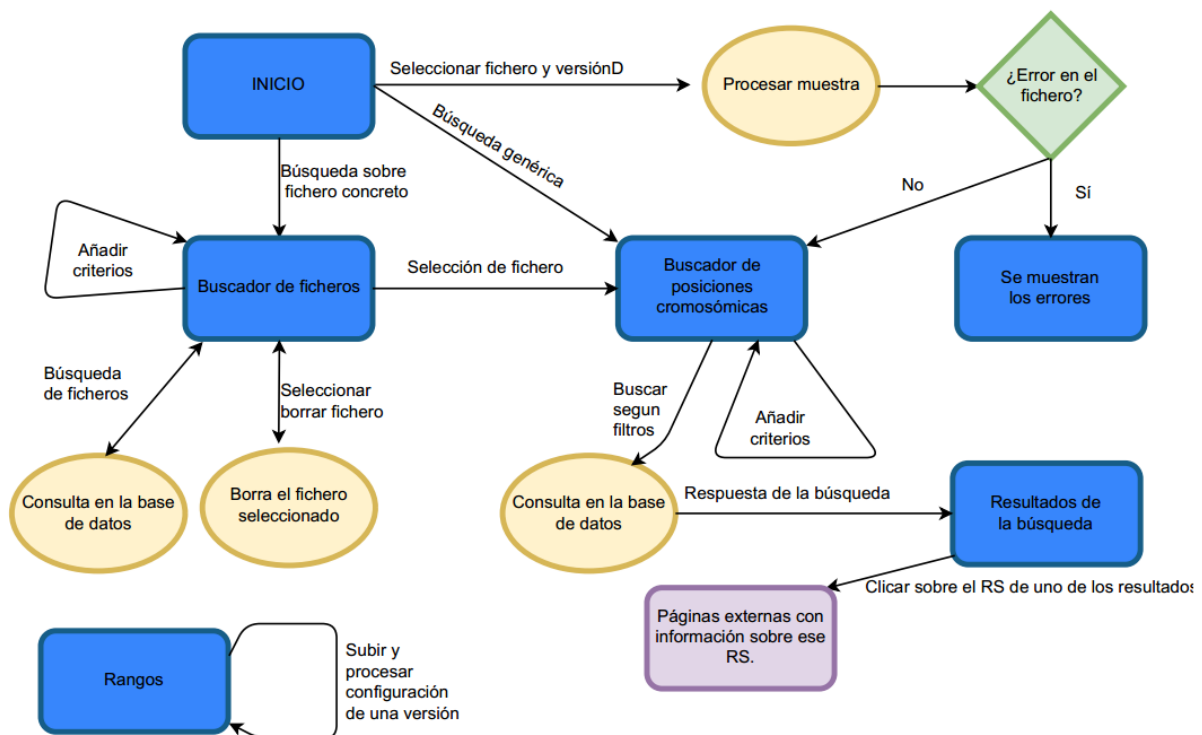


Ilustración 4. Descripción del flujo de trabajo

3.1. Inicio

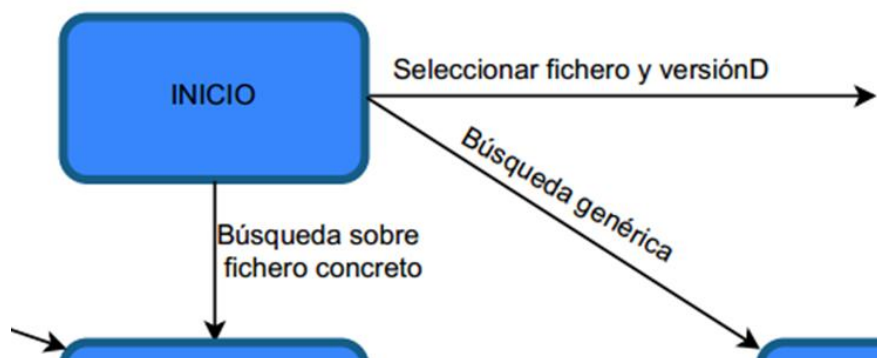


Ilustración 5. Flujo de trabajo, *inicio* al detalle

Esta es la página desde la cual se parte. En la pantalla del estado inicial se puede elegir entre dos opciones principales:

- Procesar un fichero, es decir, procesar y guardar una nueva muestra y, en caso de que tenga errores, obtener la lista de errores y la línea en que se encuentran; o, en caso de que no tenga errores, sea procesado satisfactoriamente dirigiendo al estado de búsqueda individual para buscar sobre la muestra subida.
- Elegir búsqueda independiente y a continuación elegir entre si se desea sobre un fichero concreto (individual) o sobre todos los ficheros (genérica).

3.1.1. Requisitos y algoritmos

No hay requisitos para llegar a este estado, sin embargo sí los hay a la hora de procesar los ficheros. El diseño, modelado y formato de las muestras se recoge en el **ANEXO 1**, especificaciones de las BDA y el **ANEXO 3**, especificaciones CSV de muestras. Para poder “Procesar muestra” se debe haber cargado el fichero en la página ya sea arrastrándolo a la misma o eligiendo seleccionar el fichero mediante el explorador; así como haber seleccionado la versión.

Se envía mediante una petición POST al servidor y una vez recogido por el mismo y comprobado el CRFS token por parte de Django internamente, se ejecutan dos algoritmos, el segundo dependiente del primero:

- validaCSV, un algoritmo validador con el que se comprueba tanto el formato del fichero recogido como las especificaciones descritas en el fichero así como que no exista ya una muestra con ese nombre.
- enviaBD, un algoritmo que inserta la entrada del fichero validada en la tabla que hace referencia a las muestras y, a continuación, las entradas validadas anteriormente de posiciones de cromosoma indicando también si esa posición es única de una versión o es común a ambas versiones. Finalmente se registra que el campo de las frecuencias de la tabla

referente a las entradas de las posiciones cromosómicas está desactualizado alterando el único campo y entrada de la tabla *actualizado*.

- dimeVer, que devuelve el número de versión dicha posición cromosómica es única para alguna de las versiones y devuelve 0 si se trata de una posición común.

3.2. Error en el fichero

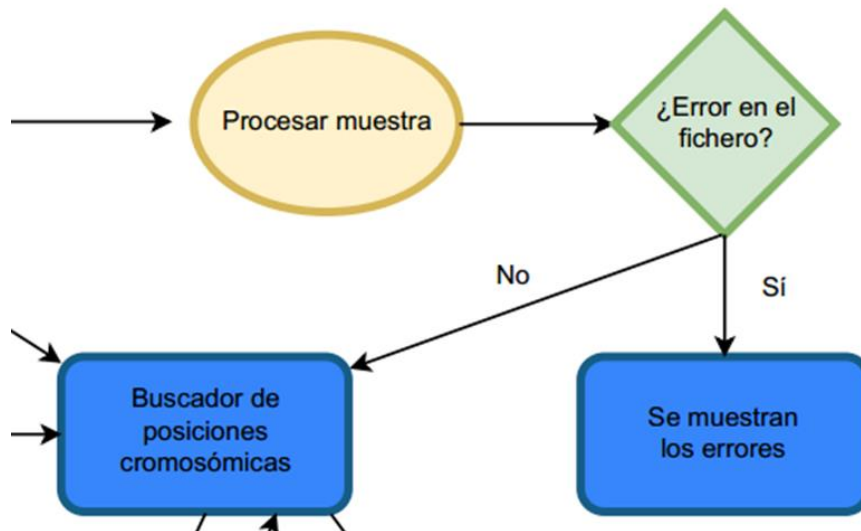


Ilustración 6. Flujo de trabajo, *errores* al detalle

Si tras procesar la muestra se ha encontrado errores en el fichero, éste no se registra y se transmite al usuario el origen del error. Este error puede ser referente al nombre o fecha de la muestra, como que ya existe el nombre o que la fecha es incorrecta, o, en el caso de que sea otra cosa, una lista de los errores de la muestra indicando qué campo falla, por qué y en que línea es en forma de tabla. En caso de error también se informa de las líneas que habitualmente se ignoran por ser lecturas duplicadas (una muestra con más de una línea para un mismo cromosoma y posición). El usuario deberá corregir la muestra y volver al estado anterior para proseguir en su objetivo.

3.2.1. Requisitos y algoritmos

El único requisito en este estado es que el listado de errores tendrá forma de tabla y recogerá la línea y la columna que sufren el error así como el motivo del mismo.

No es ejecutado ningún algoritmo de relevancia.

3.3. Búsqueda para seleccionar una muestra concreta

En caso de haber elegido la búsqueda sobre un fichero concreto, la página a la que se accede permite añadir criterios de filtrado e imprimir una lista de muestras en las cuales pueden seleccionarse dos opciones: suprimir la muestra presionando el botón que lo indica o hacer una búsqueda sobre dicha muestra presionando sobre su recuadro. Los posibles filtros a la hora de listar las muestras se componen de una combinación de las siguientes opciones:

- Rango de fecha, desde y/o hasta que fecha registrada en las muestras se desea que sean listadas.
- Nombre del fichero, completo o parcial.
- Elección de la versión del fichero, por defecto ambas versiones.

También se puede seleccionar no rellenar ningún filtro y añadir dicho criterio, es decir, “todos”, por lo que al seleccionar buscar se listan todas las muestras registradas en ese momento.

Tras haber añadido un criterio y hacer la búsqueda, se muestran en un cuadro las muestras que concuerden con el filtro pudiendo clicar sobre ellas y acceder a una búsqueda sobre una muestra concreta o al botón de suprimir que acompaña a cada muestra y que, tras pedir confirmación, procederá a borrarla de los registros.

3.3.1. Requisitos y algoritmos

En esta página los campos que se utilizan para la búsqueda se validan tanto por parte del cliente como del servidor y, en caso de error, así se indica:

- Fechas: el formato deberá ser DD-MM-AAAA o AAAA-MM-DD, por ejemplo, 2016-07-20 o 07-20-2016.
- Nombre del fichero, es tratado como un campo de texto llano.
- Versión, se escoge entre la selección de un botón o, en caso de ambas versiones, ambos botones, por lo que no da lugar a error.

Los algoritmos que se ejecutan en esta página son:

- CreaFiltro, que recoge los campos rellenos y, tras validarlos, los procesa dándoles devolviendo un array que contempla tanto si el campo ha sido relleno o no.
- DisplayFiltro, que crea e inserta nuevos elementos en el código html.

Desde el lado del servidor encontramos:

- borrarMuestra, que borra las entradas asociadas a dicha muestra de la tabla de *posiciones* y a continuación borra la entrada de dicha muestra de la tabla *muestras*.
- buscaNombre, que devuelve las entradas de la base de datos que coinciden con los filtros seleccionados.

3.4. Búsqueda sobre una muestra concreta

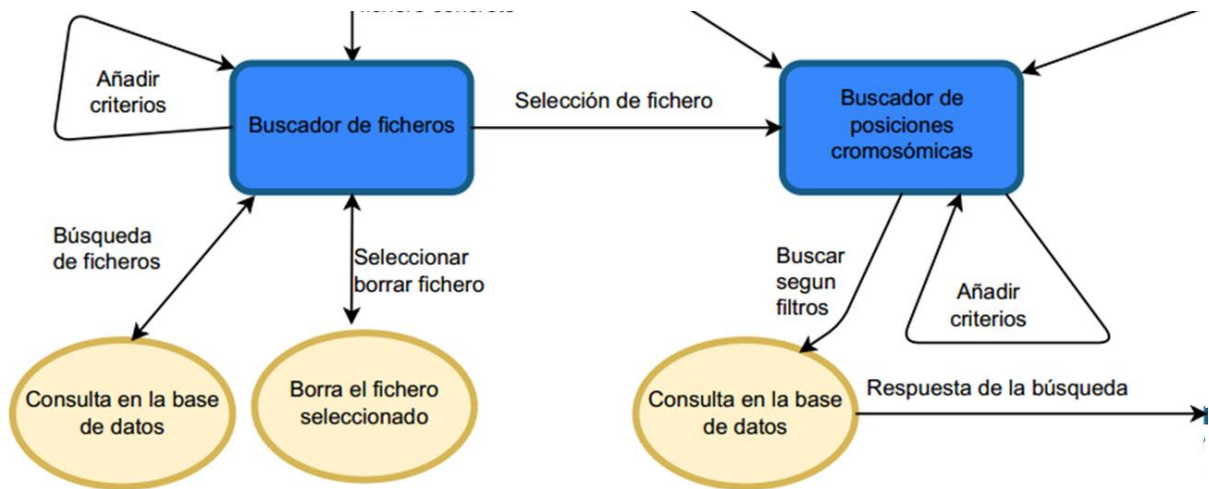


Ilustración 7. Flujo de trabajo, *buscador* al detalle

A la búsqueda sobre un fichero concreto podemos llegar tras haber subido y procesado satisfactoriamente una muestra o tras haberla seleccionado en la búsqueda para seleccionar una muestra concreta.

