



Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

(Mención computación)

Desarrollo y planificación de una aplicación de gestión de turnos de trabajo

Autor:

D. Diego Farto González



Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

(Mención computación)

Desarrollo y planificación de una aplicación de gestión de turnos de trabajo

Autor:

D. Diego Farto González

Tutor:

D. Javier Bastida Ibáñez

Quería agradecer a mi tutor Javier Bastida por su conocimiento, orientación y seguimiento durante la realización del proyecto.

Así mismo, agradecer también a mi familia todo el apoyo que me han dado. En especial, a mi madre y pareja, pues siempre ha sido una gran fuente de inspiración y motivación para continuar.

RESUMEN

En este proyecto describiremos todas las etapas que hemos llevado a cabo para desarrollar una aplicación que permita la elaboración de los turnos de trabajo en la empresa de autobuses urbanos de Valladolid (AUVASA).

Dicha aplicación necesita partir de un algoritmo adecuado que se encargue de la elaboración de los calendarios laborales. Por ello, hemos dividido este trabajo en dos partes: una primera, dedicada a la creación de dicho algoritmo y una segunda parte, correspondiente a la implementación de una interfaz visual web que la aloje y permita su utilización.

Hemos programado la aplicación en lenguaje Python, que en la actualidad es uno de los lenguajes de programación más utilizados, debido a su facilidad de uso, potencia, robustez y al gran número de librerías útiles de las que dispone.

ABSTRACT

In this project we will describe all the stages we have carried out to develop an application that allows the elaboration of work shifts in Valladolid's company of city buses (AUVASA).

Such application needs to start from an adequate algorithm that is in charge of the elaboration of work schedules. For this reason, we have divided this work into two parts: first, dedicated to the creation of such algorithm and a second part, corresponding to the implementation of a visual web interface that hosts it and allows its use.

We have programmed the application in Python language, which is currently one of the most used programming languages, due to it is easy to use, powerful, robust and it has a large number of useful libraries.

Índice

1. Introducción y objetivos	9
2. Plan de desarrollo del proyecto	10
2.1. Estudio previo de mercado	10
2.2. Planificación	12
2.2.1. Requisitos	12
2.2.2. Normativa de la empresa	15
2.2.3. Infraestructura del proyecto	18
2.2.4. Actividades del proyecto	20
2.2.5. Esfuerzo de cada etapa	22
2.2.6. Riesgos	23
2.2.7. Diagrama de Gantt	27
2.3. Seguimiento del proyecto	28
3. Diseño de Software	30
3.1. Tecnología utilizada	30
3.2. Patrones de diseño	32
3.3. Modelo de dominio	33
3.4. Casos de uso	34
3.4.1. Diagrama de casos de uso	34
3.4.2. Descripción de los casos de uso	35
3.4.3. Diagramas de secuencia de los casos de uso	44
3.5. Diagrama de datos	63
3.6. Diagrama de paquetes	64
3.7. Bocetos de la aplicación	65
4. Diseño del algoritmo	83
4.1. Algoritmo de asignación de líneas	83
4.2. Algoritmo de creación de calendarios anuales	85
5. Implementación de la aplicación	88
5.1. Creación del modelo, algoritmo y la base de datos	88
5.2. Casos de uso (Iniciar y cerrar sesión)	88
5.3. Casos de uso del gestor	89
5.4. Casos de uso del conductor	96
6. Test realizados	98
6.1. Test realizados en la Fase 1 de la implementación	98
6.2. Test realizados en la Fase 2 de la implementación	98
6.3. Test realizados en la Fase 3 de la implementación	99
6.4. Test realizados en la Fase 4 de la implementación	99
7. Manuales	100
7.1. Manual de uso del Gestor	100
7.2. Manual de uso del Conductor	104
7.3. Manual de Instalación	105
7.3.1. Paso 1: Instalación de Python	105
7.3.2. Paso 2: Instalación de las librerías necesarias	105
7.3.3. Paso 3: Instalación y preparación de la Base de datos	106

7.3.4. Paso 4: Activar el servidor	107
8. Conclusiones y líneas futuras	108
8.1. Conclusiones	108
8.2. Líneas futuras	108
Bibliografía	109

