TECNOLÓGICO DE COSTA RICA

ESCUELA DE COMPUTACIÓN

IC-7841 PROYECTO DE INGENIERÍA DE SOFTWARE

Prof. Dr. Jaime Solano Soto

MANUAL TÉCNICO DEL SISTEMA

RUTAS METABÓLICAS

INFORME FINAL

I. INTRODUCCIÓN

1.1. Visión general

Actualmente los cálculos necesarios para la comparación de redes y rutas metabólicas ocupan una considerable proporción del valioso tiempo de los investigadores. Si para estos investigadores no fuera necesario enfocar su atención en tareas repetitivas, las cuales pueden ser realizadas por máquinas, a una mayor velocidad, les sería posible enfocarse en lo verdaderamente importante y, de esta manera, alcanzar los resultados deseados más rápidamente, generando mayor valor a la sociedad.

El sistema podrá ser utilizado, a nivel académico, por investigadores y estudiantes afines a la biología, química y biotecnología. El desarrollo de la aplicación web, mejorará la productividad de los individuos involucrados en los trabajos de investigación que utilizarán la herramienta.

La creación de la aplicación web incrementará la productividad de investigadores nacionales, de esta manera generarán mayor conocimiento en nombre del país, asimismo la herramienta podrá ser accedida por cualquier investigador internacional al encontrarse alojada en servidores conectados permanentemente a Internet.

1.2. Definición del problema

Para un ser humano, realizar el cálculo “manualmente” de los algoritmos para la obtención de los datos necesarios para las investigaciones ocasiona el principal problema abordado por el sistema, en esta dificultad se encuentran involucrados todos los implicados en el desarrollo de artículos científicos, pues de una u otra manera una mayor eficiencia y certeza proporciona ventajas a estos investigadores.

Actualmente, las diferentes aplicaciones utilizadas por los investigadores para los cálculos no cuentan con una interfaz gráfica, todo se realiza a través de una consola, lo cual no es apropiado considerando que la media de usuarios finales no cuentan con un conocimiento computacional para manipular cómodamente un programa sin un front end apropiado.

Además de lo expuesto en los párrafos anteriores, contar con la opción de acceder a una aplicación desde cualquier parte del mundo es de vital importancia, y actualmente esto no se cumple. Los sistemas en uso se encuentran a un nivel de computadora local o en servidores privados de universidades y centros de investigación, los cuales no permiten el acceso al público en general.

1.3. Solución propuesta

Gracias al poder de procesamiento computacional con el que contamos hoy en día, las máquinas pueden realizar cálculos extremadamente complejos y, quizá, imposibles para humanos. Esta es una de las razones por las que se escogió un servidor con recursos suficiente para

la ejecución veloz de los algoritmos, de esta manera los usuarios podrán obtener los resultados en cuestión de segundos.

Se proveerá una interfaz gráfica fácil de utilizar para los usuarios, la misma será intuitiva y simple. En pantalla, el orden de los objetos será priorizado con base en la mayor utilización de cada una de las funciones, para así contar con una aplicación lo más comprensible posible para cada nivel de usuario.

El sistema se desarrolló como una aplicación web, la cual puede ser accedida a nivel global, facilitando, e inclusive promoviendo, la interacción de personas no especializadas en la computación, ya que se podrá hacer uso de la aplicación desde cualquier dispositivo móvil, computadora personal y/o escritorio.

1.4. Justificación

El desarrollo de la plataforma, generará una visión más amplia de la institución a investigadores y estudiantes de carreras relacionadas a la biología, química y biotecnología, quienes se verán atraídos por el desarrollo de esta herramienta.

La facilidad de interacción con el producto otorgará beneficios considerables a los usuarios, entre los que se encuentran el acceso desde cualquier punto del mundo si se cuenta con una conexión a Internet.

El sistema proporcionará a los investigadores las herramientas necesarias para reducir el tiempo invertido en cálculos que no deben ser ejecutados exclusivamente por humanos, en realidad siendo lo contrario lo ideal.

1.5. Descripción del documento

Este documento proporciona una descripción detallada de cómo se encuentra desarrollado el sistema a un nivel técnico, desde la arquitectura hasta la implementación. Los apartados siguientes describen al cliente, el ámbito del sistema, la arquitectura final, el sistema para la persistencia de datos, las estrategias y realización de las pruebas.

El presente documento se encuentra dirigido a personas con un conocimiento mínimo de computación y para aquellos quienes continuarán el desarrollo del sistema en el futuro.

Se recomienda leerlo en el orden usual.

