**Universidad De Guadalajara**

Centro Universitario De Ciencias Exactas E Ingenierías

División De Tecnologías Para La Integración Ciber-Humana

Departamento de Ciencias Computacionales

Proyecto final

**Reporte.**



**Estudiantes:**

Arellano Cervantes Guillermo Antonio (Tester)

Campos Ornelas Oscar (Dev. Backend)

Maldonado Melendez Diego Alberto (Administrador)

Sanchez Sanchez Jaime Ivan (Dev. Frontend)

**Materia:** Análisis de algoritmos.

**Sección:** D06.

**Profesor:** Lopez Arce Delgado Jorge Ernesto

**Fecha de entrega:** 30 de noviembre de 2023

**Índice**

[**1.** **Resumen.** 3](#_Toc152096729)

[**2.** **Descripción del proyecto.** 3](#_Toc152096730)

[**2.1** **Planteamiento del problema.** 3](#_Toc152096731)

[**2.2** **Diseño de la aplicación.** 4](#_Toc152096732)

[**2.3** **Implementación de apartados técnicos.** 4](#_Toc152096733)

[**3.** **Planificación y metodología.** 5](#_Toc152096734)

[**4.** **Roles y objetivos.** 6](#_Toc152096735)

[**4.1** **Definición de los roles:** 6](#_Toc152096736)

[**4.2** **Objetivos y entregables:** 6](#_Toc152096737)

[**4.3** **Reportes semanales:** 8](#_Toc152096738)

[**4.3.1** **Semana 1:** 8](#_Toc152096739)

[**4.3.2** **Semana 2:** 9](#_Toc152096740)

[**4.3.3** **Semana 3:** 10](#_Toc152096741)

[**5.** **Desafíos y soluciones.** 10](#_Toc152096742)

[**6.** **Desarrollo conexión.** 12](#_Toc152096743)

[**7.** **Resultado.** 15](#_Toc152096744)

[**8.** **Conclusión.** 15](#_Toc152096745)

[**9.** **Referencias.** 16](#_Toc152096746)

1. **Resumen.**

Este proyecto representa el producto integrador del curso de Análisis de Algoritmos, abarcando la planificación, diseño y desarrollo de un software capaz de organizar los horarios de una institución educativa. Nuestro equipo asumió la responsabilidad de establecer los parámetros de entrada, definir su formato y delinear las salidas esperadas. El objetivo primordial fue la creación de un programa que genere calendarios para docentes, aulas y clases, maximizando la eficiencia en la asignación de recursos y respetando las restricciones predefinidas.

Nuestra estrategia de planificación se fundamentó en un administrador de edificios específico para el edificio 'X'. Este administrador, al introducir datos en un archivo ".csv" que detallara las materias disponibles junto con sus horas semanales, sería capaz de generar, en cualquier momento, una distribución de horario. Para mayor comodidad, se permitiría al administrador asignar profesores después de la generación de horarios, con la opción adicional de editar las materias ya establecidas.

El proyecto se enfocó en replicar un entorno laboral realista, imitando los desafíos propios de trabajar en una empresa o proyecto. Como resultado, el equipo se organizó en roles definidos con precisión: un desarrollador backend, un desarrollador frontend, un tester y el administrador del proyecto, quien se encargó de la gestión de recursos humanos.

1. **Descripción del proyecto.**
   1. **Planteamiento del problema.**

El administrador del edificio ‘X’ requiere de un software capaz de ahorrarle gran parte de su trabajo. Este software debe ser capaz de leer todas las materias de las carreras de Ingeniería en Computación e Informática, que suman en total aproximadamente 60 cursos. Estos cursos tienen el factor adicional de las horas semanales que deben durar, el profesor que impartirá las clases y el día y la hora de la semana cuando se llevarán a cabo. Tomando en cuenta todos estos factores, el programa debe generar una distribución de horarios que contemple todas las materias de las carreras mencionadas, además de las restricciones establecidas.

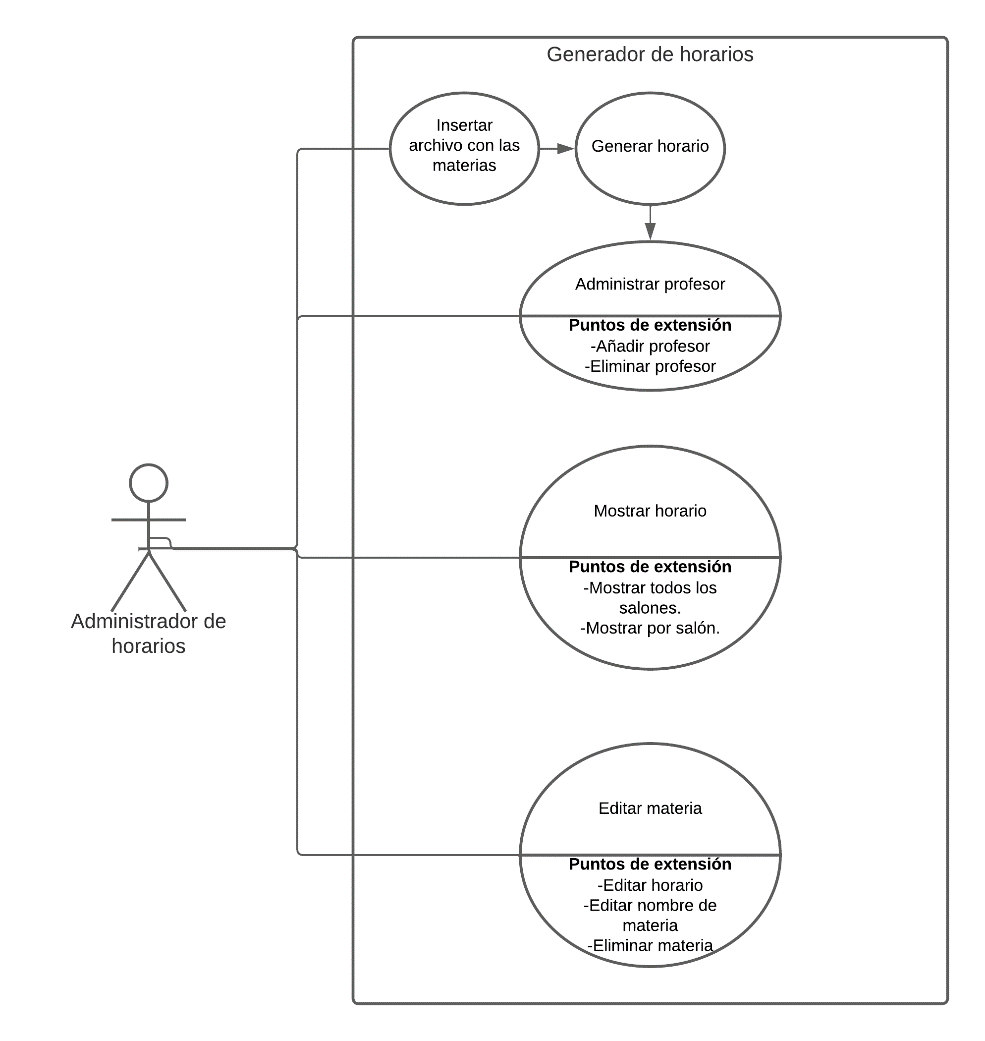
Como equipo, establecimos ciertos criterios para lograr la resolución de la actividad. En primera instancia, delimitamos la duración de las clases entre 2 y 4 horas semanales, con la intención de tener un mejor control sobre nuestro conjunto de información. Además, requerimos un formato para la importación de la información. Para esto, quisimos utilizar una herramienta con la que pudiese estar familiarizada un administrador de edificio que no tuviese relación alguna con la programación, es por ello por lo que consideramos que el formato csv, que va de la mano de Excel, sería la mejor opción.

