Hernan Daniel Garofolo Santucho

Organizador de aulas mediante el uso de algoritmos genéticos

2023

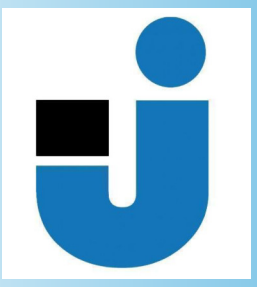
Instituto de Ingeniería y Agronomía

Carrera: Ingeniería en Informática

Universidad Nacional Arturo Jauretche

Instituto de Ingeniería y Agronomía

Carrera de Ingeniería en Informática



PRÁCTICA PROFESIONAL SUPERVISADA
Informe final

Organizador de aulas mediante el uso de algoritmos genéticos

Hernan Daniel Garofolo
Santucho

Florencio Varela, Diciembre, 2023

Estudiante

Nombre y Apellido: Hernán Garofolo
Correo electrónico: hernangarofolo@gmail.com
Organización donde se realiza la Práctica Profesional Supervisada
Nombre de la Empresa-Entidad-Institución: Universidad Nacional Arturo Jauretche
Dirección: Av. Calchaquí 6200, Florencio Varela, Buenos Aires, Argentina
Teléfono: +54 11 4275-6100
Sector: Programa TICAPPS (Tecnologías de la Información y la Comunicación en Aplicaciones de Interés Social), Instituto de Ingeniería y Agronomía

Tutor organizacional

Nombre y apellido: Martin Morales
Correo electrónico: martin.morales@unaj.edu.ar

Docente Supervisor

Nombre y apellido: Alejandro Vazquez
Correo electrónico: alevazquez.d@gmail.com

Docente tutor del Taller de Apoyo para la Producción de Textos Académicos

Nombre y apellido: Carolina Kelly
Correo electrónico: kellygcarolina@gmail.com

Coordinador de la Carrera de Ingeniería en Informática

Nombre y apellido: Martin Morales
Correo electrónico: martin.morales@unaj.edu.ar

Resumen. El presente trabajo tiene como objetivo desarrollar una solución para la problemática acerca de la asignación de aulas en un establecimiento educativo teniendo en cuenta las restricciones que comúnmente se observan. Se diseña una solución que intenta resolver la problemática de manera automática y computarizada aplicando algoritmos genéticos.

Se aplican los operadores genéticos de selección, cruza, mutación y evolución para obtener la solución óptima sin violar las restricciones obligatorias.
El sistema recibe varios inputs (aulas, materias) los cuales son analizados y ordenados, los resultados son exportados y guardados en un Excel que el usuario final puede guardarlos.
Con el sistema desarrollado se busca evitar la pérdida de tiempo en el procesamiento manual y evitar los errores humanos para obtener la mejor distribución de las aulas.
Este informe resume los resultados, objetivos, metodología, componentes técnicos, funcionalidades e impacto potencial del desarrollo.

Palabras claves. Algoritmos genéticos, Optimización, Scheduling.

Abstract. The objective of this work is to develop a solution for the problem of assigning classrooms in an educational establishment, taking into account the restrictions that are commonly observed. A solution is designed that attempts to solve the problem automatically and computerized by applying genetic algorithms.

The genetic operators of selection, crossover, mutation and evolution are applied to obtain the optimal solution without violating the binding constraints.
The system receives various inputs (classrooms, subjects) which are analyzed and ordered, the results are exported and saved in an Excel that the end user can save.
The developed system seeks to avoid wasting time in manual processing and avoid human errors to obtain the best distribution of the classrooms.
This report summarizes the outcomes, objectives, methodology, technical components, implemented functionalities, and potential impact of the development.

Keywords: Genetic algorithms, Optimization, Scheduling

Índice

Contenido

1. Introducción

En el panorama tecnológico actual, la gestión eficaz de recursos constituye un desafío incesante. Para las instituciones educativas y organizaciones que confían en la asignación efectiva de aulas, esta labor se torna aún más intricada. Como respuesta a esta exigencia, surge la iniciativa de concebir una API REST que haga uso de algoritmos genéticos para la organización dinámica y eficiente de las aulas.

1.1 Descripción del proyecto

Con el correr de los años se buscaron solucionar los problemas cotidianos basándonos en la tecnología existente e innovadora. Desde hace unos años podemos enfrentar estas problemáticas con tecnología para encontrar la solución óptima en la menor cantidad de tiempo y con el menor uso de recursos. La organización de la asignación de aulas en los establecimientos educativos siempre fue un problema difícil de resolver porque se deben tener en cuenta muchas variables. Sin embargo, encontramos en la tecnología la manera más eficiente de resolver este problema.

Hay diversos métodos que se pueden dar respuestas al problema, en este informe nos centraremos en el desarrollo de una solución para el problema de la asignación de aulas mediante la creación de una API REST, la cual se define como un conjunto de reglas y convenciones que posibilitan la comunicación eficiente y estandarizada entre sistemas distribuidos. Dicha API contiene la lógica necesaria que determina cual es el mejor resultado basándose en el uso de algoritmos genéticos, los cuales están inspirados en los procesos evolutivos naturales, convirtiéndolos en una herramienta poderosa para la optimización de diversos problemas.

Se propone desarrollar una solución que aborde el problema de asignación de aulas a través de la API REST. Este informe expone los resultados derivados de dicho desarrollo,

detallando el objetivo, la metodología aplicada, los resultados obtenidos, los componentes técnicos involucrados, las funcionalidades implementadas y el potencial impacto que esta solución podría tener en la comunidad educativa.

A lo largo del desarrollo se analizará el alcance de los algoritmos genéticos en la resolución de los problemas de asignación de aulas y de cómo la tecnología anidada con la teoría de evolución natural, puede ofrecer soluciones óptimas e innovadoras con un gran potencial para revolucionar la manera en que abordamos los desafíos relacionados con la gestión de recursos educativos.

1.2 Antecedentes

El autor propuso el modelo de algoritmos genéticos para abordar problemas de optimización en una amplia variedad de situaciones. Este modelo se fundamenta en la expresión matemática de la evolución natural, donde solo los individuos más adaptados son los que sobreviven (Holland J. ,1975).

Dado que el problema de asignación de aulas es diferente en cada establecimiento, no se encuentra una solución genérica, sino que se encuentran soluciones específicas en cada ocasión.

Algunos ejemplos donde se resuelve la problemática con algoritmos genéticos son:

- Un caso donde se implementó fue en los problemas del tipo de asignación de aulas en un centro universitario(Pidre,2002) limitado a los exámenes, mediante la utilización de algoritmos genéticos. los cromosomas fueron representados por un vector y la función de aptitud está dada por:

$$C_i = \frac{1}{d_m^2 \cdot k} \qquad \text{Si } d_m < d_{min}$$

$$C_i = 0 \qquad \text{En caso contrario}$$

- Otro caso fue la implementación de un sistema que genera horarios óptimos de una universidad utilizando algoritmos genéticos, basados en la cantidad de alumnos, la disponibilidad de los docentes y la cantidad de aulas (Vásquez, A., 2012).

Hay numerosos ejemplos que destacan la pertinencia de los algoritmos genéticos como un modelo sumamente aplicable. Su capacidad para explorar soluciones potenciales y mejorar iterativamente a lo largo del tiempo los convierte en una herramienta versátil y poderosa para abordar problemas en campos tan diversos como la planificación, la programación y la toma de decisiones. Además, su naturaleza adaptativa y su capacidad para manejar conjuntos de datos grandes hacen que los algoritmos genéticos sean una opción valiosa en el desarrollo y la mejora continua de sistemas complejos.

1.3 Planteamiento de la problemática

Realizar la asignación de aulas para cada materia implica una tarea compleja, dada la cantidad de variables involucradas. Este proceso puede ser complejo si se realiza manualmente, puede tomar desde unos pocos días hasta semanas tener una solución definitiva sin embargo con el aumento de alguna de las variables el tiempo necesario se incrementa exponencialmente y puede propiciar la aparición de numerosos errores.

Para solucionar el proceso, se propone desarrollar una API REST la cual automatiza la asignación de aulas considerando factores como la cantidad total de alumnos, la capacidad de las aulas, el tamaño de las comisiones de cada materia y el turno correspondiente.

Complementariamente, se implementará una interfaz de usuario basada en un bot de Telegram. Esta interfaz permitirá a los usuarios consultar las materias en las que se han

inscrito y conocer las aulas asignadas a dichas materias, proporcionando así una solución integral y fácil de usar.

Es fundamental que los datos relativos a las aulas, materias, alumnos y los resultados obtenidos se almacenen de manera estructurada en una base de datos, estableciendo así un repositorio centralizado y confiable que permita su almacenamiento a largo plazo y su consulta eficiente en futuras instancias.