II. DESCRIPCIÓN DE LA ESCUELA

2.1. Nombre de la Escuela

Escuela de Ingeniería en Computación

2.2. Descripción general de la Escuela

La Escuela de Ingeniería en Computación ha creado un sólido prestigio como centro de excelencia en la enseñanza y en la investigación aplicada. Ofrece opciones académicas en la Sede Central en Cartago, y en las Sedes de San Carlos, Alajuela, Limón y San José.

También cuenta con el Centro de Investigaciones en Computación (CIC), una entidad cuyo propósito es contribuir a solucionar problemas en la región de Centroamérica y el Caribe, mediante la realización de actividades orientadas a la generación, adaptación, incorporación y difusión de conocimientos informáticos.

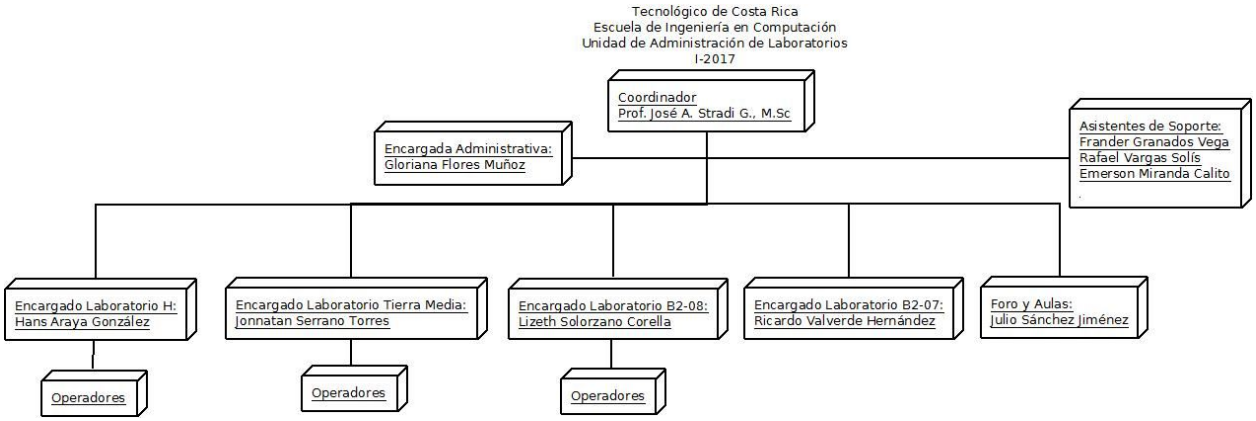
La Escuela de Computación, también es partícipe en los programas de Administración de Tecnologías de Información (ATI) y de Ingeniería en Computadores (CE).

En el área de Posgrados, se cuenta con La Maestría en Computación, que se imparte en diferentes sedes, y es dirigida a los diferentes perfiles de personal en el área de computación e informática.[1]

[1] Tomado del sitio web oficial

2.3. Organigrama

2.3.1 Unidad de Administración de Laboratorios



2.4. Nombre y teléfonos de la persona contacto

Ing. Esteban Arias Méndez

Tel. 8875 1635

2.5. Formulario de aprobación final del usuario

Ing. Esteban Arias Méndez
Profesor Escuela de Computación
Tecnológico de Costa Rica
earias@ic-itcr.ac.cr

III. ÁMBITO DEL SISTEMA

3.1 Objetivos

3.1.1 Objetivo General

Desarrollo de una aplicación web que permita a investigadores nacionales e internacionales invertir la menor cantidad de tiempo en tareas innecesarias de realizar para un humano.

3.1.2 Objetivos Específicos

1. Creación y manipulación de archivos XGML.
2. Extracción de rutas metabólicas desde archivos previamente cargados.
3. Almacenamiento de rutas metabólicas en la base de datos local.
4. Parametrización de valores para la comparación de los alineamientos.
5. Visualización del grafo generado luego del proceso de extracción.
6. Ejecución de algoritmos originales para la comparación.
7. Ejecución de la versión extendida de algoritmos para la comparación.
8. Ejecución de las cinco versiones de algoritmos con pesos.
9. Creación de un diccionario de nombres de metabolitos para la sustitución de códigos.
10. Tabulación de los resultados de los algoritmos en pantalla.
11. Definición manual de grafos.
12. Aplicación de todos los algoritmos disponibles a grafos manuales.
13. Edición del orden en las relaciones de un grafo.
14. Carga de grafos creados manualmente a la base de datos.