Índice de figuras

1.	Ejemplo de aplicación para la organización de turnos (Beam).	10
2.	Códice: Ejemplo de gestora.	11
3.	Modelo P2P	18
4.	Modelo Cliente–Servidor.	18
5.	Modelo servidor de aplicaciones.	19
6.	Representación gráfica del método iterativo e incremental.	20
7.	Diagrama de Gantt inicial.	27
8.	Diagrama de Gantt final.	29
9.	Panel de control de XAMPP.	30
10.	Logo de Flask y jinja.	31
11.	Tablas lógicas de Numpy.	31
12.	Modelo de dominio.	33
13.	Diagrama de casos de uso.	34
14.	Diagrama de secuencia del inicio de sesión.	44
15.	Diagrama de secuencia del cierre de sesión.	45
16.	Diagrama de secuencia de crear una reclamación.	46
17.	Diagrama de secuencia de la visualización del calendario individual.	47
18.	Diagrama de secuencia de la visualización de reclamaciones.	48
19.	Diagrama de secuencia de eliminación de reclamaciones.	49
20.	Diagrama de secuencia de la modificación del calendario.	50
21.	Diagrama de secuencia de la búsqueda de conductores.	51
22.	Diagrama de secuencia de la selección de calendario.	52
23.	Diagrama de secuencia de la creación de calendarios.	53
24.	Diagrama de secuencia de la modificación de los conductores.	54
25.	Diagrama de secuencia de la eliminación de conductores.	55
26.	Diagrama de secuencia de la edición de conductores.	56
27.	Diagrama de secuencia de la adición de conductores.	57
28.	Diagrama de secuencia de la asignación de líneas.	58
29.	Diagrama de secuencia de la visualización de estadísticas globales del calendario.	59
30.	Diagrama de secuencia de guardar la demanda del calendario.	60
31.	Diagrama de secuencia de guardar los turnos de trabajo del calendario.	61
32.	Diagrama de secuencia de la modificación de opciones de creación.	62
33.	Diagrama de datos.	63
34.	Diagrama de paquetes de la aplicación.	64
35.	Menú de inicio de sesión.	65
36.	Error en el menú de inicio de sesión.	66
37.	Menú principal de los conductores.	67
38.	Pantalla de creación de una notificación.	68
39.	Pantalla al enviar con éxito una notificación.	68
40.	Pantalla al enviar sin éxito una notificación.	69
41.	Menú principal de los gestores de la empresa.	70
42.	Página donde se muestran las notificaciones realizadas por los conductores.	70
43.	Página donde se muestran los datos de creación de los calendarios.	71
44.	Página donde hay un error en los datos de creación de los calendarios.	71
45.	Página donde se está ejecutando el algoritmo de creación de los calendarios.	72
46.	Página de visualización de la demanda del calendario.	72
47.	Página de Búsqueda de un conductor.	73
48.	Página de visualización del calendario del conductor.	73

49.	Página de visualización de estadísticas del calendario.	74
50.	Página de modificación del calendario.	75
51.	Página de modificación del calendario con error en los datos.	75
52.	Página de modificación del calendario con el algoritmo en ejecución.	76
53.	Página de confirmación de asignación de las líneas.	76
54.	Página de menú de guardado.	77
55.	Página de ejemplo de guardado.	78
56.	Página de selección de calendario.	79
57.	Página de error en la selección de calendario.	79
58.	Página de modificación de conductores.	80
59.	Página de creación o edición de conductores.	81
60.	Página de edición de opciones del algoritmo.	82
61.	Ejemplo de fallo 2 en la asignación de líneas.	83
62.	Etapas del algoritmo de asignación de líneas.	84
63.	Inicio sesión.	88
64.	Pantalla principal.	89
65.	Ver notificaciones.	89
66.	Opciones de creación de calendario.	90
67.	Visualización de la demanda global.	90
68.	Buscar un conductor del calendario.	91
69.	Ver estadísticas del calendario.	91
70.	Modificar el calendario.	92
71.	Menú de guardado.	92
72.	Guardar el calendario de turnos.	93
73.	Seleccionar un calendario.	93
74.	Opciones de creación de los calendarios.	94
75.	Modificación de conductores.	94
76.	Añadir un nuevo conductor.	95
77.	Pantalla principal del conductor, donde se puede ver el calendario individual del conductor.	96
78.	Pantalla de creación de notificaciones por parte del conductor.	96
79.	Pantalla de creación de notificaciones por parte del conductor cuando hay un error.	97
80.	Pantalla de ejecución del algoritmo.	101
81.	Pantalla de asignación de líneas.	102
82.	Pantalla de visualización de los calendarios individuales.	102
83.	Pantalla de Guardado de la demanda.	103
84.	Pantalla de creación de notificaciones por parte del conductor cuando se envía un mensaje con éxito.	104
85.	Salida de python.	105
86.	Crear base de datos paso 1.	107
87.	Crear base de datos paso 2.	107