Para la asignación de profesores quisimos hacerlo de la manera más eficiente posible. Tomando en cuenta que la cantidad de salones, horas, y clases no es tan grande, otorgamos al administrador la capacidad de asignar uno por uno los maestros que desee, siendo una excelente opción ya que se podría mostrar el horario generado a los profesores que impartirán dichas materias y que ellos escojan las horas que más se le facilitan, ya que estar expectante de los horarios de cada uno podría hacer que la generación de los calendarios sea más compleja.

* 1. **Diseño de la aplicación.**

Esta aplicación está pensada en su totalidad para el administrador de horarios de un edificio. Este necesita, en la primera columna de un archivo csv, plasmar el nombre de las materias a contemplar, y en la segunda columna la cantidad de horas que tienen sus clases a la semana. Posteriormente, el usuario debe ingresar al software, importar el archivo previamente creado y generar la distribución del calendario. Existirá la pestaña de consulta, la cual le permite al administrador obtener una vista general y otra vista por cada salón de los horarios generados. En caso de no estar conforme con la distribución proporcionada por la aplicación, el usuario puede generar horarios ilimitadamente.

La siguiente pestaña se encarga de añadir profesores. El usuario, indicando el salón y la clase donde desea realizar esta modificación, puede ingresar el nombre del profesor y de este modo ya estaría asignado a dicha clase. Y, por último, está la pestaña de edición de materias. En esta, el usuario tiene la capacidad de editar el nombre de la materia, el horario, y también puede eliminar cualquier materia que desee.



**Figura 1.** Diagrama de caso uso.

* 1. **Implementación de apartados técnicos.**

Uno de los aspectos más importantes para el desarrollo de este proyecto fue el modelado de archivos. En este caso, el desarrollador backend se encargó de esto. El código trabaja con 3 estructuras principales, que son las clases disponibles, los salones y los profesores. Luego de la generación, se crea una estructura de datos conjunta, la cual manejamos como diccionario, capaz de almacenar toda la información correspondiente.

Uno de los tipos de algoritmos utilizados es la búsqueda heurística. Este tipo de algoritmos hace referencia a una técnica que intenta hallar una solución aproximada de manera rápida, sobre todo cuando se enfrenta a problemas complejos. Para esto, se hace una clasificación de las alternativas de solución en cada paso de la bifurcación para así decantarse por la óptima. Es considerado como un atajo para encontrar la solución al problema.

En el caso de nuestro programa, en una de las partes de la función generadora de horarios se emplea una estrategia heurística al mezclar la lista de clases disponibles (clases\_disponibles\_hoy) para seleccionar aleatoriamente las clases que se asignarán a lo largo del día. Además, a la hora de la verificación de las clases disponibles, las funciones encargadas de obtener el horario y de determinar si hay algún espacio para la clase, utilizan una estrategia heurística para encontrar espacios libres en el horario y asignar clases de manera eficiente.

Por otro lado, también usamos un enfoque de asignación. Este tipo de algoritmos son utilizados para predecir cómo se comportarán elementos particulares en un conjunto, concretamente empleamos el enfoque de asignación de recursos, en este caso para los horarios, salones, días, entre otros elementos.

Se consideraron estos algoritmos como los óptimos debido a su simplicidad y eficiencia, sobre todo en tiempo computacional. Esto sale a relucir en nuestro programa, ya que, a pesar de trabajar con un conjunto de datos bastante grande, ofrece una solución bastante aceptable. Además, es un software muy adaptable para posibles cambios dinámicos, y se nota a la hora de utilizar el programa.

Para el apartado del frontend, se utilizó una herramienta enfocada en el diseño de GUI llamada PyQT5. Esta, en conjunto con su plugin Designer permitió generar las ventanas necesarias con facilidad. La conexión fue un apartado complejo. Para esto, el frontend le pide al usuario que ingrese un archivo con extensión csv. El programa copia y pega el archivo en la ruta donde yace el programa, para, a partir de allí, comenzar a trabajar. La codificación del backend fue integrada en la lógica del frontend, esto para evitar el uso de señales y otros elementos con los que no estábamos tan familiarizados.

Para la actualización y registro de todo lo realizado en el frontend requerimos trabajar con otro tipo de archivo, y por la simplicidad de este decidimos utilizar archivos de texto. Otro aspecto importante es que el modelado de archivo determinó que trabajaríamos con diccionarios, por lo que requerimos sobrescribir el archivo de texto con este formato. Para ello utilizamos el formato json, el cual nos facilita el trabajar con estos diccionarios y que no se pierda la información.