15. Generación de reportes en archivos.

3.2 Criterios de éxito

# de objetivo específico	Criterio de éxito	Finalizado
1	Es posible crear, eliminar y modificar el nombre de los archivos XGML en su ventana respectiva.	Sí
2	Es posible ejecutar los algoritmos seleccionando uno de los archivos cargados.	Sí
3	Es posible almacenar la estructura de datos utilizada para la representación de la ruta metabólica en la base de datos local.	Sí
4	Es posible transmitir los parámetros de comparación a la ejecución de los algoritmos desarrollados en Python.	Sí
5	Es posible mostrar en la interfaz gráfica la imagen generada como resultado de la creación del grafo.	No
6	Es posible solicitar y visualizar los resultados correctos de la ejecución de la primera versión de algoritmos.	Sí
7	Es posible solicitar y visualizar los resultados correctos de la ejecución de la versión extendida de algoritmos.	Sí
8	Es posible solicitar y visualizar los resultados correctos de la ejecución de las versiones extendidas de los algoritmos con pesos.	No
9	Es posible para el usuario seleccionar si desea que los resultados sean mostrados con los códigos usuales o con los nombres reales de los metabolitos.	Sí
10	Es posible visualizar los resultados generados de la ejecución de los algoritmos.	Sí
11	Es posible para el usuario crear un grafo (definir nodos y sus relaciones).	Sí

12	Es posible ejecutar cualquier de los algoritmos a los grafos creados manualmente.	Sí
13	Es posible para un usuario hacer cambios en el orden de las relaciones.	Sí
14	Es posible almacenar los grafos creados manualmente en la base de datos local.	Sí
15	Es posible descargar un archivo PDF que contenga los resultados de la última ejecución de algoritmo.	Sí

3.3 Alcances y suposiciones

El software de Comparación de Rutas Metabólicas será una aplicación desarrollada completamente en web, la cual ayudará a investigadores en los cálculos de necesarios para sus trabajos. La aplicación deberá ser gratis, código open source y deberá poder ser accedida sin importar la ubicación geográfica del usuario.

Los investigadores podrán cargar sus propios archivos KGML provenientes de bases de datos científicas, esta información podrá ser utilizada por cualquier otro usuario en otro momento. Será responsabilidad de los administradores del sitio mantener la base de datos limpia de archivos indeseados.

Cada usuario tendrá la posibilidad de ejecutar cualquiera de las tres versiones de algoritmos de comparación que se implementarán; de igual manera se encontrará disponible la opción de cambiar los nombres de cada metabolito a la descripción común o a los códigos utilizados en la base de datos Kegg. Una función importante será la definición y edición de grafos manualmente y/o reordenamiento en las relaciones entre cada nodo.

Para el correcto funcionamiento de la aplicación será necesario contar con una conexión estable a Internet, ya que el sistema se encontrará alojado en servidores web. Toda la información del sistema es mantenida en una base de datos. La aplicación también cuenta con la opción de descargar reportes de los resultados o visualizarlos en pantalla.

3.4 Restricciones

La aplicación web será afectada por la disponibilidad del servidor donde se encontrará alojada, ya que el tiempo de ejecución del servidor será compartido con múltiples otras tareas que son importantes para la institución.

La conexión a Internet también será una restricción para la aplicación. Dado que la aplicación obtiene información de la base de datos a través de Internet, es crucial que exista una conexión estable para el correcto funcionamiento de la aplicación.

El portal web estarán limitados por la capacidad de la base de datos. Ya que la base de datos se comparte con otras aplicaciones del mismo servidor y puede ser forzada a poner en cola las solicitudes entrantes y por lo tanto aumentar el tiempo que lleva recuperar los datos y realizar los cálculos de comparación.

3.5 Funcionamiento

Componente	Espacio	Tiempo
Interfaz gráfica	~ 50 MB	~ 45 segundos
MongoDB	~ 1 GB	~ 2 segundos
Python	~ 1 MB	~ 3 segundos
Node	~ 500 MB	~ 2 segundos

IV. ARQUITECTURA DEL SISTEMA

4.1 Descripción

El sistema está hecho con varios servicios, entre se encuentra el uso de MEAN Stack el cual es un framework o conjunto de subsistemas de software para el desarrollo de aplicaciones, y páginas web dinámicas.

Este está compuesto por Mongo que es la base de datos que aloja las rutas metabólicas del proyecto.

ExpressJS que es un módulo de NodeJS y como tal funciona sobre esta plataforma; este módulo ofrece los métodos suficientes en JavaScript, para poder manejar las solicitudes o peticiones que se hacen por medio de los métodos del protocolo HTTP (GET, POST, etc.).