Antes de hacer la búsqueda, aquí se añaden los criterios de filtrado para la muestra. Pueden añadirse diversos filtros que se combinan para resultar en una lista de mutaciones de posiciones cromosómicas que son la suma sin repeticiones de la búsqueda para cada filtro. Cada filtro se compone de los siguientes campos asimismo combinables:

- Cromosoma o rango de cromosomas
- Posición o rango de posiciones cromosómicas
- Nombre parcial o completo del gen
- La mutación que ha sucedido, matizando en caso de inserción
- Frecuencia de aparición, que puede ser 'mayor que', 'menor que' o 'igual a'.

Tras haber añadido al menos un criterio por filtro y al menos un filtro, al seleccionar buscar se pasa al estado de resultados.

3.4.1. Requisitos y algoritmos

Los campos que se utilizan en esta página tienen las siguientes especificaciones:

- Cromosomas o rango escogido se limita a un entero del 1 al 22.
- Las posiciones cromosómicas o su rango no tiene más delimitación que ser enteros.
- El nombre del gen será un campo de texto.

- La mutación que ha sucedido, matizando en caso de inserción. El formato es [*'símbolo inicial' > 'símbolos resultantes'*] o, en caso de inserción, [*'símbolos resultantes'*]. E. g. 'A>AT' o 'ACT' en caso de inserción.
- La frecuencia de aparición mayor, menor o igual será un número real del 0 al 100.

Las funciones y algoritmos utilizados son durante una búsqueda concreta son:

- Desde el lado del cliente hay diferentes algoritmos de validación con nombres autodescriptivos:
 - validaNumORangoChr, que valida el número o rango de cromosomas.
 - validaNumORango, que comprueba que sea un número o rango.
 - creaFiltro, con la misma función que en búsqueda sobre una muestra concreta.
 - displayFiltro, que tras validar los campos del filtro devuelve un diccionario con los criterios seleccionados por el usuario.
 - borraFiltro, que borra el filtro seleccionado de los previamente añadidos.
 - checkActualizacion, que, si hemos sido redirigidos tras subir una muestra, envía un POST al servidor recordándole comprobar si las frecuencias han sido actualizadas y se encarga del aviso de que la muestra ha sido procesada correctamente.
 - getCookie que devuelve la cookie solicitada
- Desde el servidor son varios los algoritmos que pueden activarse:
 - se recibe un método POST dirigido a '/actualizaFrec' que comprueba si en la tabla *actualizado* el campo está a 0, en ese caso, se procede a actualizar la frecuencia de todas las entradas de posiciones cromosómicas (la tabla *posiciones*)
 - actualizaFrecuencias, que recorre todas las muestras ordenadas por cromosoma, posición y mutación y recalculando para cada una la nueva frecuencia comprobando que no se calcula dos veces para un mismo cromosoma-posición-mutación.

3.5. Búsqueda genérica

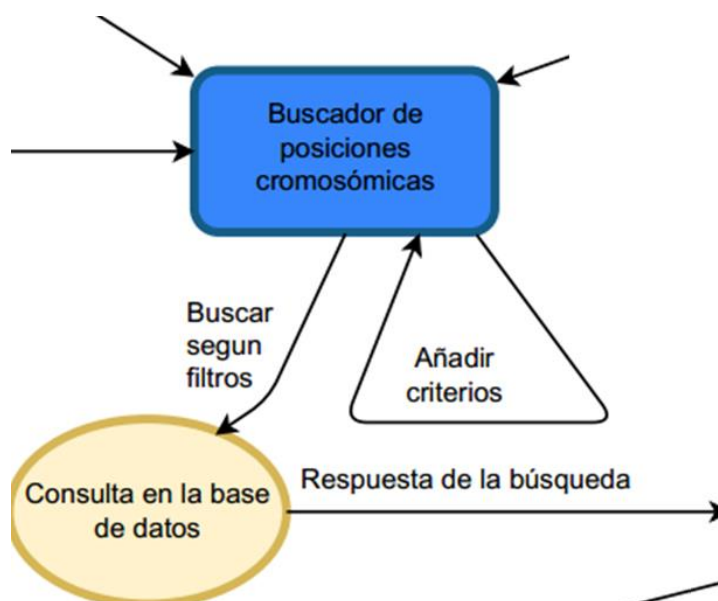


Ilustración 8. Flujo de trabajo, *buscador* al detalle 2

A la búsqueda genérica llegamos desde la página inicial. Es exactamente igual que la búsqueda sobre una muestra concreta solo que, en lugar de ser sobre un fichero concreto cuyo nombre ya está registrado, ésta se realiza sobre las muestras perteneciente a un rango de fechas ahora seleccionable o sobre todas las muestras del registro en el caso de no determinar fechas concretas para los filtros de la búsqueda.

Tras enviar la búsqueda se pasa al estado de resultados.

3.5.1. Requisitos y algoritmos

Funciona del mismo modo que la búsqueda sobre un fichero concreto solo que en este caso se tienen también en cuenta rangos de fecha, cuyo formato de fecha es el mismo que en la selección de muestras y se valida del mismo modo. El resto de campos también son validados como en el modo de búsqueda sobre un fichero concreto.

3.6. Resultados

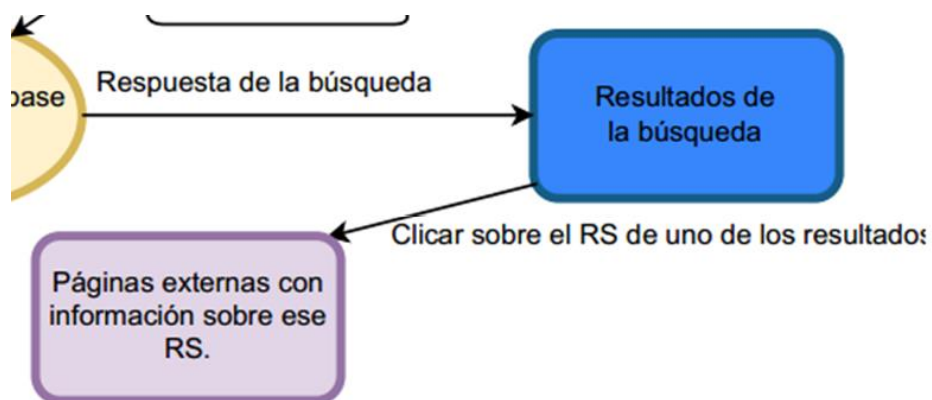


Ilustración 9. Flujo de trabajo, *resultados* al detalle

Tras haber hecho una búsqueda individual o genérica se muestra una tabla con todos los campos de las entradas de las posiciones cromosómicas que cumplen las condiciones del filtrado. Esta tabla ordenada por cromosoma, posición y finalmente mutación, se puede ser redirigido a páginas de información del National Center for Biotechnology Information clicando sobre el campo RS del gen.

3.6.1. Requisitos y algoritmos

Para llegar a esta página se ejecuta el algoritmo *búsqueda* que, tras hacer una búsqueda por cada uno de los filtros conjuntos de criterios añadidos, devuelve la unión de todas las entradas de posiciones cromosómicas resultantes sin repetición y ordenadas por número de cromosoma, posición cromosómica y mutación.

3.7. Rangos

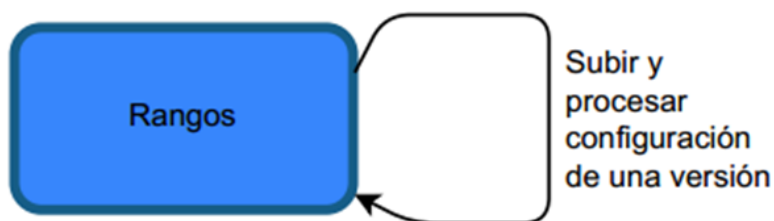


Ilustración 10. Flujo de trabajo, *rangos* al detalle

A esta página cuyo acceso está pensado para administradores no se es redirigido desde otra, sino que debe entrarse directamente con conocimiento de causa. Puesto que la diferencia entre una versión y otra son los rangos de posiciones cromosómicas que cada una abarca, la subida de un fichero de configuración mal elaborado ensuciará los cálculos de las frecuencias. No es por tanto una página a la que deba llegar un mero usuario curioso. Siguiendo el mismo proceso de subida de archivo que la página principal, la función de esta página es configurar o actualizar los rangos característicos de cada versión, una vez procesado dicho archivo toda la base de datos se ve afectada.

3.7.1. Requisitos y algoritmos

El fichero que especificará los rangos de cada versión seguirá el formato de la tabla de rangos especificado en el **ANEXO 2**: Formato del csv de RANGOS. Así mismo, ejecutará dos algoritmos:

- *actualizaRangos*, que recibe la versión a registrar y el CSV de rangos procesado y cambia el estado de los rangos comunes a rangos no pertenecientes a la versión a actualizar.
- *registraRangos*, que basándose en los rangos existentes los modificará para indicar que se tratan de rangos comunes o añadirá los nuevos rangos correspondientes a la versión que se configura o actualiza.

4. Diseño

Tras la descripción de la solución centrándose en los requisitos y no la implementación, en este capítulo tratamos cómo se da forma internamente a la solución y cómo funciona. Para cumplir con los requisitos establecidos y modelar la aplicación se utiliza el *framework* Django. Este decide en gran medida la estructura de la aplicación.

4.1. Diseño del sistema: DJANGO y el patrón MVC

El sistema de una aplicación web está basado en la comunicación entre un cliente que quiere cumplir una tarea y un servidor que ofrece unos servicios. El framework Django se escoge para ayudar a cumplir esta tarea por los siguientes motivos entre otros:

- La filosofía de re-utilización, conectividad y extensibilidad de componentes, el desarrollo rápido y el principio *No te repitas*.
- Una API de base de datos robusta en sintonía con su filosofía que facilita la integración con otras bases de datos.
- Una estricta separación entre las piezas de la aplicación promoviendo el acoplamiento débil.
- Sistema de *templates*, plantillas como herramienta de apoyo para reducir el código de la capa de presentación y homogeneizarla, así como simplificar su funcionamiento. Basado en etiquetas con bucles y condicionales simples y herencia de plantillas.
- Un *despachador* de *urls* basado en expresiones regulares.
- Django se preocupa por la seguridad, su sistema *middleware* se encarga de evitar errores comunes de seguridad, como *SQL injection*, *cross-site scripting*, *cross-site request forgery* y *clickjacking*. Y permite también desarrollar características adicionales; por ejemplo, soporte para archivos rss, administración de contenido, generador de sitemaps, configuración de la cache, compresión de la salida, normalización de URLs, protección CSRF y soporte de sesiones.
- Incorpora documentación accesible a través de la aplicación administrativa, además de la generada automáticamente de los modelos y las bibliotecas de plantillas añadidas por aplicaciones.
- Sistema de aplicaciones que permite integrar o comunicarse fácilmente otras aplicaciones webs elaboradas con este framework.