1. **Planificación y metodología.**

Debido a la falta de experiencia y de conocimientos que teníamos acerca de ciertos temas, necesitábamos una metodología de trabajo flexible, la cual contemplara posibles cambios e inconvenientes y nos permitiese afrontarlos. Por consiguiente, nos sentimos muy cómodos con la metodología Agile. Esta se basa en la colaboración cruzada de todos los integrantes del proyecto, así como la subdivisión del problema en tareas más pequeñas, un reduce y vencerás en términos de administración de proyectos.

La fecha clave fue el 9 de noviembre del presente año. Ese día nos reunimos durante 2 largas y tediosas horas los cuatro integrantes de este equipo. Estuvimos confundimos, estresados y con muchas dudas. Pero viéndolo hoy en retrospectiva, considero este día como el momento clave de este proyecto. Aquí, logramos estar todos los integrantes en un mismo canal, asignamos roles, objetivos y definimos cómo lograríamos los objetivos y cómo se vería el resultado final.

Para una correcta planificación decidimos utilizar la herramienta Monday, ya que era con la que más familiarizados estábamos los integrantes del equipo. Además, para almacenar el código y todo lo correspondiente con el apartado técnico, decidimos utilizar Github, iniciando así con el trabajo.

1. **Roles y objetivos.**
   1. **Definición de los roles:**

|  |  |
| --- | --- |
| Nombre completo | Rol |
| Arellano Cervantes Guillermo Antonio | Tester |
| Campos Ornelas Oscar | Backend |
| Maldonado Melendez Diego Alberto | Administrador |
| Sanchéz Sánchez Jaime Iván | Frontend |

* 1. **Objetivos y entregables:**

Este apartado presentó muchas modificaciones. La propuesta inicial era la siguiente:  

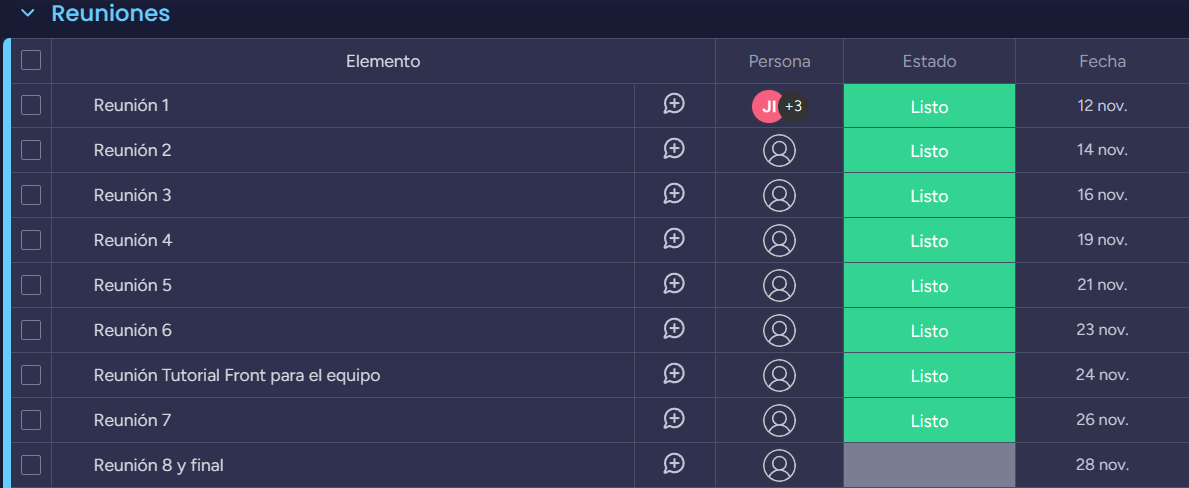

No obstante, gracias al enfoque Agile y los inconvenientes a los que nos fuimos enfrentamos, los objetivos cambiaron progresivamente, quedando de la siguiente manera:









Otro punto de suma importancia fueron las reuniones. La planificación fue la siguiente:  


Se anexan evidencias de la realización de estas reuniones:

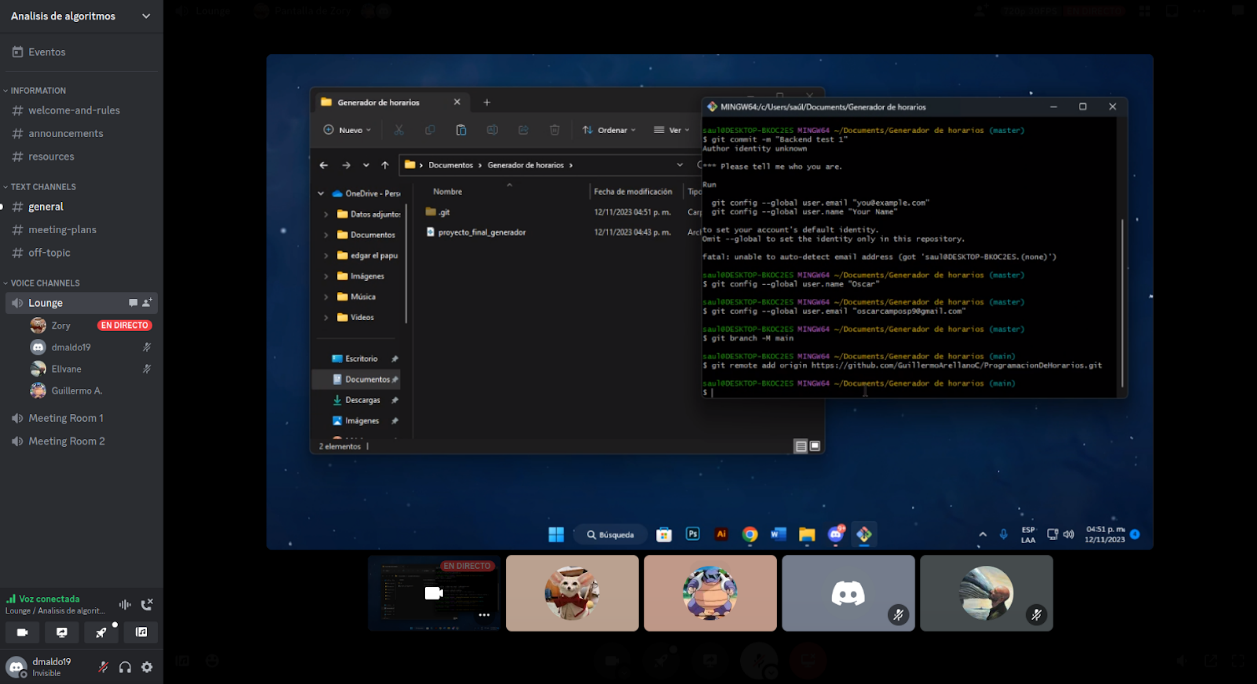


Figura . Evidencia reunión virtual

Figura . Evidencia reunión virtual

* 1. **Reportes semanales:**

La distribución de los trabajos se hizo con un enfoque semanal. Donde se pensó que la primera semana estuviese enfocada en el backend, la segunda en el frontend, y la última en la conexión de ambos apartados.