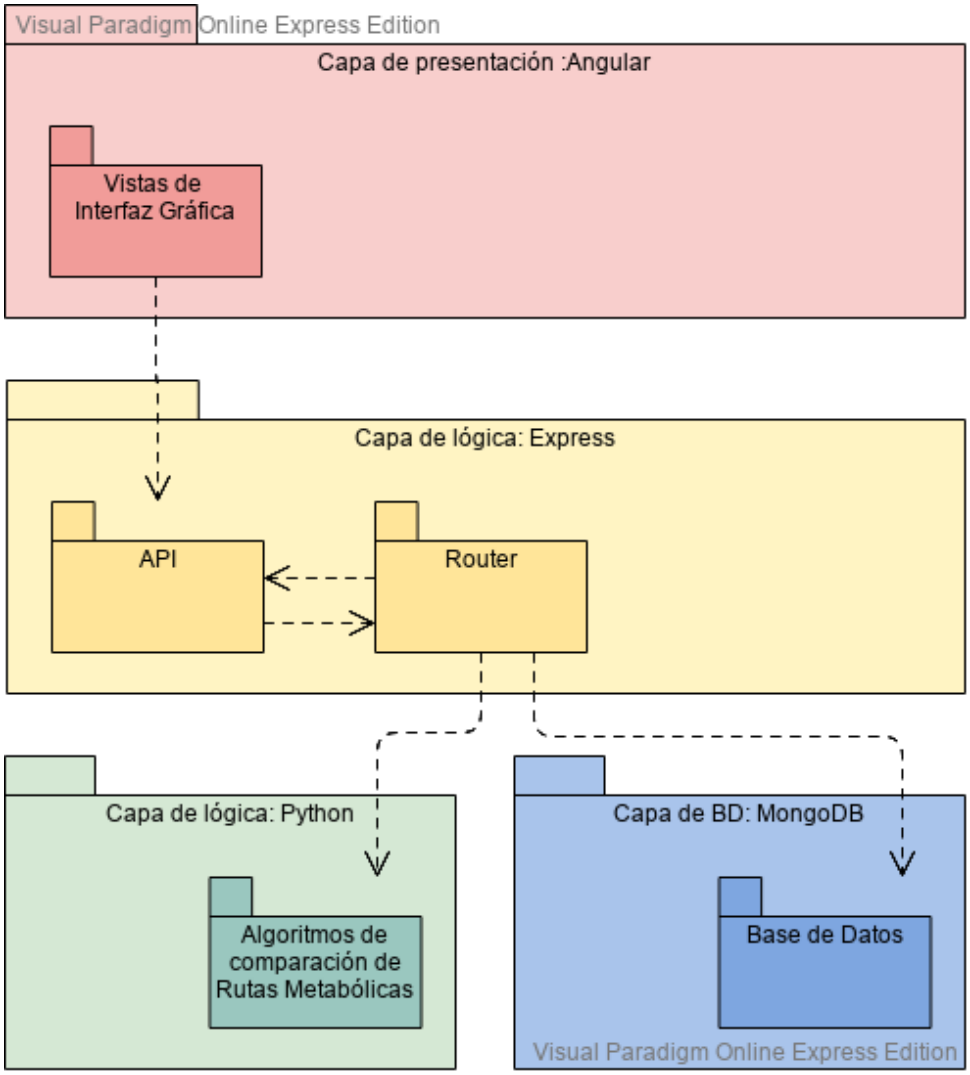
AngularJS que es es un framework que se utiliza para trabajar en el front end.

NodeJS que es una plataforma encargada del funcionamiento del servidor.