1.4 Objetivo

El objetivo principal consiste en diseñar y desarrollar un algoritmo genético que solucione de manera eficiente el problema de la asignación de aulas en los diferentes turnos. Además, se buscan objetivos secundarios que, aunque menos prioritarios, contribuyen a mejorar la experiencia de usuario. Entre estos objetivos adicionales se incluye la implementación de una interfaz de usuario basada en un bot de telegram donde los alumnos pueden consultar diversas cuestiones referidas a la universidad, como las inscripciones a las materias, mapa de aulas, links de webs oficiales y el calendario académico.

El proyecto consta de 5 características claves: El algoritmo genético, la API REST, la interfaz de usuario, la persistencia de datos y la interfaz de resultados. De esta manera se busca que la experiencia del usuario final, al igual que la de los alumnos, sea eficiente, transparente, adaptable y escalable.

2. Marco Teórico

En esta sección, se revisará el marco teórico esencial que respalda el desarrollo de la solución propuesta. Se explorarán conceptos clave, comenzando con los Algoritmos Genéticos, una potente herramienta de optimización inspirada en la evolución biológica. Posteriormente, se analizará la Telegram Bot API, que funciona como la interfaz de usuario basada en Telegram para la consulta y gestión de datos relacionados con el sistema. Seguidamente, se examinará MongoDB, una base de datos NoSQL empleada para almacenar de manera eficiente la información necesaria. Finalmente, se abordarán los JWT (JSON Web Tokens), cuyo papel es crucial en la seguridad y autenticación del sistema.

Algoritmo genético

Los algoritmos genéticos están basados en la teoría de la evolución y buscan dar solución a una amplia gama de problemas de optimización y búsqueda. Están inspirados en la evolución biológica y tienen en cuenta procesos como la selección natural, evolución y mutación.

La idea general está acompañada de diversos procesos naturales que son codificados para ser simulados, comenzando con la codificación del problema en forma de ***cromosoma***, una secuencia de genes, que refleja una posible solución.

Para poder realizar la aproximación a una solución, se debe generar aleatoriamente una ***población inicial*** de cromosomas. Esta población va a tener que ser procesada por una ***función de aptitud***, comúnmente llamada función de ***fitness***, donde se evalúa que tan buena es la solución obtenida de los cromosomas en la población inicial. Este valor es de suma importancia porque identifica cuales son los cromosomas con posibilidad de sobrevivir y reproducirse.

Una vez obtenidos los cromosomas más aptos, son seleccionados para la ***reproducción***. Esta selección se realiza mediante el uso de probabilidades equivalente al proceso de selección natural.

Tomando los cromosomas más aptos, se realiza un ***cruzamiento*** de a pares, donde cada cromosoma combina e intercambian segmentos de sus genes para generar uno nuevo, permitiendo combinar las características favorables.

Como en la naturaleza, existe una probabilidad de ***mutación***, algunos genes de un cromosoma pueden cambiar aleatoriamente, esto intenta evitar las soluciones subóptimas. Una vez realizados los pasos de mutación y cruzamiento, se busca reemplazar parte de la población anterior con la nueva descendencia, intentando que los cromosomas menos aptos sean ***reemplazados***. Los últimos 3 pasos se repiten para generar múltiples generaciones, lo que hace que el algoritmo converja en soluciones más óptimas, que está definido por un criterio de terminación donde puede considerarse un número finito de generaciones, una aptitud deseada o cuando el progreso se estanca.

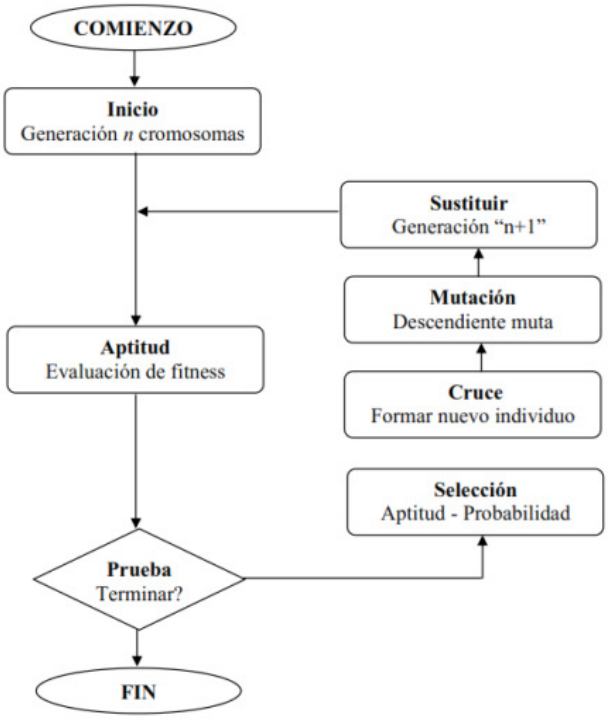


Imagen 1. Diagrama de flujo del algoritmo genético básico

Fuente: (Naupari & Rosales, 2010)

En pseudocodigo:

- Inicializar población aleatoria (población inicial)
- Evaluar aptitud de cada individuo en la población (fitness)
- Repetir hasta que se alcance un criterio de terminación:

  Seleccionar padres mediante algún método de selección (selección)

  Aplicar cruzamiento a los padres seleccionados para crear descendencia(cruzamiento)

  Aplicar mutación a algunos individuos de la descendencia(mutación)

Evaluar aptitud de los nuevos individuos (descendencia + algunos miembros de la población actual)

Seleccionar individuos para la próxima generación (reemplazo)

- Fin del bucle
- Devolver la mejor solución encontrada

Telegram Bot API

Telegram es una plataforma de mensajería instantánea gratuita donde los usuarios pueden enviar mensajes de texto o con algún archivo multimedia. Además, proporciona servicios automatizados que pueden interactuar con los usuarios, llamados bots.

Para poder conectar el servicio de bots, se requiere una cuenta de telegram desde la cual se obtendrá el acceso al bot y a las claves correspondientes de configuración para procesar y responder solicitudes desde un servidor específico

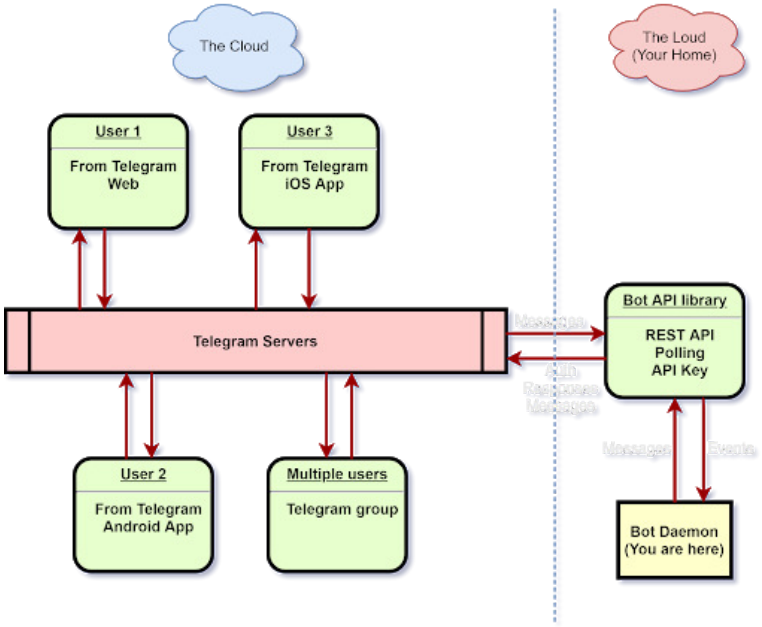


Imagen 2. Arquitectura de un bot de telegram

MongoDB

MongoDB se basa en la arquitectura NoSQL (No solo lenguaje de consulta estructurado) orientadas a documentos, a diferencia de las bases SQL que utilizan tablas, mongo utiliza documentos en formato BJSON que permite tener una estructura más flexible y escalable.

Las entradas en MongoDB son flexibles permitiendo ingresar texto, documentos, objetos o matrices en el mismo campo. Permite tener datos duplicados.

MongoDB admite el uso de índices para acelerar las búsquedas dado que mejoran la velocidad de búsqueda y recuperación de información, ampliando enormemente el rendimiento, siempre que su uso sea el adecuado

Uno de los beneficios más grandes que tiene MongoDB es su escalabilidad, dado que se escala horizontalmente agregando más servidores en vez de verticalmente (se agrega más recursos a un único servidor).

MongoDB es una base de datos distribuida que permite la creación de réplicas que mejoran la disponibilidad y la tolerancia a fallos. También admite la fragmentación cuando se procesan grandes volúmenes de datos, permitiendo una distribución eficiente de carga y almacenamiento.

El lenguaje de consulta es muy flexible y permite realizar cálculos y transformaciones avanzadas de datos.

Al igual que otras bases de datos, MongoDB proporciona seguridad mediante métodos de autenticación y autorización.

JSON Web Token

JSON Web Token (JWT) es un estándar abierto establecido en la RFC 7519 (Sakimura, 2015) que define una forma compacta y autónoma de transmitir información de forma segura entre partes como un objeto JSON. Esta información se puede verificar y confiar porque está firmada digitalmente.