Índice de tablas

1.	Costes de cada etapa.	22
2.	Plantilla de registro de riesgos.	24
3.	Riesgo 1. Enfermedad del desarrollador.	24
4.	Riesgo 2. Incompatibilidad con otras tareas.	25
5.	Riesgo 3. Modificación de los requisitos.	25
6.	Riesgo 4. Problemas con el material de trabajo.	25
7.	Riesgo 5. Retrasos en las tareas.	26
8.	Riesgo 6. Errores de la aplicación.	26
9.	Riesgo 7. Problemas de comunicación.	26

1. Introducción y objetivos

El objetivo de este trabajo es desarrollar una aplicación con la que podamos generar los calendarios de los trabajadores de la empresa de autobuses urbanos de Valladolid (AUVASA) de forma semi-automática y respetando su normativa internas. Además, se debe poder modificar de forma parcial tanto un calendario generado, como uno preseleccionado por el usuario encargado de la gestión, así como ver algunos datos estadísticos del calendario y con ello poder aceptarlo o rechazarlo. Por último, un trabajador que acceda a dicha aplicación debe poder visualizar los turnos de trabajo que se le han asignado.

La finalidad de la aplicación es satisfacer la necesidad de la empresa de crear los calendarios laborales de los trabajadores de forma cómoda y rápida. Dicha aplicación debe ser web y en una primera aproximación en lengua castellana.

Hemos partido de un algoritmo que habrá que adaptarlo a las necesidades y normas internas de la empresa. Además, cabe destacar, que disponemos de total libertad tanto para la elección del lenguaje de programación, como de la arquitectura web.

Por otro lado, los objetivos a nivel personal tienen carácter formativo y de ampliación de la experiencia obtenida a lo largo de los estudios en algunos campos de la informática. Más en concreto, estos objetivos son los siguientes:

- Ampliar los conocimientos relacionados con aplicaciones web.
- Mejorar el uso de los lenguajes de programación elegidos.
- Aplicar y ampliar los conocimientos de metodologías para la creación de turnos.
- Empleo de los conocimientos de gestión de proyectos adquiridos durante la formación universitaria.

La memoria de este trabajo se divide principalmente en cuatro partes. En la primera de ellas, hemos planificado el desarrollo de la aplicación para que este trabajo tenga una mayor probabilidad de éxito cumpliendo los requisitos pedidos por la empresa. En la segunda parte, se expone el estudio de diseño de software de la aplicación a desarrollar y el funcionamiento del algoritmo. En la tercera parte, hemos procedido, tras la corrección ciertos errores surgidos, a la elaboración de un manual de uso de la aplicación. Finalmente, exponemos las conclusiones del trabajo realizado, así como futuras ampliaciones y mejoras de la aplicación. Al final de este documento se mostrará toda la bibliografía utilizada.

2. Plan de desarrollo del proyecto

En esta sección vamos a describir los requisitos de la aplicación y los pasos necesarios para llevarla a cabo, estudiando tanto sus actividades y etapas, como su duración, los riesgos que pueden surgir y la tecnología que vamos a utilizar.

2.1. Estudio previo de mercado

En el mercado existe un gran número tanto de empresas gestoras como de aplicaciones web que se ofrecen para solucionar los problemas de aquellas empresas que requieran sus servicios para la gestión de los turnos laborales de sus trabajadores garantizando encontrar los mejores calendarios de trabajo, es decir, los óptimos.

