

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



**“DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN
DE AYUDANTES UTILIZANDO MODELOS BASADOS EN
RESTRICCIONES”**

JORGE FRANCISCO CABRERA BRUNA

**MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA**

Profesor Guía: Carlos Castro
Profesor Correferente: José Luis Martí

Noviembre - 2023

DEDICATORIA

A todos los que me importan.

AGRADECIMIENTOS

Agradezco en primer lugar a mis padres y familia, los cuales me dieron todo el apoyo y la oportunidad de dedicarme en exclusividad al estudio. A los profesores de la universidad y el departamento por su dedicación a la excelencia en la educación de los futuros profesionales. A mis compañeros y amigos con los cuales compartí sangre, sudor y lágrimas y por último a mi círculo social por ayudarme a despejar la mente cuando lo necesitaba.

RESUMEN

Resumen— Las universidades, en general buscan la ayuda de sus estudiantes en forma de asistentes de aprendizaje, comúnmente llamados ayudantes. Cada semestre inicia un proceso de búsqueda y selección de candidatos, donde cada coordinador elige según su propio criterio a sus ayudantes. De aquí nace el problema raíz ¿Cómo selecciono a un buen candidato? Este es un problema clásico de asignación, el cual ha sido modelado integrando diferentes posibles objetivos. Ahora nuestro dilema es como llevar este modelo teórico a la realidad, para ello utilizaremos herramientas de desarrollo de software para diseñar y crear una plataforma que permita recopilar todo lo necesario por este modelo teórico, así posteriormente utilizar un solver que nos entregue la solución de manera automática.

Palabras Clave— Desarrollo de software, Fullstack, Optimización, Solver, Ayudantes

ABSTRACT

Abstract— Universities generally seek help from their students in the form of teaching assistants, commonly called assistants. Each semester a process of search and selection of candidates begins, where each coordinator chooses his or her assistants according to his or her own criteria. This is where the root problem arises: How do I select a good candidate? This is a classic assignment problem, which has been modeled by integrating different possible objectives. Now our dilemma is how to bring this theoretical model to reality, for this we will use software development tools to design and create a platform to collect everything necessary for this theoretical model, and then use a solver to deliver the solution automatically.

Keywords— Software development, Fullstack, Optimization, Solver, Teaching Assistants

GLOSARIO

Backend: Parte de un sistema computacional encargado generalmente de las operaciones del servidor y aquellas no visibles al usuario.

CSP: Constraint satisfaction problem.

FrontEnd: Parte de un sistema computacional encargado de la interacción con el usuario

FullStack: Conjunto que incluye tanto frontend como backend.

GAP: Generalized Assignment Problem

Solver: Herramienta tecnológica para la resolución de CSP.

Stack Tecnológico: Conjunto de herramientas para la creación de software

INDICE DE CONTENIDOS

Contenido

RESUMEN	4
ABSTRACT.....	4
INDICE DE FIGURAS	8
INDICE DE TABLAS	9
INTRODUCCIÓN	10
1.1 Objetivo General	13
1.2 Objetivos Específicos	13
CAPÍTULO 2: MARCO CONCEPTUAL.....	14
2.1 Programación con Restricciones	14
2.1.1 Conceptos Y Objetivos Básicos CSP	14
2.1.2 Generalized Assignment Problem (GAP)	15
2.2 Modelo	16
2.3 Solver	19
2.3.1 Branch and bound	20
2.4 Desarrollo web	22
CAPÍTULO 3: PROPUESTA DE SOLUCION	24
3.1 Implementación	24
3.1.1 Requisitos Alumnos.....	24
3.1.2 Requisitos Profesores.....	25
3.1.3 Requisitos Administrador.....	26
3.1.4 Requisitos generales	27
3.2 Stack tecnológico	28
3.2.1 MongoDB	28
3.2.2 ExpressJS	31

3.2.3 React	31
3.2.4 NodeJS	32
3.2.5 NextJS	32
3.3 Modelamiento del problema	32
3.3.1 Construcción y solución de modelos	34
CAPÍTULO 4: ANALISIS DE RESULTADOS	37
4.1 Portal Estudiantes	37
4.2 Portal Profesores.....	45
4.3 Portal Administrador.....	50
CAPÍTULO 5: CONCLUSIONES Y TRABAJO FUTURO.....	58
5.1 Conclusiones	58
5.1.1 Diseño e implementación	58
5.1.2 Modelo.....	59
5.2 Trabajo futuro	60
5.2.1 Integración otros sistemas	60
5.2.3 Funcionalidades extra	60
5.2.4 Modificación de parámetros	62
5.2.5 Casos particulares	63
REFERENCIAS BIBLIOGRÁFICAS	64

INDICE DE FIGURAS

Figura 1: asignación subóptima	11
Figura 2: Proceso de postulación de ayudantías.	12
Figura 3: Proceso Selección y Resultados	12
Figura 4: Arbol del problema.....	13
Figura 5: Diagrama de flujo Branch and bound.....	21
Figura 6: Representación visual stack tecnológico.....	22
Figura 7: Ejemplo documento	29
Figura 8: Ejemplo de colección	29
Figura 9: Bases de datos implementada	30
Figura 10: Conversion del arbol DOM intermedio a HTML DOM.....	31
Figura 11: Comparación de formatos modelamiento problemas de optimización	34
Figura 12: Modelo generado para un caso 3x3	35
Figura 13: Output entregado por el solver.....	36

INDICE DE TABLAS

Tabla 1: Software y algoritmos disponibles para encontrar solución.	20
Tabla 2: Comparación entre stack tecnológicos.....	23
Tabla 3: Historia de usuario 01	24
Tabla 4: Historia de usuario 02	25
Tabla 5: Historia de usuario 03	25
Tabla 6: Historia de usuario 04	26
Tabla 7: Historia de usuario 05	26
Tabla 8: Historia de usuario 06	27
Tabla 9: Historia de usuario 07	27
Tabla 10: Historia de usuario 08	28
Tabla 11: Historia de usuario 09	28
Tabla 12: Resumen diferencias bases de datos SQL y MongoDB.....	30
Tabla 13: Preferencias de ejemplo para profesores y alumnos	36

INTRODUCCIÓN

En cada semestre académico existe un periodo de selección de ayudantes de docencia, el cual, se realiza de manera manual y a lo largo de varias iteraciones hasta que se cumplan las plazas necesarias. En un principio simple, a medida que el rango de variables aumenta, esto empieza a ser laborioso y encontrar una solución no perfecta pero adecuada ya puede significar un gasto de tiempo y recursos no despreciables. Este es un problema clásico de asignación de recursos, dentro de la disciplina de optimización es por ello que, es posible modelarlo mediante problemas de satisfacción de restricciones, dentro de los cuales ya existen propuestas de modelos que describen el proceso, es así como se produce la necesidad de dar el siguiente paso y llevarlo de la teoría a una forma práctica mediante el desarrollo de una plataforma que cumpla con los requisitos necesarios para automatizar el proceso y entregar mejores resultados.

Iniciamos con una introducción al desafío en el capítulo 1, y la necesidad de resolver el problema ahora practico y no tanto teórico, de una implementación de modelos de optimización para la asignación de ayudantes y su consiguiente diseño y desarrollo de una plataforma informática.

Así en el capítulo 2 se revisan los modelos y métodos de resolución estudiados y propuestos previamente por otros autores y como integrar estos a una plataforma que abarque todo el problema mediante un abanico de posibles herramientas de creación de software. Una vez hemos decidido nuestro stack informático este se muestra y presentan sus aspectos técnicos en el capítulo 3, donde además se listan requisitos tanto de diseño como funcionales mediante técnicas de ingeniería de software, tales como, historias de usuario.

Una vez elaborada la plataforma esta se muestra en el capítulo 4 mediante una sucesión de imágenes donde se observan los aspectos de diseño y el cumplimiento de criterios de aceptación para la correcta solución del problema, siguiendo un orden lógico de lo que sería un proceso real.

Finalmente, en el capítulo 5, se revisa el trabajo realizado y que el cumplimiento de sus resultados está en línea con los objetivos propuestos, así como intenciones, problemas y soluciones que se encontraron a lo largo de la elaboración de este. Por último, se mencionan posibles mejoras, modificaciones o cambios que podrían ser beneficios para la investigación en iteraciones o trabajos futuros.

CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA

Los ayudantes de docencia son un recurso muy valioso en el aprendizaje, por lo que saber aprovecharlos es de vital importancia, y un paso clave es la selección de estos, ya que el proceso es voluntario y los recursos escasos. Actualmente el proceso de postulaciones y selección es totalmente manual, cada interesado debe postular y subir información relevante para que posteriormente un profesor revise su postulación y decida a quien escoger, este proceso si bien sencillo, puede causar problemas al momento de tener una visión global del problema, creando asignaciones subóptimas o dejando vacantes sin ocupar debido a conflictos de preferencias, y quien realiza primero su selección, quitando oportunidad a otros, un ejemplo de esto se muestra en la figura 1.

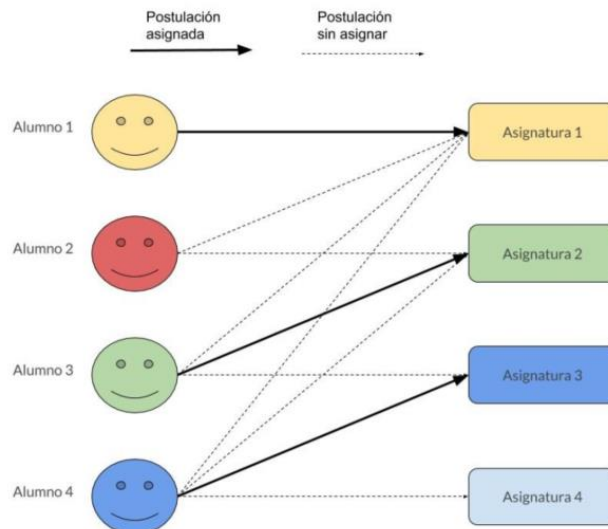


Figura 1: asignación subóptima
Fuente: (Marino García, 2020)

La principal causa de lo anterior es debido a la implementación existente donde no hay estandarización ni regulación del método de selección, es decir, los profesores escogen, cuando quieren, y como quieren, lo que lleva a situaciones desde su punto de vista optimas, sin embargo, este *first come first serve*, trae problemas como el de la figura 1: donde un postulante es seleccionado por un profesor para una asignatura, cuando este también postulaba a otra donde era el único, dejando así una vacante abierta que no será cubierta, generando una asignación global deficiente. Por otro lado, el proceso es inflexible, es decir, elijo a los n estudiantes que necesite, no se tienen en cuenta, preferencias del postulante como cuantas horas quiere dedicarle a la ayudantía, o quiere participar en más de una asignatura, pero se ve obligado a solo dictar una por restricciones de los departamentos y los casos especiales deben verse uno por uno. Las figuras 2 y 3 resumen el proceso tomado por estudiantes y profesores.

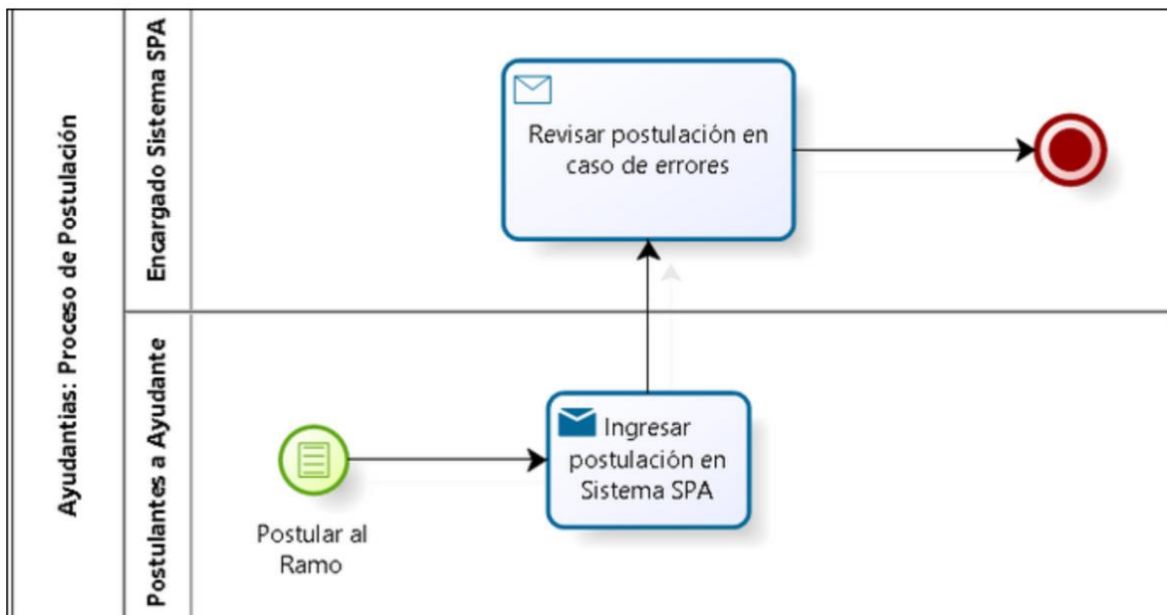


Figura 2: Proceso de postulación de ayudantías.

Fuente: Modelado de Procesos de la Subdirección de Pregrado del Departamento de informática de la UTFSM, Mauricio Muñoz.

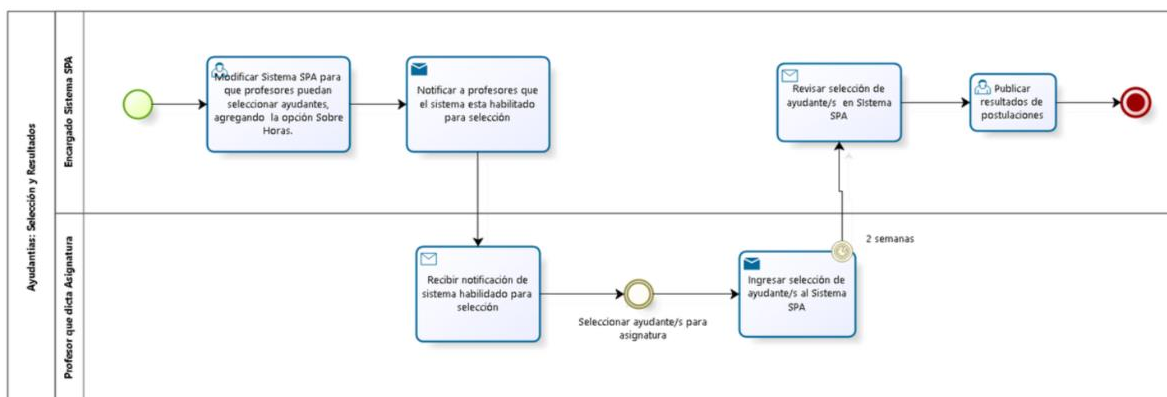


Figura 3: Proceso Selección y Resultados

Fuente: Modelado de Procesos de la Subdirección de Pregrado del Departamento de informática de la UTFSM, Mauricio Muñoz.

Los efectos del problema son directos, dado que quien elige primero tiene ventaja, puede existir una competición por los recursos humanos, o incitar a que alumno y profesor deban a comprometerse antes del proceso oficial. Además de que se generan asignaciones incompletas, lo que lleva a obligar a tener que repetir el proceso, haciendo segundos llamados. Por último, cómo es posible y natural, al comenzar un nuevo semestre y comenzar las postulaciones se pueden repetir asignaciones, pero no tenemos manera de determinar si aquella fue buena o no, tal vez es más conveniente otro postulante, o viceversa. Los puntos anteriores son resumidos por la figura 4.