Django cumple un patrón MVC (modelo-vista-controlador) un tanto particular separando la lógica de negocios de la lógica de presentación mediante la utilización del sistema de plantillas, es decir, “una abstracción del acceso a la base de datos, una capa que se encarga de mostrar los datos y una capa en el medio que regule esto”. A continuación se explica punto por punto dicho modelo MVC:

- M de modelo, la parte de acceso a la base de datos, manejada por la capa de la base de datos de Django, tiene como resultado una API que facilita la integración de la BD una vez especificados unos modelos.

Desarrollo de una aplicación web de análisis y consultas en una base de datos biológica

- V de vista, esta parte se encarga de seleccionar qué datos mostrar y cómo mostrarlos y es manejada en conjunto por la vista y la utilización de las plantillas.
- C de controlador, la parte que delega a la vista dependiendo de la entrada del usuario, es manejada por el framework mismo siguiendo la URLconf mediante el *despachador* de URL y es la parte que se encarga de los aspectos de seguridad de Django internamente.

Al hablar de Django también se puede hacer referencia a él como un framework MTV, debido a que la “C” es manejada por el mismo framework y la esencia de la aplicación viene producido por los modelos, las plantillas y las vistas donde:

- M significa “Model” (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.
- T significa “Template” (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web o otro tipo de documento.
- V significa “View” (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: puedes pensar en esto como un puente entre los modelos y las plantillas.

Si estás familiarizado con otros frameworks de desarrollo web MVC, como Ruby on Rails, quizás consideres que las vistas de Django pueden ser el “controlador” y las plantillas de Django pueden ser la “vista”. Esto es una confusión desafortunada a raíz de las diferentes interpretaciones de MVC. En la interpretación de Django de MVC, la “vista” describe los datos que son presentados al usuario; no necesariamente el cómo se mostrarán, pero sí cuáles datos son presentados. En contraste, Ruby on Rails y frameworks similares sugieren que el trabajo del controlador incluya la decisión de cuales datos son presentados al usuario, mientras que la vista sea estrictamente el cómo serán presentados y no cuáles. Ninguna de las interpretaciones es más “correcta” que otras. Lo importante es entender los conceptos subyacentes. (Moss, 2015, p. 76)

4.2. Diseño de la capa de negocios y persistencia.

Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada junto a su “entorno”, que asigna las variables de las plantillas. Se trata de la capa intermedia entre la de persistencia, los modelos; y la de presentación, las plantillas y otros ficheros asociados como la hoja de estilos CSS o el fichero de scripts correspondientes en lenguaje Javascript. Tras el manejo interno de Django, la configuración de las vistas viene dado por el fichero `url.py` que determina qué función maneja cada dirección mediante expresiones regulares:

```

r'^admin/' -> admin.site.urls),
r'^buscador'-> buscador),
r'^seleccionFichero'-> seleccion),
r'^fecha'-> fecha_actual),
r'^prueba'-> prueba),
r'^$'-> nueva_muestra),
r'^subiendo'-> subiendo),
r'^analizando$'-> analiza),
r'^rangos$'-> rangos),
r'^actualizaFrec$'-> actualiza)

```

Ilustración 11. Despachador de URLs Django

Cada una de estas funciones constituyen módulo diferenciados que manejan todas las posibles peticiones a una dirección o conjunto de direcciones concretas determinadas por una expresión regular. A continuación se describen las peticiones contempladas y cómo se tratan así como el funcionamiento de las funciones auxiliares principales:

El fichero views.py engloba por tanto las siguientes funciones, las cuales todas reciben el parámetro request de tipo HttpRequest ya procesado internamente por Django:

- **nueva_muestra**, la página inicial de la aplicación que devolverá *"nueva_muestra.html"* al entrar de forma normal y solo en caso de petición POST que tiene un tipo fichero y un parámetro "versión", procesará el fichero y tras validarlo con *validaCSV*, lo enviará a la base de datos a través de *enviaBD*. En caso de error tras la validación devolverá *"error.html"* con la lista de errores .
- **rangos**, si la petición es de tipo POST comprueba que hay un parámetro de tipo *file* (fichero) y lo procesa convirtiendolo en un array que se envía a *actualizaRangos* junto con la versión del fichero, si la actualización se realiza con éxito se devuelve un mensaje que así lo confirma, en cualquier otro caso devuelve el *"rangos.html"* inicial.
- **actualiza**, si analiza recibe una petición POST comprueba si la base de datos está actualizada y si no es así ejecuta *actualizaFrecuencias*, sin embargo no devuelve nada y si se intenta acceder a */actualiza* devolverá no encontrado (error 404).
- **seleccion**, si se trata de un método GET y contiene el parámetro "f", hace una búsqueda de las muestra que contengan ese nombre parcial o totalmente y la devolverá configurando la plantilla *"resultNombre.html"*; si además está pasando un parámetro "d", borra la muestra con el nombre exacto de esa variable llamando a la función *borrarMuestra* a la que se le pasa el nombre exacto como parámetro. Si no se pasa ningún parámetro con GET, se devuelve la plantilla *"resultNombre.html"* dispuesta para realizar la búsqueda.
- **buscador**, si es una petición GET con el parámetro "f" se devuelve *"condiciones.html"* configurado para la búsqueda de un fichero individual; si no tiene dicho parámetro se devuelve *"condiciones.html"* de modo que haga una búsqueda genérica.

- **analiza**, si es una petición GET con el parámetro “b” se decodifica de base64 su valor y se llama a *busqueda*, si además tiene el parámetro “f” a *busqueda* también se le pasa su valor; después enviará los datos al template “*analiza.html*” y lo enviará al usuario una vez renderizado. En cualquier otro caso se devuelve un mensaje con “*Has subido un formulario vacío.*”

A continuación se describen algunas de las funciones que contempla el fichero *funcAux.py* cuyo funcionamiento se ha descrito en el análisis:

- **registraRangos**, recibe dos variables modelos de la entrada de la tabla rangos con su cromosoma asociado y un rango de posiciones y devuelve 0 si el rango es común a ambas versiones o el número de la versión si no es así.
- **actualizaRangos**, recibe una lista de listas con el csv que contiene la información del fichero de rangos procesada y la versión a la que pertenece dicha muestra y no devuelve nada.
- **validaCSV**, recibe un parámetro con la versión a la que pertenece la muestra y una lista de listas con toda la información a registrar y devuelve un array cuya primera posición es True en caso de que no haya errores seguido de las filas creadas a partir de la muestra listas para enviar a la base de datos y el nombre del fichero como último parámetro; o en caso de error, devuelve False y la lista de errores.
- **enviaBD**, recibe las entradas directamente a insertar en la base de datos y no devuelve nada.

4.3. Diseño de la capa de presentación

La capa de presentación viene dada por la estructura de templates, ficheros de código javascript y CSS que alberga el servidor.

El árbol de herencia según el sistema de plantillas de Django es el siguiente:

- “Base.html”, heredan:
 - o *analiza.html*
 - o *condiciones.html* (incluye *buttons1.html*)
 - o *error.html*
 - o *nueva_muestra.html* (incluye *buttons1.html*)
 - o *rangos.html* (incluye *buttons1.html*)
 - o *resultNombre.html*

Por otro lado, todos los ficheros cargan el fichero de estilos “*CSS.css*” y cada uno carga un fichero javascript distinto según se presenta:

- *analiza.html* → *anali.js*
- *condiciones.html* → *condi.js*
- *nueva_muestra.html* → *subida.js*
- *rangos.html* → *rangos.js*
- *resultNombre.html* → *selección.js*

- error.htm → ninguno.

4.4. Tecnologías utilizadas

4.4.1. Tecnologías utilizadas en el servidor

En este punto se describen las tecnologías utilizadas en el proyecto sobre las que se ha hecho menos hincapié. Primero se explica por qué se ha escogido esta base de datos y a continuación otras tecnologías utilizadas en la capa de presentación por parte del cliente.

4.4.1.1. MariaDB

MariaDB es uno de los servidores de bases de datos más populares del mundo. Construido de la mano de los desarrolladores originales de MySQL y con garantías de permanecer como código abierto. Entre los usuarios notables se incluye Wikipedia, WordPress.com y Google.

MariaDB convierte datos en información estructurada para una amplia gama de aplicaciones que van desde la banca hasta páginas web. Es un reemplazo mejorado para MySQL. MariaDB se utiliza debido a su rapidez, escalabilidad y robustez, demostrando un rico ecosistema de motores de almacenamiento, complementos y muchas otras herramientas que lo hacen muy versátil para una amplia variedad de casos de uso.

MariaDB es desarrollado como software de código abierto y como una base de datos relacional que proporciona una interfaz SQL para el acceso a los datos. Las últimas versiones de MariaDB incluyen también características de GIS y JSON. (MariaDB Foundation, 2016)

¿Y por qué usar una base de datos MariaDB en lugar de MySQL? A continuación se exponen solo algunas de las múltiples razones encontradas:

- **El desarrollo de MariaDB es más abierto y activo**

A diferencia de muchos otros proyectos de código abierto que Oracle heredó de la adquisición de Sun Microsystems, Oracle todavía desarrolla MySQL y, hasta donde se sabe, han contratado a nuevos desarrolladores competentes después de que la mayoría de los desarrolladores originales renunciaron. La próxima versión importante de MySQL, la 5.7, tendrá una mejora significativa sobre MySQL 5.6. Sin embargo, el log de los commits realizados muestra que todos los contribuyentes son de @oracle.com. Es decir, la mayoría de mensajes de confirmación hacen referencia a números de problemas que sólo se encuentran en un rastreador interno de Oracle y, por lo tanto, no están abiertos para el debate público y no parece que Oracle esté dispuesto a actualizar el repositorio de código público, mostrando como no pretende beneficiarse del bucle de retroalimentación del público.

- **Publicaciones de seguridad más rápidas y transparentes**

Oracle sólo tiene una política de para publicar lanzamientos de seguridad cada tres meses para todos sus productos. Sin embargo, MySQL tiene una nueva versión cada dos meses. A veces esto lleva a

situaciones en las que las actualizaciones de seguridad y la información sobre seguridad no se sincronizan. Además, las notas de la versión de MySQL no enumeran todos los identificadores de CVE que se corrigen posteriormente. Muchos se han quejado de que los anuncios de seguridad son muy vagos y no identifican problemas reales o los commits que los arreglaron, lo que dificulta la retroalimentación pública.

MariaDB sigue sin embargo los buenos estándares de la industria liberando sus publicaciones de seguridad y actualizaciones al mismo tiempo y manejando el pre-secreto y la post-transparencia de una manera adecuada. Las notas de la versión de MariaDB también enumeran los identificadores de CVE a diferencia de Oracle.

- **Más motores de almacenamiento**

MariaDB en particular sobresale por la cantidad de motores de almacenamiento y otros plugins que se subministran con los motores de almacenamiento Connex and Cassandra para backends NoSQL o migraciones sucesivas de bases de datos heredadas, Spider, TokuDB con índices fractales... etc. Son plugins también disponibles para MySQL a través de terceros, pero parte del lanzamiento oficial en el caso de MariaDB, garantizando que los plugins están bien integrados y son fáciles de usar.

- **Mejor rendimiento**

MariaDB asegura que tiene un optimizador de consultas mucho mejor y muchas otras mejoras relacionadas con el rendimiento. Ciertos puntos de referencia muestran como MariaDB es radicalmente más rápido que MySQL. Aunque cabe admitir que estos no siempre son traducidos directamente a situaciones de la vida real, cuando en Seravo se migró de MySQL a MariaDB, se observaron mejoras moderadas del 3-5% reales en el día a día. Sin embargo, cuando todo se suma, ese 5% es relevante en particular para backends del servidor web, donde cada milisegundo cuenta. Más rápido siempre es mejor incluso si se trata de sólo “un poco” más rápido.