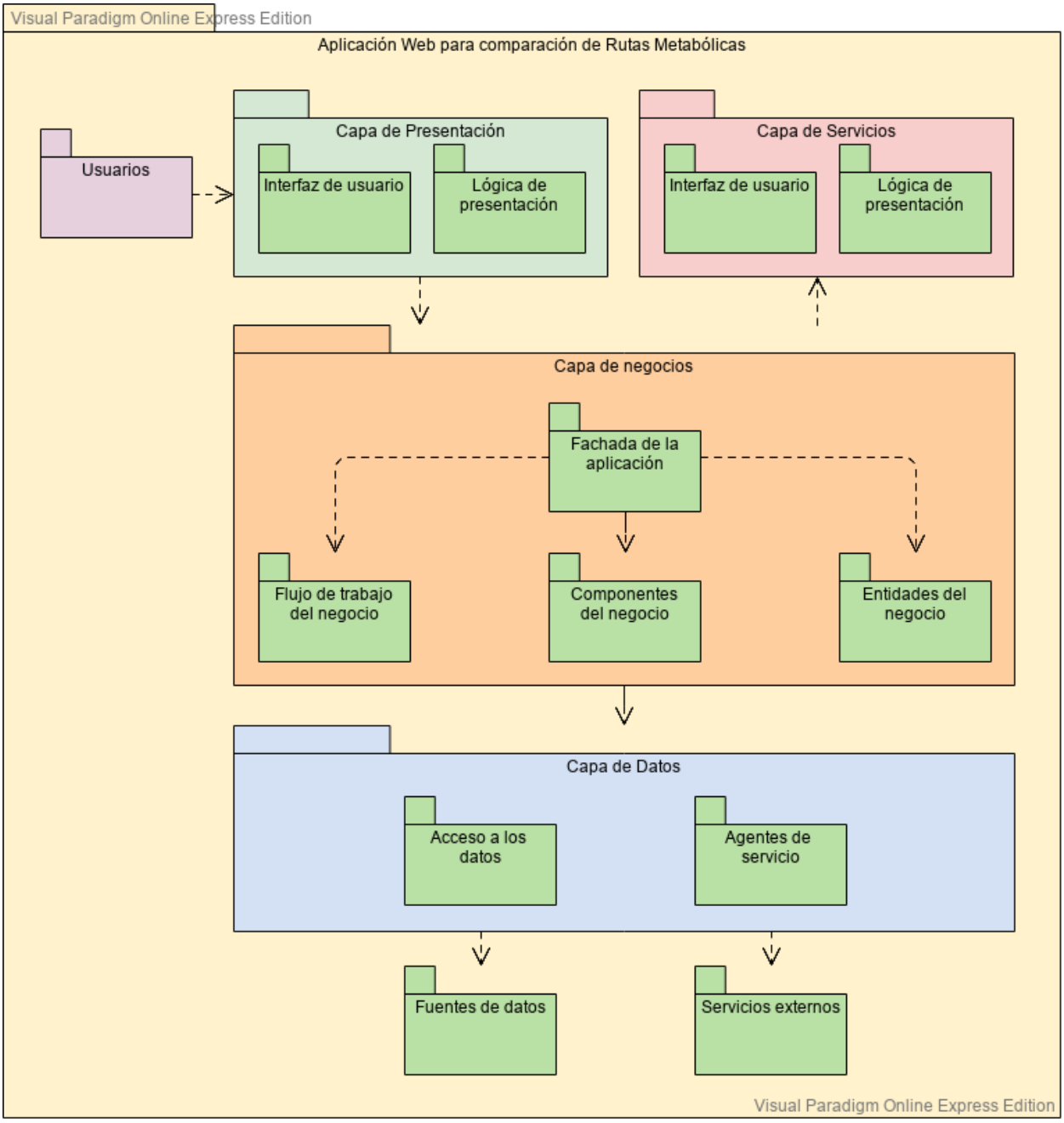
Aparte de mean se trabaja utilizando python para el manejo de las funciones para la comparación de rutas metabólicas.

Toda la aplicación se encuentra montada en el servidor de la escuela de computación llamada ebro.