Es así como surgen las preguntas:

- ¿Qué toman en consideración los profesores al momento de elegir un ayudante?
¿Qué toman en consideración los alumnos al momento de postular a una ayudantía?
- ¿Hay un método de selección de preferencias que optimice la distribución de ayudantes? ¿Son estos parámetros generales para cada usuario o específicos y diferentes?

Podemos representar el problema en sus partes esenciales en el siguiente árbol del problema:

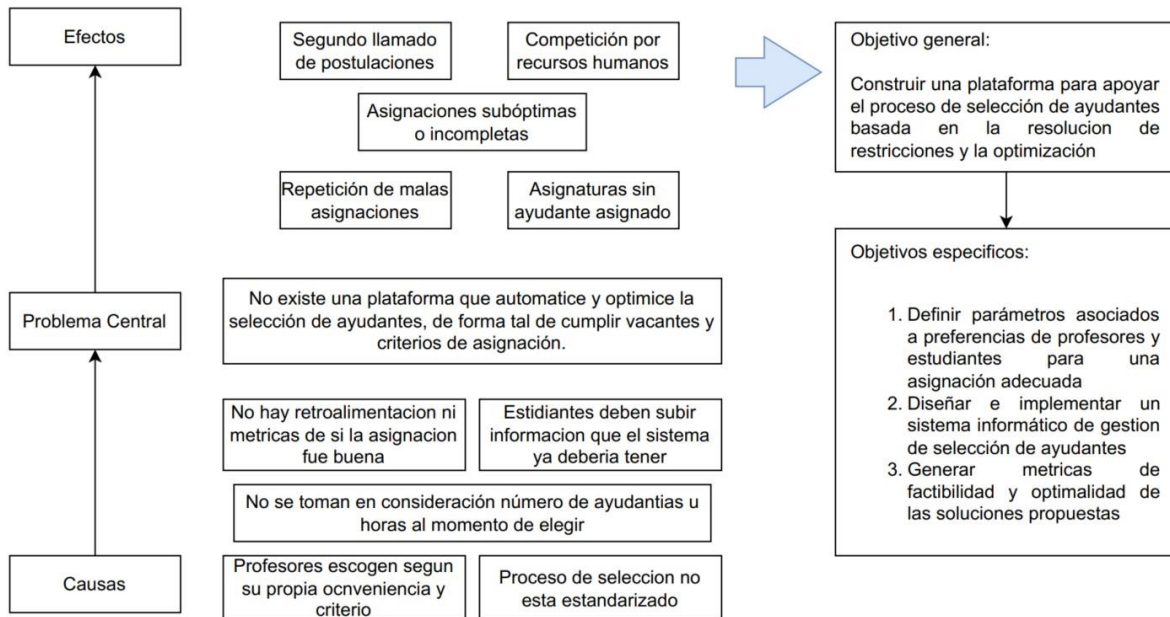


Figura 4: Arbol del problema

Fuente: Elaboración propia

1.1 Objetivo General

Construir una plataforma para apoyar el proceso de selección de ayudantes basada en la resolución de restricciones y la optimización.

1.2 Objetivos Específicos

1. Definir parámetros asociados a preferencias de profesores y estudiantes para una asignación adecuada.
2. Diseñar e implementar un sistema informático de gestión de selección de ayudantes.
3. Generar métricas de factibilidad y optimalidad de las soluciones propuestas.

CAPÍTULO 2: MARCO CONCEPTUAL

En general, el problema se puede modelar utilizando un conjunto de herramientas de optimización, las cuales deben integrarse conjuntamente para producir una solución. En esta línea tenemos los temas de problema de asignación, programación con restricciones, utilización de *solver*, y software que permita una conjunción de las herramientas.

2.1 Programación con Restricciones

Hoy en día muchas decisiones del mundo cotidiano están sujetas a restricciones que, si bien son tomadas en segundos o minutos, cuando nos vemos enfrentados a un problema de una magnitud mucho más elevada, nuestra agilidad mental ya no es suficiente, y si bien podemos encontrar soluciones que satisfacen el problema pueden llegar a no ser las más apropiadas.

Dado lo anterior existe la necesidad de llevar estos problemas a un mundo más formal, de donde nace la programación con restricciones, donde la resolución de un problema de satisfacción de restricciones (CSP) viene dada en dos fases:

- Modelar el problema como un problema de satisfacción de restricciones, es decir, llevar el problema a un lenguaje técnico matemático mediante el uso de un conjunto de variables, dominios y restricciones. En la sección 2.1.1 se presentan más detalles de los aspectos necesarios para realizar un modelo CSP.
- Procesar el problema de satisfacción de restricciones resultante, es decir, buscar una solución. Hay dos maneras de encontrar soluciones:
 - I. Técnicas de consistencia. Se trata de técnicas para la resolución de CSPs basadas en la eliminación de valores inconsistentes de los dominios de las variables.
 - II. Algoritmos de búsqueda. Estos algoritmos se basan en la exploración sistemática del espacio de soluciones hasta encontrar una solución o probar que no existe tal solución. En la sección 2.3 se presentan posibles métodos a utilizar para la búsqueda de soluciones.

Así podemos sintetizar la programación con restricciones como una tecnología de software utilizada para la descripción y posterior resolución efectiva de grandes y complejos problemas, particularmente combinatorios (Barber & Salido, 2003).

2.1.1 Conceptos Y Objetivos Básicos CSP

Un problema de satisfacción de restricciones (CSP) tiene 3 partes esenciales:

- Un conjunto de n variables $\{x_1, \dots, x_n\}$ al que llamaremos X .

- Un vector de dominios $\langle D_1, \dots, D_n \rangle$ al que llamaremos D . Donde D_i es el vector que contiene todos los valores posibles de la variable x_i .
- Un conjunto C finito de restricciones n -arias definidas sobre el conjunto X restringiendo todos los posibles valores que pueden tomar dichas variables.

Además, tenemos los siguientes conceptos claves:

- Asignación: llamaremos asignación o instanciación, (x, a) a un par variable-valor que representa la asignación del valor a a la variable x . De la misma manera si existe un conjunto de variables, su instanciación corresponderá a una tupla de pares ordenado, donde cada par representa una asignación variable-valor.
- Solución: La solución de un CSP corresponde a una asignación correcta de las variables, es decir, que la instancia actual satisface todas las restricciones, y todas las variables tienen una asignación.
- Restricciones: Una restricción corresponde a una relación entre una o más variables, la cual restringe los valores que se le pueden ser asignados.
- Variables: Las variables son el método en el cual representamos de forma matemática una situación o relación del mundo real.
- Dominio: El dominio corresponde a los valores que puede tomar una variable, sin considerar las restricciones impuestas por el modelo.

2.1.2 Generalized Assignment Problem (GAP)

Los CSP modelan diferentes situaciones y problemas de muchos ámbitos distintos, pero la derivación que nos interesa es el problema de asignación general, el cual en palabras sencillas intenta optimizar una asignación de tareas y agentes.

El problema de asignación generalizado corresponde a la tarea de asignar trabajos o *jobs* a agentes o maquinas, de tal forma que cada *job* es asignado a exactamente un agente sujeto a alguna restricción de capacidad. Este corresponde a un problema de optimización combinatorio del tipo NP-HARD (Cattrysse & Van Wassenhove, 1990).

La formulación matemática generalmente es presentada en forma de programación entera binaria (Nauss, 2003), es decir el valor de sus variables son 0 o 1, y se muestra a continuación:

Sea n el número de tareas para ser asignadas a m agentes ($n \geq m$) y se define $N = \{1, 2, \dots, n\}$. Se requieren los siguientes elementos:

c_{ij} = Costo de la tarea j al ser asignada al agente i

r_{ij} = Cantidad de recursos utilizados para la tarea j por el agente i

b_i = Recursos disponibles para el agente i

Las variables de decisión son definidas como:

$$x_{ij} = \begin{cases} 1, & \text{si la tarea } j \text{ es asignada a un agente } i \\ 0, & \text{en caso contrario} \end{cases}$$

Entonces escribimos el modelo como:

$$\text{minimizar } \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}$$

Sujeto a:

$$\sum_{j=1}^n r_{ij} x_{ij} \leq b_i \quad \forall i$$

$$\sum_{i=1}^m x_{ij} = 1 \quad \forall j \in N$$

$$x_{ij} = 0 \text{ o } 1, \forall i, j$$

2.2 Modelo

En trabajos anteriores (Marino García, 2020), se han propuesto algunos modelos que simulan la situación del departamento, a continuación, se presenta y revisa el modelo más prometedor y *ad hoc* a nuestros requerimientos.

1. Parámetros

- m : Cantidad de estudiantes postulantes a ayudantías.
- n : Cantidad de asignaturas.
- ha_j : Cantidad total de horas de ayudantía requeridas para la asignatura j .
- na_{max_j} : Cantidad máxima de ayudantes deseada para la asignatura j .
- na_{min_j} : Cantidad mínima de ayudantes deseada para la asignatura j .
- pp_{ij} : Preferencia del profesor j por el alumno i .
- he_{max_i} : Cantidad máxima de horas de ayudantía que puede realizar el estudiante i .
- he_{min_i} : Cantidad mínima de horas de ayudantía que puede realizar el estudiante i .
- ne_{max_i} : Cantidad máxima de ayudantías deseadas por el estudiante i .
- ne_{min_i} : Cantidad mínima de ayudantías deseadas por el estudiante i .
- pe_{ij} : Preferencia del estudiante i por la asignatura j .

2. variables

- x_{ij} : Se asigna alumno i a asignatura j ; $x \in \{0,1\} \forall i = 1, \dots, m; j = 1, \dots, n$
- y_{ij} : Cantidad de horas de ayudantía asignadas para el estudiante i en la asignatura j ; $y_{ij} \forall i = 1, \dots, m; j = 1, \dots, n$
- ya_j : Cantidad de horas de ayudantía satisfechas en la asignatura j ; $\forall j = 1, \dots, n$
- za_j : Cantidad de horas de ayudantía no satisfechas en la asignatura j ; $\forall j = 1, \dots, n$
- xpe : Valor total de las preferencias de alumnos asignadas.
- xpp : Valor total de las preferencias de profesores asignadas

3. Restricciones

- Cantidad total máxima de horas de ayudantía a asignar para cada asignatura:

$$\sum_{i=1}^m y_{ij} \leq ha_j \quad \forall j = 1, \dots, n$$

- Cantidad máxima de ayudantes a asignar para cada asignatura:

$$\sum_{i=1}^m x_{ij} \leq na_max_j \quad \forall j = 1, \dots, n$$

- Cantidad mínima de ayudantes a asignar para cada asignatura:

$$\sum_{i=1}^m x_{ij} \geq na_min_j \quad \forall j = 1, \dots, n$$

- Cantidad máxima de horas de ayudantía para cada estudiante:

$$\sum_{j=1}^n y_{ij} \leq he_max_i \quad \forall i = 1, \dots, m$$

- Cantidad mínima de horas de ayudantía para cada estudiante:

$$\sum_{j=1}^n y_{ij} \geq he_min_i \quad \forall i = 1, \dots, m$$

- Cantidad máxima de ayudantías para cada estudiante:

$$\sum_{j=1}^n x_{ij} \leq ne_max_i \quad \forall i = 1, \dots, m$$

- Cantidad mínima de ayudantías para cada estudiante:

$$\sum_{i=1}^m \sum_{j=1}^n x_{ij} \geq ne_min \quad \forall i = 1, \dots, m; j = 1, \dots, n$$

- Preferencias de estudiantes:

$$xpe = \sum_{i=1}^m \sum_{j=1}^n x_{ij} pe_{ij} \quad \forall i = 1, \dots, m; j = 1, \dots, n$$

- Preferencias de profesores:

$$xpp = \sum_{i=1}^m \sum_{j=1}^n x_{ij} pp_{ij} \quad \forall i = 1, \dots, m; j = 1, \dots, n$$

- Cantidad total de horas de ayudantía asignadas a la asignatura:

$$ya_j = \sum_{i=1}^m y_{ij} \quad \forall j = 1, \dots, m$$

- Cantidad de horas de ayudantía no satisfechas en la asignatura:

$$za_j = ha_j - ya_j \quad \forall j = 1, \dots, n$$

- No se pueden asignar horas de ayudantía a un estudiante no asignado, cota superior:

$$y_{ij} \leq x_{ij} he_max_i \quad \forall i = 1, \dots, m; j = 1, \dots, n$$

- No se pueden asignar horas de ayudantías a un estudiante no asignado, cota inferior:

$$y_{ij} \geq x_{ij} he_min_i \quad \forall i = 1, \dots, m; j = 1, \dots, n$$

4. Posibles funciones objetivo

$$\max f1 = \alpha \sum_{j=1}^n ya_j + \beta \cdot xpp + \gamma \cdot xpe$$

$$\max f2 = -\alpha \sum_{j=1}^n za_j + \beta \cdot xpp + \gamma \cdot xpe$$

$$\max f3 = -\alpha \sum_{j=1}^n \frac{za_j}{ha_j} + \beta \cdot xpp + \gamma \cdot xpe$$

Este modelo representa el sistema de postulación de ayudantías y, además toma en consideración parámetros que agregan valor a la asignación final que actualmente no son utilizados, pero entregan un mayor abanico de posibilidades. Específicamente el agregar las

horas disponibles por asignatura permite una distribución más fraccionaria y no tan entera, evitando el completo descarte de un alumno a otra asignatura.

Por otro lado, se nos presentan 3 opciones de funciones objetivo, lo que nos permite adaptarnos a situaciones particulares, si queremos maximizar las horas de ayudantía, minimizar las horas no asignadas o minimizar las horas no asignadas de forma uniforme. Finalmente, se nos entrega otro parámetro de control de preferencias mediante las constantes α , β y γ que nos permite manipular la importancia de las preferencias de profesores o alumnos en el cálculo final.

2.3 Solver

Hasta ahora hemos visto como modelar el problema por lo que el siguiente paso es la búsqueda de soluciones, para ello tenemos software especializado en la optimización y problemas combinatorios que utilizan diferentes técnicas de resolución. A continuación, se listan algunas:

1. OR-TOOLS: Perteneciente a Google, el cual está especializado en programación entera, programación lineal, problemas clásicos como enrutamiento de vehículos, pero sobre todo lo que nos interesa programación con restricciones. Este además de permitirnos llevar el modelo a la programación nos entrega distintos mecanismos para encontrar la solución, es más, nos da compatibilidad de utilizar otros solver. Utiliza múltiples métodos entre ellos tenemos tree search, branch-and-bound, local search con metaheurísticas del tipo mejor vecino (Shaw, Furnon, & De Backer).
2. Lp_solve: Es un solver de programación lineal entera mixta (Berkelaar, s.f.). Es capaz de resolver problemas puramente lineares, enteros, binarios, mixtos y semicontinuos. Para encontrar soluciones utiliza métodos como Branch-and-bound y revised simplex.
3. LINGO: Perteneciente a la familia de LINDO systems (Lindo systems, s.f.), es una herramienta no solo de búsqueda de soluciones sino de modelamiento. Es capaz de resolver problemas del tipo lineal, no-lineal, cuadráticos, cuadráticos restringidos, entre otros. Utiliza métodos como DualSimplex, branch-and-bound, Generalized Reduced Gradient.

Tabla 1: Software y algoritmos disponibles para encontrar solución.