- **La administración de Oracle es incierta**

Muchas personas han expresado desconfianza en las verdaderas motivaciones de Oracle y el interés por mantener vivo MySQL. Oracle no fue autorizado inicialmente a adquirir Sun Microsystems, que poseía MySQL, debido a la legislación de competencia de la UE. MySQL era el mayor competidor de la base de datos original de Oracle. La Comisión Europea, sin embargo, aprobó el acuerdo después de que Oracle publicase una promesa oficial de mantener MySQL vivo y competitivo. Este documento incluía una fecha de vencimiento, el 14 de diciembre de 2014, por lo que dicha promesa ahora está en el aire.

- **Compatible y fácil de migrar**

MariaDB es un reemplazo completo para MySQL. Migrar a MariaDB es tan fácil como ejecutar “apt-get install mariadb-server” o el comando equivalente a su versión de Linux escogida. De cualquier modo y a pesar de que la migración es fácil, siempre es recomendable que los administradores de bases de datos realicen sus propias pruebas y respalden sus bases de datos, por seguridad.

Siendo estas sólo algunas de las razones, parece saltar a la vista que efectivamente MariaDB es, claramente, una opción preferible a MySQL. (SERAVO, 2015)

4.4.2. Tecnologías utilizadas en la aplicación

4.4.2.1. HTML

El lenguaje de marcado para la elaboración de páginas web por excelencia, es un estándar que, mediante etiquetas, permite al navegador del cliente interpretar y dar forma y estructura al contenido web, pese a que actualmente va acompañado de otros lenguajes como CSS o Javascript que complementan su funcionalidad.

4.4.2.1. JavaScript

Lenguaje script por excelencia para el uso en página web, da funcionalidad al código *hueco* del HTML, en este proyecto se han utilizado algunas de sus librerías más importantes:

- **Ajax**, una biblioteca que permite hacer peticiones al servidor dejándolas en segundo plano, es decir, sin bloquear el cliente/página web, para la transmisión de información entre cliente-servidor sin necesidad de recargar la página.
- **Jquery**, se trata de una biblioteca de JavaScript que sirve como herramienta de apoyo para la interacción entre los documentos HTML, el árbol de objetos del documento perteneciente a una página web, y que permite manejar eventos, animaciones o interacciones con la técnica AJAX.

4.4.2.2. Bootstrap

El framework HTML, CSS y JS más conocido para desarrollar páginas web responsive. Se trata de un software responsive de apoyo para desarrollar webs de visualización agradable facilitando la capacidad de ser responsive.

4.4.2.3. CSS

Se trata de un lenguaje de diseño gráfico para definir y crear la presentación de documentos estructurados escritos con lenguaje de marcado, como HTML. Este conocido lenguaje de ejecución en cascada se convierte hoy día en una herramienta raramente obviada en el mundo de las páginas web.

5. Evaluación del desarrollo

Tras hacer un repaso al funcionamiento de la aplicación y su flujo de trabajo así como a su desarrollo más al detalle, en este capítulo se mostrarán diversos ejemplos de las funcionalidades de la aplicación y una evaluación del sistema basada en el cálculo de tiempos y la carga de la actualización y consulta de la base de datos mostrando así su crecimiento lineal y permitiendo elucubrar sobre el crecimiento de los tiempos a corto plazo. Estos tiempos se recogen en el **ANEXO 4: Tablas de tiempos**.

5.1. Varios ejemplos concretos

5.1.1. Buscando sobre una muestra concreta

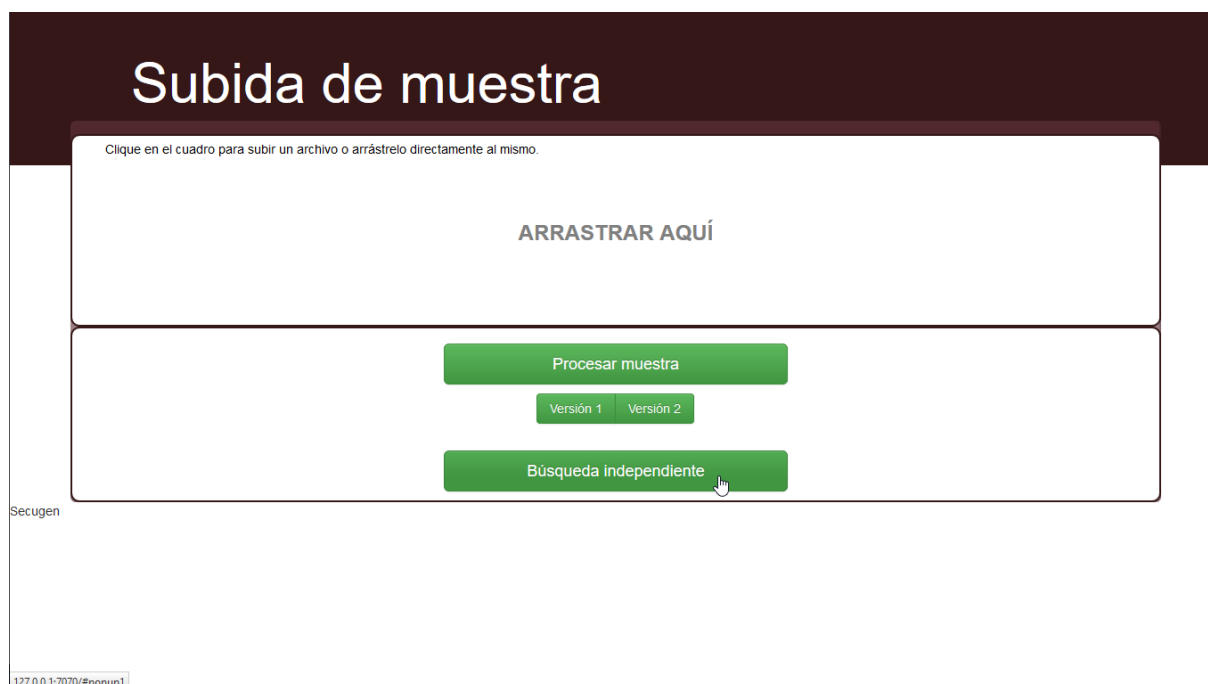


Ilustración 12. Pantalla inicial

Para realizar una búsqueda concreta el primer objetivo será seleccionar sobre qué muestra se pretende hacer la búsqueda. A continuación se accederá al buscador y, tras añadir los filtros pertinentes y decidirse a buscar, el servidor devolverá los resultados correspondientes.

Empezamos en la página inicial, donde clicamos sobre búsqueda independiente, esto hace aparecer una ventanita emergente.



Ilustración 13. Selección del botón *búsqueda independiente*



Ilustración 14. Menú pantalla emergente

Para entrar en el buscador de muestras, en la ventana emergente se clica a “Sobre un fichero”



Ilustración 15. Buscador, detalle fechas

En este ejemplo se añade la fecha, que debe ser en un formato de DD-MM-AAAA o AAAA-MM-DD.

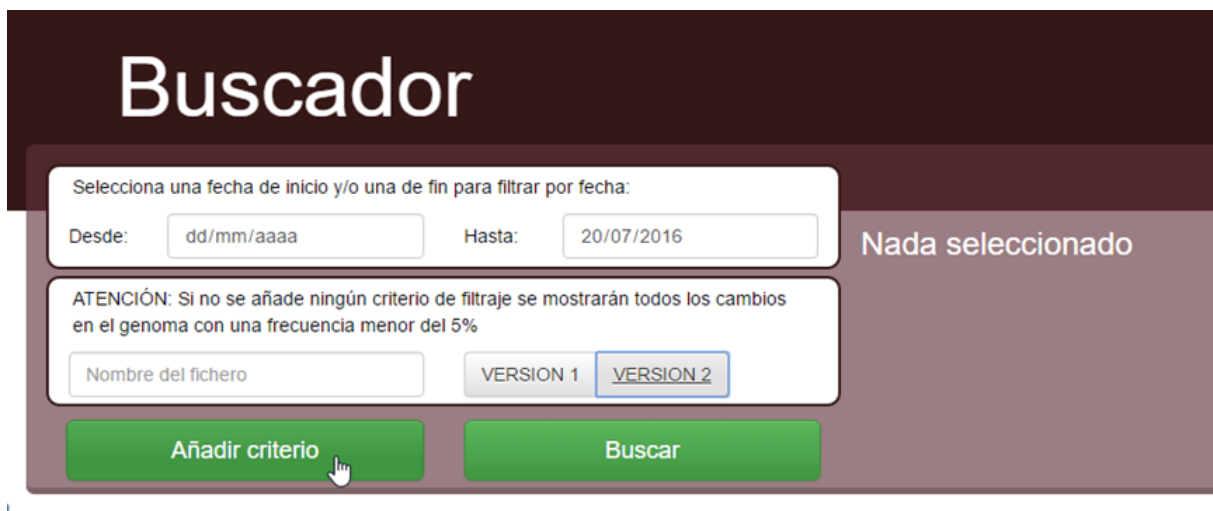


Ilustración 16. Buscador de muestras, seleccionando criterio

En este ejemplo se selecciona una fecha concreta y que los ficheros mostrados sean de la versión dos y a continuación se clica a "Añadir criterio".

Buscador

Selecciona una fecha de inicio y/o una de fin para filtrar por fecha:

Desde:

Hasta:

Hasta: 2016-07-20

Versión: 2

ATENCIÓN: Si no se añade ningún criterio de filtraje se mostrarán todos los cambios en el genoma con una frecuencia menor del 5%

Nombre del fichero

VERSION 1

VERSION 2

Añadir criterio

Buscar

Ilustración 17. Buscador de muestras, criterio añadido

Cuando se ha decidido el criterio para la búsqueda se clic a buscar y el servidor devuelve el resultado de acuerdo al criterio seleccionado.

Nº	file_name	fecha	version	
0	NGS-337-H610_S12_L001_R1_001_mean20_Output.pjt	19/07/16	2	SUPR
1	NGS-368-GN187_S1_L001_R1_001_mean20_Output.pjt	19/07/16	2	SUPR

Ilustración 18. Resultado buscador de muestras

Tras escoger la búsqueda deseada se clic encima del resultado escogido. Obsérvese que podría borrarse la muestra de la base de datos clicando en “SUPR”

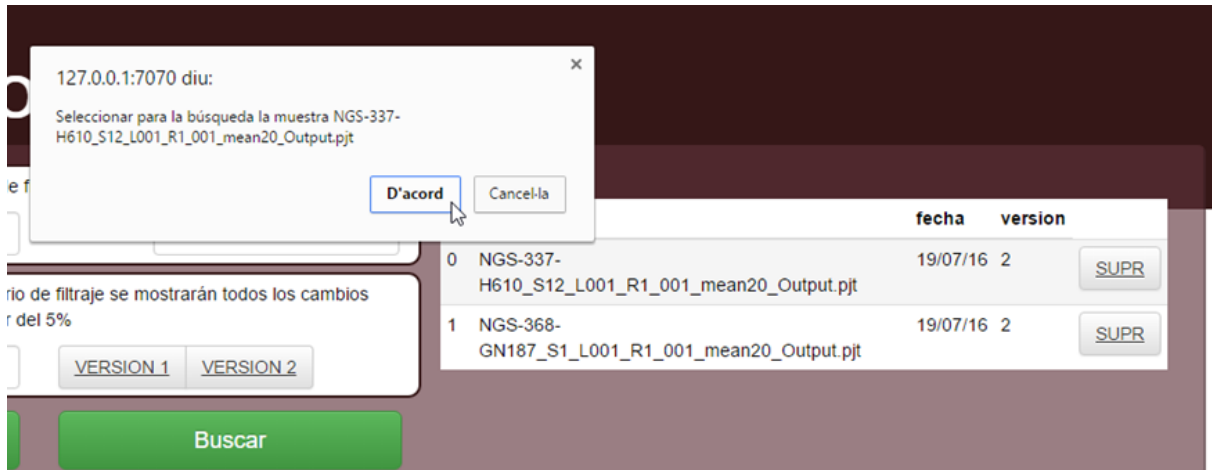


Ilustración 19. Borrando una muestra

Tras haber escogido la búsqueda, aceptar la confirmación nos llevará al buscador principal.