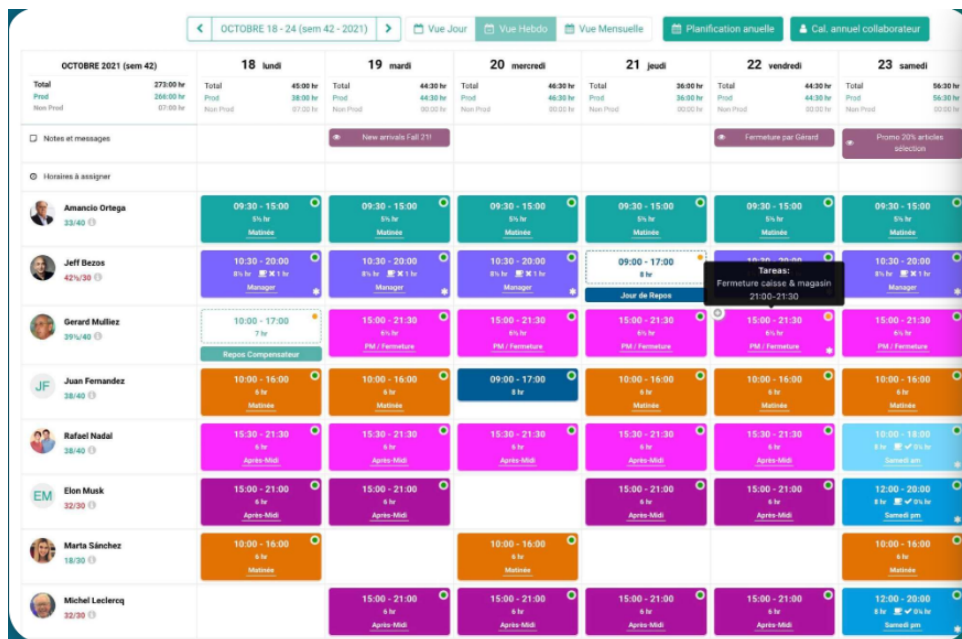


Figura 1: Ejemplo de aplicación para la organización de turnos (Beam).

La Figura 1 se corresponde con un ejemplo de la aplicación *Beam* cuya dirección web es: <https://es.beamhr.com/logiciel-gestion-planning> y se encarga de la distribución de turnos. Dicha aplicación recoge los ciclos o grupos en los que se encuentra cada uno de los empleados así como sus turnos de trabajo. Esta aplicación puede usarse de forma anual, pero su principal orientación es mostrar los turnos de forma semanal o mensual. Además, no genera un calendario que se ajusta a una demanda, sino a unos grupos o ciclos de trabajo. Estas son las principales razones por las que esta aplicación no sirve para afrontar el problema que nosotros nos planteamos, pero sí para darnos una idea de distintas visualizaciones que nos puedan ser útiles.

Otras gestoras y aplicaciones sí que buscan ajustarse a ciertos requisitos dados previamente para generar los turnos de trabajo, pero tienen un coste elevado y no obtienen un resultado que tenga en cuenta toda la normativa de la empresa. Además, en el caso de las gestoras, estas no suelen proporcionar ninguna aplicación o algoritmo con las particularidades de la propia empresa, y cuando los hacen se trata de programas que ofrecen calendarios más genéricos de turnos, no gratuitos y que es preciso renovar de forma periódica. Por otro lado, suelen obtener resultados inadecuados cuando el problema de turnos es complejo.



Figura 2: Códice: Ejemplo de gestora.

Como ejemplo, *Códice* (véase Figura 2), una de las gestoras y consultoras más conocida a nivel nacional, con sede en Extremadura, busca ayudar a empresas en distintas labores de gestión. El trabajo que hemos realizado es semejante al que lleva a cabo *Códice*, pero particularizado al campo de la gestión de turnos y sin dependencia de ninguna empresa externa. Para más información véase [12].