Fuente: Elaboración propia

Software	Algoritmos disponibles
OR-TOOLS	Branch-and-bound, local search, tree search, best neighbor
Lp_solve	Branch-and-bound, simplex, revised simplex
LINGO	Branch-and-bound, DualSimplex, Generalized Reduced Gradient

2.3.1 Branch and bound

Branch and bound es el algoritmo por defecto utilizado por la mayoría de los softwares de optimización, por lo que veremos en detalle su funcionamiento, a través de los pasos que sigue para encontrar soluciones.

1. Usando alguna heurística, encontrar alguna solución x_h . Se guarda este valor, $B = f(x_h)$ como la mejor solución hasta ahora. (Una alternativa a usar heurísticas es establecer el primer valor como $B = \infty$)
2. Se inicializa una *cola* que contiene una solución parcial correspondiente a ninguna variable asignada.
3. Hacemos un bucle hasta que la *cola* está vacía:
 - i. Se quita un nodo N de la *cola*
 - ii. Si N es un candidato a solución x y $f(x) < B$, entonces x es la mejor solución hasta ahora. $B \leftarrow f(x)$
 - iii. Si no, hago un *Branch* en N para producir nuevos nodos N_i . Para cada uno de ellos:
 - a. Si $Bound(N_i) > B$, se ignora, puesto esta rama lleva a una solución peor a la almacenada.
 - b. Si no, almaceno N_i en la *cola*.

Los conceptos claves son *Branch* que significa una nueva instancia como un subset de todas las posibles soluciones, y *Bound* que calcula si la instancia actual es mejor o peor que la mejor hasta ahora, decidiendo si continua esa rama o no. En la figura 5, se muestra un diagrama de flujo siguiendo los pasos antes mencionados.

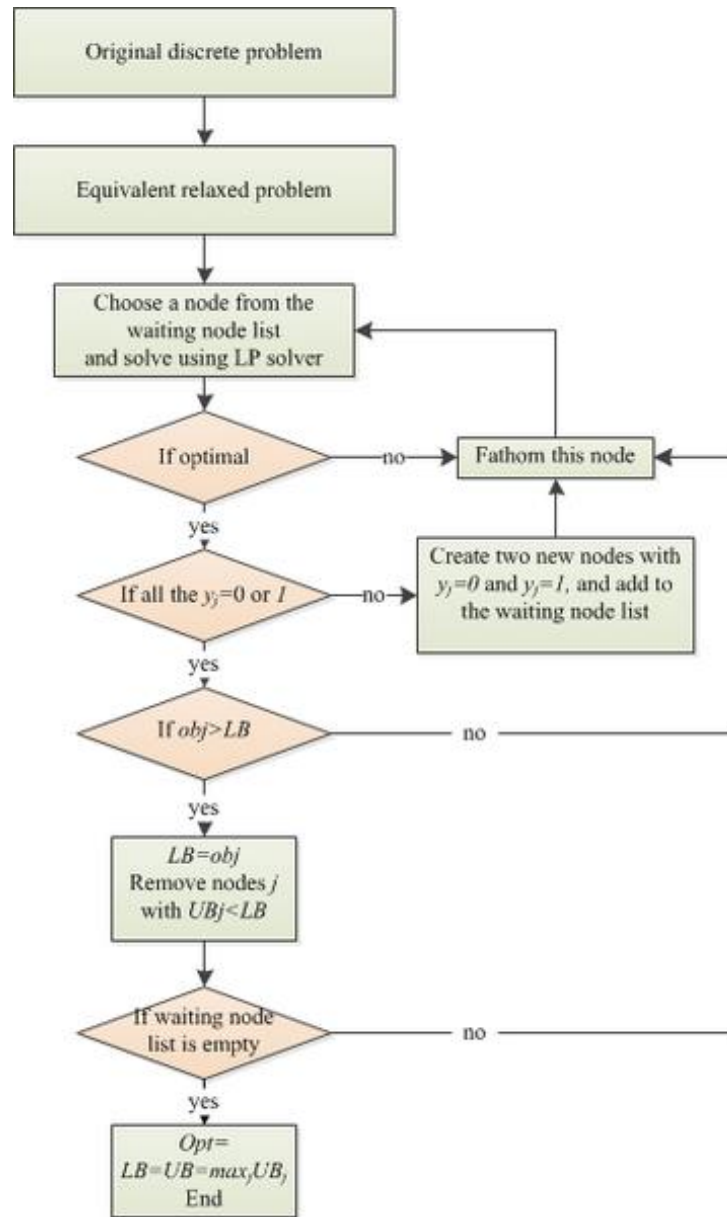


Figura 5: Diagrama de flujo Branch and bound
Fuente: (Gao, 2014)

2.4 Desarrollo web

En la ingeniería de software reciente, una de las industrias que más ha avanzado es el desarrollo de aplicaciones web. Como resultado de esta evolución también han surgido nuevos lenguajes de programación, herramientas y metodologías para desarrollar aplicaciones web.

Para el desarrollo existen stacks tecnológicos, conjuntos de servicios que se integran para facilitar el desarrollo de software, generalmente integrando, pero no limitándose a lenguajes de programación, framework, bibliotecas, entre otros. El stack tecnológico definirá el alcance y forma que puede llegar a tener la aplicación web final, por lo que elegir un stack adecuado es vital para la creación de software.

El stack puede ser completo o estar dividido en partes dedicadas hacia las funcionalidades de servidor, que lo llamamos backend, o el diseño y experiencia de lo que se encuentra el usuario, frontend. El backend se encarga de los servidores, aplicaciones y bases de datos e incluye lenguajes de programación dedicados a la lógica de la operación, mientras que el frontend lo conforman tres componentes esenciales HTML, que le da la estructura al diseño, CSS que le da los estilos a elementos HTML y un lenguaje de programación encargado de manejar las interacciones dentro de la página. Un stack que incorpora todo en su conjunto se denomina fullstack y en general tiene el aspecto mostrado en la figura 6. A continuación, se listan y comparan algunas de las tecnologías más populares.

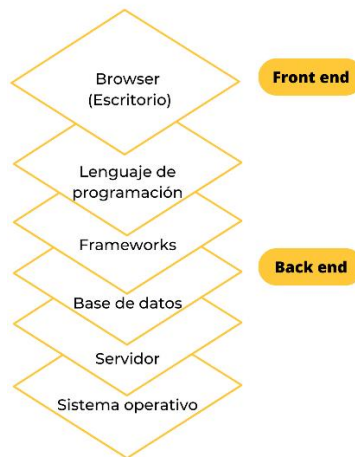


Figura 6: Representación visual stack tecnológico

Fuente: (syntonize, 2023)

- XAMPP: Corresponde a uno de los primeros stack creados, permite el desarrollo multiplataforma (X) y junta las herramientas de Apache (A) para la operación del servidor, MariaDB (M) en bases de datos y los lenguajes de PHP (P) y Perl (P). Este está pensado principalmente para el desarrollo local y requiere de complementos de seguridad para ser utilizado en la red.

- Django: Django está construido en Python y ofrece un desarrollo fullstack, entregando rapidez, seguridad y escalabilidad. Entrega herramientas para crear la lógica de la interacción con el usuario mediante su capa de vistas y manipulación y abstracción de datos mediante su capa de modelos. Da la libertad de elegir la base de datos más apropiada, pues este soporta la mayoría de ellas.
- Ruby on rails: Comúnmente conocido como Rails, permite el desarrollo fullstack a través del patrón modelo-vista-controlador, está basado en el lenguaje Ruby y sus Gemas las que corresponden a plugins, códigos que nos expanden las funcionalidades. Este permite la conexión con la mayoría de bases de datos, por lo que queda a decisión del desarrollador.
- MEAN/MERN/MEVN: Es otro fullstack que integra las herramientas de MongoDB en bases de datos, ExpressJS para el manejo de peticiones HTTP y NodeJS como la base de todo el conjunto, encargado del servidor y gestor de paquetes, finalmente para la capacidad de creación de vistas existen múltiples opciones como lo son Angular, React o Vue, que se encargan de facilitar el diseño y manipulación de HTML.

Tabla 2: Comparación entre stack tecnológicos
Fuente: elaboración propia

	XAMPP	Ruby on rails	Django	MEAN/MERN/MEVN
Lenguajes	PHP, Perl	Ruby	Python	JavaScript
Bases de Datos	MariaDB	Varias	Varias	MongoDB
Servidor	Apache	Apache	Apache	Node, Express
Vistas	HTML	Ruby HTML	HTML	Angular, React, Vue

CAPÍTULO 3: PROPUESTA DE SOLUCION

Para lograr los objetivos planteados es necesario crear una plataforma que integre tanto sistemas informáticos de interacción con el usuario como un solver de optimización que entreguen resultados según las interacciones realizadas por los usuarios.

3.1 Implementación

Para llevar a cabo la implementación en primer lugar se definen los requisitos necesarios por el software, así para una división mejor estos se separan por tipo de usuario, definidos a continuación:

- Alumnos: Estudiantes que han cursado las asignaturas y están interesados en participar como ayudantes.
- Profesores: Encargados de coordinar o dictar las asignaturas, los cuales definen los criterios para aceptar ayudantes.
- Administrador: Encargado de supervisar el proceso, activar y cerrar las postulaciones y solucionar eventualidades.

Según cada usuario tendremos requisitos específicos los cuales serán expresados a través de historias de usuario, finalmente también se tendrán requisitos generales, los cuales son transversales a todo el sistema.

3.1.1 Requisitos Alumnos

Tabla 3: Historia de usuario 01
Fuente: elaboración propia

Nombre historia	Postulación a ayudantías
Código historia	HU01
Descripcion	Como estudiante quiero postular al cargo de ayudante para usar mi experiencia en la asignatura enseñando a mis compañeros.
Criterios de aceptación	La página tendrá una sección de postulaciones donde mostrará todas las asignaturas disponibles para postular.
	La página solicitará seleccionar las asignaturas de interés e indicar un nivel de un nivel de preferencia.
	El alumno podrá eliminar, modificar o cambiar su postulación.

Tabla 4: Historia de usuario 02

Fuente: elaboración propia

Nombre historia	Visualizaciones estadísticas para alumnos
Código historia	HU02
Descripción	Como estudiante quiero ver estadísticas de mis postulaciones y resultados generales para conocer las tendencias de los alumnos.
Criterios de aceptación	La página tendrá una sección de estadísticas donde se mostrará un recuento de los procesos ya finalizados.
	La página mostrara en forma gráfica datos anónimos de los procesos ya terminados.
	La página mostrara estadísticas sobre las postulaciones del alumno y generales.

3.1.2 Requisitos Profesores

Tabla 5: Historia de usuario 03

Fuente: elaboración propia

Nombre historia	Definir preferencia del profesor
Código historia	HU03
Descripción	Como profesor quiero escoger parámetros específicos para seleccionar al ayudante más adecuado.
Criterios de aceptación	La página tendrá una sección de ajustes de preferencias donde mostrará las asignaturas del profesor.
	La página solicitara seleccionar las asignaturas de interés e indicar un nivel de un nivel de preferencia.
	El alumno podrá eliminar, modificar o cambiar su postulación.

Tabla 6: Historia de usuario 04

Fuente: elaboración propia

Nombre historia	Visualizaciones estadísticas para profesores
Código historia	HU04
Descripción	Como profesor quiero ver estadísticas sobre las postulaciones y resultados para mejorar la definición de parámetros y ver el éxito del proceso.
Criterios de aceptación	La página tendrá una sección de estadísticas donde mostrará un recuento de los procesos ya finalizados.
	La página mostrara en forma gráfica datos anónimos de los procesos ya terminados.
	La página mostrara estadísticas de las asignaturas del profesor, así como generales del proceso.

3.1.3 Requisitos Administrador

Tabla 7: Historia de usuario 05

Fuente: elaboración propia

Nombre historia	Configuración proceso
Código historia	HU05
Descripción	Como administrador quiero controlar el proceso de postulaciones para lograr una asignación efectiva.
Criterios de aceptación	La página tendrá una sección de configuración única para usuario administrador donde se podrá cargar asignaturas
	La página le permitirá iniciar, pausar o terminar el proceso.
	La página le entregara una vista previa de las asignaciones al administrador cuando este de por terminado el proceso.
	La página le permitirá publicar los resultados una vez este conforme con ellos.

Tabla 8: Historia de usuario 06

Fuente: elaboración propia

Nombre historia	Configuración y control de asignaturas y postulaciones
Código historia	HU06
Descripción	Como administrador quiero poder manipular las asignaturas y postulaciones para tener mayor control del proceso.
Criterios de aceptación	La página le entregará al administrador un menú de asignaturas donde podrá agregar, eliminar o modificar las mismas.
	La página le entregará al administrador un historial de asignaturas de procesos anteriores con sus parámetros y ayudantes seleccionados.
	La página le entregará al administrador un menú donde podrá visualizar, modificar, o eliminar las postulaciones de estudiantes.

3.1.4 Requisitos generales

Tabla 9: Historia de usuario 07

Fuente: elaboración propia

Nombre historia	Página home
Código historia	HU07
Descripción	Como usuario quiero ver una página de inicio para acceder a todas las opciones disponibles.
Criterios de aceptación	La página de inicio mostrará el estado del proceso actual e información relevante a él.
	La página de inicio permitirá navegar al proceso de postulaciones, estadísticas, modificaciones e informaciones.

Tabla 10: Historia de usuario 08

Fuente: elaboración propia

Nombre historia	Pagina perfil
Código historia	HU08
Descripción	Como usuario quiero ver una página perfil para verificar mis datos.
Criterios de aceptación	La página perfil mostrara información básica del usuario.
	La página perfil permitirá ingresar una breve descripción del usuario.
	La página perfil permitirá ingresar información adicional, y decidir si mostrar o no que tipo de información.

Tabla 11: Historia de usuario 09

Fuente: elaboración propia

Nombre historia	Pagina informaciones
Código historia	HU09
Descripción	Como usuario quiero ver una página con preguntas frecuentes sobre el proceso por si tengo alguna duda.
Criterios de aceptación	La página de informaciones tendrá respuestas a preguntas frecuentes sobre el proceso.
	La página de informaciones mostrara información según el usuario que acceda a esta.

3.2 Stack tecnológico

Se ha seleccionado el stack MERN (MongoDB, ExpressJS, React y NodeJS) ya que este cumple con todo lo necesario para construir dicha plataforma, además de encontrarse ampliamente documentado y con una de las comunidades más grandes de colaboradores lo que nos agrega facilidad ante la posible aparición de complicaciones o dudas al momento de la construcción.

3.2.1 MongoDB

MongoDB es una base de datos no relacional de código abierto, que entrega gran disponibilidad, rendimiento y escalabilidad.