* + 1. **Semana 1:**

En esta semana, la mayoría de las tareas estaban enfocadas en el área del backend, ya que la visión era primero estructurar el funcionamiento del programa, para que, a raíz de allí, se pueda construir. No obstante, todas las áreas tuvieron tareas que cumplir, y todo se realizó de manera satisfactoria. Para desglosar de manera más comprensible, se presentarán las tareas desarrolladas por cada integrante.

**Developer backend:**

* Realizó el modelado de objetos necesarios para el desarrollo del proyecto, que en nuestro caso fueron “salones”, “cursos” y “profesores.
* Realizó un diagrama UML de las clases mencionadas en el punto anterior, con la finalidad de comprender la estructura de cada una de ellas.
* Indagó en los posibles algoritmos a realizar, para posteriormente implementar aquellos que le parecieron más sencillos, eficaces y capaces de proporcionar el resultado que esperamos.
* Creó un menú modularizado, mediante el cual, el usuario, pueda navegar cuantas veces sean necesarias a través del programa.

**Developer frontend:**

* El principal objetivo del desarrollador de front era determinar la herramienta que iba a utilizar para la creación, desarrollo y diseño de la interfaz gráfica necesaria.
* Posteriormente, realizó una serie de diseños, en forma de bosquejos, de tal forma que pudimos, en equipo, discutir cual enfoque nos parecía más acertado para la aplicación, y tener una guía para el momento de la codificación de la interfaz.
* Otro punto de suma importancia fue el establecimiento de los requisitos para la importación y exportación de datos, así como la manera en la que será la comunicación con entre el frontend y backend. Esto se realizó en equipo, pero profundizando en las opiniones e intereses del desarrollador de frontend.

**Tester:**

* Realizó un trabajo destacable, creando el repositorio de Github en el que íbamos a plasmar todo el proyecto, además de estudiar la información necesaria para que, en nuestra primera reunión virtual, pudiese darnos a todos un tutorial sobre el control de versiones con git y github.
* Creó el canal para la comunicación virtual del equipo, concretamente un canal de Discord.
* En conjunto con el desarrollador backend y el administrador, colaboró en la definición de la sintaxis y el formato para la importación de datos. Gracias a esta labor es que determinamos los archivos “.csv” como los que estaremos usando para el desarrollo del proyecto.
* Inició con los testing para el programa realizado por el desarrollador backend.
  + 1. **Semana 2:**

En esta semana, la mayoría de las tareas estaban enfocadas en el área del frontend, debido a que se consiguió un correcto funcionamiento del área del backend corroborado por el tester. Las tareas realizadas fueron:

**Developer backend:**

* Inició con la documentación de su apartado, indicando los algoritmos y enfoques utilizados, el modelado de datos, detallado de las funciones utilizadas, entre otros.
* Colaboró en la explicación técnica de lo logrado en el backend en favor de una correcta conexión entre frontend y backend.

**Developer frontend:**

* Diseñó las ventanas que serán necesarias para el funcionamiento del programa.
* Se trabajó en las funciones de las ventanas, que tiene más importancia para nosotros que el diseño.

**Tester:**

* Halló y corrigió pequeños fallos en las validaciones del backend.
* En conjunto con el administrador, se realizaron mejoras al código existente, con la intención de lograr el mejor producto posible.
  + 1. **Semana 3:**

En esta semana, la mayoría de las tareas estuvieron enfocadas en su totalidad en la conexión, que fue el mayor reto que tuvimos:

**Administrador:**

* Debido a un desafío que mencionaré más adelante, tuve que colaborar con mi equipo en el apartado de la conexión. Logré complementar la lógica establecida por mi desarrollador backend junto con el diseño creado con el desarrollador frontend.
* A lo largo de las semanas estuve encargado de la gestión, administración y colaboración de mis compañeros, por lo que tuve que plasmar todo esto en las presentaciones de la revisión de operaciones, además de la realización de la presentación y reporte final.

**Developer backend:**

* Realizó la documentación particular de su apartado.

**Developer frontend:**

* Modificó las ventanas pertinentes, de acuerdo con las necesidades que demandaba la lógica de la conexión.
* Creó el ejecutable del proyecto.
* Realizó la documentación particular de su apartado.

**Tester:**

* Realizó las pruebas correspondientes al producto final.
* Realizó la documentación particular de su apartado.

1. **Desafíos y soluciones.**

* **Confusión con el formato de importación de los datos:** Debido a nuestra experiencia, teníamos pensado trabajar con archivos .txt, no obstante, debido a la poca estructura que se puede tener, nos dimos a la tarea de investigar una alternativa que nos permitiese tener un mayor control sobre nuestra información.

**Soluciones aplicadas:** El tester se encargó de investigar acerca de los archivos “.csv”, su funcionamiento y su utilización en Python. Posteriormente, en conjunto con el desarrollador backend, se logró la optimización para la creación de la estructura necesaria para el desarrollador y la implementación de su código.

* **Repetición de horarios a la hora de ejecutar el backend:** Una vez era ejecutado el backend, no existía tanta distribución de los horarios, es decir, una clase de 4 horas era distribuida en su totalidad un mismo día, y ni siquiera se establecían de manera consecutiva, sino que podía ser una clase de 9 a 11, y la siguiente de 3 a 5.