El principal uso está determinado para la autenticación, es decir, en cada petición se envía adjunto un token válido para poder acceder a rutas, recursos, etc. Sin embargo, también se puede usar en la transferencia de datos entre partes para mantener la seguridad de las firmas.

Tiene una estructura de 3 partes:

Encabezado

El encabezado consta de dos partes: el tipo de token, que es JWT, y el algoritmo de firma que se utiliza, como HMAC SHA256 o RSA.

Carga útil

La carga útil contiene los reclamos, que son declaraciones sobre una entidad que proporcionan un conjunto de reclamaciones útiles e interoperables. Existen 3 tipos de reclamaciones: las registradas, las públicas y las privadas.

Firma

La firma es la parte más importante dado que es donde estará codificado el encabezado, la carga útil, el secreto, el algoritmo utilizado en el encabezado y la firma en sí. La firma se utiliza para validar que el token no se modificó en el camino.

Estas tres partes se fusionan en una cadena separada por puntos en Base64 lo que hace que sean más fácil de tratar con HTTP y además más cortas que las utilizadas en XML o Saml.

Cuando el usuario inicia sesión se genera un token de autorización que será utilizado en los próximos envíos de solicitudes desde el cliente con esas credenciales, básicamente se agrega el token en un header de la solicitud y este será analizado para saber si es válido o no. La validez va a depender de las características asociadas al producto, puede ser por tiempo, por cantidad de solicitudes o por ambas para ello varía la lógica de la validación de datos.

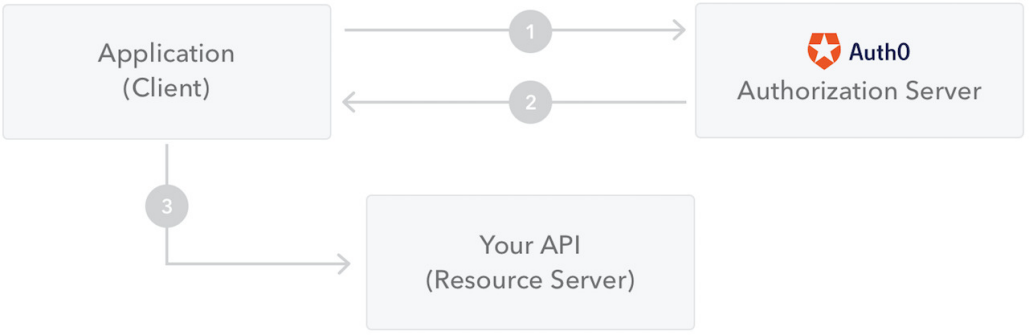


Imagen 3. Arquitectura JWT

3. Desarrollo

La creación de un sistema que utilice algoritmos genéticos para asignar y organizar aulas no solo promete una distribución eficiente de recursos físicos, sino que también se adapta dinámicamente a las cambiantes necesidades y restricciones. Este enfoque no solo simplifica la logística, sino que también maximiza la utilidad de los espacios, mejorando así la experiencia educativa y optimizando los recursos disponibles.

3.1 Alcance

El proyecto se enfoca en desarrollar una API Rest que incorpora un algoritmo genético para optimizar la disposición de aulas y asignación de materias. Este proceso

considerará diversos factores, como horarios, equipamiento de las aulas, requerimientos específicos de cada materia, capacidad de las aulas, cantidad de alumnos por materia y la naturaleza de las aulas (como laboratorios). La finalidad es lograr una solución automatizada y eficiente para la organización universitaria.

Los objetivos generales son:

- Creación de una api rest para la gestión de aulas: La api está destinada para ser fácilmente integrada y posee endpoints para la generación de soluciones son un simple llamado.
- Implementación del algoritmo genético: Se desarrolla un algoritmo genético específico que satisface la asignación de aulas de manera eficiente y automatizada.
- Integración de bot de Telegram para consultas de alumnos acerca de las aulas donde deben cursar según las materias que eligieron. Permite que los alumnos puedan consultar las aulas de las materias donde se inscribieron mediante un bot de telegram llamado Materias_Alumnos_Unaj (@materiasAlumnosUnajBot).
- Implementación de un login para la ejecución y consultas del algoritmo genético.
- Exposición de endpoints para registros de usuarios finales, alumnos y carga de materias y aulas.
- Guardado de datos de tiempos de ejecución para análisis estadístico.

Es importante destacar que en esta versión del proyecto no se tendrán en cuenta características específicas de docentes y alumnos, ni se considerará la duración de las materias en el proceso de optimización.

Estos servicios tienen el potencial de evolucionar al incorporar nuevos endpoints, permitiendo la captura de datos desde diversas fuentes, exponer informes, expandir su escala para satisfacer las necesidades de diferentes usuarios, entre otras posibilidades.

3.2 Metodologías a utilizar

Dada la envergadura del proyecto, se llevó a cabo una exhaustiva búsqueda de las metodologías más apropiadas para asegurar una realización y control eficiente del mismo. Tras un análisis detenido, se optó por la implementación de la metodología ágil Scrum, la cual se distingue por su facilidad en la gestión de proyectos. La elección de Scrum se fundamenta en su capacidad para dividir el proyecto en sprint cortos, proporcionando así una herramienta esencial para el análisis preciso de los tiempos y la adaptación continua a medida que avanza el desarrollo.

La estructuración y ejecución de tareas se encuentran organizadas de manera sistemática en un diagrama de Gantt. Esta representación visual facilita la comprensión global del proyecto, permitiendo una lectura ágil y eficiente de las fases de trabajo y sus interrelaciones (ver anexo 1).

Para garantizar una gestión de código efectiva y colaborativa, el proyecto se encuentra versionado en Git. Esta elección estratégica no solo simplifica el control de versiones, sino que también posibilita la futura colaboración entre distintos miembros del equipo de desarrollo. El uso de Git, como sistema de control de versiones distribuido, permite un seguimiento preciso de los cambios realizados en el código, facilitando la identificación de contribuciones individuales y asegurando la coherencia del desarrollo en conjunto.(ver https://github.com/garofoloh/genetic).

3.3 Tecnologías utilizadas

Para el desarrollo de la api rest se utilizaron las siguientes tecnologías:

- Lenguaje de programación: Kotlin
- Versión de java: 16
- Frameworks: Spring boot, JWT
- Telegram bot
- Base de datos: MongoDB
- Contenedor: Docker
- Herramientas de pruebas: Junit, Mockito, Postman

La elección de las tecnologías para este proyecto se basó en criterios específicos que priorizan la eficiencia, la flexibilidad y la compatibilidad con los objetivos planteados. Es importante comprender que la elección de las tecnologías adecuadas es fundamental para el éxito de cualquier proyecto de software.

A continuación, se detallan las razones detrás de la elección de cada tecnología:

Lenguaje de programación: Kotlin fue seleccionado como el lenguaje de programación principal debido a su interoperabilidad con Java, lo que facilita la integración con otras tecnologías y bibliotecas existentes en el ecosistema Java. Además, Kotlin ofrece una sintaxis concisa y expresiva, lo que conduce a un desarrollo de código más limpio y eficiente.

Versión de Java: La elección de Java 16 se fundamenta en la necesidad de aprovechar las últimas características y mejoras introducidas en esta versión, garantizando así un rendimiento optimizado y el acceso a las últimas innovaciones del lenguaje.

Frameworks:

Spring Boot: Este framework fue elegido por su capacidad para simplificar el desarrollo de aplicaciones Java, proporcionando un entorno robusto y modular. Su amplia comunidad y el soporte para el desarrollo de microservicios hacen de Spring Boot una elección sólida para la construcción de sistemas escalables y eficientes.

JWT (JSON Web Tokens): Se optó por JWT como un estándar para la autenticación y autorización debido a su capacidad para transmitir información de manera segura entre partes. Esto es crucial para la implementación de un sistema seguro y confiable.

Base de datos: MongoDB fue elegido como el sistema de gestión de bases de datos debido a su naturaleza NoSQL y su capacidad para manejar grandes volúmenes de datos de manera eficiente. Además, MongoDB es altamente escalable y ofrece una gran flexibilidad en la estructura de los datos.

Integración de un bot de Telegram La integración de un bot de Telegram se decidió para ofrecer una interfaz de comunicación amigable y accesible. Telegram proporciona una API robusta y fácil de usar, permitiendo la interacción eficiente con los usuarios y facilitando la notificación y gestión de eventos relevantes para el proyecto.

Docker: La adopción de Docker como plataforma de contenedorización está basada por su capacidad para encapsular aplicaciones y sus dependencias, lo que simplifica el despliegue y la gestión de entornos de desarrollo y producción. La portabilidad y la consistencia que proporciona Docker son fundamentales para garantizar un despliegue sin problemas en diferentes entornos.

La elección de estas tecnologías se basa en la sinergia entre sus características individuales, apuntando a construir un sistema eficiente, escalable y fácilmente mantenible.