MongoDB basa su funcionamiento en documentos JSON (véase figura 7), el cual es un formato para el intercambio de datos, estos están encerrados entre llaves { } y en su interior existen pares clave: valor separados por comas, internamente se almacenan en formato BSON el cual es una representación binaria de JSON y no es estrictamente necesaria sigan el mismo esquema. Haciendo el símil a una base de datos relacional el documento correspondería a una fila y los pares a los atributos.

```
{
  name: "sue",
  age: 26,
  status: "A",
  groups: [ "news", "sports" ]
}
```

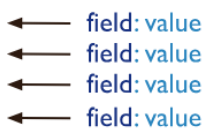


Figura 7: Ejemplo documento
Fuente: Documentación MongoDB

Los documentos JSON se agrupan en lo que llamamos colección (véase figura 8), una colección es equivalente a una tabla en las bases de datos SQL, y múltiples colecciones pueden pertenecer a la misma base de datos, pero, aunque no lo hagan de igual manera pueden interactuar entre sí. En la tabla 12 se establece un resumen de los paralelismos mencionados entre bases de datos SQL y mongoDB

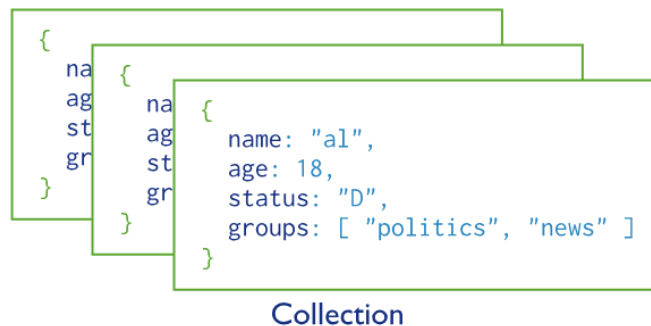


Figura 8: Ejemplo de colección
Fuente: Documentación MongoDB

Por supuesto esto trae consigo también desventajas, al no seguir un esquema relacional, no es normalizada, lo que, si bien esto entrega flexibilidad, implica que la lógica de los esquemas este dictada en su materia por la aplicación y no la base de datos misma. Por otro lado, existe el riesgo de aumentar la redundancia de datos, al existir una misma variable en diferentes documentos de diferentes colecciones o inclusive al no seguir necesariamente todos los documentos la misma estructura.

Tabla 12: Resumen diferencias bases de datos SQL y MongoDB
Fuente: (Chauhan, 2019)

SQL	MongoDB
Base de datos	Base de datos
Tabla	Colección
Índice	Índice
Fila	Documento BSON
Columna	Campo BSON
Join	Documentos empotrados y links
Llave primaria	Llave primaria
Group By	Agregation

En la figura 9 se presenta el modelo a implementar. Teniendo la colección de USUARIOS, donde se almacena los datos específicos de cada usuario, POSTULACIONES, conteniendo la información de cada postulación hecha por los estudiantes, ASIGNATURAS, guardando la información de los cursos impartidos y preferencias de los profesores, y finalmente AYUDANTIAS, que contiene la información de los seleccionados mediante el proceso.

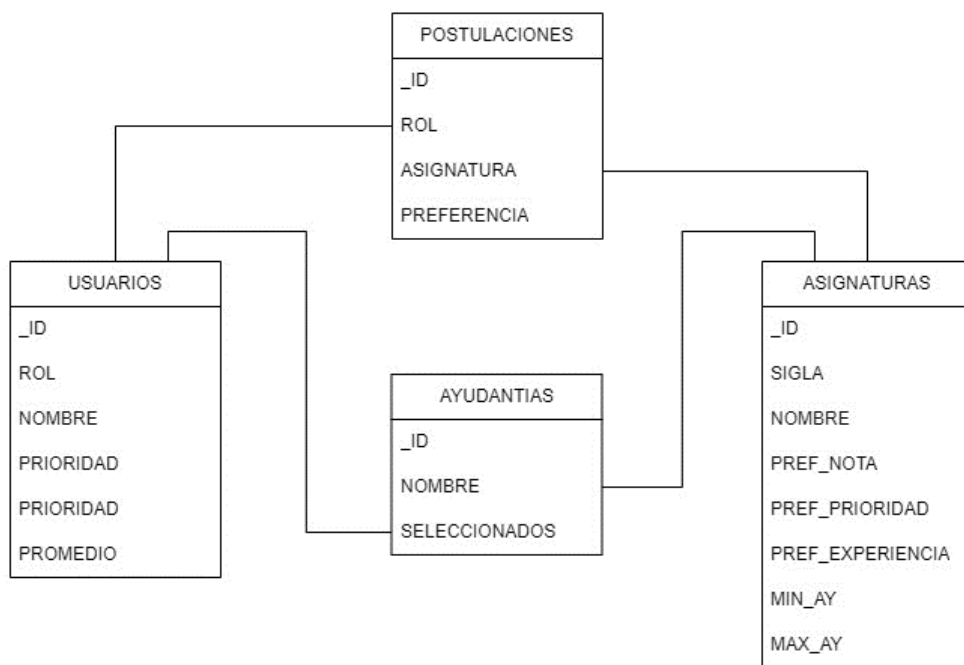


Figura 9: Bases de datos implementada
Fuente: Elaboración propia

3.2.2 ExpressJS

Express es un módulo de NodeJS especializado en la creación de infraestructura mínima y flexible que proporciona un conjunto de características para el desarrollo web. Este ofrece los métodos para manejar las solicitudes del protocolo HTTP, la comunicación entre programas mediante APIs y un sistema simple de enrutamiento.

3.2.3 React

React es una librería de JavaScript que permite desarrollar componentes reusables de interfaces de usuario. React se define a sí mismo como una librería para construir interfaces de forma modular. React corresponde a la herramienta que nos permite crear las vistas en el método de trabajo MVC y por lo tanto lo que denominamos *frontend*.

Entre sus principales características están su árbol DOM ligero, lo que entrega un rendimiento rápido y robusto de la aplicación. React utiliza una extensión de JavaScript conocida como JSX, similar a código XML, estos son conocidos como componentes y permite acelerar y facilitar la escritura de código gracias a estos ser trozos de código independientes y reusables.

Los componentes corresponden a nuestros bloques de construcción, las interfaces serán desarrolladas mediante la unión de diferentes componentes generando un árbol intermedio de componentes el cual es posteriormente transformado al DOM de HTML (figura 10), esto es lo que diferencia a React de otros *frameworks* mientras otros hacen los cambios de forma dinámica y en vivo, renderizando el DOM a cada cambio, React hace una copia en memoria del original para que la comparación de cambios sea muy rápida.

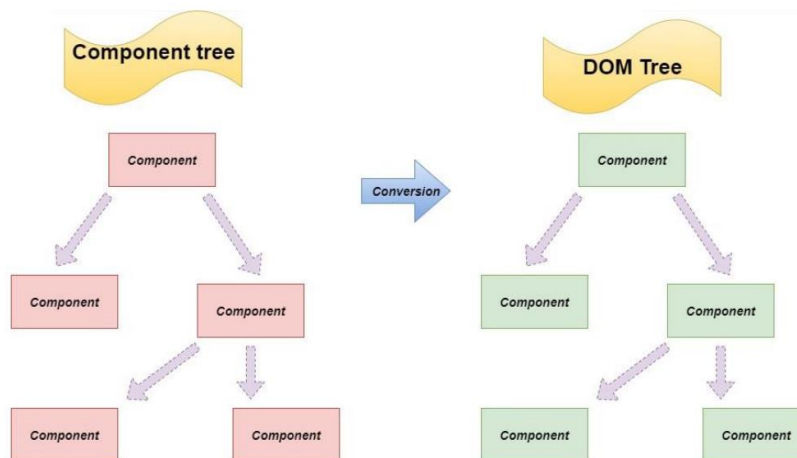


Figura 10: Conversión del árbol DOM intermedio a HTML DOM
Fuente: Modern Web-Development using ReactJS

Adicionalmente existen herramientas que nos permiten acelerar más el desarrollo como lo son Material UI y Tailwind CSS. Material UI corresponde a una librería de componentes de React, lo que nos permite ahorrarnos el diseño y funcionamiento de secciones de la interfaz,

incluye además iconos, tipografías y plantillas. El segundo corresponde a un *framework* de CSS que nos entrega estilos predefinidos ayudando a crear diseños más eficientes y limpios.

3.2.4 NodeJS

El último término del acrónimo, sin embargo, una parte esencial de él ya que es la base donde se construye todo, en palabras sencillas se encarga del funcionamiento tras bambalinas y se define como “un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor, basado en el lenguaje de programación JavaScript”, es decir, nos entrega herramientas para construir nuestro *backend*.

Una parte importante de node es su gestor de paquetes (npm) el cual nos permite integrar módulos de forma rápida, esto es esencial pues utilizaremos muchos de ellos en el desarrollo, pues nos entregan herramientas ya elaboradas.

3.2.5 NextJS

Para crear la aplicación en React se utiliza NextJS, el cual es un framework especializado para crear aplicaciones renderizadas en servidor. Es por ello que muchas de las prestaciones que nos entrega se solapan con las ya mencionadas como el *routing* de ExpressJS o el servidor de NodeJS, sin embargo, su uso lo limitamos a los meramente relacionados con React.

3.3 Modelamiento del problema

En los modelos presentados por (Marino García, 2020) se entrega un coeficiente de preferencia el cual es usado para determinar la mejor asignación, sin embargo, no se ahonda en este en cuanto a cómo asignarlo, su escala o representación, es más, su uso es puramente según uno considere adecuado, por lo que una de las principales modificaciones que se propone, es definir este parámetro en función de 3 características, nota estudiante, prioridad académica estudiante y experiencia del estudiante.

La nota del estudiante (de la asignatura a la cual está postulando) y su prioridad académica son datos numéricos exactos, sin embargo, la experiencia es más difícil de cuantificar, por lo que se optará por lo simple y se definirá como el número de ayudantías previamente realizadas.

Así los profesores tendrán la capacidad de definir la importancia que tendrá cada variable para el cálculo del parámetro “preferencia del profesor”, lo cual queda definida matemáticamente de la siguiente forma:

$$pp = \alpha \cdot nota_alumno + \beta \cdot prioridad_academica + \delta \cdot experiencia$$

Donde:

$$\alpha + \beta + \gamma = 1$$

$$0 \leq \alpha \leq 1$$

$$0 \leq \beta \leq 1$$

$$0 \leq \gamma \leq 1$$

Para el modelo a implementar, se priorizan dos objetivos, completar el total de ayudantes requeridos, y maximizar las preferencias de profesores y estudiantes. Además, se toma la asignación en cuanto a unidad de ayudantía y no por horas como en otras propuestas. Así el modelo a implementar es el siguiente:

1. Parámetros

- m : Cantidad de estudiantes postulantes a ayudantías.
- n : Cantidad de asignaturas.
- na_{max_j} : Cantidad máxima de ayudantes deseada para la asignatura j .
- na_{min_j} : Cantidad mínima de ayudantes deseada para la asignatura j .
- pp_{ij} : Preferencia del profesor j por el alumno i .
- ne_{max_i} : Cantidad máxima de ayudantías deseadas por el estudiante i .
- ne_{min_i} : Cantidad mínima de ayudantías deseadas por el estudiante i .
- pe_{ij} : Preferencia del estudiante i por la asignatura j .

2. Variables

- x_{ij} : Se asigna alumno i a la asignatura j
- $x \in \{0, 1\} \forall i = 1, \dots, m; j = 1, \dots, n$

3. Restricciones

- Cantidad máxima de ayudantes a asignar para cada asignatura

$$\sum_{i=1}^m x_{ij} \leq na_{max}$$

- Cantidad mínima de ayudantes a asignar para cada asignatura

$$\sum_{i=1}^m x_{ij} \geq na_{min}$$

- Cantidad máxima de ayudantías para cada estudiante

$$\sum_{j=1}^n x_{ij} \leq ne_{max}$$

- Cantidad mínima de ayudantías para cada estudiante

$$\sum_{j=1}^n x_{ij} \geq ne_{min}$$

4. Función Objetivo

$$\text{maximizar} \sum_{j=1}^n \sum_{i=1}^m x_{ij} \cdot p_{e_{ij}} + \sum_{j=1}^n \sum_{i=1}^m x_{ij} \cdot p_{p_{ij}}$$

Este modelo si bien sencillo nos entrega una solución muy buena, y es más podemos dar por sentado la restricción 4, ya que realmente no existe una obligación de que un alumno sea asignado a una ayudantía, designando el mínimo como 0, por otro lado tenemos un conflicto “administrativo” con la restricción 3, dado que si un estudiante desea tener muchas ayudantías esto será negado por dirección de estudios, o inclusive en casos bajos como 2, requieren de aprobaciones especiales, por lo que para estos casos es necesario limitar automáticamente a 1 o 2 ayudantías como límite. Por otro lado, la sencillez del modelo nos permite gastar recursos en el cálculo de las preferencias ya que estas son la parte más importante para poder hacer la mejor asignación final.

3.3.1 Construcción y solución de modelos

En general la construcción de modelos de optimización depende del software que se utilice, en este caso, se utilizara LP SOLVE, ya que este nos entrega una facilidad al momento de escribir el modelo en lenguajes de programación, su formato LP.

El formato LP de LP SOLVE, es una forma intuitiva de escribir los modelos ya que este formato es muy similar al formato analítico manual y se aleja un poco de la abstracción de la programación del formato más común “MPS” pudiendo transferir el modelo descrito en la sección anterior de manera directa. Como se aprecia en la figura 11, podemos observar que un formato MPS es más complejo de leer mientras que en el formato LP es muy similar al formato escrito de ecuaciones, se aprecian los parámetros, restricciones y funciones objetivo, tal cual como si estuviéramos leyendo uno escrito a mano.

MPS FORMAT						EQUATION ORIENTED		LP	
ROWS						Optimize		max: XONE + 4*YTWO + 9*ZTHREE	
N	COST					COST: XONE + 4*YTWO + 9*ZTHREE			
L	LIM1					Subject To		XONE + YTWO <= 5	
G	LIM2					LIM1: XONE + YTWO <= 5		XONE + ZTHREE >= 10	
E	MYEQN					LIM2: XONE + ZTHREE >= 10		XONE <= 4	
COLUMNS						MYEQN: - YTWO + ZTHREE = 7		-1 <= YTWO <= 1	
	XONE	COST	1	LIM1	1	Bounds			
	XONE	LIM2	1			XONE <= 4			
	YTWO	COST	4	LIM1	1	-1 <= YTWO <= 1			
	YTWO	MYEQN	-1			End			
	ZTHREE	COST	9	LIM2	1				
	ZTHREE	MYEQN	1						
RHS									
	RHS1	LIM1	5	LIM2	10				
	RHS1	MYEQN	7						
BOUNDS									
	UP BND1	XONE	4						
	LO BND1	YTWO	-1						
	UP BND1	YTWO	1						
ENDATA									

Figura 11: Comparación de formatos modelamiento problemas de optimización

Fuente: Elaboración propia

De esta forma podemos escribir el modelo de forma intuitiva y dinámica solamente conociendo los parámetros de profesores y estudiantes. Para ello se utiliza el lenguaje de programación Python, usando sus capacidades más básicas logramos escribir el modelo en el formato LP.

Para ejemplificar la construcción de los modelos, se fabrica uno con 3 asignaturas y 3 postulantes, representado en la figura 12. Aquí cada par X_{ab} significa el estudiante a es asignado a la asignatura b , y el valor de X será 0 si la condición no se cumple o 1 en caso contrario.

Además, como se mencionó necesitamos calcular las preferencias de profesores según los parámetros entregados, aprovecharemos esta instancia de la construcción para hacerlo en Python esto es visible en la imagen 12, específicamente dentro de la función objetivo, donde se aprecia un valor numérico junto a las variables, estos pueden ser vistos además en la tabla 13.

```
max: 4 X11 + 4 X21 + 4 X31 + 2 X11 + 2 X21 + 2 X31 +  
4 X12 + 4 X22 + 4 X32 + 2 X12 + 2 X22 + 2 X32 +  
4 X13 + 4 X23 + 4 X33 + 2 X13 + 2 X23 + 2 X33;  
  
X11 + X21 + X31 <= 1;  
X12 + X22 + X32 <= 1;  
X13 + X23 + X33 <= 1;  
  
X11 + X21 + X31 >= 1;  
X12 + X22 + X32 >= 1;  
X13 + X23 + X33 >= 1;  
  
X11 + X12 + X13 <= 1;  
X21 + X22 + X23 <= 1;  
X31 + X32 + X33 <= 1;  
  
X11 + X12 + X13 >= 1;  
X21 + X22 + X23 >= 1;  
X31 + X32 + X33 >= 1;  
  
bin X11 , X12 , X13 , X21 , X22 , X23 , X31 , X32 , X33;
```

Figura 12: Modelo generado para un caso 3x3
Fuente: Elaboración propia

Tabla 13: Preferencias de ejemplo para profesores y alumnos
Fuente: Elaboración propia

Profesor	1	2	3
Preferencia	2	2	2
Alumno	1	2	3
Preferencia	4	4	4

Las líneas siguientes corresponden a las restricciones del modelo, cantidad máxima de ayudantes a asignar a cada ayudantía, Cantidad mínima de ayudantes a asignar para cada asignatura, Cantidad máxima de ayudantías para cada estudiante, Cantidad mínima de ayudantías para cada estudiante. Finalmente dado la naturaleza binaria de X es necesario indicarlo dentro del modelo en la última línea.