Ilustración 20. Buscador general de una muestra concreta

Una vez en el buscador principal pueden añadirse tantos filtros como se desee.



Ilustración 21. Añadiendo un filtro en el buscador dada una muestra concreta

Para ello, se seleccionan los criterios del filtro y se clic a al botón de “Añadir criterio” tal y como muestr a la imagen. En este caso, posiciones cromosómicas de los genes entre el 10 y el 15 con una frecuencia menor del 3% y una mutación A>AT.

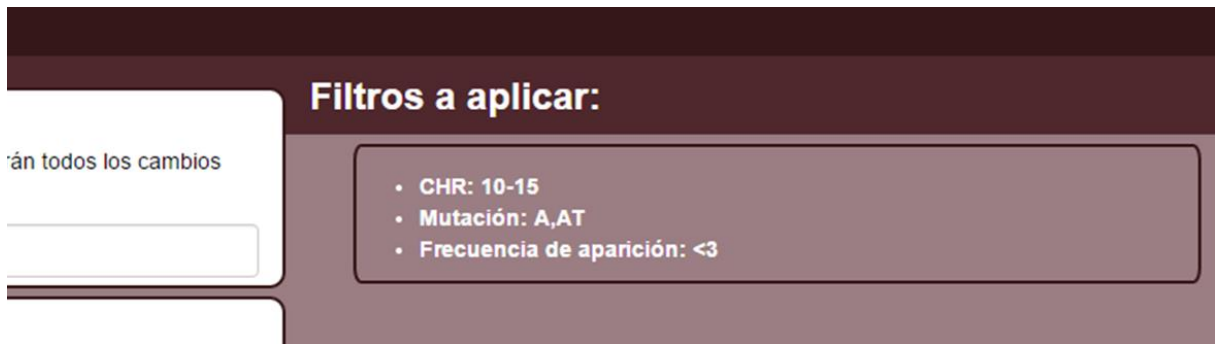
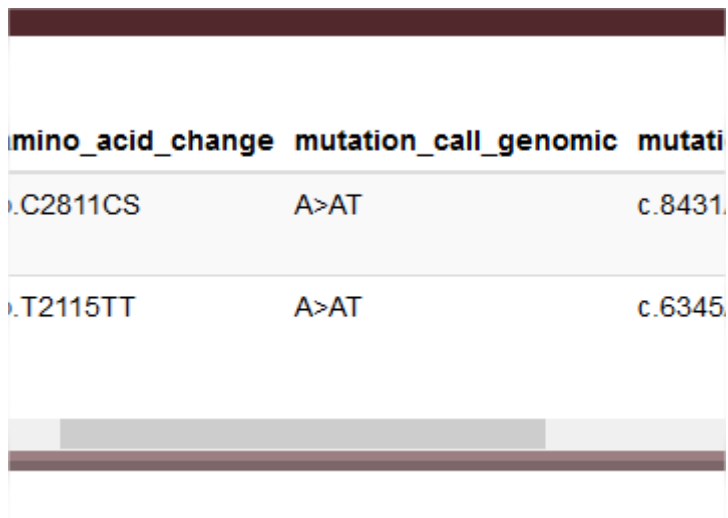


Ilustración 22. Filtro añadido en el buscador por fichero

Los filtros agregados se irán añadiendo agrupándose uno tras otro. Al clicar a buscar los el cliente es redirigido a una tabla que puede recorrerse con las flechas del teclado o con el mismo ratón con los resultados devueltos por el servidor según el criterio de la búsqueda

file_name	chr	version	chromosome_position	gene	rna_accession	cds	reference_nucleotide	coverage	score	snp_db_xref
NGS-337-H610_S12_L001_R1_001_mean20_Output.pjt	12	0	6058192	VWF	NM_000552.3	51	A	619	21.6	
NGS-337-H610_S12_L001_R1_001_mean20_Output.pjt	12	0	6103281	VWF	NM_000552.3	36	A	1737	25.8	rs11537642

Ilustración 23. Resultados respuesta del servidor a una consulta



amino_acid_change	mutation_call_genomic	mutati
C2811CS	A>AT	c.8431
T2115TT	A>AT	c.6345

Ilustración 24. Mutación de las posiciones buscadas

Como podemos observar, se trata de un gen que ha mutado, efectivamente, de A a AT, así como su frecuencia es menor del 3%. Es decir, el individuo de la muestra seleccionada solo contempla, del cromosoma 10 al 15 dos posiciones cromosómicas que hayan sufrido dicha mutación con dicha frecuencia dentro de los rangos estudiados por las versiones 1 y 2.



categoria	confirmacion_stranger	frecuencia
ie	None	2,04081632653061
ie	None	2,04081632653061

Ilustración 25. Frecuencia de las mutaciones buscadas

5.1.2. Una búsqueda genérica



Ilustración 26. Detalle botón “Búsqueda independiente”

Para realizar una muestra genérica partiendo de la página inicial debemos clicar sobre búsqueda independiente y a continuación en la ventana emergente clicar sobre “Genérica (sobre todos)” tal y como muestra la figura 14.

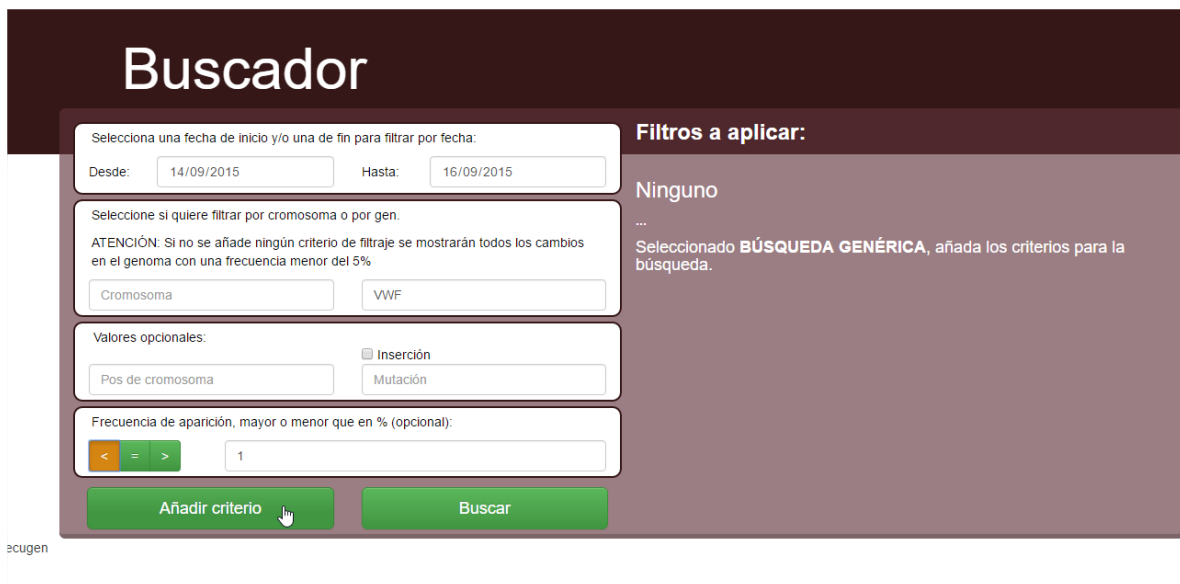


Ilustración 27. Buscador genérico detalle añadiendo filtro

Esta vez el buscador dejará seleccionar rangos de fechas. Para este ejemplo se escoge realizar la búsqueda sobre las fechas comprendidas entre el 14/09/2015 y el 16/09/15, así como que se muestren solo las posiciones pertenecientes al gen VWF y cuya frecuencia sea menor del 1%.



The screenshot shows a web interface for searching biological data. On the left, there are several filter sections: 'para filtrar por fecha:' with a 'Hasta:' field containing 'dd/mm/aaaa'; 'por gen:' with a 'Gen' field; 'le filtraje se mostrarán todos los cambios' with a 'Mutación' field; and 'e en % (opcional):' with a 'ción' field. A green 'Buscar' button is at the bottom. On the right, under 'Filtros a aplicar:', the selected filters are: 'Desde: 2015-09-14', 'Hasta: 2015-09-16', 'Gen: VWF', and 'Frecuencia de aparición: <1'.

Ilustración 28. Buscador genérico, detalle filtro añadido

Tras añadir el filtro escogido y seleccionar “Buscar” somos redirigidos a la página de resultados



The screenshot shows a web browser window with the URL '127.0.0.1:7070/analizando?b=dGltZTEsIHRpbWUyICwgaWQsIGdlbiwgcG9zLCBtdXQsIHByb2IKMjAxNS0wOS0xNCYyMDE1LTA5LTE2KyZWV0YmJlY8MSYr&'. The page title is 'Formulario de búsqueda'. The main content area is titled 'Resultados' and contains a table with the following columns: 'file_name', 'chr', 'version', 'chromosome_position', 'gene', 'ma_accession', 'cds', 'reference_nucleotide', 'coverage', 'score', 'snp_db_xref', 'genotype', 'amino_acid_change', and 'muta'. The table is currently empty, indicating a search with no results.

Ilustración 29. Resultado de búsqueda vacía

En este caso vemos que no existen filas para nuestra búsqueda, es decir, no existe en la base de datos ninguna posición cromosómica con esas características.

5.1.3. Subiendo un fichero con errores

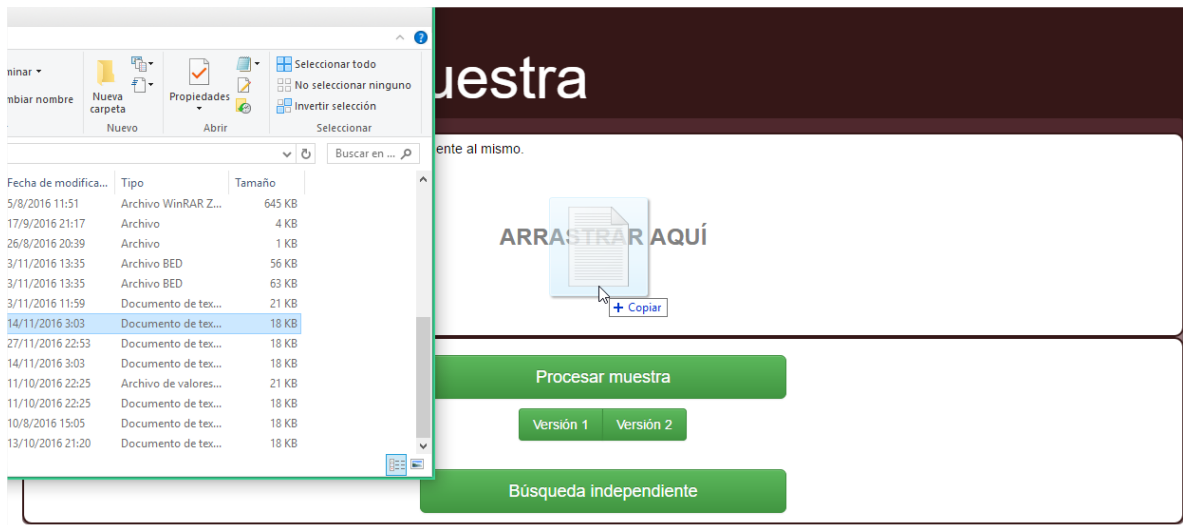
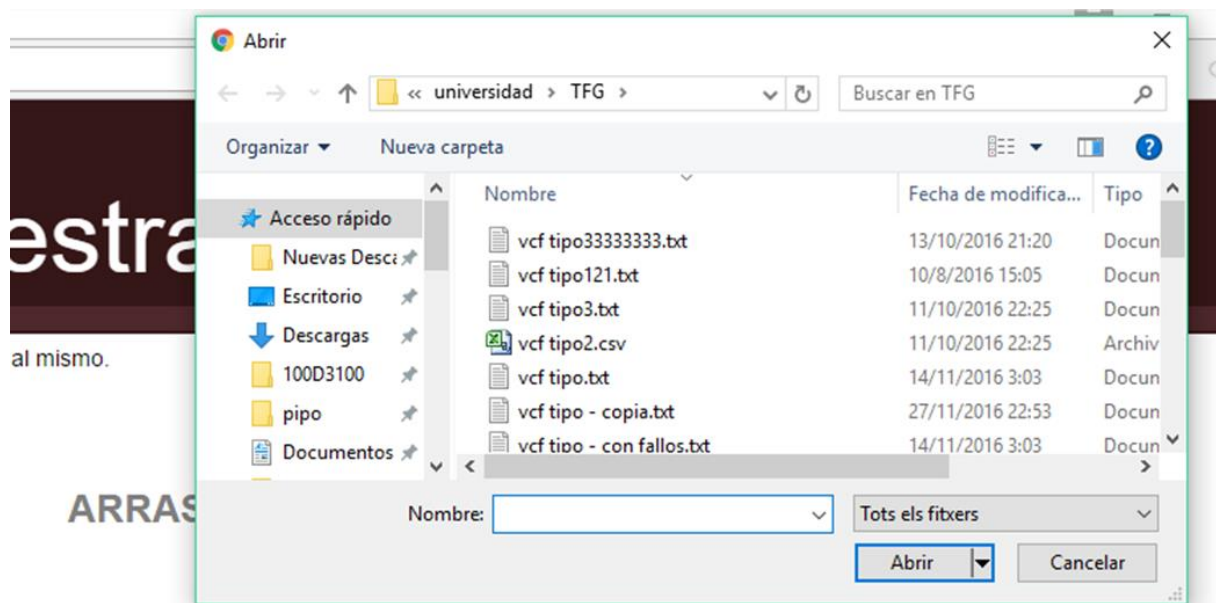