Por lo tanto, gracias a la aplicación desarrollada, queremos conseguir que la empresa considerada en este proyecto (AUVASA), con su normativa y requisitos específicos (que describiremos más adelante) no dependa de gestoras externas para la creación y gestión de sus turnos de trabajo y además que los pueda generar en el menor tiempo posible, alcanzando un compromiso entre eficacia y eficiencia. Eficacia, ya que se deben conseguir los calendarios que mejor se ajusten a los requisitos dados por la empresa y, eficiencia, ya que se deben obtener invirtiendo el menor número de recursos y priorizando el tiempo de ejecución.

2.2. Planificación

En la planificación se mostrarán todos los pasos que se han tenido en cuenta para el desarrollo del proyecto, comenzando por los requisitos solicitados por la propia empresa y concluyendo con el diagrama de Gantt que resulta tras el estudio de esta sección.

Para la realización del proyecto lo primero que hemos tenido en cuenta es que consta de dos fases diferentes. Por un lado, el desarrollo de un algoritmo adecuado para la elaboración de los calendarios de trabajo y por otro, la propia interfaz de la aplicación.

Tenemos que poner de manifiesto que hemos utilizado la metodología de iteración incremental que escribiremos de forma detallada más adelante. Para más información ver el libro Hughes y Cotterell [6].

2.2.1. Requisitos

Es muy importante conocer los requisitos de un proyecto de software, ya que estos, describen el comportamiento de forma completa y precisa que debe tener el sistema que se va a desarrollar. Destacamos los tres tipos de requisitos siguientes:

- **Requisitos funcionales.** Describen los servicios que el programa debe proporcionar a los usuarios. Es el apartado donde se pueden ver todos los casos de uso que se deben desarrollar.
- **Requisitos no funcionales.** Estos requisitos deben describir características del funcionamiento del sistema, como la accesibilidad, el rendimiento, la disponibilidad... Estos son los llamados requisitos de calidad del sistema. Sin embargo, no tienen que describir funciones del sistema ni tan siquiera, cómo guardar la información.
- **Requisitos de información.** Explican cómo se debe obtener, gestionar y almacenar la información de la aplicación.

A continuación, se muestran los requisitos identificados para el sistema concreto que se vamos a desarrollar.

Funcionales:

- **RF-01**
Crear calendarios. El sistema deberá permitir al usuario encargado de la gestión la creación de los calendarios.
- **RF-02**
Modificar calendarios. El sistema deberá permitir al usuario gestor la modificación de los calendarios de los trabajadores en unas fechas determinadas.
- **RF-03**
Visualización de calendarios. El sistema deberá permitir la visualización de los calendarios a todos los usuarios autorizados por la empresa.
- **RF-04**
Guardar calendario. El sistema deberá permitir al usuario gestor guardar el calendario en la base de datos.

- **RF-05**
Guardar demanda. Se permitirá que el usuario gestor guarde la demanda resultante de un calendario.
- **RF-06**
Crear reclamaciones: Se debe permitir que los usuarios ordinarios puedan realizar reclamaciones.
- **RF-07**
Ver reclamaciones. El sistema deberá permitir al usuario gestor ver las reclamaciones realizadas por los conductores.
- **RF-08**
Restricciones de creación. El sistema permitirá al usuario gestor la modificación de las opciones de generación automática de calendarios.
- **RF-09**
Recogida de datos. Permitirá al usuario gestor seleccionar la ubicación de la información.
- **RF-10**
Análisis de los calendarios. El sistema deberá permitir al usuario gestor visualizar datos estadísticos de los calendarios de todos los empleados incluidos en el estudio.
- **RF-11**
Datos de calendarios individuales. Se permitirá a los usuarios ordinarios la visualización de los datos estadísticos de su calendario particular.
- **RF-12**
Inicio de sesión. El sistema deberá permitir a todos los usuarios autorizados que puedan iniciar sesión.
- **RF-13**
Gestión de conductores. El sistema deberá permitir al usuario gestor la creación o inclusión de nuevos conductores.
- **RF-14**
Gestión de conductores 2. Se permitirá al usuario gestor editar la información de los conductores existentes.
- **RF-15**
Gestión de conductores 3. El sistema deberá permitir al usuario gestor eliminar algunos conductores ya existentes.