3.4 Arquitectura

La elección de la arquitectura modelo-vista-controlador, más conocida como MVC, se fundamenta en su capacidad demostrada para proporcionar una estructura organizada y modular en el desarrollo de software. Otro aspecto a tener en cuenta es su capacidad para proporcionar una estructura mantenible y escalable, así como su habilidad para mejorar la colaboración y fomentar la reutilización del código (ver Imagen 4).

Consta de 3 capas:

- **Modelo**: Es la capa donde se representa la lógica de negocio. Es el responsable de gestionar los datos necesarios como aulas, alumnos y materias. También es el encargado de manejar el algoritmo genético y la optimización de este. Las principales tareas que tiene asignada esta capa son almacenar y gestionar los datos a utilizar, ejecución de algoritmos genéticos, actualización de datos en la base, manejo de las consultas de los alumnos desde el bot de telegram y la generación del archivo con los resultados de la ejecución.
- **Vista**: La vista representa la forma en la cual los usuarios obtendrán la respuesta. En nuestro caso utilizamos 2 vistas, un json en respuesta a una consulta desde el bot de telegram que será mostrado en telegram y el archivo Excel generado con el resultado de la ejecución del algoritmo genético.
- **Controlador**: Es el encargado de mediar entre ambas capas anteriores, para ello maneja las solicitudes HTTP entrantes, se comunica con el modelo y formatea la respuesta de acuerdo a las necesidades del cliente. El endpoint utilizado para ejecutar el algoritmo genético es un método post a “/v1/generate”.

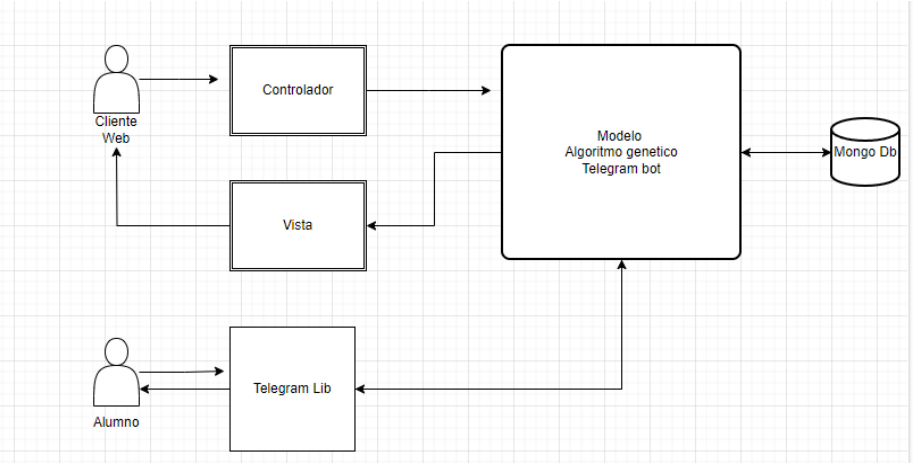


Imagen 4. Arquitectura de la api rest

3.5 Algoritmo genético

Los algoritmos genéticos son una técnica de optimización que simula la evolución natural para encontrar soluciones eficientes a problemas complejos. En la organización de aulas universitarias, los algoritmos genéticos desempeñan un papel fundamental en la asignación óptima de aulas a materias. A continuación, se detallan los aspectos más importantes de su implementación:

Representación de Genes y Cromosomas:

Se representan los cromosomas como un par de genes, uno que tiene las características de las aulas y otro las características de las materias. Se busca que los cromosomas evaluados en la función de aptitud tengan la puntuación más alta, lo que resulta en la mayor de afinidad entre genes.

Genes: En la solución definida existen dos tipos de genes diferentes que formarán parte del cromosoma. Un gen Aula con las propiedades específicas de cada aula, es decir, un listado de equipamientos disponibles, el tamaño del aula, el turno, un booleano que indica si se trata de un laboratorio y el número de aula.

El segundo gen está destinado a representar las características específicas de las materias, englobando información como el nombre de la materia, el número de comisión, el turno asignado, el límite máximo de estudiantes, los equipos necesarios, y un indicador booleano que señala si la materia requiere cursarse en un laboratorio.

Cromosoma: Un cromosoma es una solución completa y se compone de una lista de genes que representan la asignación de aulas para todas las clases.

Función de Aptitud:

La función de aptitud (fitness function) es crucial para evaluar cuán buena es una solución candidata en términos de optimización. La evaluación de los atributos de un gen y la comparación con los atributos del otro gen devuelve un valor numérico de la compatibilidad de la solución, mientras más alto es el número más compatible son los genes y más óptimo el cromosoma.

La función está determinada por la diferencia entre el tamaño del aula y el número de estudiantes máxima permitida por la materia, mientras mayor sea la diferencia menor es el puntaje, otorgando la mayor puntuación a las aulas que no tengan diferencia con la cantidad máxima de alumnos de la materia, si se supera esta capacidad del aula, se castiga quitando puntos.

En segundo punto se evalúa los equipamientos necesarios y requeridos, si el aula no requiere ningún equipamiento, se otorga un valor positivo, si el aula contiene el equipamiento que requiere la materia, la puntuación es el doble de la anterior y si el aula no contiene los equipamientos requeridos se castiga quitando puntos. El tercer punto a evaluar está dado por el turno, si el turno de la materia es igual al turno del aula, se otorga un valor positivo, en caso contrario se penaliza con el mismo valor, pero negativo.

El último punto a evaluar es si el curso requiere estar en un aula laboratorio, si comparten característica se puntúa positivamente, es decir si el aula es laboratorio y si la materia requiere que sea laboratorio o si el aula no es laboratorio y la materia no requiere que sea laboratorio, en caso contrario, se penaliza con la quita de puntos.

Operadores Genéticos:

Selección: La selección, al igual que en la naturaleza hace referencia a la elección de dos cromosomas con la mejor aptitud para reproducirlos en búsqueda de un cromosoma mejorado. Se utiliza el método de clasificación donde se eligen al azar dos cromosomas con una aptitud alta para reproducirlos.

Cruce (Crossover): Se define el método por el cual los cromosomas seleccionados se "reproducen" generando un nuevo cromosoma. Al igual que en la naturaleza, hay un porcentaje de posibilidad de cruzamiento y es definido con un 50% de posibilidades. Si este se produce, lo hará de forma de cruce de punto, donde se cruzarán un gen diferente de cada cromosoma seleccionado .

Mutación: La mutación introduce variabilidad en los cromosomas al cambiar aleatoriamente algunos de sus genes. Con una probabilidad muy baja, se produce el cambio de genes en los cromosomas con un resultado de aptitud bajo.

Proceso de Evolución e Integración con la Asignación de Aulas:

Inicialización: Se genera la población aleatoriamente con los listados de aulas y materias, creando los cromosomas como pares de aula-materia.

Evaluación de Aptitud: Utiliza la función de aptitud para evaluar cada cromosoma de la población.

Selección: Selecciona cromosomas para la reproducción en función de su aptitud.

Cruce y Mutación: Aplica operadores de cruce y mutación para crear una nueva generación de cromosomas.

Evaluación de Aptitud: Evalúa la aptitud de los nuevos cromosomas.

Evolución: Selecciona cromosomas para la próxima generación, utilizando el elitismo como estrategia para preservar los mejores cromosomas.

Iteración: Repite los pasos a partir de la selección durante un número especificado de generaciones (en nuestro caso 1000).

Resultado: El listado de los cromosomas con la mejor aptitud en la última generación representa la solución óptima para la asignación de aulas de la universidad

```
private fun geneticAlgorithm(
    rooms: List<Room>,
    courses: List<Course>,
    students_count: Int,
    generations: Int
): List<Pair<Course, Room>> {
    val population = Population(rooms, courses, students_count)
    population.generateIndividuals()
    var bestFitness = 0.0
    var bestIndividual: List<Pair<Course, Room>> = mutableListOf()
    var listTime: MutableList<ExecutionTime> = mutableListOf()
    repeat(generations) {
        val startTime = Instant.now()
        for (individual in population.individuals) {
            val fitness = calculateFitness(individual)
            if (individual.size >= bestIndividual.size){
            if (fitness > bestFitness) {
                bestFitness = fitness
                    bestIndividual = individual.distinctBy { it.first }.distinctBy { it.second }
                }
            }
        }
        val endTime = Instant.now()
        val executionTime = Duration.between(startTime, endTime).toMillis()
        listTime.add(ExecutionTime(endTime,executionTime))
        evolution(population)
    }
    val gaTime = GATime( id: null,listTime)
    gaTiemRepository.save(gaTime)
    return bestIndividual
}
```

Imagen 5. Etapa final del desarrollo del algoritmo genético

Para una revisión más abarcadora y comprensiva, se sugiere dirigirse a la Imagen 1, la cual proporciona una visión más global y completa del análisis en cuestión.