Ilustración 30. Arrastrando una muestra para subirla

A la hora de subir un fichero puede hacerse de dos modos, arrastrándolo hasta el recuadro o clicando sobre el recuadro y seleccionándolo a través del explorador de archivos.



14/11/2016



Ilustración 31. Seleccionando una muestra, desde el explorador de archivos, para subirla

No hay que olvidar seleccionar una versión o una advertencia nos recordará que no se puede continuar si no es seleccionada primero.



ARRASTRAR AQUÍ

modified: 14/11/2016



Ilustración 32. Botón de validación y registro de la muestra

Así pues, tras seleccionar finalmente “Procesar muestra”, se redirigirá a la pantalla de error donde nos indicará la línea y el tipo de error.

ERROR EN EL FICHERO

Listado de errores

```
linea:6 Error: {'chr': ['El valor'sf1' debe ser un entero.']}
linea:8 Error: {'chromosome_position': ['El valor'11090asdf916' debe ser un entero.'], 'score': ['El valor '23as.f5' debe ser un float.'], 'chr': ['El valor'fa1' debe ser un entero.']}
linea:10 Error: {'chromosome_position': ['El valor'57aaa340727' debe ser un entero.'], 'chr': ['El valor'1fsdaf' debe ser un entero.']}
linea:11 Error: {'chromosome_position': ['El valor'573asdf78149' debe ser un entero.'], 'chr': ['El valor'1as' debe ser un entero.']}
linea:29 Error: {'reference_nucleotide': ['Asegúrese de que este valor tenga menos de 1 caracter (tiene 5).']}
linea:34 Error: {'genotype': ['Asegúrese de que este valor tenga menos de 10 caracteres (tiene 11).']}
```

ecugen

Ilustración 33. Salida de errores al subir muestra corrupta

Listado de errores

```
linea:6 Error: {'chr': ['El valor'sf1' debe ser un entero.']}
linea:8 Error: {'chromosome_position': ['El valor'11090asdf916' debe ser un entero.'], 'score': ['El valor '23as.f5' deb
linea:10 Error: {'chromosome_position': ['El valor'57aaa340727' debe ser un entero.'], 'chr': ['El valor'1fsdaf' debe s
```

5.2. Evaluación del sistema:

Para la evaluación de tiempos se ha optado por calcular la escalabilidad de la base de datos midiendo los tiempos del recálculo de frecuencias a medida que añadíamos una nueva muestra así como del tiempo de devolución de todas las posiciones cromosómicas de todas y cada una de las muestras según aumentaba el volumen de la base de datos.

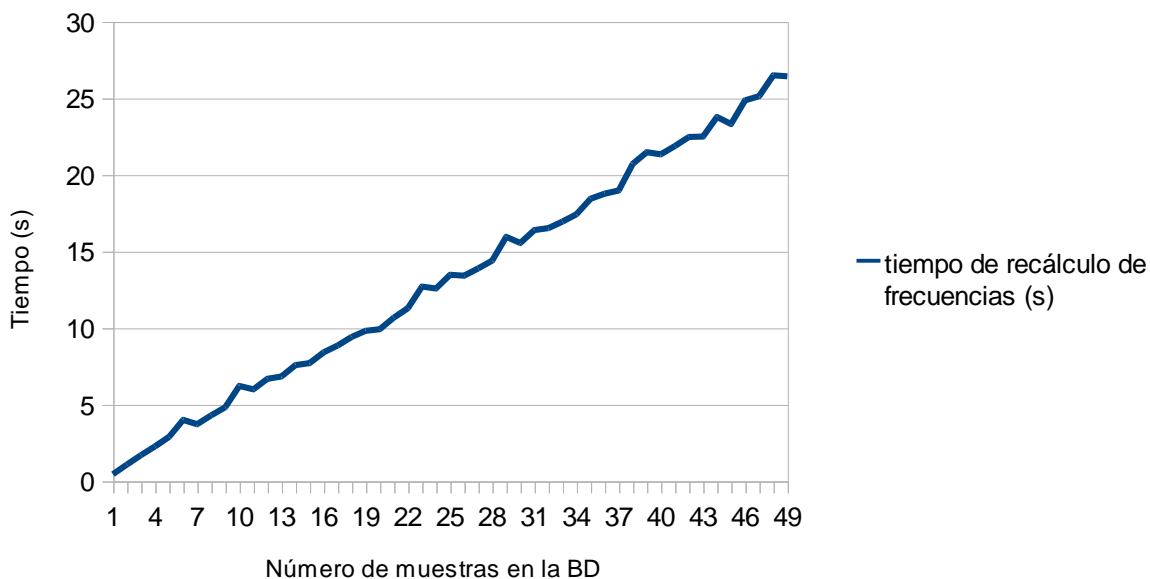


Ilustración 34. Gráfica de tiempo de recálculo de frecuencias (s)

Con un cálculo de media de 0,5336 segundos por muestra en la BD, se calcula que tarde aproximadamente 107 con una base de datos con un volumen de muestras similar al actual que consta de unas 200 muestras. A continuación se justifica por qué se supone un crecimiento lineal de coste de la actualización de las frecuencias por parte del servidor.

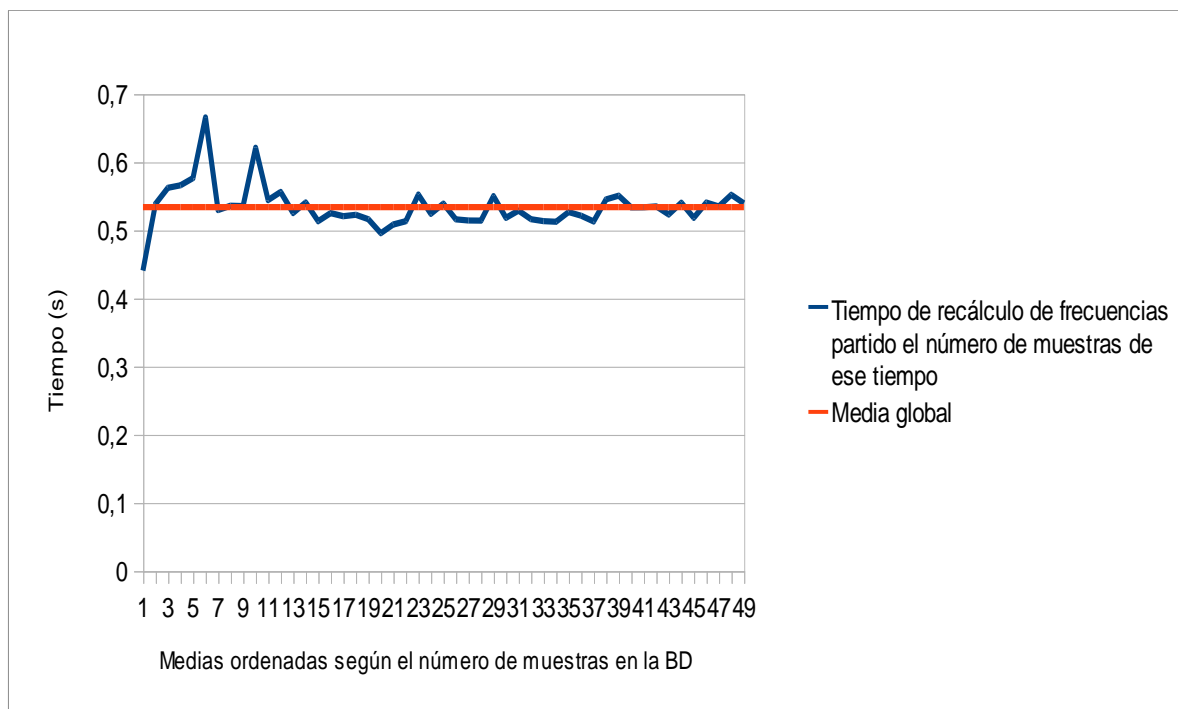


Ilustración 35. Gráfico de la media de las frecuencias de la 2 a la 49 en comparación con la media de las medias (s)

En este gráfico se observa cómo el conjunto de medias se mantiene constante a la media global a medida que ha crecido la base de datos y estas no van en aumento ni disminución. Es decir, el aumento del coste del tiempo se mantiene constante al número de muestras, por lo que decir que 200 muestras tardarían aproximadamente 107 es correcto dado que el crecimiento es lineal respecto a la carga de la BD.

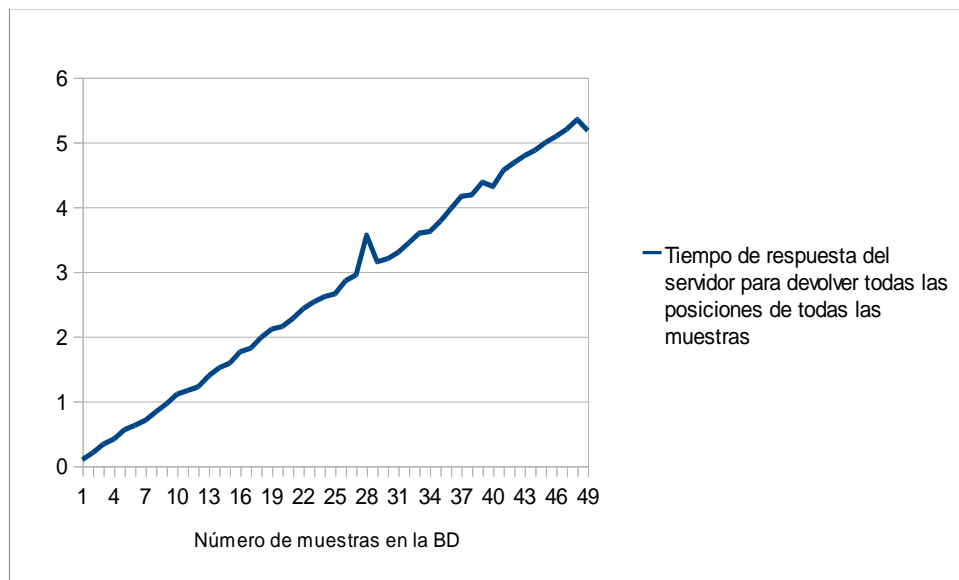


Ilustración 36. Gráfica de tiempo de listado de todas las posiciones de las muestras (s)

El siguiente gráfico permite estudiar la evolución del tiempo de respuesta del servidor respecto a la cantidad de filas a devolver. Con una media de 0,1089 segundos por muestra registrada en el servidor, esta vez comprobamos que con 20 muestras, devolver la búsqueda entera costaría unos 17 segundos al servidor. De igual modo, cabe destacar que debería disponerse de un equipo potente para recibir una web tan pesada.