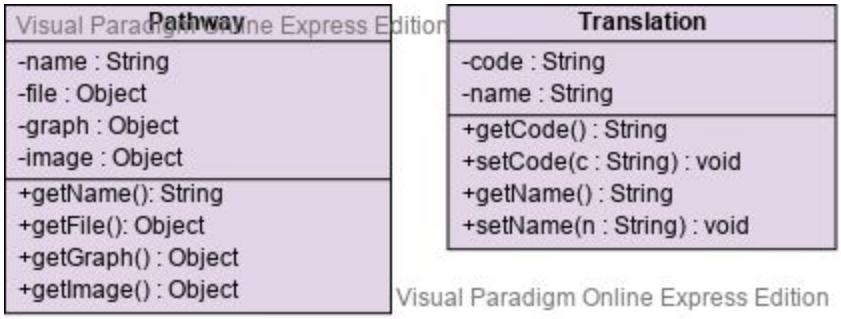
4.2 Diagrama de capas-paquetes



4.3 Diagrama de componentes



4.4 Diagrama de clases

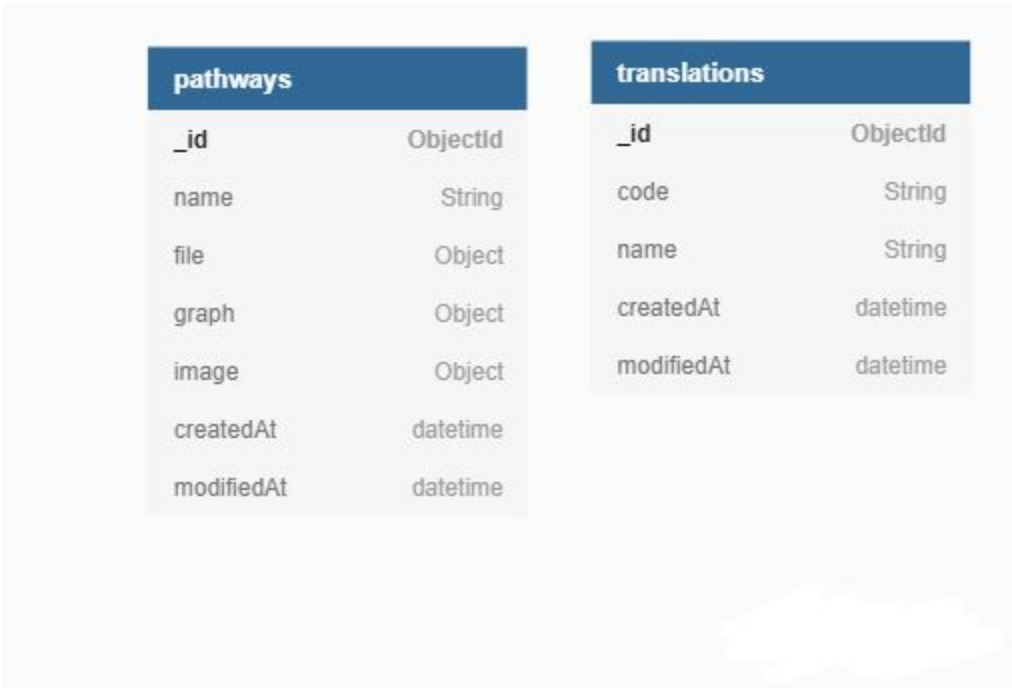


Pathway es una clase que representa una Ruta Metabólica, cuando un usuario decide cargar un archivo de formato XGML, el sistema genera un grafo en formato de JSON y una imagen que representa dicho grafo a partir de tal archivo. Toda esta información existe en los atributos de la clase Pathaway. Sus métodos son sencillos y se encargan de devolver la información contenida en los atributos.

Translation es otra clase que se encarga de representar una traducción del diccionario de código de metabolitos a nombres de uso común, de ahí sus atributos *code* y *name*. Sus métodos son simples *setters* y *getters*.

V. BASE DE DATOS

5.1 Esquema



5.2 Descripción en alto nivel de cada tabla

La tabla pathways contiene las rutas metabólicas almacenadas por los usuarios de la aplicación. Cada documento contiene un id autogenerado para el control de ítems duplicados, un nombre que corresponde al nombre del archivo original junto con un timestamp del momento en el que el archivo fue cargado en la aplicación, un archivo codificado como un buffer de datos que sería el XGML original de esa ruta, el grafo obtenido de esa ruta, representado en un JSON, y una imagen codificada como un buffer de datos también; además del momento de fecha de creación y modificación de de ese documento.

La tabla translations maneja dos atributos esenciales: name y code, estos indican una traducción del diccionario de metabolitos de código a nombre de uso común, ambos de tipo String. La tabla además contiene un id autogenerado al igual que la anterior para el control de duplicados y también fechas de creación y modificación del documento respectivo.

VI. PRUEBAS

6.1 Estrategia

Las pruebas se realizaron mediante la librería Chai en conjunto con el framework Mocha, ambos utilizados en NodeJS.

Las métricas que se utilizaron fueron si las pruebas pasaron o no para definir que las funcionalidades no presentaron alguna irregularidad, esto dado en un porcentaje del total de las pruebas que pasaron correctamente entre el total de las pruebas realizadas, que definirá la proporción de pruebas exitosas.

En la Gestión de la Configuración se pretendió llevar de la mano los casos de uso propuestos para el desarrollo de las iteraciones con casos de prueba específicos para cada uno los requerimientos para así poder identificar problemas en la funcionalidad del componente por entregar y realizar las correcciones pertinentes.

Los casos de uso orientados a ejecución de algoritmos tomaron en cuenta por lo menos cinco pruebas cada uno, con archivos seleccionados aleatoriamente que se ajusten a las necesidades de las pruebas. Los casos de tabular resultados de los algoritmos en pantalla se probaron bajo el

criterio experto de los desarrolladores de acuerdo con el resultado mostrado en pantalla y los datos recibidos por los algoritmos, de la misma forma se probó la traducción de códigos de metabolitos a nombres pues estos se evidencian en la nueva imagen de grafo generado con estas traducciones. Las pruebas realizadas fueron únicamente de software. Pruebas orientadas al hardware no fueron consideradas dentro del respectivo documento.