3.6 Integración de Telegram Bot

La integración de un bot de la plataforma de telegram como interfaz de usuario mejora la experiencia de los alumnos en las consultas sobre las aulas en la que se inscribieron. Con solo enviar el número de alumno a Materias_Alumnos_Unaj (@materiasAlumnosUnajBot) se analiza la existencia del número de este y se procesa los datos de las materias a las cuales se encuentra inscripto. Se compara las materias obtenidas con el resultado del último algoritmo genético guardado en la base de datos para obtener las aulas donde se cursa la materia y se envía la respuesta al alumno.

Para implementar un bot de telegram hay crearlo y registrarlo mediante un token de acceso para ello lo primero que se debe hacer es ingresar a telegram y buscar el bot "BotFather". Se inicia una nueva conversación y se envía el comando "/newbot". Este te guiará en la creación del bot donde hay que configurar el nombre del bot y el usuario.

Una vez configurado el bot, enviará información sobre el token de acceso.

En la API la integración consta de varios pasos. El primero y principal es registrar el bot configurando el usuario y el token de acceso.

Después se debe procesar las respuestas dependiendo de los comandos que utilices, en este caso, se utiliza el comando vacío (por defecto) donde el alumno debe enviar el número de alumno.

Para que el alumno pueda realizar consultas en la base de datos debe primero registrarse en el bot, para ello se tiene que suministrar los datos que el bot le solicitará, los

cuales se irán validando de acuerdo a los datos proporcionados en la inscripción a la universidad. Una vez realizado el registro se podrá consultar el estado de las materias mediante el uso del comando "/materias" enviando el número de alumno para solicitar la información.

La información proporcionada se procesa en la base de datos, obteniendo el listado de materias a las cuales está inscripto el alumno. Tomando en cuenta el último resultado guardado del algoritmo genético, se busca cada materia y se extrae los datos del aula, que será procesado y enviado al alumno como respuesta del bot.

Ejemplos de la consulta:

Registro

Hay que agregar alguna palabra para que el bot te guie con los comandos necesarios para el uso. Siempre se comienza con /iniciar, si se le pasa otro dato, el bot envía un mensaje de error y cuál es el comando correspondiente para comenzar (ver Imagen 6).

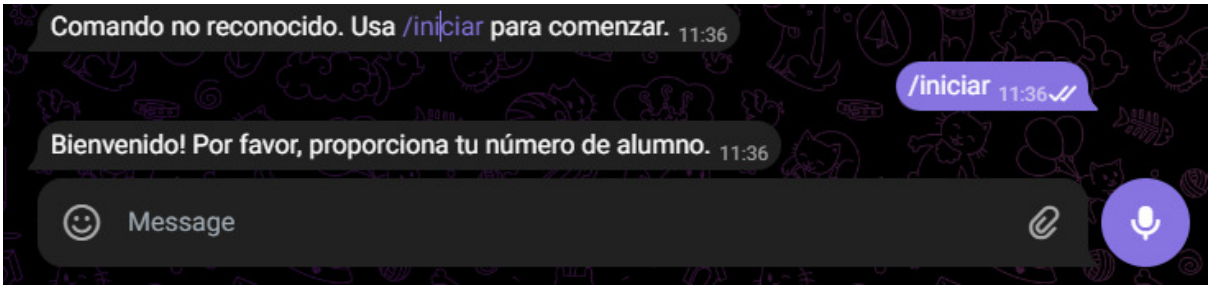


Imagen 6. Inicio del bot de Telegram

Iniciado el proceso, lo primero que se solicita es el número de alumno para verificar el estado del registro, si el registro está completo el alumno ya puede comenzar a operar, sin embargo, si no está registrado se deben proporcionar más datos que serán solicitados y verificados en cada paso (ver Imagen 7).

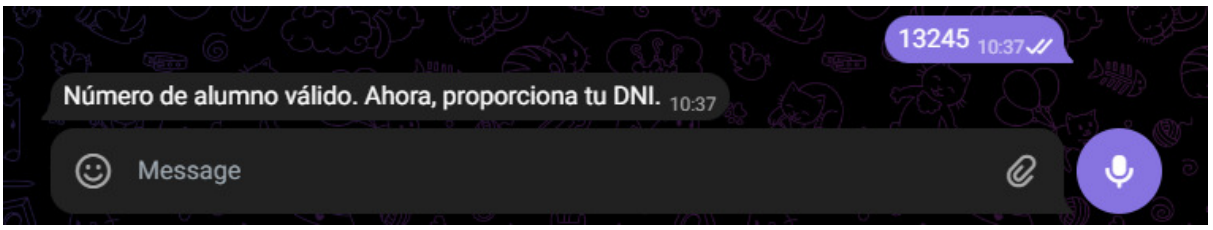


Imagen 7. Validación dni del bot de Telegram

Se valida que el número de alumno es correcto, pero no está registrado, para continuar con el registro se solicitan más datos, en este caso es el DNI (ver Imagen 8).

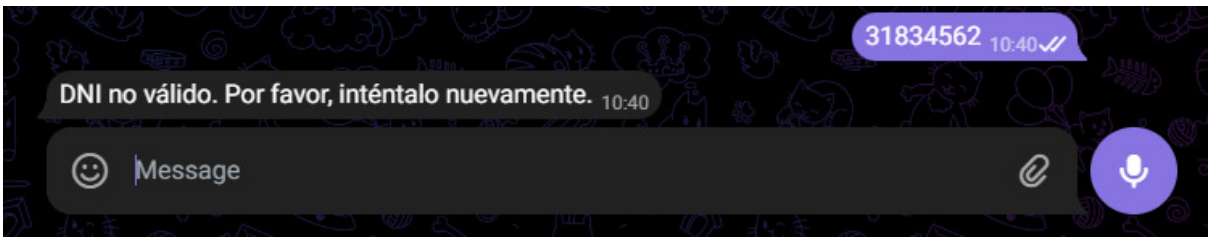


Imagen 8. Dni no válido del bot de Telegram

Validamos el DNI y vemos que no es correcto, se volverá a solicitar el número de DNI (ver Imagen 9).

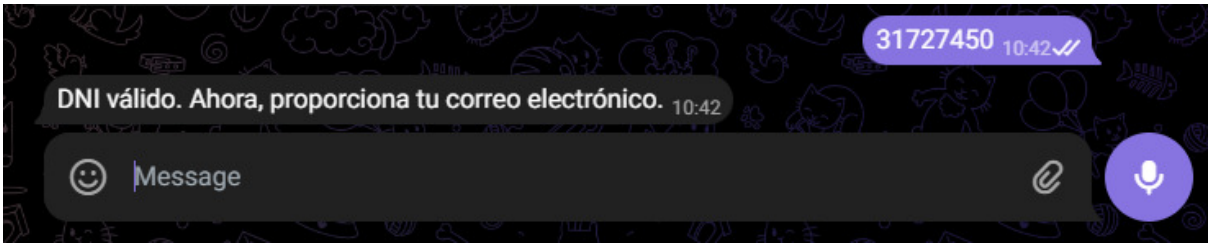


Imagen 9. Validación del dni correcta del bot de Telegram

Proporcionamos el DNI correcto, el cual está validado y aceptado y nos pide el último dato que es el mail (ver Imagen 10).

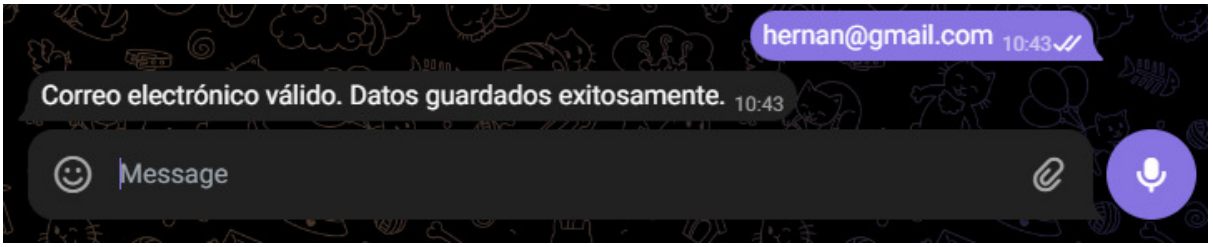


Imagen 10. Guardado de dato correcto del bot de Telegram

En caso de enviar un dato incorrecto, se dará el aviso y lo solicitará de nuevo. En este caso el mail fue validado correctamente y el usuario fue dado de alta en el registro.

Imagen 11. Flag de base de datos modificada

Vemos cómo se actualizó el dato en la base y a partir de ese momento el usuario ya se encuentra registrado y en condiciones de usar las otras opciones (ver Imagen 11).

Para salir del registro tenemos que volver a iniciar con /iniciar con lo cual inicia el proceso y solicita el número de alumno (ver Imagen 12).