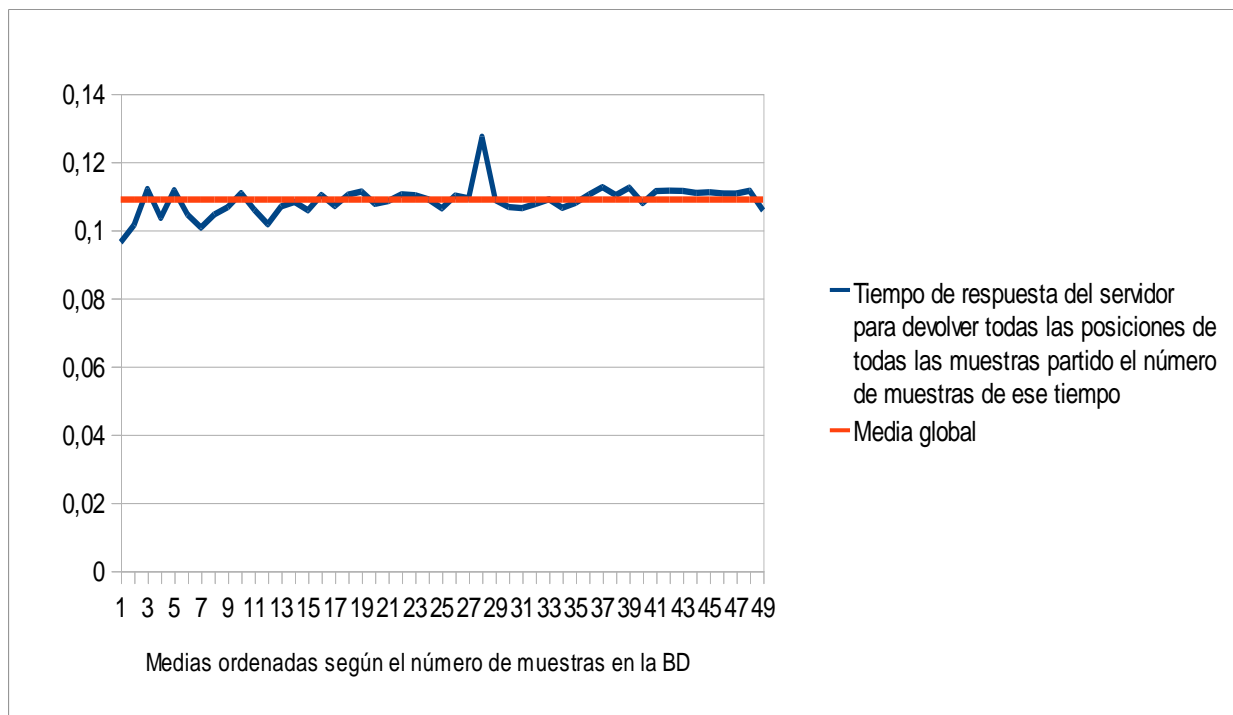


Ilustración 37. Gráfico de la media del tiempo de devolución de todas las entradas en comparativa con la media de las medias obtenidas (s)

Comprobamos una vez más que la media global parece mantenerse más o menos constante con el número de muestras, mas no tiene sentido tener en cuenta tiempos más altos puesto que sería extraño devolver una consulta tan grande al cliente.

Los tiempos de consulta son muy satisfactorios, pues rara tomará 18 segundos una consulta, pero los tiempos de actualización, no obstante, deberían de ser relativamente fáciles de mejorar. Tras tantear y testear la opción de calcular las frecuencias solo para las consultas realizadas pero cada vez que se realizara una consulta sobre las frecuencias, se comprobó como ralentizaba drásticamente las grandes consultas. Finalmente se ha optado por un sistema que anime a comprobar a la base de datos si necesita ser actualizada y, si es menester, se actualice, cada vez que se sube un fichero.

Este método acaba tomando 214 segundos con 400 muestras, por lo que quizá sería interesante calcular la frecuencia solo para las muestras subidas en el momento en que se sube y en un momento concreto día a día hacer la tarea de mantenimiento sobre una tabla auxiliar. En el momento que se haya actualizado la tabla completamente, redirigir las búsquedas a la tabla actualizada para en 24 horas actualizar la tabla que ahora se da por desactualizada, cada día una en un procedimiento cíclico.

Respecto al resto de aspectos, cabe decir que la memoria que tienen las tablas de la base de datos es muy manejable, la navegación entre una página y otra de la aplicación web es inmediata.

6. Conclusiones

En este último capítulo tratamos los puntos de la completitud del trabajo realizado de acuerdo a los requisitos previos, observaciones y conclusiones sobre el periodo de desarrollo y trabajos futuros, donde se plantean recomendaciones sobre la posible evolución del proyecto. .

6.1. Trabajo realizado

Pese a ser una aplicación web con una gran cantidad de posibles mejoras en diferentes ámbitos, cumple gratamente con el objetivo principal, que es la consulta en base a las frecuencias y con un potente filtrado sobre un conjunto de muestras. Al cumplir las funcionalidades básicas para dicho análisis, tal y como se detallan en el análisis, se convierte en una aplicación útil y que cumple su cometido, pasando de resultar una tarea realizada artesanalmente con Excel gen a gen a que pueda disfrutarse de un potente buscador para la comparación y estudio de las mutaciones genéticas. Así mismo, también cuenta con una interfaz amigable con un potente sistema explicativo y la navegación es rápida y segura, por lo que, también cumple las características básicas que debería tener toda aplicación.

Para concluir se recuerda que, gracias a haber sido programado con Django, es fácilmente combinable con otras aplicaciones Django, así como fácil de modificar internamente debido a su filosofía de débil acoplamiento, convirtiéndolo en una herramienta personalizable e imprescindible para los profesionales del sector.

6.2. Posibles mejoras y versiones futuras

Muchas son las posibles mejoras, algunas más inmediatas y otras recomendables a largo plazo. Desde la recién mentada en trabajo realizado sobre hacer invisible para el usuario el tiempo de mantenimiento de la BD como otras funcionalidades a nivel de interfaz. A continuación se comentan algunas otras posibles mejoras:

- Mejorar el sistema de interacción con las tres penúltimas columnas, antes de la frecuencia, convirtiéndolas en un desplegable, una etiqueta on/off o un comentario editable con formato.
- Activación y configuración del sistema de usuarios por defecto de Django. Usuarios normales y administradores. Solo los administradores podrán editar las mentadas tres columnas, subir y borrar muestras así como acceder a la página de rangos y actualizarlos. Los usuarios por defecto serán libres únicamente de realizar cuantas consultas quieran.
- Existirá un registro sobre la actividad de cada usuario, en especial énfasis a las inserciones y borrados.

Desarrollo de una aplicación web de análisis y consultas en una base de datos biológica

- Un botón en la página de resultados permitirá descargar en PDF el resultado de la búsqueda.
- Distinguir entre filtros inclusivos y exclusivos a la hora de buscar posiciones cromosómicas. Por ejemplo, que puedan buscarse todos los genes menores de X frecuencia pero que NO tengan la mutación Y.
- Permitir búsquedas que filtren por otras columnas que actualmente se recogen y se muestran pero aún no se tienen en cuenta, como por ejemplo "Function", que puede ser "Synonymous", "Noncoding" o "Missense".

7. Bibliografía

- Allele Frequencies. (4 de enero de 2015). *The Allele Frequency Net Database - Allele, haplotype and genotype frequencies in Worldwide Populations*. Recuperado el 23 de noviembre de 2016, de <http://www.allelefrequencies.net/default.asp>
- dbSNP Short Genetic Variations NCBI. (2016). *Submitted SNP(ss) Details: ss167704337*. Recuperado el 29 de noviembre de 2016, de https://www.ncbi.nlm.nih.gov/SNP/snp_ss.cgi?ss=ss167704337
- EUROPA PRESS. (19 de noviembre de 2014). *En marcha la primera base de datos que recoge mutaciones genéticas de los españoles*. Recuperado el 2016 de noviembre de 22, de http://www.lainformacion.com/salud/cancer/en-marcha-la-primera-base-de-datos-que-recoge-mutaciones-geneticas-de-los-espanoles_jSwRfgg3zhaSHgnX55kSC4/
- FINDbase. (2016). *Gene Mutation Map*. Recuperado el 25 de noviembre de 2016, de <http://www.biodata.gr/findbase/GeneMutationMap/>
- FINDbase. (2016). *Mutation Dependency Graph*. Recuperado el 20 de octubre de 2016, de <http://www.biodata.gr/findbase/MutationDependencyGraph/>
- Genome Variation Allele Frequencies Database Worldwide. (2016). *FINDbase.org - Genome Variation Allele Frequencies Worldwide*. Recuperado el 29 de octubre de 2016, de <http://www.findbase.org/>
- ISCIII. (2016). *Spainmdb*. Recuperado el 20 de noviembre de 2016, de <http://spainmdb.isciii.es/>
- ISCIII, Gabinete de prensa. (2014 de 11 de 20). Recuperado el 20 de 11 de 2016, de The Instituto de Salud Carlos III starts the first database for documenting the genetic mutations of the Spanish people: http://www.eng.isciii.es/ISCIII/es/contenidos/fd-el-instituto/fd-comunicacion/fd-noticias/20_11_2014_SpainMutationDataBase.shtml
- MariaDB Foundation. (2016). *About MariaDB - MariaDB.org*. Recuperado el 5 de noviembre de 2016, de <https://mariadb.org/about/>
- Moss, A. H. (2015). *La guía definitiva de django. Desarrolla aplicaciones Web de forma rápida y sencilla*. Celayita México.
- SERAVO. (9 de enero de 2015). *10 reasons to migrate to MariaDB (if still using MYSQL) - Seravo*. Recuperado el 8 de noviembre de 2016, de <https://seravo.fi/2015/10-reasons-to-migrate-to-mariadb-if-still-using-mysql>
- U.S. National Science Foundation. (2016). *ALFRED*. Recuperado el 23 de noviembre de 2016, de <https://alfred.med.yale.edu/>

UPV. (2015). *Bases de datos biológicas - Bioinformatics at COMAV 0.1 Documentation*.
Recuperado el 24 de noviembre de 2016, de
https://bioinf.comav.upv.es/courses/intro_bioinf/bases_datos.html