**Soluciones aplicadas:** En un trabajo conjunto entre tester y administrador se observó que la sintaxis para el llenado de la lista que contemplaba los horarios estaba mal implementada. Por consiguiente, se corrigió este pequeño error técnico, dando lugar a un muy buen código realizado por el desarrollador backend.

* **Fallos hallados en el testing de backend:** Tras realizar las primeras pruebas por parte del tester se hallaron pequeñas fallas en el código. Son básicamente falta de validaciones y errores mínimos que requieren ser solucionados.

**Soluciones por aplicar:** El desarrollador backend evaluó y solucionó los errores proporcionados por el tester.

* **Descomposición de la computadora del desarrollador backend:** Debido a problemas con su computadora, no le fue posible al desarrollador backend ayudar en la complementación del código con la GUI.

**Soluciones por aplicar:** Como administrador, tuve la oportunidad de colaborar igualmente en la codificación del proyecto, concretamente en la conexión de la lógica y la interfaz

* **Desconocimiento de la conexión:** Una vez finalizado el diseño de las ventanas, no teníamos muy en claro cómo continuar.

**Soluciones por aplicar:** Se organizó una reunión de emergencia entre el desarrollador frontend y el administrador para determinar las soluciones a seguir.

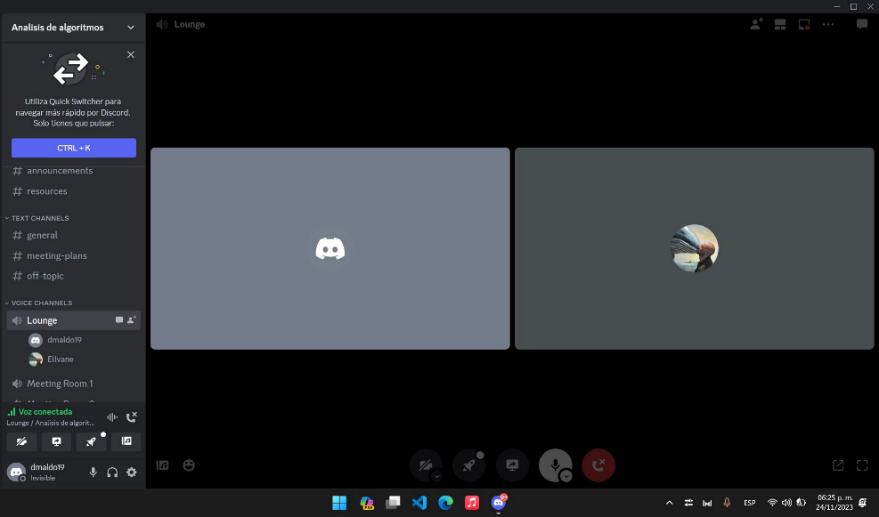


Figura . Reunión de emergencia

* **Resguardo de la información:** La aplicación funcionaba, pero no era dinámica, es decir, cualquier cosa que se hiciera no se guardaba para su posterior uso.

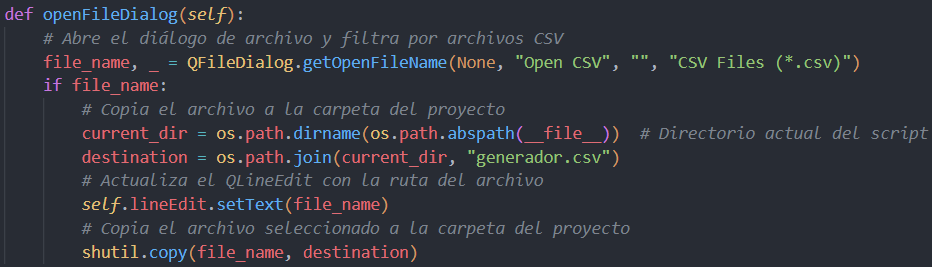
**Soluciones por aplicar:** Requerimos utilizar otro método para el resguardo de la información. Como encargado de la conexión, consideré que lo más importante de todo era el salvaguardar la estructura de datos utilizada, en este caso, el diccionario que alojaba los horarios generados. Por ende, lo que consideré óptimo fue utilizar un archivo de texto, y sobrescribir este archivo con la librería json, lo que permite que la información guardada permanezca en su formato original.

* **Corrección de una función:** La función de añadir materias fue replanteada, esto debido a que la consideramos redundante.

**Soluciones por aplicar:** Se mejoró la opción de editar materia, en donde decidimos dirigir al usuario a que cualquier modificación realizada se hiciese en base a la distribución generada.

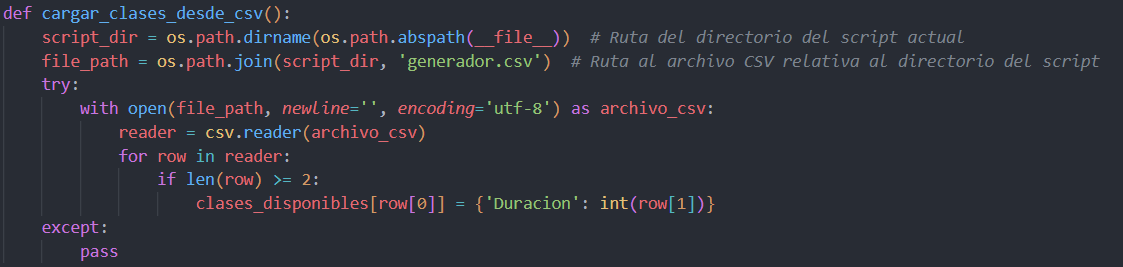
1. **Desarrollo conexión.**

Tuve la responsabilidad de implementar la conexión entre el backend y el frontend. Afortunadamente, mi desarrollador frontend se encargó de implementar una pestaña en la que el usuario debe importar el archivo csv, por lo que buena parte del trabajo estaba hecho. No obstante, debido a que había constante intercambio de datos, si requería existir una conexión más específica. Primeramente, fue necesario el crear una manera en la que el archivo que eligiese el usuario ingrese a la ruta del programa. Para ello, realicé lo siguiente:

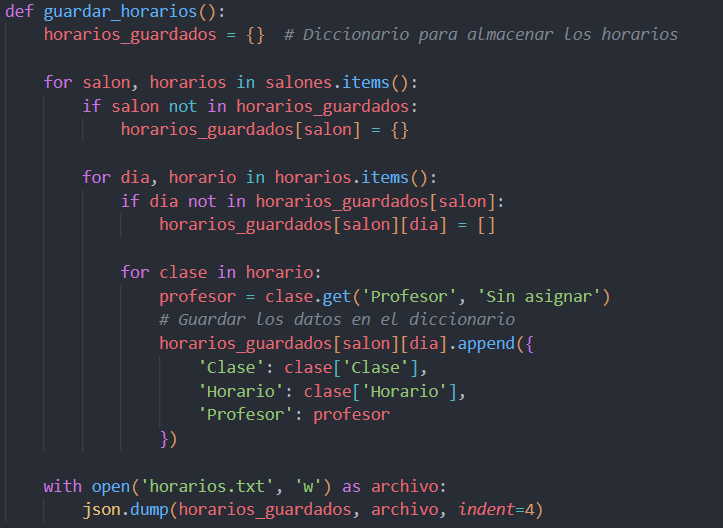


En el apartado de generar horarios, el programa copia el archivo de su ruta original y lo pega en la ruta donde se ejecuta, para finalmente asignarle el nombre de generador.csv, para así trabajar con él. Posteriormente, tuve que tomar una importante decisión, la cual fue ingresar el código del backend en el frontend. Entiendo que esta no es la mejor práctica, no obstante, otras opciones como la creación de señales era mucho más tedioso y complejo, por lo que optar por una opción más eficiente era necesario.

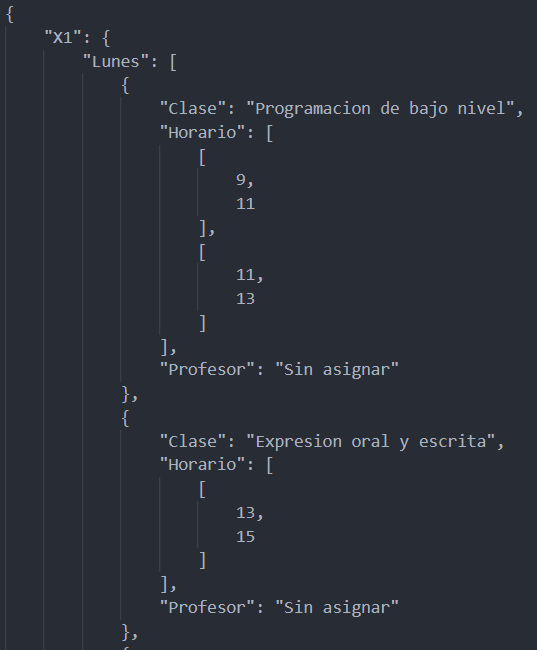
Para la lectura de este archivo, mejoré la función que carga la información desde el csv y la implementé en muchas de las pestañas del frontend:



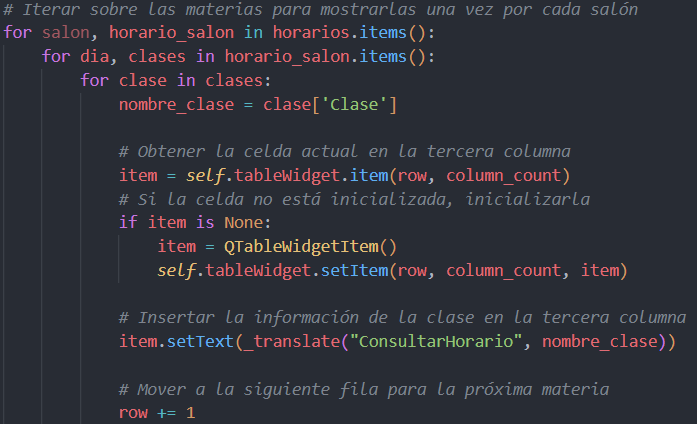
Haciéndola un poco más dinámica, ya que no necesita que se le especifique la ruta del archivo, sino que detecta el directorio en donde se encuentre el código. Posteriormente implementé las funciones encargadas de generar los horarios tal cual lo hizo mi desarrollador de backend, solo realizando un pequeño cambio en la función encargada de almacenar los datos en el diccionario:



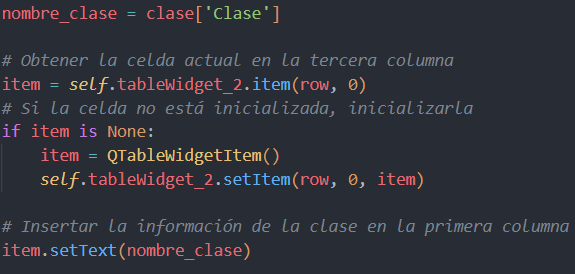
La principal razón es porque, al realizar pruebas, me di cuenta de que la información no era dinámica, y requeríamos de otra estructura capaz de almacenar y actualizar los horarios creados. Consideré que utilizar nuevamente un archivo csv era algo tedioso, así que me decidí por un archivo de texto común, donde el verdadero reto era mantener la estructura de diccionario que teníamos en el código. Es aquí donde entra el formato json, el cual sirve únicamente para que, al guardar el diccionario en el txt, permanezca como diccionario, y se ve algo como esto:



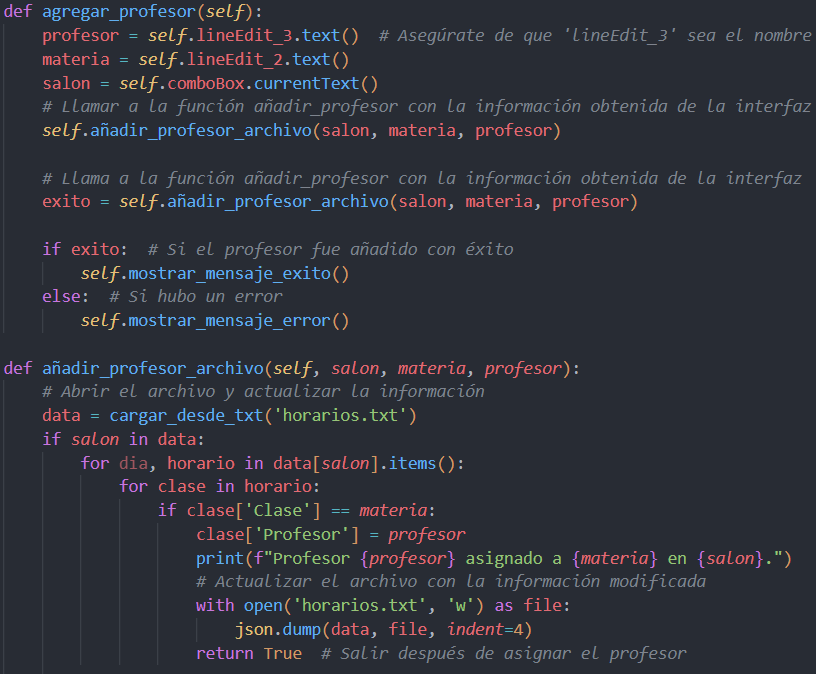
El apartado más complejo fue el de consulta de horarios, donde la solución la hallé en la iteración de mi diccionario para posteriormente ir mostrando mis valores en las columnas correspondientes, algo más o menos así:



Lo mismo para el día y la hora. Cabe destacar que todas las funciones obtienen la información de los horarios generados que yacen en el txt. Se hizo el mismo procedimiento para mostrar el día de la semana y el horario de la clase. Para mostrar las clases por salón se hizo algo similar, se asignaron los valores de la primera, segunda y tercera columna, haciendo un reduce y vencerás, ya que nos incumbía únicamente las clases de un salón en específico. Algo como esto para cada atributo:



Para agregar profesor y para la edición de las materias fue más sencillo. Se invocaba el diccionario generado, se buscaba la clase que debía ser editaba, se añadía o editaban los atributos pertinentes, para posteriormente actualizar el archivo txt. Para añadir profesor se hizo esto:



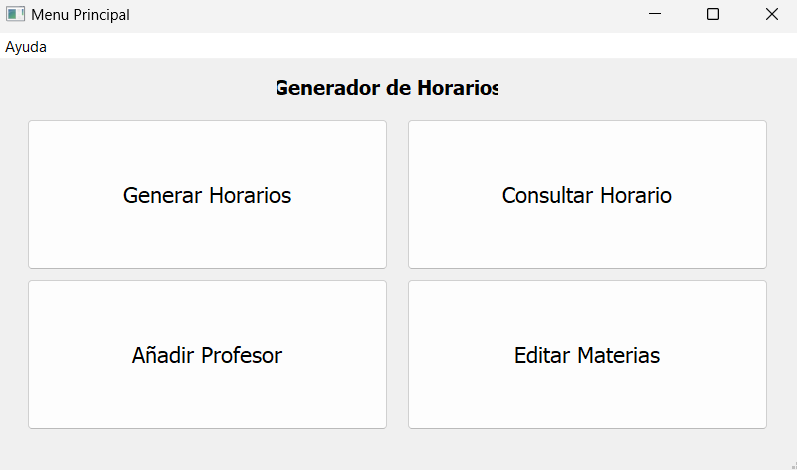
La edición de los horarios fue quizás el apartado más complejo. Tuve que realizar unas correcciones debido a que la generación de horarios estaba establecida como que debía buscar si había horarios disponibles, cosa imposible ya que todos los salones estaban ocupados todas las horas, por lo que decidí otorgarle libertad al usuario que, respetando el rango de horarios de 9 a 17 con clases de 2 horas, pudiese asignarle la que la plazca. Otro aspecto importante es que, para las clases de 4 horas, es necesario mantener dicha estructura, esto para salvaguardar la generación de los horarios. Por último, se actualiza el archivo que contiene el diccionario:



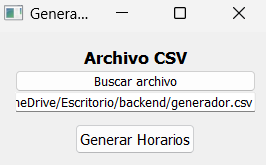
Para lograr un mejor acabado, se creó un nuevo entregable para el desarrollador frontend en el cual debía buscar la manera de convertir toda la aplicación creada en un archivo ejecutable. Cabe destacar que la documentación entregada por cada integrante del equipo profundiza en cada una de sus funciones y de lo que desarrollaron.

1. **Resultado.**

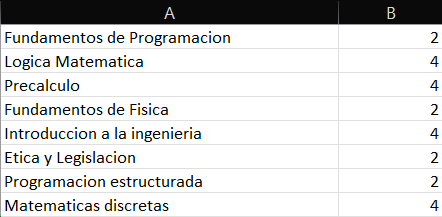
Como resultado, tuvimos el siguiente programa:



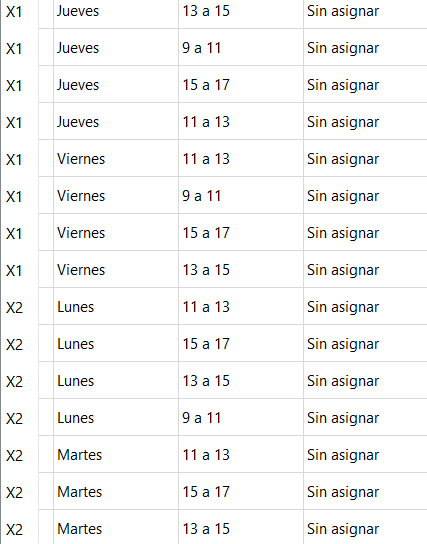
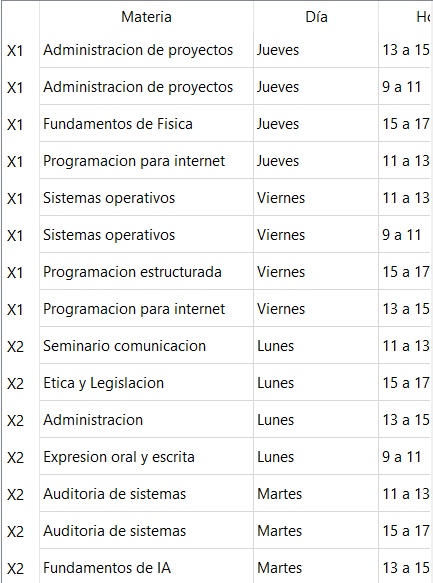
El menú principal sirve perfectamente, guiando al usuario por cualquier funcionalidad deseada. La función de generar horarios se ve así:



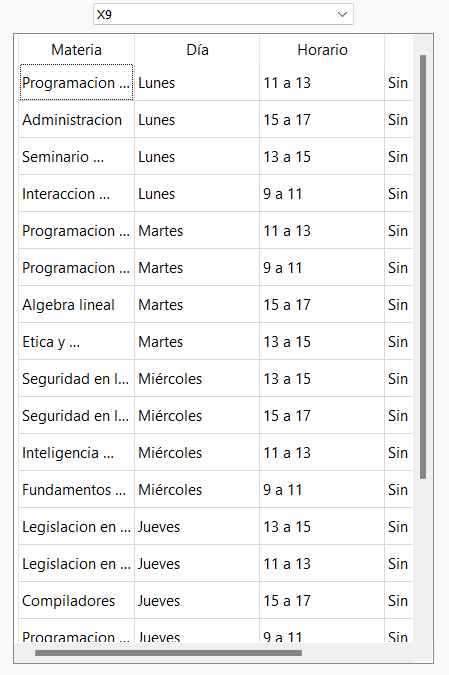
Le permite al usuario buscar el archivo csv el cual tomará como referencia para la generación de los horarios. En caso de no seleccionar un archivo, el usuario puede generar horarios y el programa lo hará en base al generador que ya se le haya ingresado previamente. Cabe destacar que el archivo csv se debe observar así:



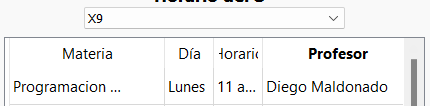
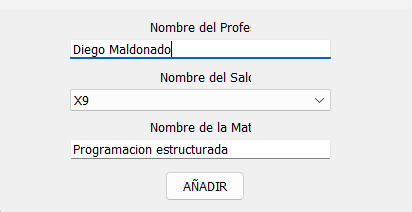
La primera columna es el nombre de las materias, mientras que la segunda es la duración de sus clases semanalmente.



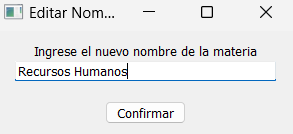
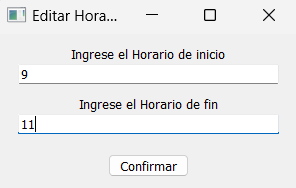
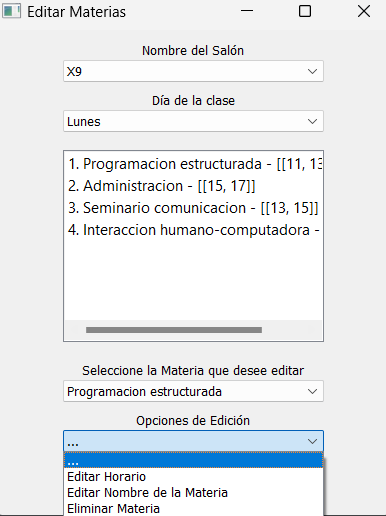
La pestaña encargada de mostrar el horario general se ve así, donde se muestran todas las clases generadas, su salón, el día, la hora, y el profesor que la imparta. La pestaña que muestra el horario particular de los salones es la siguiente:

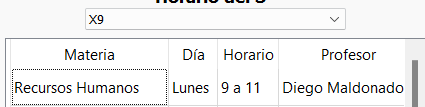


Muestra básicamente la misma información, pero de cada salón. Para asignar un profesor hay que indicar el salón, el nombre de la clase y el del profesor:

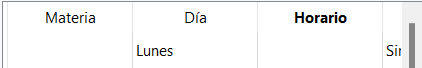


Y el resultado es mostrado. Se puede también editar la información de la materia, como su nombre y su horario:





También, la materia puede ser eliminada, con la intención de dejar ese espacio libre y poder ingresar la que desee el administrador:



El programa desarrollado cumple con lo planteado y con las expectativas. Para nada se considera como un programa perfecto, presenta ciertas fallas y estamos conscientes de que la conexión no se realizó de la mejor manera posible. No obstante, estamos muy contentos con el resultado, ya que es el fruto del empeño entregado en estas 3 semanas.

1. **Conclusión.**

La realización de este proyecto fue una verdadera experiencia. El dirigir a un grupo humanos fue mucho más complejo de lo pensado. Llevó a momentos de estrés, desesperación y frustración, muchas dudas se presentaron y en ciertos pasajes consideramos que completar el proyecto con éxito iba a ser difícil. No obstante, quiero hacer un reconocimiento a mi equipo, ya que desde el segundo 1 mostraron empatía y consideración por el resto de los integrantes. Tuvieron una actitud proactiva y colaboradora, lo que facilitó mi trabajo como administrador, y eso no es fácil conseguir en un equipo académico.

Consideramos que la realización del proyecto fue satisfactoria debido a que el programa desarrollado cumple con las funciones impuestas por la actividad, con un funcionamiento correcto, un diseño simple pero intuitivo y siendo bastante agradable a la hora de utilizarlo. Pero considero que el apartado más valioso no yace en el programa, sino en el hecho de que hacer proyectos de este tipo nos nutre en muchos aspectos. Como desarrolladores, tenemos que desarrollar la capacidad del aprendizaje rápido, ya que debemos aprender a utilizar herramientas que eran desconocidas; utilizar conceptos vistos durante la carrera, como estructuras de datos, manejos de archivos, o los algoritmos.

Y ajeno a aplicaciones técnicas, es de suma importancia la comunicación entre todos, el trabajo en equipo, la proactividad, creatividad y el optimismo, ya que, sin esto, estoy seguro de que el proyecto no se hubiese finalizado. En síntesis, este proyecto nos hace desarrollarnos en cosas más allá de Python o las librerías utilizadas, y eso tiene mucha valía.

1. **Referencias.**

Atlassian. (s. f.). *What is Agile?* https://www.atlassian.com/agile

GraphEverywhere. (2019). *Algoritmo de asignación de recursos*. https://www.grapheverywhere.com/algoritmo-de-asignacion-de-recursos/

Softjourn Inc. (2023). *Heuristics Unleashed: A comprehensive guide to heuristics in computer science and programming*. https://softjourn.com/insights/heuristic-programming#Definition-of-Heuristics