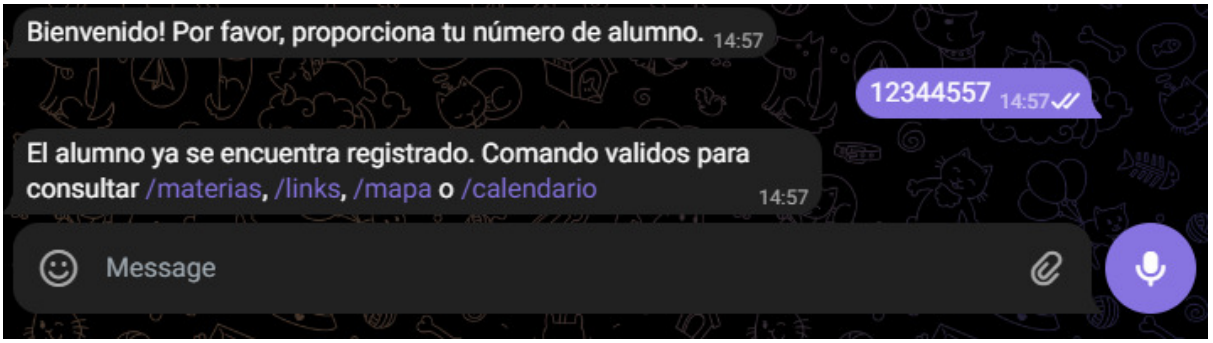


Imagen 12. Listado de opciones del bot de Telegram

Valida que el usuario está registrado y sugiere el uso de una de las opciones, "/materias" que tiene como finalidad ver las aulas donde se debe cursar las materias a las que

se anotó para cursar. Para el correcto funcionamiento de este comando se tuvo que generar el archivo con la solución de la ejecución del algoritmo genético. No es necesario que el alumno esté anotado en materias dado que la respuesta será que el alumno no se encuentra registrado en ninguna materia, en cambio sí está anotado en 1 o más materias se dará la información del aula en que le toca cursar la materia correspondiente (ver Imagen 13).

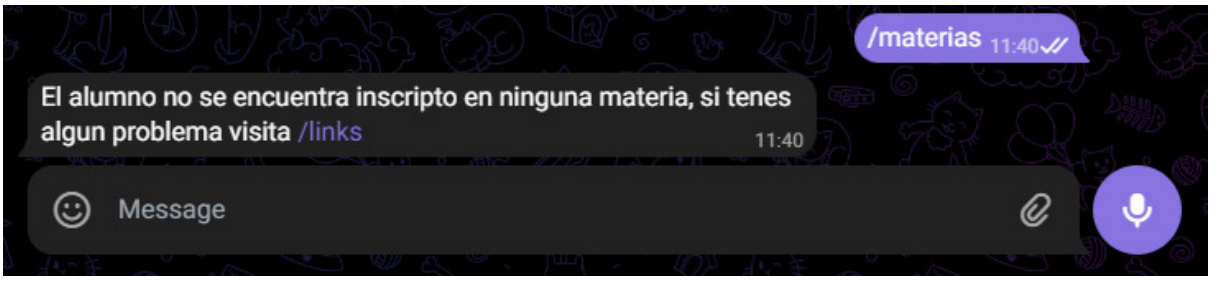


Imagen 13. Alumno sin materias inscriptas del bot de Telegram

También sugiere el uso de /links para obtener las direcciones de las páginas oficiales de la universidad (ver Imagen 14).

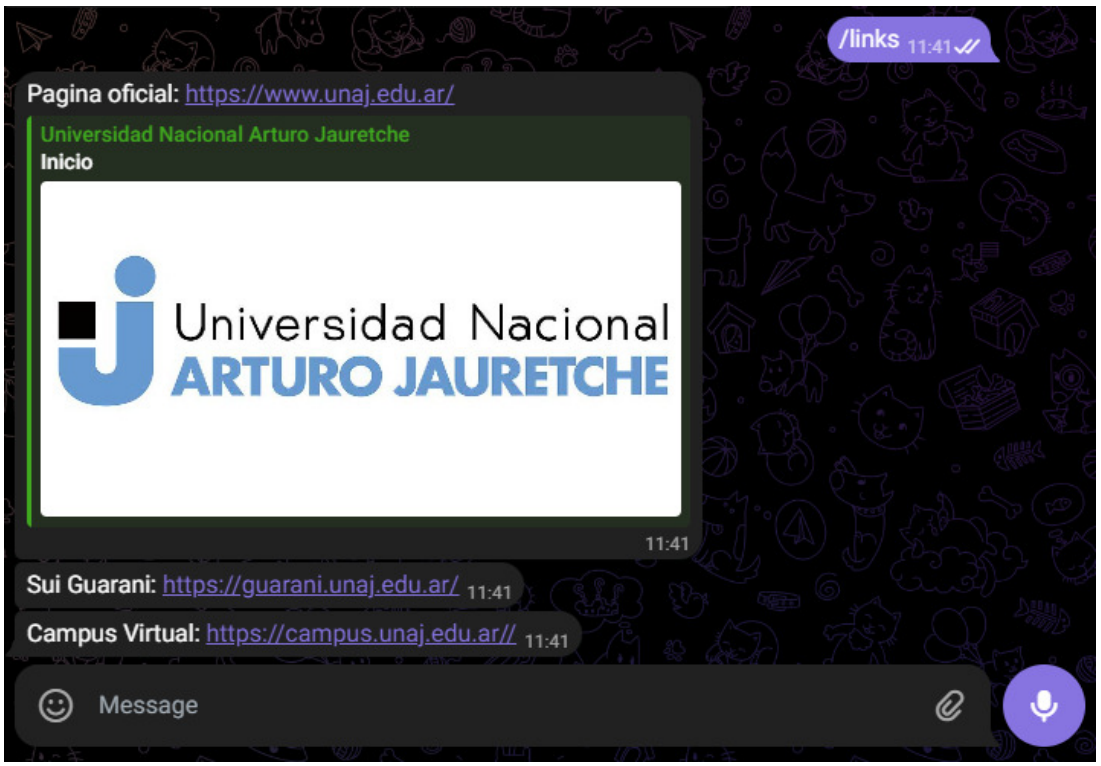


Imagen 14. Redes oficiales del bot de Telegram

En caso que el alumno se encuentre anotado a alguna materia la respuesta del bot es (ver Imagen 15).

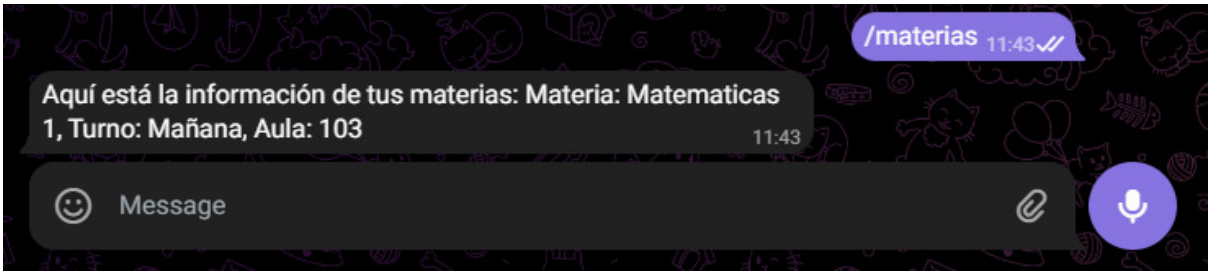


Imagen 15. Listado materias del bot de Telegram

3.7 Frontend y validación de datos

Para poder generar un archivo o consultar el último archivo guardado, se debe iniciar sesión al igual que en otra aplicación oficial de la universidad, se solicita número de DNI y contraseña (ver Imagen 16), si los datos son correctos se genera un token de autorización para el uso de la herramienta a través de JWT. Cada solicitud enviada desde el frontend al backend, sea consultas del resultado guardado, ejecución del algoritmo genético, agregado manual de materias en los alumnos, o cualquier posibilidad en los endpoints deberá viajar con un token válido, sino no se podrá completar la acción y devolverá un HTTP status 401(***UNAUTHORIZED***).

Imagen 16. Inicio de sesión en el front

Una vez iniciada la sesión, ya se puede ingresar a la página principal de la aplicación donde se encuentran opciones para realizar las operaciones (ver Imagen 17).

Imagen 17. Página principal

Para acceder al detalle de los tiempos de desarrollo ver en la sección Anexo, Anexo 1.

Se facilita el acceso directo a la especificación completa en formato JSON a través de swagger, lo que permite un análisis más profundo y detallado de la funcionalidad de la API. Esta estructura organizativa y documentación exhaustiva aportan una mayor claridad y transparencia al desarrollo, facilitando a los desarrolladores y usuarios finales la comprensión y utilización efectiva de la API (ver Imagen 18).