8. Anexos

Anexo 1: Especificaciones de la BD

```
1 class Actualizado(models.Model):
2     actual = models.CharField(max_length=1, blank=True, null=True)
3     primaria = models.IntegerField(primary_key=True)
4
5     class Meta:
6         managed = False
7         db_table = 'actualizado'
8
9
10 class Muestras(models.Model):
11     file_name = models.CharField(db_column='File_Name', primary_key=True, max_length=80) # Field name made lowercase.
12     fecha = models.DateTimeField(db_column='Fecha') # Field name made lowercase.
13     version = models.IntegerField(db_column='Version', blank=True, null=True) # Field name made lowercase.
14
15     class Meta:
16         managed = False
17         db_table = 'muestras'
18
19
20 class Posiciones(models.Model):
21     file_name = models.ForeignKey(Muestras, models.DO_NOTHING, db_column='File_Name')
22     chr = models.IntegerField(db_column='Chr') # Field name made lowercase.
23     version = models.IntegerField(blank=True, null=True)
24     chromosome_position = models.IntegerField(db_column='Chromosome_Position')
25     gene = models.CharField(db_column='GENE', max_length=30, blank=True, null=True)
26     rna_accession = models.CharField(db_column='RNA_Accession', max_length=30, blank=True, null=True)
27     cds = models.CharField(db_column='CDS', max_length=4, blank=True, null=True)
28     reference_nucleotide = models.CharField(db_column='Reference_Nucleotide', max_length=1, blank=True, null=True)
29     coverage = models.IntegerField(db_column='Coverage', blank=True, null=True)
30     score = models.FloatField(db_column='Score', blank=True, null=True)
31     snp_db_xref = models.CharField(db_column='SNP_db_xref', max_length=30, blank=True, null=True)
32     genotype = models.CharField(db_column='Genotype', max_length=10, blank=True, null=True)
33     amino_acid_change = models.CharField(db_column='Amino Acid Change', max_length=15, blank=True, null=True)
34     mutation_call_genomic = models.CharField(db_column='Mutation Call:Genomic', max_length=10, blank=True, null=True)
35     mutation_call_relative_to_cds = models.CharField(db_column='Mutation Call:Relative To CDS', max_length=30, blank=True, null=True)
36     mutation_call_relative_to_mrna = models.CharField(db_column='Mutation Call:Relative To mRNA', max_length=30, blank=True, null=True)
37     mutant_allele_frequency = models.DecimalField(db_column='Mutant Allele Frequency', max_digits=5, decimal_places=2, blank=True, null=True)
38     function = models.CharField(db_column='Function', max_length=10, blank=True, null=True)
39     siftpred = models.CharField(db_column='SIFTpred', max_length=3, blank=True, null=True)
40     polyphen2_hdiv_pred = models.CharField(db_column='Polyphen2 HDIV pred', max_length=10, blank=True, null=True)
41     polyphen2_hvar_pred = models.CharField(db_column='Polyphen2 HVAR pred', max_length=10, blank=True, null=True)
42     lrt_pred = models.CharField(db_column='LRT pred', max_length=3, blank=True, null=True)
43     mutationtaster_pred = models.CharField(db_column='MutationTaster pred', max_length=3, blank=True, null=True)
44     mutationassessor_pred = models.CharField(db_column='MutationAssessor pred', max_length=3, blank=True, null=True)
45     fathmm_pred = models.CharField(db_column='FATHMM pred', max_length=3, blank=True, null=True)
46     number_1000gpl_af = models.CharField(db_column='1000Gpl_AF', max_length=200, blank=True, null=True)
47     number_1000gpl_afr_af = models.CharField(db_column='1000Gpl_AFR_AF', max_length=200, blank=True, null=True)
48     number_1000gpl_eur_af = models.CharField(db_column='1000Gpl_EUR_AF', max_length=200, blank=True, null=True)
49     number_1000gpl_amr_af = models.CharField(db_column='1000Gpl_AMR_AF', max_length=200, blank=True, null=True)
50     number_1000gpl_asn_af = models.CharField(db_column='1000Gpl_ASN_AF', max_length=200, blank=True, null=True)
51     comentario = models.CharField(db_column='Comentario', max_length=500, blank=True, null=True)
52     categoria = models.IntegerField(db_column='Categoria', blank=True, null=True)
53     confirmacion_stranger = models.IntegerField(db_column='Confirmacion Stranger', blank=True, null=True)
54     frecuencia = models.FloatField(blank=True, null=True)
55
56     class Meta:
57         managed = False
58         db_table = 'posiciones'
59
60
61 class Rangos(models.Model):
62     cont = models.AutoField(primary_key=True)
63     version = models.IntegerField(blank=True, null=True)
64     chr = models.IntegerField(blank=True, null=True)
65     desde = models.IntegerField(blank=True, null=True)
66     hasta = models.IntegerField(blank=True, null=True)
67
68     class Meta:
69         managed = False
70         db_table = 'rangos'
```

ANEXO 2: FORMATO DE LOS CSV DE RANGOS

El formato generado por la empresa es el que se utiliza y acuerda utilizar para el proyecto. Dicha representación hace referencia a todas las columnas y cada secuencia entre corchetes es la composición de cada columna:

[Nº cromosoma] [Posición cromosómica inicial] [Posición cromosómica final] [Columna irrelevante] [Diferencia entre posición inicial y final] [Columna irrelevante]

Tal que así:

chr1	11086924	11087725	chr1:11086944:11087705:MASP2:CDSExon(71150830)	761	+
chr1	11090212	11090327	chr1:11090232:11090307:MASP2:CDSExon(71150832)	75	+
chr1	11090784	11090959	chr1:11090804:11090939:MASP2:CDSExon(71150834)	135	+
chr1	11094864	11094983	chr1:11094884:11094963:MASP2:CDSExon(71150829)	79	+
chr1	11097729	11097888	chr1:11097749:11097868:MASP2:CDSExon(71150831)	119	+
chr1	11102911	11103099	chr1:11102931:11103079:MASP2:CDSExon(71150823)	148	+
chr1	11103375	11103612	chr1:11103395:11103592:MASP2:CDSExon(71150837)	197	+
chr1	11104979	11105030	chr1:11104999:11105010:MASP2:CDSExon(71150836)	11	+
chr1	11105444	11105616	chr1:11105464:11105596:MASP2:CDSExon(71150839)	132	+
chr1	11106592	11106810	chr1:11106612:11106790:MASP2:CDSExon(71150833)	178	+

ANEXO 3: FORMATO DE LOS CSV DE MUESTRAS

El formato generado por la empresa es el que se utiliza y acuerda utilizar para el proyecto. Se divide la descripción en las cinco primeras filas y el resto, donde cada columna se describe entre corchetes. Dicho formato es el siguiente:

Fila 1: ["Software"] ["NetGENe V2,4"]

Fila 2: ["Project Name"] [Nombre del proyecto]

Fila 3: ["Date/Time:"] [Fecha del proyecto] [Hora del proyecto]

Fila 4 : --Vacía--

Fila 5 (etiquetas): ["Index"] ["Chromosome Position"] ["GeneRNA Accession"]
["CDS"] ["Chr"] ["Reference Nucleotide"] ["Coverage"] ["Score"] SNP
db_xref"] ["Genotype"] ["Amino Acid Change"] ["Mutation Call:Genomic"]
["Mutation Call:Relative To CDS"] ["Mutation Call:Relative To mRNA"] ["Mutant
Allele Frequency"] ["Function"] ["SIFT_pred"] ["Polyphen2_HDIV_pred"]
["Polyphen2_HVAR_pred"] ["LRT_pred"] ["MutationTaster_pred"]
["MutationAssessor_pred"] ["FATHMM_pred"] ["1000Gp1_AF"]
["1000Gp1_AFR_AF"] ["1000Gp1_EUR_AF"] ["1000Gp1_AMR_AF"]
["1000Gp1_ASN_AF"]

La fila 6 y el resto cumplen con las etiquetas que en la fila 5 están escritas literalmente.

ANEXO 4: TABLA DE TIEMPOS

Los cálculos realizados en la evaluación del sistema vienen recogidos aquí:

Tiempo de recálculo de frecuencias partido el número de muestras de ese tiempo	Media global		Tiempo de respuesta del servidor para devolver todas las posiciones de todas las muestras	Media global
0,4405	0,5336		0,096340895	0,1089
0,5383	0,5336		0,202824354	0,1089
0,562	0,5336		0,335893154	0,1089
0,5656	0,5336		0,414241314	0,1089
0,5762	0,5336		0,55805254	0,1089
0,6651	0,5336		0,626850128	0,1089
0,5295	0,5336		0,704895496	0,1089
0,5356	0,5336		0,836313009	0,1089
0,5352	0,5336		0,960050106	0,1089
0,6207	0,5336		1,107707977	0,1089
0,544	0,5336		1,164343596	0,1089
0,5559	0,5336		1,220054627	0,1089
0,525	0,5336		1,389790058	0,1089
0,5404	0,5336		1,514778376	0,1089
0,513	0,5336		1,586523294	0,1089
0,5249	0,5336		1,763720751	0,1089
0,5204	0,5336		1,819026709	0,1089
0,5225	0,5336		1,985954046	0,1089
0,516	0,5336		2,113614082	0,1089
0,4954	0,5336		2,153103352	0,1089
0,5081	0,5336		2,277936697	0,1089
0,5131	0,5336		2,429728031	0,1089
0,552	0,5336		2,534325123	0,1089
0,5239	0,5336		2,615007639	0,1089
0,5387	0,5336		2,658463717	0,1089
0,5157	0,5336		2,861261368	0,1089
0,5141	0,5336		2,949965715	0,1089
0,5139	0,5336		3,564744473	0,1089
0,5497	0,5336		3,1544559	0,1089
0,518	0,5336		3,200391054	0,1089
0,5285	0,5336		3,298042297	0,1089
0,516	0,5336		3,443876505	0,1089
0,5131	0,5336		3,596139669	0,1089

0,5122	0,5336		3,620112896	0,1089
0,5264	0,5336		3,778586626	0,1089
0,521	0,5336		3,973491669	0,1089
0,5126	0,5336		4,163614512	0,1089
0,545	0,5336		4,187789679	0,1089
0,5505	0,5336		4,382694721	0,1089
0,5333	0,5336		4,315871	0,1089
0,5334	0,5336		4,56704545	0,1089
0,5348	0,5336		4,684187412	0,1089
0,5229	0,5336		4,791893005	0,1089
0,5401	0,5336		4,877128124	0,1089
0,518	0,5336		4,997111559	0,1089
0,5403	0,5336		5,092003059	0,1089
0,5346	0,5336		5,200950623	0,1089
0,5517	0,5336		5,351331472	0,1089
0,5392	0,5336		5,17951107	0,1089

8.1. Glosario

Framework: O entorno de trabajo, se trata de una infraestructura, en términos generales, conceptual y tecnológica de soporte definido que sirven de base para la organización y desarrollo de software, resultando como herramienta que ayuda a desarrollar y unir diferentes componentes en un proyecto.

Token: es una firma cifrada que permite al API identificar al usuario

Capo RS: Número único que identifica posiciones cromosómicas para todas sus variantes

NextGENe: Next Generation Sequencing Software for Biologists. Se trata de un software utilizado para que analizas muestras en formato BAM, .fasta, Roche .fna o SOLiD System .csfasta, para dar lugar, respecto a la aplicación que nos atañe, a listas de rangos de posiciones cromosómicas y el cromosoma al que pertenecen.

Diseño responsive: Es decir, que se adapta a la forma de la pantalla, ya sea una pantalla de baja resolución como un pequeño móvil, como una gran pantalla como pueda ser una televisión.

Plugin: Complemento aplicación que, al relacionarlo con otra aplicación, sirve para agregar una función nueva y generalmente muy específica.

Middleware: Hace referencia a la lógica y funcionamiento del intercambio de información entre aplicaciones, siendo un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, programas, redes, hardware, ... etc

SQL injection: Método de infiltración de código intruso que aprovecha errores del software para realizar acciones no autorizadas a ese nivel de usuario en una base de datos.

No te repitas: (DRY, del inglés Don't Repeat Yourself). Filosofía de programación que promueve la reducción de la duplicación de código en pos de maximizar la claridad y minimizar las inconsistencias y la dificultad de los cambios y evolución.

8.2. Definición de abreviaturas

BD: Iniciales que significan Base de Datos. Se trata del software encargado de almacenar y administrar las muestras para su posterior consulta y utilización.

MVC: *Model-View-Controller*, patrón de diseño caracterizado por separar los datos y la lógica de negocio y el módulo encargado de gestionar los eventos y comunicaciones.

HTML: *HyperText Markup Language*, lenguaje de marcas de hipertexto que se utiliza estructura base en el diseño de páginas web.

CSS: *Cascading Style Sheet*, hojas de estilo en cascada, su uso es dar una apariencia agradable a la página web.

CVE: Common Vulnerabilities and Exposures, es decir, vulnerabilidades comunes y exposiciones (a dichas vulnerabilidades).

URL: Uniform Resource Locator, es un identificador de recursos uniforme, cuyos recursos pueden cambiar, es decir, que dicha dirección apunte a recursos variables en el tiempo.

CFRS: Cross-site request forgery, falsificación de petición en sitios cruzados, es decir, un tipo de ataque que busca hacer creer al servidor que se es un cliente distinto al que es en realidad.