Así podemos entregarle este modelo al solver, el cual nos devuelve una asignación óptima, como la mostrada en la figura 13. Aquí observamos los valores 0 y 1 según el alumno ha sido asignado o no. Para nuestro caso el alumno 1 ha sido asignado a la asignatura 1, el alumno 2 a la asignatura 2, y el alumno 3 a la asignatura 3.

X11	1
X21	0
X31	0
X12	0
X22	1
X32	0
X13	0
X23	0
X33	1

Figura 13: Output entregado por el solver
Fuente: Elaboración propia

Así finalmente tendremos la siguiente línea de acción, se inicia la petición de distribución, se recuperan de la base de datos las postulaciones y preferencias, se construyen los parámetros, se construye el modelo, se entrega el modelo al solver, se encuentra una solución, se publica el resultado.

CAPÍTULO 4: ANALISIS DE RESULTADOS

Para comprobar la efectividad de la solución propuesta, se hará una comprobación de cumplimiento de requisitos, basado en las historias de usuario propuestas en el capítulo anterior.

4.1 Portal Estudiantes

A continuación, se muestran las imágenes 1 a 18, donde se puede apreciar que siguen los lineamientos propuestos en las historias de usuario HU01, HU02, correspondientes a los requisitos del usuario estudiante, además se aprovecha de utilizar este usuario para mostrar la realización de los requerimientos generales propuestos en las historias de usuario HU06, HU07 Y HU08.

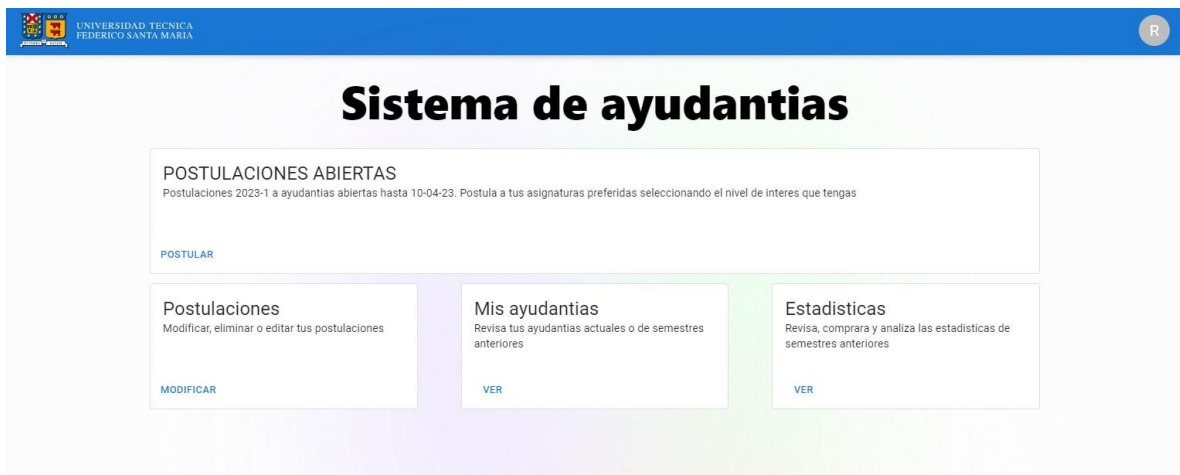


Imagen 1: Menú principal usuario estudiante.
Fuente: Elaboración propia.



Imagen 2: Menú fechas periodo.
Fuente: Elaboración propia.

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES

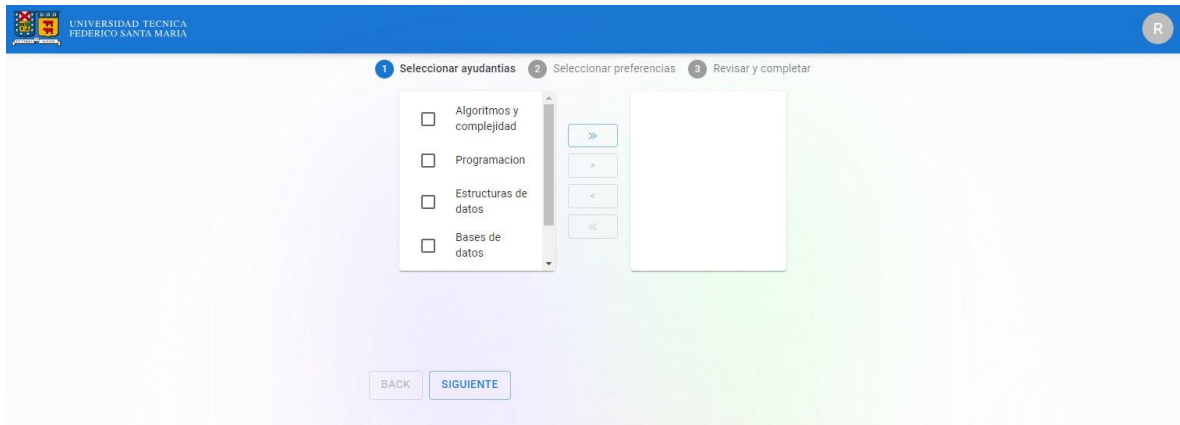


Imagen 3: Menú postulación, paso 1 selección asignaturas.
Fuente: Elaboración propia.

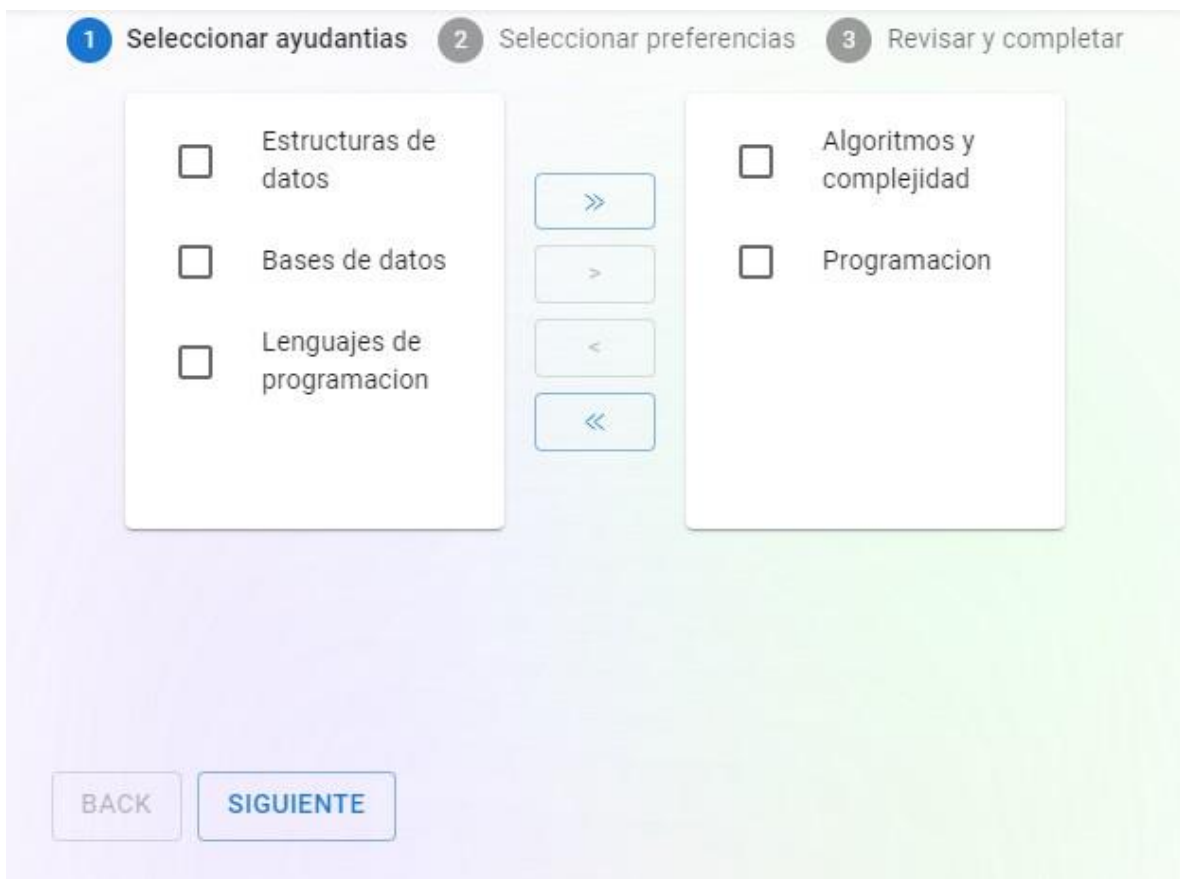


Imagen 4: Menú paso 1 asignaturas seleccionadas.
Fuente: Elaboración propia.

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES

The screenshot shows the application interface for the selection of assistants. The header includes the logo of Universidad Técnica Federico Santa María and a user profile icon. The main content area displays a progress bar with three steps: 1. Seleccionar ayudantias (checked), 2. Seleccionar preferencias (active), and 3. Revisar y completar. Below the progress bar, there is a table with two columns: 'Asignatura (Ayudantia)' and 'Preferencia'. The table has two rows: 'Algoritmos y complejidad' and 'Programacion'. Each row has three radio buttons for 'Alta', 'Media', and 'Baja'. At the bottom, there are 'BACK' and 'SIGUIENTE' buttons.

Asignatura (Ayudantia)	Preferencia
Algoritmos y complejidad	☐ Alta ☐ Media ☐ Baja
Programacion	☐ Alta ☐ Media ☐ Baja

BACK SIGUIENTE

Imagen 5: Menú postulación paso 2, selección preferencia
Fuente: Elaboración propia

The screenshot shows the application interface for the selection of assistants. The header includes the logo of Universidad Técnica Federico Santa María and a user profile icon. The main content area displays a progress bar with three steps: 1. Seleccionar ayudantias (checked), 2. Seleccionar preferencias (checked), and 3. Revisar y completar (active). Below the progress bar, there is a table with two columns: 'Asignatura (Ayudantia)' and 'Preferencia'. The table has two rows: 'Algoritmos y complejidad' and 'Programacion'. Each row has three radio buttons for 'Alta', 'Media', and 'Baja'. The 'Alta' radio button for 'Algoritmos y complejidad' and the 'Media' radio button for 'Programacion' are selected. At the bottom, there are 'BACK' and 'SIGUIENTE' buttons.

Asignatura (Ayudantia)	Preferencia
Algoritmos y complejidad	☒ Alta ☐ Media ☐ Baja
Programacion	☐ Alta ☒ Media ☐ Baja

BACK SIGUIENTE

Imagen 6: Menú postulación paso 2, preferencia elegida.
Fuente: Elaboración propia

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

✓ Seleccionar ayudantías — ✓ Seleccionar preferencias — 3 Revisar y completar

Asignatura (Ayudantía)	Preferencia
Algoritmos y complejidad	Alta
Programacion	Media

BACK TERMINAR

Imagen 7: Menú postulación paso 3, revisar y completar.
Fuente: Elaboración propia

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

✓ Seleccionar ayudantías — ✓ Seleccionar preferencias — ✓ Revisar y completar

POSTULACION REALIZADA

SALIR REALIZAR OTRA POSTULACION

Imagen 8: Menú postulación paso 4, retroalimentación postulación.
Fuente: Elaboración propia

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

Resumen Postulaciones

Asignatura	Preferencia	
Programacion	Baja	EDITAR
Estructuras de datos	Alta	EDITAR
Lenguajes de programacion	Media	EDITAR
Computacion Cientifica	Media	EDITAR

VOLVER

Imagen 9: Menú edición postulaciones.
Fuente: Elaboración propia

Resumen Postulaciones

Asignatura

Preferencia

Editar preferencia

Programacion

☐ Alta ☒ Media ☐ Baja

CONFIRMAR

Computacion Cientifica

Media

EDITAR

VOLVER

Imagen 10: Menú edición postulaciones, modal edición asignatura.
Fuente: Elaboración propia

UNIVERSIDAD TÉCNICA
FEDERICO SANTA MARÍA

R

Resultados

Curso (Ayudantía)	Minimos	Maximos	Seleccionados
Programacion	5	12	10
Bases de datos	2	4	4
Informatica teorica	1	2	2
Estructuras de datos	1	3	3
Ingenieria de software	1	5	5

Imagen 11: Resultado postulaciones, vista general todas las asignaturas.
Fuente: Elaboración propia

Resultados

	Curso (Ayudantia)	Minimos	Maximos	Seleccionados
▼	Programacion	5	12	10
▼	Bases de datos	2	4	4
▼	Informatica teorica	1	2	2
▲	Estructuras de datos	1	3	3
Seleccionados				
	Nombre	Rol	Email	
	Jose Montana	202074560-8	jose.montana@usm.cl	
	Miguel Guerrero	202174512-9	miguel.guerrero@usm.cl	
	Carla Gomez	202074580-1	carla.gomez@usm.cl	
▼	Ingenieria de software	1	5	5

Imagen 12: Resultado postulaciones, vista expandida con detalle asignatura.
Fuente: Elaboración propia

 UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA		J	
Historial			
2023-2	Programacion	Aceptado	▲
2023-2	Estructuras de datos	Rechazado	
2023-2	Lenguajes de programacion	Rechazado	
2023-2	Computacion Cientifica	Rechazado	
2023-1	Estructuras de datos	Rechazado	
2023-1	Estructuras discretas	Aceptado	
2023-1	Programacion	Aceptado	▼
VOLVER			

Imagen 13: Menú mis ayudantías, historial de postulaciones.
Fuente: Elaboración propia

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES



Imagen 14: Menú estadísticas generales.
Fuente: Elaboración propia



Imagen 15: Menú estadísticas asignaturas
Fuente: Elaboración propia

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES



Imagen 16: Menú estadísticas personales.
Fuente: Elaboración propia

ESTUDIANTE
JOSE EDUARDO
MONTANA SALAZAR

EMAIL
jose.montana@usm.cl

ROL
20235689-7

DEPARTAMENTO
Informatica

CAMPUS
San Joaquín

TELEFONO
+56955684429

DIRECCION
Ray Prairie Apt. 87870

CIUDAD
SANTIAGO

REGION
METROPOLITANA

DESCRIPCION
Descripción

EDITAR DESCRIPCION

Imagen 17: Menú perfil.
Fuente: Elaboración propia

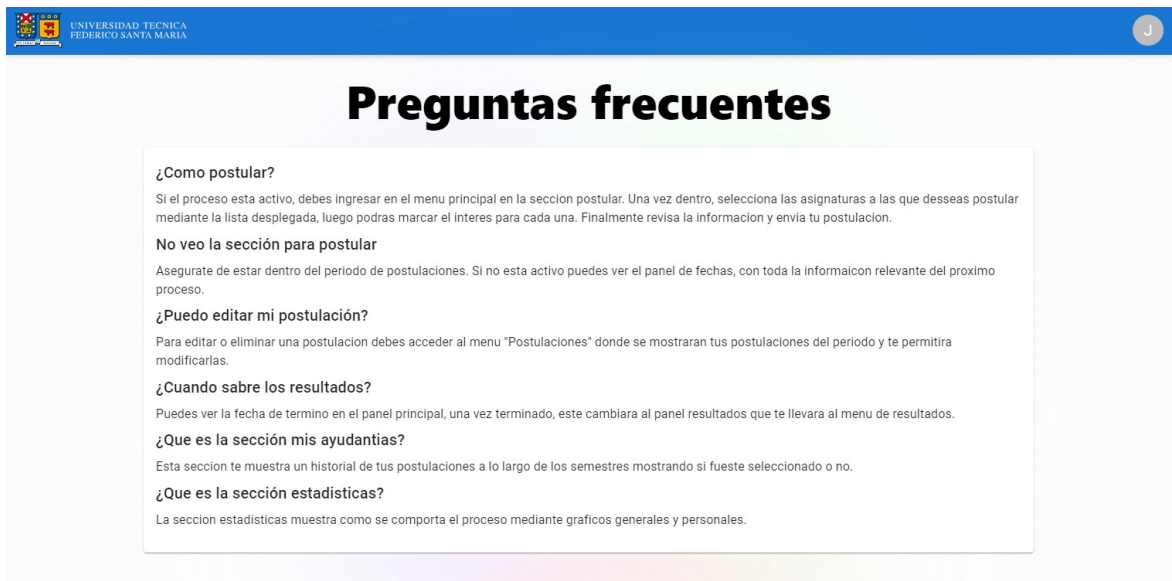


Imagen 18: Menú preguntas frecuentes alumno.