Los casos de uso orientados a archivos tomaron en cuenta por lo menos cinco pruebas cada uno, con archivos seleccionados aleatoriamente que se ajusten a las necesidades de las pruebas. El caso de visualizar en pantalla el grafo generado se probó bajo el criterio experto de los desarrolladores en conjunto con el cliente. Las pruebas realizadas fueron únicamente de software. Pruebas orientadas al hardware no fueron consideradas dentro del respectivo documento.

Los casos de uso orientados a ejecución de algoritmos tomaron en cuenta por lo menos cinco pruebas cada uno, con archivos seleccionados aleatoriamente que se ajusten a las necesidades de las pruebas. Los casos de tabular resultados de los algoritmos en pantalla se probaron bajo el criterio experto de los desarrolladores de acuerdo con el resultado mostrado en pantalla y los datos recibidos por los algoritmos, de la misma forma se probó la traducción de códigos de metabolitos a nombres pues estos se evidencian en la nueva imagen de grafo generado con estas traducciones. Las pruebas realizadas fueron únicamente de software. Pruebas orientadas al hardware no fueran consideradas dentro del respectivo documento.

Los casos de uso orientados a grafos definidos manualmente (definir grafos, aplicar algoritmos existentes, editar el orden de las relaciones, cargar en base de datos), tomaron en cuenta por lo menos cinco pruebas cada uno, que se ajusten a las necesidades de las mismas. El caso de prueba de generar reporte en archivo se probó bajo el criterio experto de los desarrolladores de acuerdo con el resultado mostrado en pantalla y el archivo generado por el sitio. Las pruebas realizadas fueron únicamente de software. Pruebas orientadas al hardware no fueron consideradas dentro

del respectivo documento.

La coordinación interna del equipo de desarrollo, y además, en conjunto con el cliente se dio mediante grupos en redes sociales tales como Telegram en donde se discutían problemas encontrados en el proceso, así como asuntos pendientes, organización de reuniones y acuerdos de menor impacto. Tales grupos también se usaban para tratar temas relativos a pruebas de software.

6.2 Resultados del test

Se ha realizado pruebas a diferentes funcionalidades del sistema, entre estas están: los casos de uso orientados a ejecución de algoritmos, el caso de visualizar en pantalla el grafo generado, los casos de uso orientados a archivos, los casos de tabular resultados de los algoritmos en pantalla, la traducción de códigos de metabolitos a nombres, los casos de uso orientados a grafos definidos manualmente y el caso de prueba de generar reporte en archivo.

Todas estas pruebas realizadas al sistema nos mostró resultados que nos permitió verificar que el sistema es confiable para su uso.

Y los únicos cambios realizados fue por decisión del cliente no tanto por errores.

Como se indica antes, solo se realizaron pruebas al software y no al hardware.

VII. REFLEXIÓN

7.1 Evaluación

La conexión de todo el sistema es un éxito, y la manera en la que se realizó la interfaz y la conexión con las funciones fue un aprendizaje completo que claramente nos hizo invertir mucho tiempo aprendiendo, esto junto con la herramienta de angular fue algo que quisiéramos haber conocido desde un inicio para no tener que haber invertido tanto tiempo en errores que de cierta manera eran simple de solucionar con un conocimiento regular de la herramienta que conocemos ahora.

Algo que nos quitó algo de tiempo fue entender las funciones de python encargadas de realizar la comparación, entonces tuvimos que convertir estas funciones de una manera que nos sirviera para nuestro proyecto y para ser usada por MEAN stack.

Al inicio comenzamos de una manera que creíamos que iba a ser más fácil y por el contrario nos equivocamos y tuvimos que hacer una reestructuración del proyecto para ponerlo en buen camino y no seguir gastando tiempo en errores que ocurrieron simplemente por no haber iniciado de una manera correcta.

7.2 Lecciones

Todo lo que aprendimos trabajando con las herramientas de node y angular es muy valiosa y nos permite utilizar este conocimiento en un futuro para tareas o proyectos que requieran el uso de estas.

Si tuviéramos que volver a realizar un proyecto que utilice lo mismo definitivamente lo reestructurariamos de una manera que más adelante nos facilite todo. Ya que en este proyecto aprendimos que ciertas cosas no hacen como creíamos.

Problemas como el manejo de errores y casos alternos de respuestas fue algo que al inicio por no saber cómo manejar se nos complicó y nos dio mucho problemas, con el tiempo implementamos algunos métodos para manejar esto.

También el envío de datos entre las diferentes herramientas fue algo que por no conocer nos dio muchos problemas y nos hizo gastar mucho tiempo para aprenderlas. Claramente con el conocimiento previo el proyecto hubiera avanzado mucho más rápido de lo esperado.