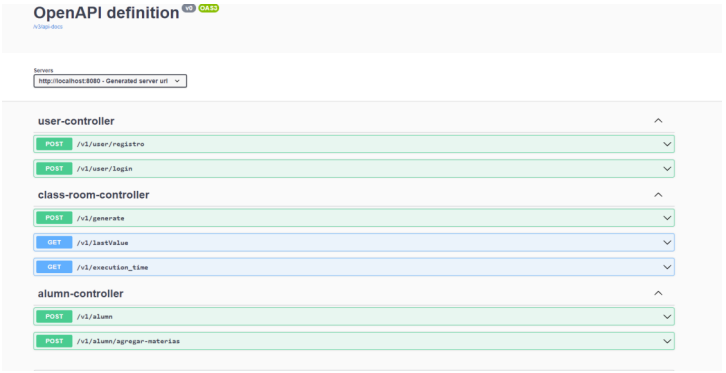


Imagen 18. Página de swagger con los endpoints disponibles

4. Conclusiones

En este trabajo se ha desarrollado una API REST para la optimización de aulas en la universidad mediante el uso de algoritmos genéticos. La finalidad de este desarrollo fue ofrecer una solución que permita a la universidad asignar de forma óptima los recursos disponibles a las demandas de los planes de estudio. Para ello, se ha utilizado una metodología de desarrollo ágil.

La aplicación está optimizada para la Unaj, pero con algunos cambios mínimos sobre los datos se puede integrar en otros centros universitarios.

La aplicación toma la información de las aulas, de las materias y de los alumnos desde la base de datos, y devuelve como salida una propuesta de asignación óptima que maximiza el aprovechamiento de los recursos y minimiza los conflictos. La API utiliza algoritmos genéticos para resolver el problema de optimización, ya que son capaces de explorar el espacio de soluciones de forma eficiente y adaptativa.

Los resultados obtenidos han sido evaluados mediante un conjunto de pruebas que han demostrado su correcto funcionamiento y su capacidad para generar soluciones óptimas o cercanas al óptimo. Se ha comprobado que la API es capaz de resolver problemas de diferente tamaño y complejidad, y que los algoritmos genéticos ofrecen un buen rendimiento y una buena calidad de las soluciones.

La integración con telegram permite que los alumnos tengan una amigable experiencia intercambiando datos con el bot y así realizar consultas más fácilmente y tener la respuesta en un click. La validación de los datos permite que los alumnos debían enviar datos correctos otorgando seguridad en cada consulta.

La implementación de un sistema de inicio de sesión y la utilización de tokens de acceso desempeñan un papel fundamental al restringir posibles usos indebidos de la

herramienta. Al establecer este mecanismo de autenticación, se contribuye significativamente a mitigar potenciales amenazas y a salvaguardar la integridad y confidencialidad de los datos sensibles asociados con la herramienta. En esencia, esta medida de seguridad no solo fortalece la protección contra posibles riesgos, sino que también añade un nivel adicional de control y supervisión sobre el acceso y uso de la herramienta, asegurando así un manejo seguro y responsable de la información confidencial.

La totalidad de los datos empleados en el sistema se extraen de las bases de datos, siendo esenciales para el correcto funcionamiento del mismo. Estos datos incluyen información crítica sobre aulas, alumnos y materias, constituyendo elementos fundamentales para la operatividad integral del sistema. Se diseñó con características básicas que simplifican la integración, permitiendo realizar ajustes mínimos para adaptarse eficientemente a las particularidades de cualquier establecimiento educativo. Este enfoque garantiza no solo la flexibilidad del sistema, sino también su capacidad para ajustarse a las necesidades específicas de diversos contextos educativos con cambios mínimos y una fácil adaptabilidad.

Para la ejecución del algoritmo genético se considera la cantidad de alumnos inscritos y activos en la universidad, excluyendo los casos con características especiales como los alumnos que no cumplen con la mínima cantidad de materias aprobadas requeridas en el año lectivo o los que fueron dados de baja por finalización de estudios.

La utilización de bases de datos es esencial para asegurar que la información sobre materias, aulas, alumnos y usuarios finales de la universidad esté actualizada, permitiendo así la extracción de datos más recientes y precisos. Eso hace que los cambios, las bajas de materias o aulas o en alumnos no generen ningún inconveniente en la ejecución del algoritmo.

Quedó fuera del alcance la posibilidad de integrar consultas a Apis externas para traer los datos necesarios, así como el análisis de los horarios particulares de las materias quedando todo sujeto a solo a los turnos.

Si bien se guardan los datos de los tiempos de ejecución en cada una de las generaciones en la base de datos, el análisis estadístico quedó fuera del alcance del proyecto.

La introducción del bot de Telegram para ciertos alumnos ha generado consecuencias sumamente positivas que tienen un impacto notable en las perspectivas futuras de investigación y desarrollo. La apertura de este canal de comunicación ha suscitado un marcado interés entre los estudiantes, quienes han expresado su solicitud de obtener información adicional sobre temas específicos. Además, se ha planteado de manera destacada la posibilidad de oficializar el uso del bot como un canal de comunicación institucional. La eficiencia en la velocidad de respuesta a las consultas ha desempeñado un papel fundamental en la experiencia de los estudiantes, contribuyendo significativamente a su comodidad con la implementación. Esta prontitud en las respuestas no solo ha mejorado la experiencia del usuario, sino que también ha sido determinante para cultivar la confianza en el sistema, subrayando la importancia del registro que garantiza seguridad a las consultas realizadas. En conjunto, estos aspectos positivos respaldan de manera contundente la efectividad y la recepción favorable del bot de Telegram como una herramienta valiosa para la interacción y la comunicación dentro del entorno educativo.

Se realiza un minucioso registro de los endpoints disponibles en la interfaz Swagger, accesible a través del enlace proporcionado: "link" (swagger-ui/index.html#/). Este recurso se convierte en una herramienta fundamental para la comprensión integral de la API, ya que ofrece una documentación exhaustiva y detallada de cada endpoint disponible. Al explorar este enlace, los usuarios pueden sumergirse en una interfaz interactiva que no solo presenta

una visión general de la API, sino que también desglosa información esencial sobre cada punto final, incluyendo descripciones, parámetros y ejemplos de uso.

Dado que el software no es estático, es posible que en el futuro se realicen inclusiones y modificaciones en algunos desarrollos existentes, los cuales se habían limitado con el fin de facilitar la formulación de la presente propuesta de desarrollo y solución a la problemática abordada.

5. Glosario

Algoritmo: Un conjunto de instrucciones o reglas que una computadora sigue para realizar una tarea específica.

API (Interfaz de Programación de Aplicaciones): Un conjunto de reglas que permite que dos aplicaciones se comuniquen entre sí. Facilita la integración de diferentes sistemas.

Autenticación: Proceso de verificar la identidad de un usuario antes de permitirle el acceso a un sistema.

Backend: Parte de una aplicación que gestiona la lógica de negocio, la base de datos y la interacción con el servidor.

Base de datos: Un lugar donde se almacena información de manera organizada y accesible.

BJSON (Binary JSON): Un formato de datos similar a JSON pero en una representación binaria.

Endpoint: Un punto final o URL específica en una API que se puede utilizar para realizar operaciones específicas.

Frameworks: Un conjunto de herramientas y reglas que facilitan el desarrollo de aplicaciones.

Frontend: Parte de una aplicación con la que interactúa el usuario, incluyendo la interfaz gráfica y la experiencia de usuario.

HMAC (Código de Autenticación de Mensajes con Función Hash): Un método de autenticación que utiliza una función hash para verificar la integridad y autenticidad de los datos.

HTTP (Protocolo de Transferencia de Hipertexto): Un protocolo utilizado para la transmisión de datos en la web.

JSON (Notación de Objetos de JavaScript): Un formato de intercambio de datos fácil de leer y escribir para humanos, y fácil de analizar y generar para las máquinas.

RSA: Un algoritmo de cifrado utilizado para garantizar la seguridad en la transmisión de datos.

SHA256: Una función hash criptográfica que produce un valor hash de 256 bits.

Token: Un identificador único que se utiliza para autenticar a un usuario.

XML (Lenguaje de Marcado Extensible): Un formato de texto que permite estructurar datos de manera legible tanto para humanos como para máquinas.

6. Bibliografía

Holland, J. H. (1975). Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. U Michigan Press.

Telegram(Abril 18, 2023).Telegram Apis, https://core.telegram.org/

MongoBD(Oct 16, 2018), Mongo Datebase, https://www.mongodb.com/es

JSON Web Tokens(Jun 11, 2020), https://jwt.io/

Naupari, R. E., & Rosales, G. K. (2010). Aplicación de algoritmos genéticos para el diseño de un sistema de apoyo a la generación de horarios de clases para la Facultad de Ingeniería de Sistemas e Informática de la UNMSM. E.A.P. de Ingeniería de Sistemas, 52.