Fuente: Elaboración propia

4.2 Portal Profesores

Al igual que en la sección anterior, se presentan en las imágenes 19 a 27, el resultado de los requisitos propuestos en las historias de usuario HU03, HU04 Y HU08.

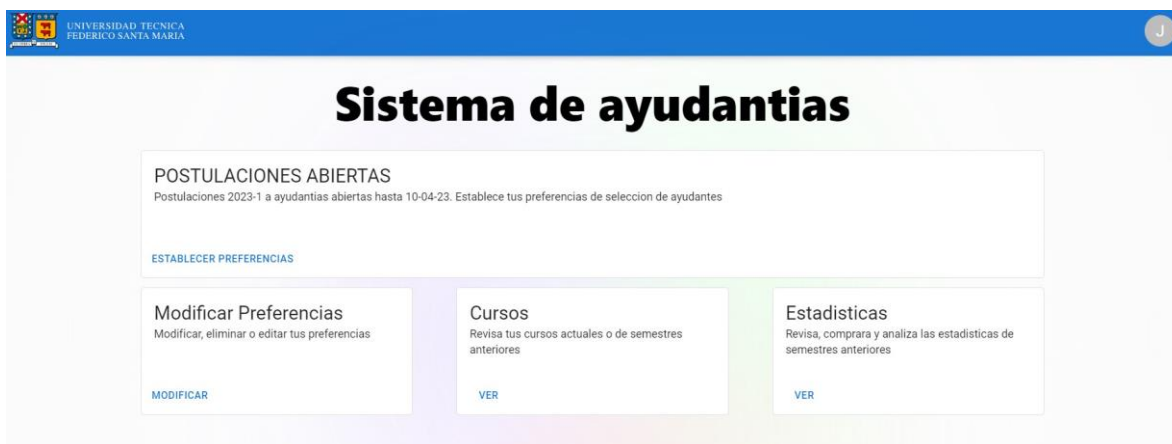


Imagen 19: Menú principal profesores

Fuente: Elaboración propia



Imagen 20: Menú especificación de preferencias profesor
Fuente: Elaboración propia

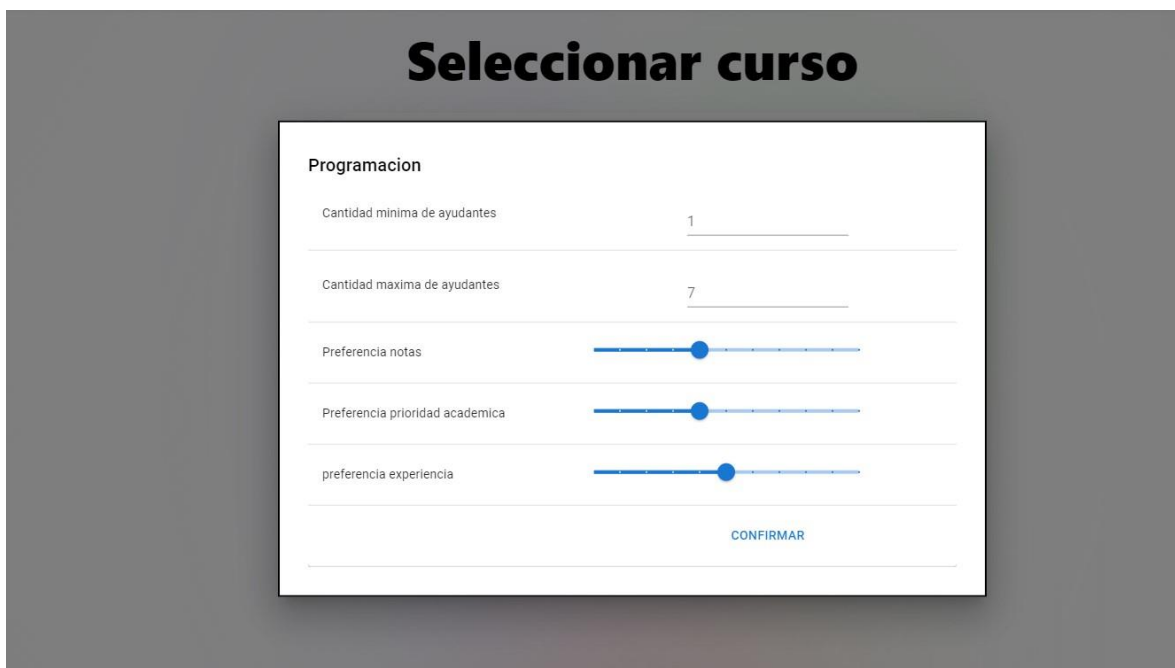


Imagen 21: Menú modal especificación preferencias asignatura
Fuente: Elaboración propia

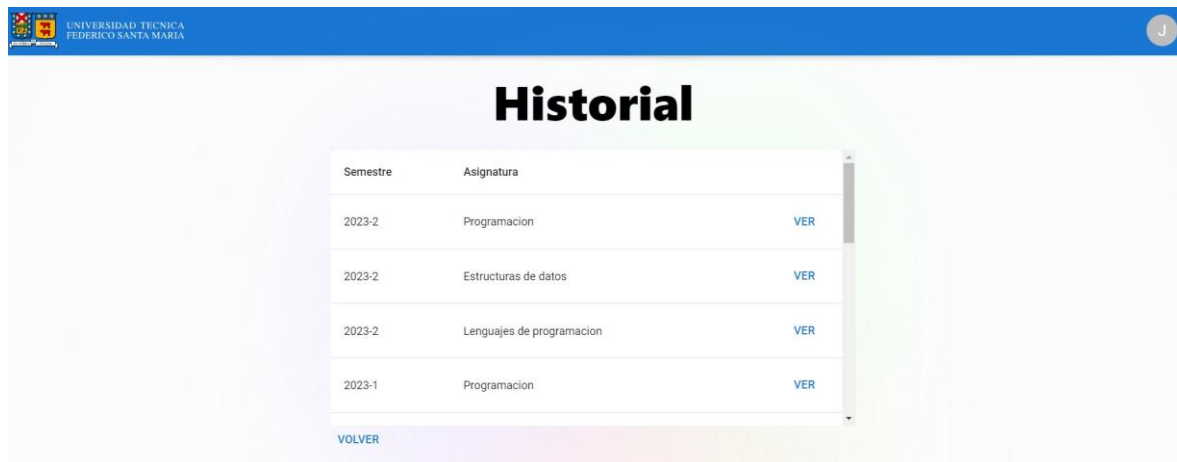


Imagen 22: Menú mis cursos, historial de ayudantías
Fuente: Elaboración propia

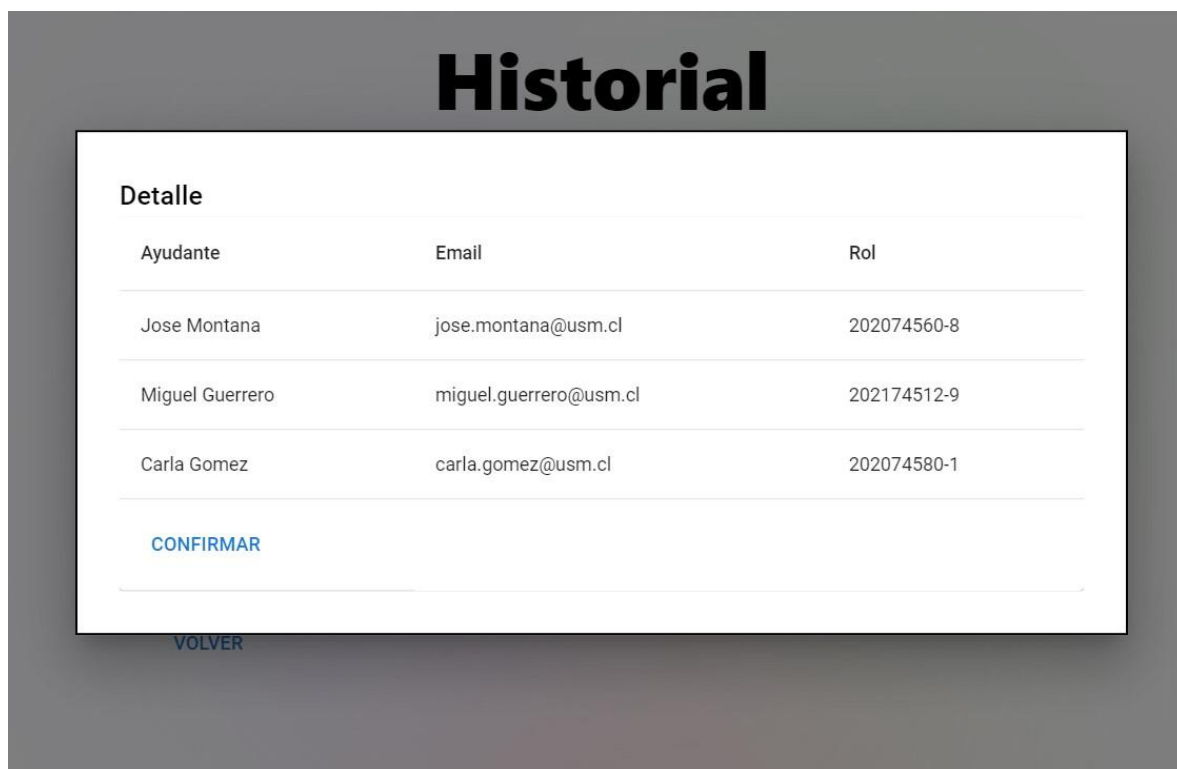


Imagen 23: Menú modal historial asignatura.
Fuente: Elaboración propia

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES



Imagen 24: Menú estadísticas generales profesor
Fuente: Elaboración propia



Imagen 25: Menú estadísticas asignaturas profesor
Fuente: Elaboración propia

DESARROLLO DE UNA PLATAFORMA DE SELECCIÓN DE AYUDANTES UTILIZANDO MODELOS BASADOS EN RESTRICCIONES



Imagen 26: Menú estadísticas sobre ayudantes seleccionados profesor.
Fuente: Elaboración propia

Preguntas frecuentes

¿Como configuro mis preferencias de selección?
Si el proceso esta activo, debes ingresar en el menu principal en la seccion Establecer preferencias. Una vez dentro, se desplegaran las asignaturas de las cuales eres coordinador y tendras la opcion de ingresar las preferencias mediante el boton Preferencias.

No veo la sección Establecer preferencias
Asegurate de estar dentro del periodo de postulaciones. Si no esta activo puedes ver el panel de fechas, con toda la informacion relevante del proximo proceso.

¿Puedo editar mis preferencias?
Para editar o eliminar las preferencias debes acceder al menu "Modificar preferencias" donde se mostraran las asignaturas con preferencias seteadas.

¿Que pasa si no selecciona mis preferencias?
Si no estableciste preferencias, se designaran valores por defecto a tus asignaturas.

¿Cuando sabre los resultados?
Puedes ver la fecha de termino en el panel principal, una vez terminado, este cambiara al panel resultados que te llevara al menu de resultados.

¿Que es la sección mis cursos?
Esta seccion te muestra un historial de tus asignaturas a lo largo de los semestres mostrando los ayudantes seleccionados.

¿Que es la sección estadísticas?
La seccion estadísticas muestra como se comporta el proceso mediante graficos generales y personales.

Imagen 27: Menú informaciones y preguntas frecuentes profesor.
Fuente: Elaboración propia

4.3 Portal Administrador

Finalmente, en las imágenes 28 a 42, se encuentran los resultados al seguir lo estipulado por las historias de usuario HU06 Y HU07.