7.3 Errores y limitaciones conocidos.

Algunas funciones que nos hizo falta de implementar fueron: la visualización de las rutas en pantalla ya que se nos pedía hacerle un zoom a esta imagen y eso no se pudo implementar a tiempo.

Y convertir los códigos a nombres no se terminó, ya que la función en Python que hacía esto no fue terminada a tiempo y no había cómo hacer esta conversión, aparte como esto sucedió tampoco se implementó una función en Javascript que enviará el diccionario para su respectivo uso.

VIII. APÉNDICES

A. OTROS DOCUMENTOS

1. Minutas con el usuario

2. Cartas de aprobación y calificación del usuario

3. Cronograma original y modificado del proyecto

gantter™

Project Name **Vision del proyecto**

	Nombre	Duración	Inicio	Fin
1	Definir el Project Charter	6.5días?	02/11/2019	02/17/2019
2	Terminar Canvas Social	1día?	02/18/2019	02/19/2019
3	Terminar Mapa de empatía	1día?	02/19/2019	02/20/2019
4	Completar la visión del proyecto	4días?	02/20/2019	02/24/2019
5	Crear Prototipo completo	6.5días?	02/25/2019	03/03/2019
6	Terminar el ERS	6.5días?	03/04/2019	03/10/2019
7	Definir el plan de pruebas 1	6.5días?	03/11/2019	03/17/2019
8	Ejecución 1 con manual y carta de usua...	6.5días?	03/18/2019	03/24/2019
9	Definir el plan de pruebas 2	6.5días?	03/25/2019	03/31/2019
10	Ejecución 2 con manual y carta de usua...	6.5días?	04/01/2019	04/07/2019
11	Completar el SAD	6.5días?	04/08/2019	04/14/2019
12	Definir el plan de pruebas 3	13.5días?	04/15/2019	04/28/2019
13	Ejecución 3 con manual y carta de usua...	6.5días?	04/29/2019	05/05/2019
14	Plan de pruebas aceptación del sistema	6.5días?	05/06/2019	05/12/2019
15	Manual usuario completo	6.5días?	05/13/2019	05/19/2019
16	Inf pruebas aceptación del sistema	6.5días?	05/20/2019	05/26/2019
17	Manual técnico	6.5días?	05/27/2019	06/02/2019

B. TABLAS DE LA BASE DE DATOS

1. Pathways

La tabla pathways contiene las rutas metabólicas almacenadas por los usuarios de la aplicación. Cada documento contiene un id autogenerado para el control de ítems duplicados, un nombre que corresponde al nombre del archivo original junto con un timestamp del momento en el que el archivo fue cargado en la aplicación, un archivo codificado como un buffer de datos que sería el XGML original de esa ruta, el grafo obtenido de esa ruta, representado en un JSON, y una imagen codificada como un buffer de datos también; además del momento de fecha de creación y modificación de de ese documento.

2. Translations

La tabla translations maneja dos atributos esenciales: name y code, estos indican una traducción del diccionario de metabolitos de código a nombre de uso común, ambos de tipo String. La tabla además contiene un id autogenerado al igual que la anterior para el control de duplicados y también fechas de creación y modificación del documento respectivo.

C. PROGRAMACIÓN

1. Descripción programas capa interfaz usuario

Para la capa de interfaz/usuario se utiliza Angular el cual es un framework que brinda una gran variedad de efectos, de una forma sencilla, reduciendo contundentemente la cantidad de código, lo que permite que sea mucho más sencillo de mantener.

2. Descripción programas capa reglas negocio

Para la capa de negocio se utiliza node y python.

Node: Es la plataforma encargada del funcionamiento del servidor la cual funciona totalmente con JavaScript.

Python: Es un lenguaje de programación interpretado, encargado en este proyecto de realizar todas las algoritmos aplicados a las rutas.

3. Tipos de datos abstractos

3.1. Pathway

Este tipo de dato representa una ruta metabólica. Por el momento no es mutable.

Name: Es de tipo String, representa el nombre del archivo XGML junto con un timestamp del momento en que el archivo fue cargado en el sistema.

File: Es de tipo Object, representa los datos contenidos en el archivo XGML. La idea es utilizar estos datos para generar un archivo XGML exactamente igual al cargado en el sistema.

Graph: Es de tipo Object, representa el grafo generado del archivo XGML en forma de JSON.

Image: Es de tipo Object, representa los datos contenidos en la imagen generada que visualiza el grafo también generado de la ruta metabólica.

3.2. Translation

Este tipo de dato representa una traducción del diccionario de metabolitos. Es mutable.

Code: Es de tipo String, representa un código en la entrada del diccionario que se asocia a un nombre.

Name: Es de tipo String, representa el nombre de uso común que un código de metabolito puede tomar.