Sistema de apoyo a la generación de horarios basado en algoritmos genéticos, Augusto Cortez Vásquez1,2, Gissela Rosales Gerónimo1 , Raúl Naupari Quiroz1 ,Hugo Vega Huerta Z, Universidad Nacional Mayor de San Marcos, ISSN 1816-3823, https://sisbib.unmsm.edu.pe/bibvirtual/Publicaciones/risi/2010_n1/v7n1/a05v7n1.pdf , Enero 2012

Aplicación de Algoritmos Genéticos al Problema de Asignación de Aulas para Exámenes en un Centro Universitario, Ing. José Cidrás Pidre, Ing. Eloy Díaz Dorado, Ing. Antonio García Lorenzo, http://www.adingor.es/congresos/web/uploads/cio/cio2002//metodos_cuantitativos/C089.pdf , II Conferencia de Ingeniería de Organización Vigo, 5-6 Septiembre 2002

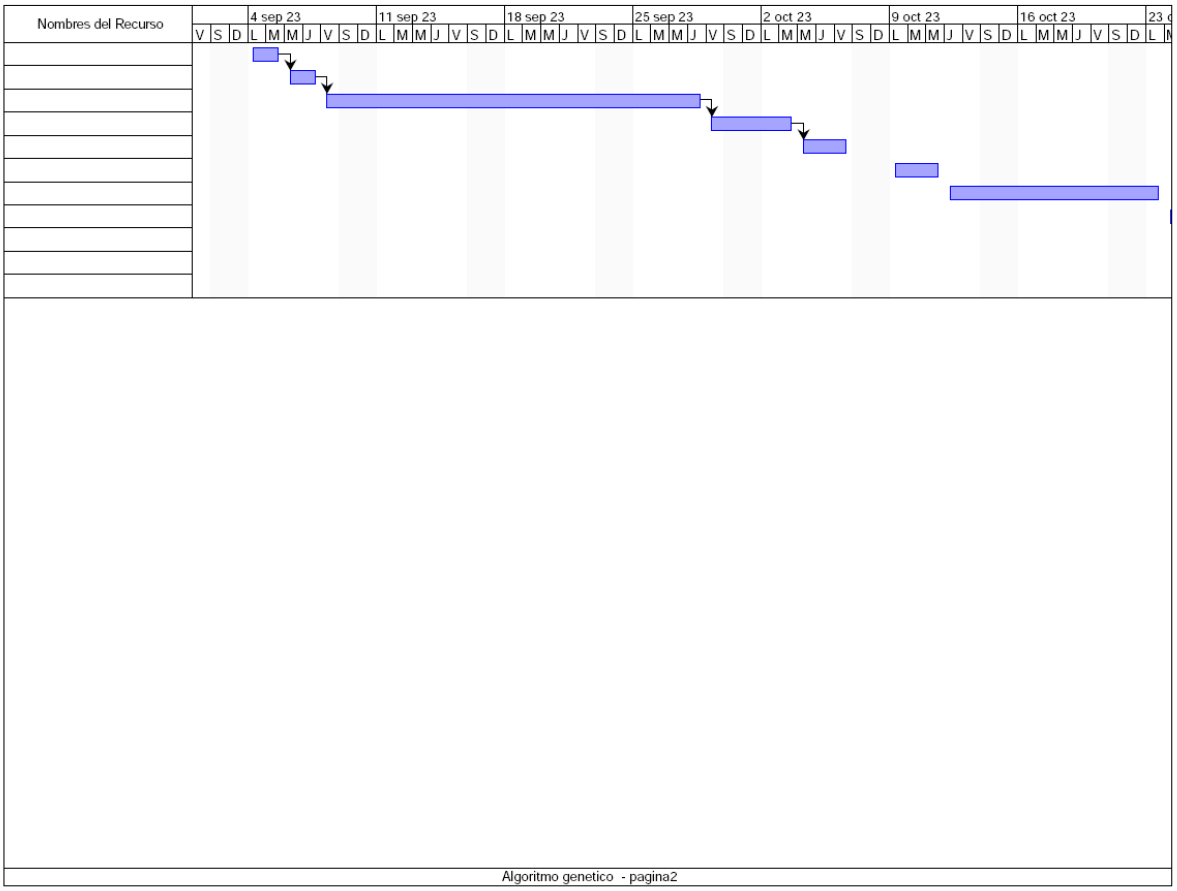
https://datatracker.ietf.org/doc/html/rfc7519, Sakimura N., Mayo 2015

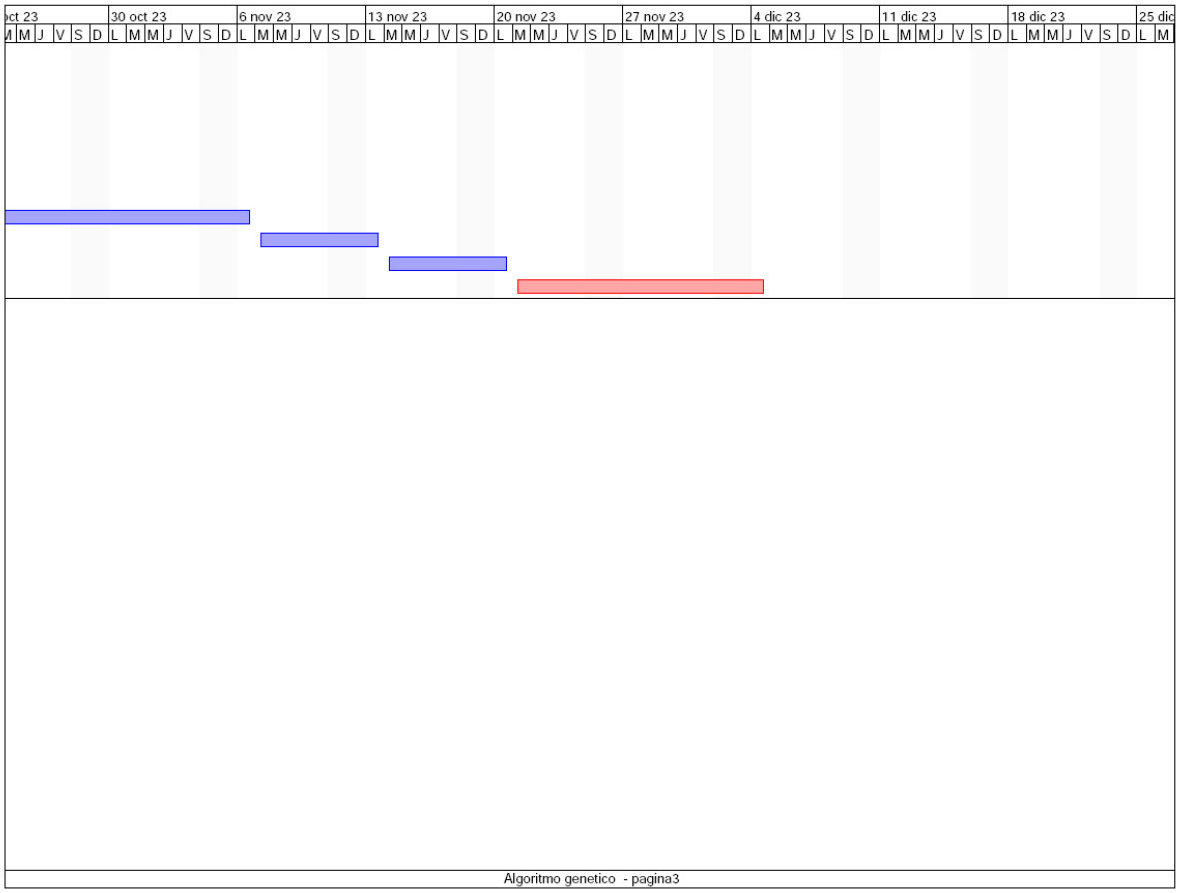
7. Anexo 1

El diagrama de Gantt constituye una representación gráfica que detalla y organiza la planificación y programación del desarrollo de la aplicación. A través de este diagrama, se logra visualizar de manera estructurada y cronológica las diversas fases, tareas e hitos asociados con el proceso de creación de la aplicación.

		Nombre	Duracion	Inicio	Terminado	Predecesores
1		Diseño arquitectura	2 days?	04/09/23 08:00	05/09/23 17:00	
2		Analisis de herramientas de desarrollo	2 days?	06/09/23 08:00	07/09/23 17:00	1
3		Desarrollo del algoritmo genetico	15 days?	08/09/23 08:00	28/09/23 17:00	2
4		Desarrollo de servicio	3 days?	29/09/23 08:00	03/10/23 17:00	3
5		Desarrollo controladores	3 days?	04/10/23 08:00	06/10/23 17:00	4
6		Configuracion de base de datos	3 days?	09/10/23 08:00	11/10/23 17:00	
7		Modulo de alumnos	8 days?	12/10/23 08:00	23/10/23 17:00	
8		Integracion de telegram bot	10 days?	24/10/23 08:00	06/11/23 17:00	
9		Desarrollo de la pagina de login	5 days?	07/11/23 08:00	13/11/23 17:00	
10		Desarrollo de la pagina principal	5 days?	14/11/23 08:00	20/11/23 17:00	
11		Test y puesta a punto	10 days?	21/11/23 08:00	04/12/23 17:00	

Algoritmo genetico - pagina1


Nombres del Recurso
4 sep 23
11 sep 23
18 sep 23
25 sep 23
2 oct 23
9 oct 23
16 oct 23
V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M
Algoritmo genetico - pagina2


ct 23
30 oct 23
6 nov 23
13 nov 23
20 nov 23
27 nov 23
4 dic 23
11 dic 23
18 dic 23
25 dic
Algoritmo genetico - pagina3