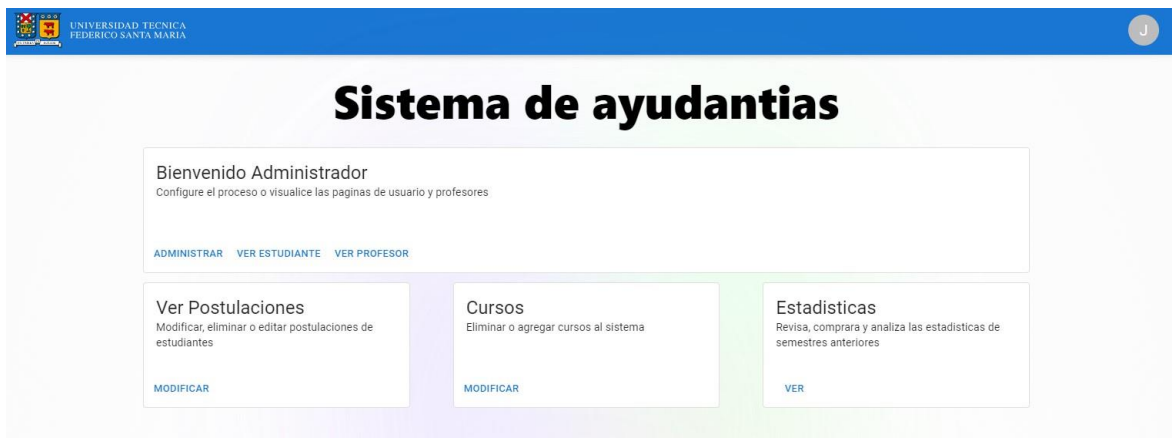


Imagen 28: Menú principal administrador
Fuente: Elaboración propia

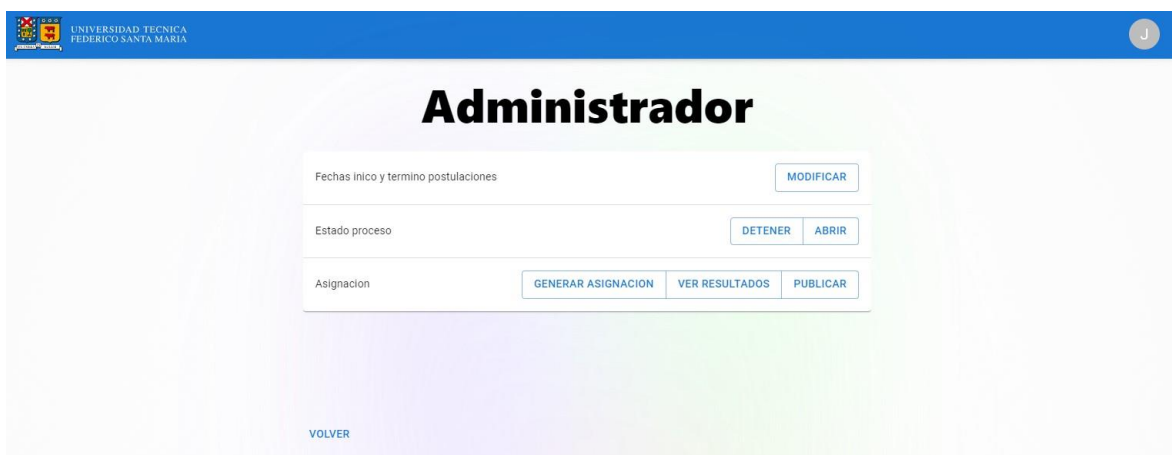


Imagen 29: Menú de control del proceso administrador.
Fuente: Elaboración propia



Imagen 30: Menú administración postulaciones.
Fuente: Elaboración propia



Imagen 31: Menú administración postulaciones, detalle y funciones.
Fuente: Elaboración propia



Imagen 32: Menú modal edición preferencia alumno.
Fuente: Elaboración propia

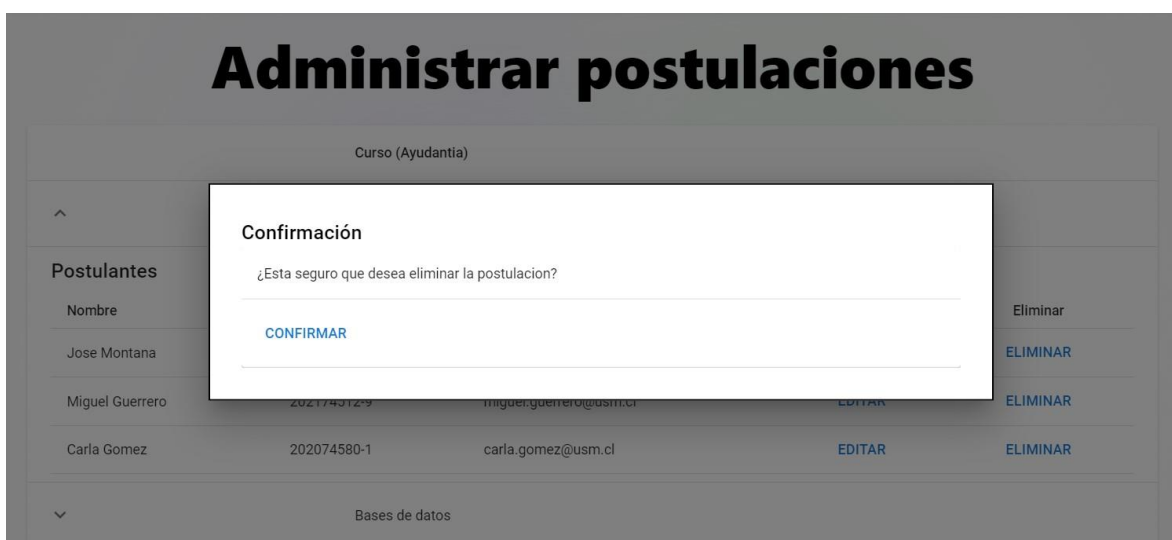


Imagen 33: Menú modal confirmación eliminación postulación.
Fuente: Elaboración propia

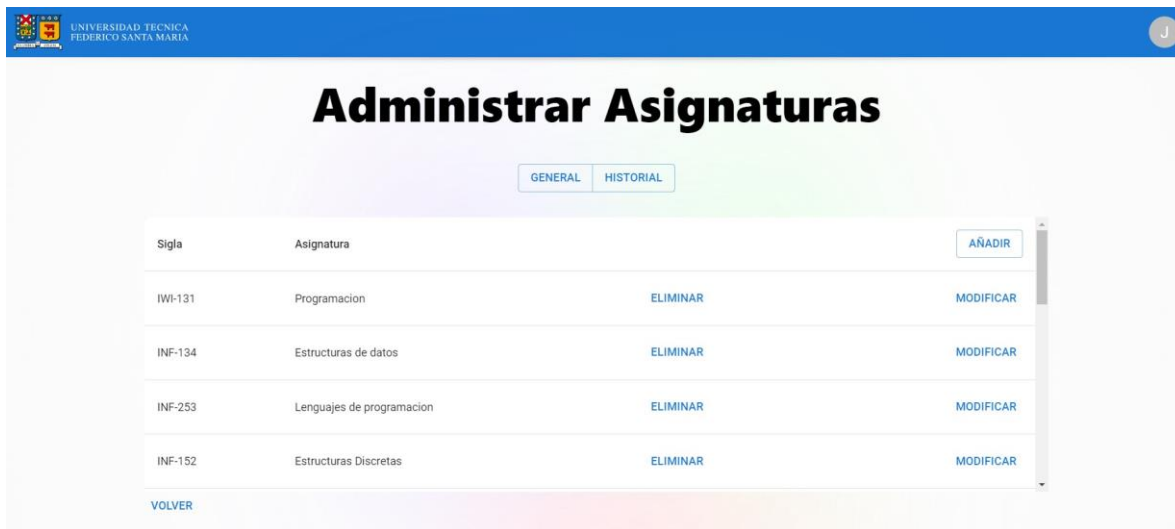


Imagen 34: Menú administración asignaturas.
Fuente: Elaboración propia

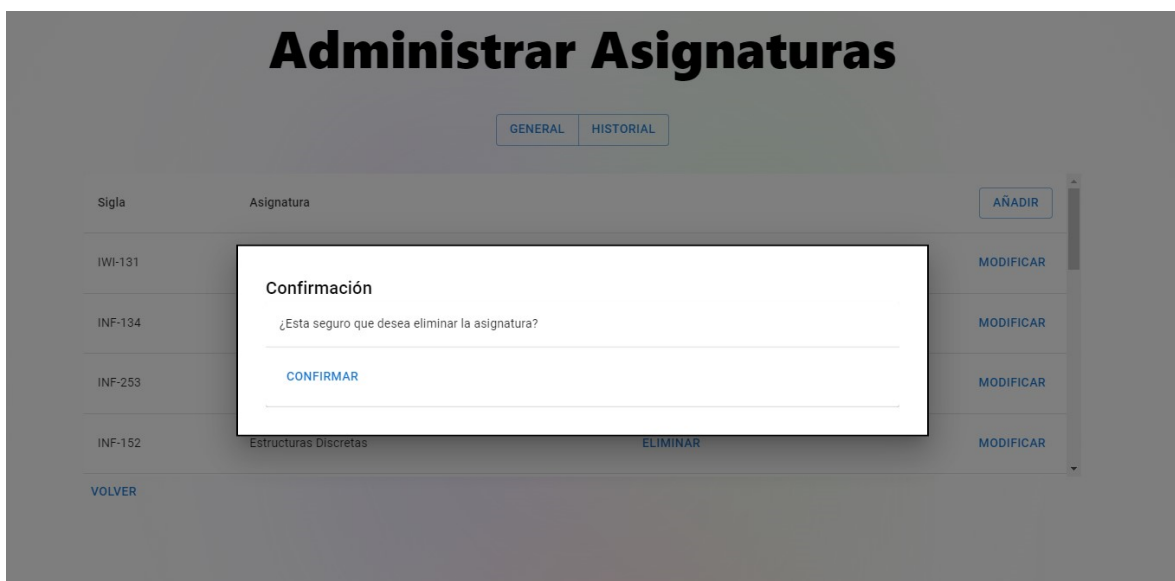


Imagen 35: Menú modal confirmación eliminación asignatura.
Fuente: Elaboración propia

The screenshot shows a modal titled "Administrar Asignaturas" with a "Detalle" section. It contains several input fields and sliders. On the left, there is a sidebar with a list of subjects (Sigla) and a "VOLVER" button. On the right, there are buttons for "AÑADIR" and "MODIFICAR".

Detalle	
Nombre	Programacion
Sigla	IWI-131
Cantidad minima de ayudantes	1
Cantidad maxima de ayudantes	5
Preferencia notas	<input type="range"/>
Preferencia prioridad academica	<input type="range"/>
preferencia experiencia	<input type="range"/>

Imagen 36: Menú modal edición asignatura.
Fuente: Elaboración propia

The screenshot shows a modal titled "Administrar Asignaturas" with tabs for "GENERAL" and "HISTORIAL". It contains a table with columns "Sigla" and "Asignatura". A modal titled "Agregar asignatura" is open, showing input fields for "Nombre" and "Sigla". On the right, there are buttons for "AÑADIR" and "MODIFICAR".

Sigla	Asignatura
IWI-131	
INF-134	
INF-253	
INF-152	

Imagen 37: Menú modal adición de asignatura.
Fuente: Elaboración propia

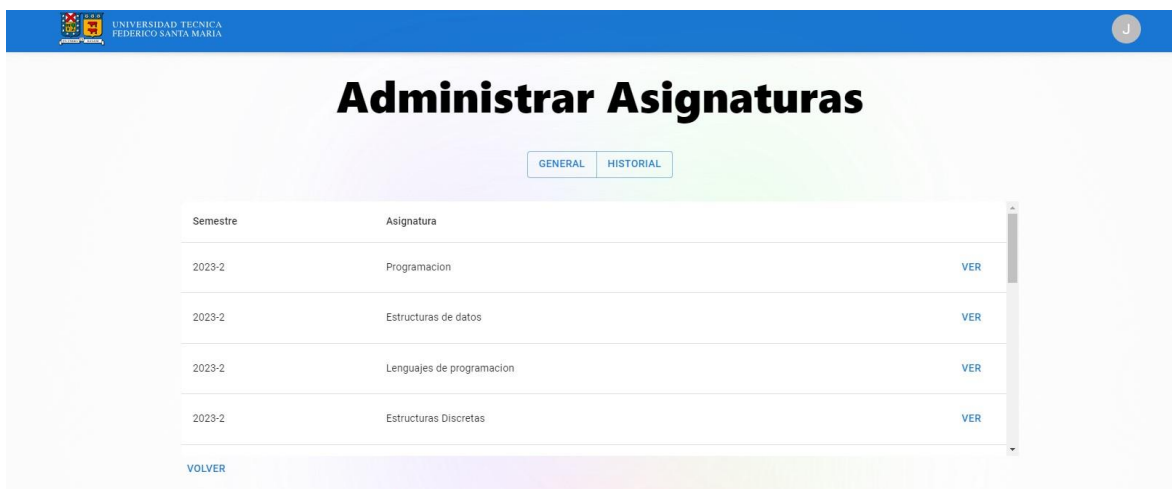


Imagen 38: Menú historial asignaturas y ayudantes asignados.
Fuente: Elaboración propia

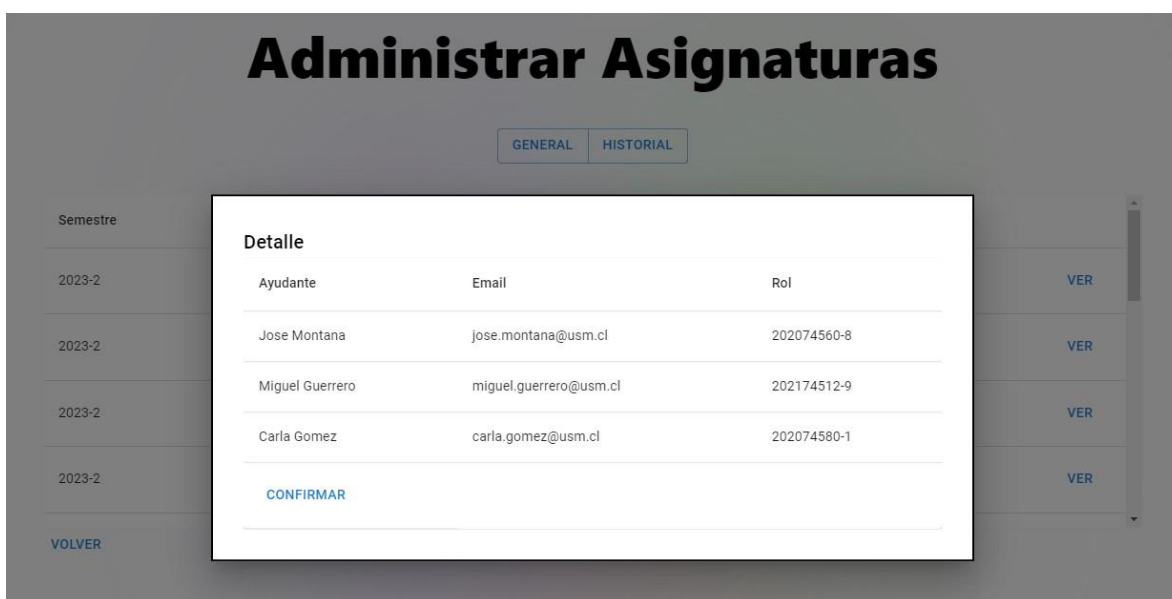


Imagen 39: Menú modal detalle histórico ayudantes seleccionados.
Fuente: Elaboración propia



Imagen 40: Menú estadísticas generales administrador.
Fuente: Elaboración propia



Imagen 41: Menú estadísticas asignaturas administrador.
Fuente: Elaboración propia

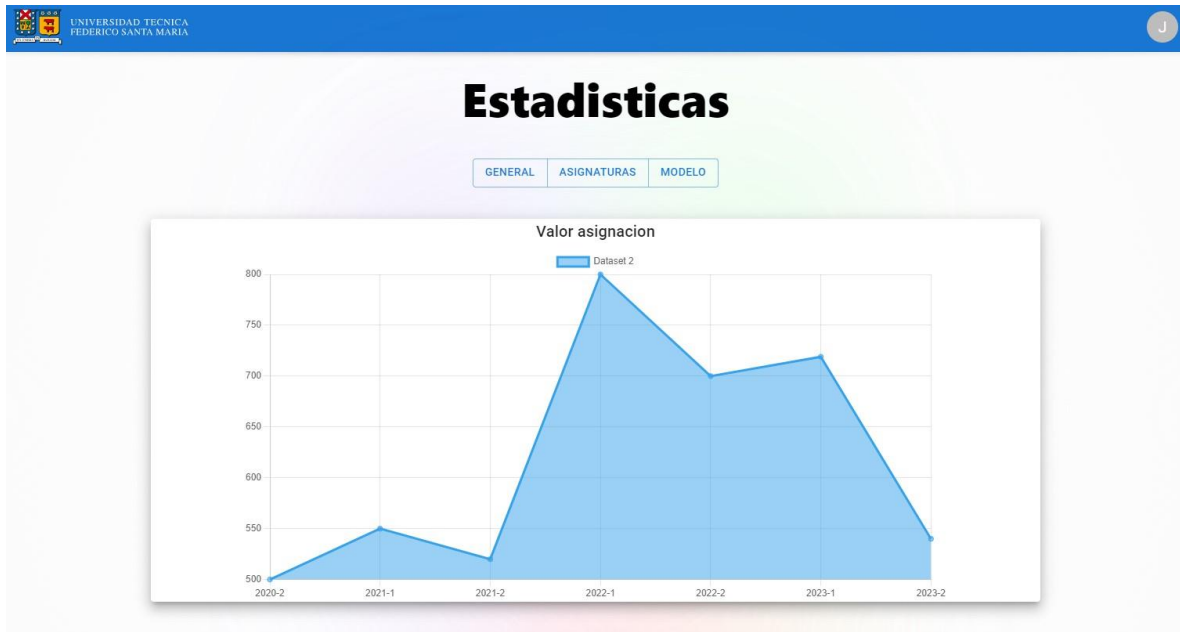


Imagen 42: Menú estadísticas valor asignación.
Fuente: Elaboración propia

CAPÍTULO 5: CONCLUSIONES Y TRABAJO FUTURO

5.1 Conclusiones

De forma general, podemos concluir que se logró el objetivo de construir una plataforma que apoye el proceso de selección de ayudantes. Tal como vimos en el capítulo 4, se consiguió componer una nueva plataforma que integre un modelo de resolución de restricciones aplicado a nuestro caso dentro de un sistema informático dirigido al usuario.

De forma particular tenemos:

- Se logró incluir los parámetros de preferencias de profesores y estudiantes dentro del modelo, siendo sus niveles de importancia ponderados por el mismo usuario según su nivel de interés o requisitos para ser seleccionado, lo que proporciona un nexo entre las preferencias subjetivas y el modelo matemático alcanzando mejores asignaciones.
- Se incluyeron métricas y estadísticas que permiten revisar la efectividad de las asignaciones, entendiendo si un proceso fue exitoso, o dilucidando fallas, y entregando una vista global del estado de ayudantes y asignaturas permitiendo así agregar correcciones en procesos subsiguientes.
- Se ha comprobado que el modelo entrega buenas soluciones y es más eficiente que una asignación manual, sobre todo en casos de muchas variables, sin embargo, este no puede solucionar problemas ajenos a él, como, por ejemplo, que una asignatura no reciba postulantes.
- El diseño del sistema cumple con los requisitos propuestos mediante las historias de usuario y les entregan a los usufructuarios, componentes que ayudan a la navegación de la página, componentes informativos del cómo y para que usarla, controles de entrada que permiten introducir información al sistema por parte de los usuarios para interactuar con él y este posee contenedores que mantienen el contenido organizado.

5.1.1 Diseño e implementación

El punto visible de la solución viene dado por el desarrollo del sistema web presentado en el capítulo 4. El sistema logra cumplir con los requisitos propuestos por las historias de usuario logrando así el objetivo principal de este texto.

El desarrollo web con tecnologías modernas facilita la producción gracias a la modalidad de trabajo con componentes, los cuales pueden ser adquiridos mediante librerías y modificados según nuestras propias necesidades, decreciendo el tiempo total invertido en la creación de los mismo desde cero. Por ello confiamos en que utilizar el stack MERN fue una buena decisión.

El incorporar métricas mediante visualizaciones gráficas, nos ayuda a entender mejor el pensamiento de los postulantes y profesores. Al mostrarnos las tendencias que se van generando nos permiten ir corrigiendo falencias a lo largo de los semestres.

Finalmente podemos decir que la plataforma es totalmente funcional para los efectos necesarios del proceso de selección de ayudantes, sin embargo, existe trabajo que puede ser agregado para darle una mayor comodidad, calidad de vida y usabilidad para los usuarios y resolución de conflictos administrativos o casos borde, los que se repasan en la sección 5.2.

5.1.2 Modelo

El modelo utilizado corresponde a los presentados en trabajos anteriores, donde se agregaron dos modificaciones principales:

1. la inclusión de parámetros de preferencias de profesores y estudiantes para cada postulación. Estos parámetros deben ser preprocesados para luego ser utilizados en el modelo como valores numéricos. El mayor desafío al plantear estos parámetros es el traspasar sentimientos subjetivos a valores numéricos, es por ello que aun dentro de todo el modelo matemático existe un componente de subjetividad humana. La agregación de estos nuevos parámetros trae consigo dos consecuencias, una asignación más acertada desde el punto de vista subjetivo al “calcular” al mejor postulante y no llanamente llenar una vacante, y un costo extra de cálculo al momento de realizar su construcción.
2. El mapeo de valores numéricos a integrantes de la comunidad y asignaturas, véase la variable x_{11} “alumno 1” corresponde a “Jhon Doe” postulando a la “asignatura 1” perteneciente a “IWI-131 Programación” es necesario quitar esta abstracción matemática para la correcta lectura de los resultados para los participantes puesto ellos no conocen el trasfondo técnico.

Por otro lado, tenemos diferentes maneras para dividir los recursos, es decir, como tramamos a una asignatura, ¿requiere cierta cantidad de horas? ¿cierta cantidad de estudiantes? O de manera invertida ¿Cuántas horas quiere dedicar un estudiante? ¿Para cuántas asignaturas quiere ser seleccionado?

En esta iteración se tomó la decisión de tratar cada asignatura como unitaria, es decir, si se asignó un estudiante, este debe cumplir con todo el tiempo destinado normalmente a ella y viceversa. Esto debido a que se deben tener en cuenta varias restricciones no matemáticas y ajenas al cálculo, como lo son las presentadas por el reglamento interno de la institución que indica un número máximo de horas o de ayudantías. Esto implico realizar algunas relajaciones al modelo o establecer valores predeterminados a ciertos parámetros. Debido a que estas restricciones pueden ser retiradas para permitirse en ocasiones especiales, fue necesario permitir a un agente administrador con facultades para realizar modificaciones manuales al modelo para lograr aprobar tales asignaciones.

Por último, podemos notar que, si bien tanto estudiantes como profesores tienen preferencias, las de profesores serán quien entreguen la ponderación decisiva, puesto que en un evento donde existan dos postulantes con preferencias idénticas, estos serán medidos según las preferencias de profesores (experiencia, nota asignatura y prioridad académica) y asignados según quien tenga mejor desempeño dada las preferencias del profesor, cabe resaltar que esto significa que todos los postulantes a dicha asignatura serán evaluados bajo los mismos criterios, por lo que un profesor solo debe ingresar sus preferencias una vez y no debe depender de que primero existan postulantes para luego sortear preferencias individuales.

5.2 Trabajo futuro

En este texto hemos mostrado la implementación de las funcionalidades necesarias para automatizar el proceso de postulaciones de ayudantías, sin embargo, existen incertidumbres frente a casos especiales, conflictos administrativos o funcionalidades no esenciales pero que podrían ser útiles, es por ello que se proponen posibles mejoras que agreguen valor al sistema.

5.2.1 Integración otros sistemas

El sistema está diseñado para funcionar de forma *standalone*, sin embargo, seria de muchísima utilidad, conectarlo con el resto de los sistemas de la universidad para evitar redundancia en funciones o datos.

- SIGA: Sistema de información de gestión académica, el centro principal datos de la universidad. Necesario para la recolección de información de todos los usuarios de la plataforma.
- Aula virtual: Uno de los requisitos para ser ayudantes es haber realizado el curso de ayudantes, el cual se realiza a través del aula virtual USM, por lo que es un factor clave y prerrequisito para hacer una postulación. Además de ser el centro virtual de comunicación y almacenamiento de material académico, al conectar los sistemas idealmente agregar a los ayudantes seleccionados de forma automática a los cursos correspondientes.

5.2.3 Funcionalidades extra

Uno de los problemas más comunes que podemos encontrar es la baja postulación a ciertas asignaturas es por ello que se proponen a futuro los siguientes planes de mitigación:

- Promoción de asignaturas con bajas postulaciones: Gracias a las estadísticas ahora tenemos un método de ver las materias más y menos solicitadas en procesos anteriores, podemos aplicar esta métrica al proceso en curso, y así identificar aquellos ramos que no están recibiendo atención para moverlos al principio de la lista y promoverlos pudiendo generar nuevos interesados.

- **Redistribución de asignados:** Si tomamos en consideración el plan de estudios, notamos que existe una progresión natural en ciertas áreas o seguidilla de asignaturas, por lo que, si se diera el caso en que una asignatura avanzada obtiene muchas postulaciones y los prerrequisitos menos o ninguna, se puede proponer una reasignación y buscar interesados, siguiendo la lógica de que un postulante a una asignatura avanzada, domina sus prerrequisitos. Por ejemplo en la imagen 56 se podría dar el caso que ingeniería de software tiene un máximo de 5 ayudantes y ya tiene más de 20 postulaciones, por lo que es altamente probable que muchos no queden seleccionados, y si se diera el caso que análisis y diseño de software solo tiene 3 postulantes de un mínimo de 5, se crea una notificación para los postulantes de ingeniería de software indicando que existen mayores posibilidades de asignación en una materia de la misma área en la cual están capacitados.

SEMESTRE V			SEMESTRE VI			SEMESTRE VII			SEMESTRE VIII		
INF-239 24			INF-236 30			INF-225 36			INF-322 42		
Bases de Datos			Análisis y Diseño de Software			Ingeniería de Software			Diseño Interfaces Usuarios		
12	3	5	24 18	3	5	30	3	5	36	3	5

Figura 14: Extracto progresión plan de estudios.

Fuente: (María, 2023)

- **Notificación de intereses:** En el caso de que, una vez finalizado el proceso, no se haya logrado captar la atención, dado que muchos no querrán arriesgarse a ser asignados a una preferencia menor, pero aun así quieren ser ayudantes, se puede enviar una notificación informativa para llenar ahora los cupos faltantes.
- **Tipo de ayudantía:** En general existen dos tipos de ayudantes, de catedra y revisión/tareas, el primero encargado de repasar conceptos y realizar ejercicios en horas extras fuera de las clases y el segundo en generar tareas y calificarlas. Agregar a que posición tiene intenciones de ejercer puede ayudar a evitar posteriores desconformidades al crear la coordinación de ayudantes.
- **Resolución de conflictos administrativos:** Como se ha mencionado, existen algunos conflictos administrativos que restringen al modelo en primera instancia y deben ser abordados mediante ajustes manuales. Así se propone la inclusión de una etapa extra previa al lanzamiento de los resultados, en caso de ser necesario la aprobación de estudiantes con más de una ayudantía para lograr cubrir la totalidad de cupos,

así una vez resuelto realizar un recalcu de la asignación y la final publicación de resultados.

5.2.4 Modificación de parámetros

En este trabajo se presentaron las preferencias de los profesores como una triada de parámetros los cuales son ponderados para crear un valor numérico final, estos corresponden a la nota en la asignatura, la prioridad académica y la experiencia, donde este último es simplemente el número de ayudantías que ha realizado hasta el momento.

Si nos enfocamos en el parámetro de la experiencia podemos hacer un análisis más exhaustivo. Hasta ahora su importancia es lineal, a medida más ayudantías uno ha realizado más grande su valor y por ende su importancia, sin embargo, uno podría alegar que la experiencia es un sistema complejo formado por más factores que la reiteración, por ejemplo, el nivel de la ayudantía (que podemos medir según los créditos de la asignatura, el semestre al que pertenece y los requisitos que necesita), la diversidad o especialización de un área del conocimiento (¿he dictado asignaturas que siguen un orden lógico o soy un generalista que maneja las bases de todo?). Esto lo podemos traducir en una función, al igual que los tres parámetros iniciales, donde ponderamos cada uno según la importancia que creemos estos tienen.

además, se podría argumentar que cada ayudantía extra que uno realice no entrega el mismo conocimiento que la anterior, haciendo que cada una añada menos experiencia, pues mucho de ello ya se ha aprendido. Por lo que se puede cambiar su linealidad por otra función que vaya disminuyendo el incremento entregado a medida crece en número de ayudantías.

Por otro lado, existen otras cualidades que podemos tomar como parámetros, entre ellas tenemos:

- Numéricas: Son directas y nos entregan inmediatamente un valor numérico para ser ponderado dentro del modelo.
 - I. Promedio: Existen diferentes formas de tomar el promedio, ponderado o solo de ramos aprobados. Lo que ayuda a medir un rendimiento promedio.
 - II. Créditos: Al igual que con promedios podemos jugar con créditos aprobados versus créditos totales. Lo que nos ayuda a medir el nivel de avance dentro de la carrera.
- Conceptuales: No son directas y requieren un paso extra de conversión a número para ser utilizadas por el modelo, lo que puede significar un costo mayor.
 - I. Plan de carrera: ¿Me interesa que quienes dicten la ayudantía sean aquellos en más contacto con el área a desempeñar? Por ejemplo, para Química y Sociedad (Asignatura común para todas las ingenierías) ¿prefiero a un estudiante de ingeniería civil química?

- II. Prerrequisitos: ¿Me interesa o es de interés que el ayudante haya dictado ayudantías de ramos prerrequisitos?
- III. Factor extracurricular: ¿Prefiero una persona con menos distracciones o es esto un beneficio?

En cuanto a las preferencias de estudiantes, se utiliza un sistema simple donde solo se entrega un nivel conceptual (baja, media, alta) de interés por cada asignatura. Una alternativa es una vez seleccionadas las materias a postular, entre estas crear un orden descendiente de mayor a menor determinado por el postulante, en donde la primera asignatura tiene una mayor preferencia y viceversa.

5.2.5 Casos particulares

Existen casos excepcionales donde el trabajo del modelo no tiene facultades, pues su causa se escapa al alcance del problema, entre ellos tenemos:

- No hay postulantes, es decir, una asignatura no recibió postulaciones de ningún alumno. Para el solver es imposible asignar si no hay recursos por lo que este problema no puede ser resuelto por el mismo.
- Pre-Acuerdo: Se puede dar el caso en que un alumno y profesor estén de acuerdo en trabajar juntos previo al proceso, pero el modelo es ciego frente a esta preferencia, por lo que no se puede asegurar su asignación final según el acuerdo dado.

La solución actual para estos problemas es hacer una selección a mano posterior a las asignaciones entregadas por el modelo.

REFERENCIAS BIBLIOGRÁFICAS

- Barber, F., & Salido, M. (2003). *Introducción a la Programación de Restricciones*. Valencia, España: Revista Iberoamericana de Inteligencia Artificial.
- Berkelaar, M. (s.f.). *Introduction to lp_solve*. (M. I. Technology, Productor) Recuperado el 10 de Diciembre de 2022, de <http://web.mit.edu/lpsolve/doc/>
- Cattrysse, D., & Van Wassenhove, L. (1990). *A survey of algorithms for the generalized assignment problem*. North-Holland: European Journal of Operational Research.
- Chauhan, A. (2019). *A Review on Various Aspects of MongoDB*. Faridabad, Haryana, India: International Journal of Engineering Research & Technology (IJERT).
- Gao, J. (2014). *Branch and bound (BB)*.
- Lindo systems. (s.f.). *LINGO 20 - Optimization Modeling Software for Linear, Nonlinear, and Integer Programming*. Recuperado el 10 de Diciembre de 2022, de <https://www.lindo.com/index.php/products/lingo-and-optimization-modeling>
- María, U. T. (2023). *Departamento de Informática USM*. Obtenido de <https://www.inf.ut fsm.cl/estudiantes/pregrado>
- Marino García, J. I. (2020). *Estudio de criterios y automatización del proceso de asignación de ayudantes en el departamento de informática*.
- Nauss, R. (2003). *Solving the Generalized Assignment Problem: An Optimizing and Heuristic Approach*. St. Louis, Missouri: Informs Journal on Computing.
- Shaw, P., Furnon, V., & De Backer, B. (s.f.). *A CONSTRAINT PROGRAMMING TOOLKIT FOR LOCAL SEARCH*. France.
- syntonize. (16 de 07 de 2023). *syntonize*. Obtenido de <https://www.syntonize.com/stack-tecnologico-que-es-y-como-crear-uno/>