No funcionales:

- **RNF-01**
Usabilidad. La aplicación deberá ser intuitiva y fácil de usar para el 95 % de los usuarios de la empresa.
- **RNF-02**
Idioma. Se mostrará la información en la interfaz en castellano.
- **RNF-03**
Acceso a datos. La aplicación deberá gestionar los datos de la empresa mediante su MySQL.
- **RNF-04**
Interfaz. Deberá ser una aplicación web.
- **RNF-05**
Informes. Se deben indicar los *días conflictivos* del calendario que pueden surgir ir por falta o exceso de personal.

- **RNF-06**
Sistema operativo. Se permitirá montar el servidor web desde el SO Windows.
- **RNF-07**
Datos anuales. Se podrá realizar la creación web de calendarios de forma anual.

De información:

- **RI-01**
Usuarios. El sistema deberá contar con la información de los usuarios que pueden acceder al sistema y el rol que desempeñan.
- **RI-02**
Subir los calendarios. Deberá poder almacenar la información de los calendarios en la base de datos de la empresa.
- **RI-03**
Descargar los calendarios. El sistema deberá poder adquirir la información de los calendarios desde la base de datos.
- **RI-04**
Información de los empleados. El sistema deberá poder obtener la información de los empleados involucrados en la creación de los calendarios desde MySQL.
- **RI-05**
Almacenar sugerencias. Deberá permitir recoger y almacenar las sugerencias de los trabajadores en la base de datos.

2.2.2. Normativa de la empresa

Además de los requisitos previamente expuestos, hemos tenido en cuenta y estudiado la normativa e información de la propia empresa para poder realizar las siguientes etapas de la creación de la aplicación.

La información específica de AUVASA es la siguiente:

1. Existen únicamente dos tipos de turnos de trabajo: turno de mañana (M) y turno de tarde (T).
2. Todos los conductores tienen cuatro posibles estados en el calendario: turno de mañana (M), turno de tarde (T), descanso (D) y vacaciones (V).
3. Existen tres grupos o tipos principales de conductores en la plantilla:
 - **Eventuales.** Son conductores que realizan su trabajo temporalmente, su trabajo dura 92 días comenzando el 1 de julio y terminando en septiembre. Sirven para completar los turnos en los días de verano pues es cuando se producen más ausencias dentro de la plantilla.
 - **Corretornos.** Son un grupo de conductores fijos dentro de la empresa que están disponibles para cubrir las distintas vacantes que se generen a lo largo del calendario. A su vez, estos conductores pueden ser de tres tipos dependiendo del trabajo que desempeñen.
 - **Conductores de turnos de mañana:** dichos conductores solo realizan turnos de trabajo de mañana.
 - **Conductores de turnos de tarde:** dichos conductores solo realizan turnos de trabajo de tarde.
 - **Conductores de turnos mixtos:** dichos conductores pueden realizar turnos de trabajo tanto de mañana como de tarde.
 - **Tríos:** son un conjunto de tres conductores concretos de la empresa, asignados a una sola línea, que realizan turnos de trabajo mixtos (tanto de tarde como de mañana).
4. Existen 18 líneas, pero para este proyecto se utilizarán 16 de ellas. Esto debido a que existen dos líneas de las que se encargan otros conductores no tenidos en cuenta en este proyecto.
5. Se estima la demanda anual de trabajo en cada línea y turno.
6. Las líneas que tienen mayor demanda también tienen un mayor número de tríos asignados.
7. La demanda anual suele repartirse de la siguiente manera.
 - **Menor demanda:**
 - En los meses de verano (julio y agosto).
 - En los días festivos.
 - En los fines de semana.
 - **Mayor demanda:**
 - En los días comprendidos entre festivos del mes de enero.
 - En los días hábiles (de lunes a viernes).
 - Durante la primera semana de septiembre debido a las fiestas de la ciudad de Valladolid.

La normativa interna de la empresa y que debemos cumplir en la creación de los calendarios anuales es la siguiente.

1. Todos los conductores:

- Los turnos seguidos de trabajo, en principio, oscilan entre 4 y 6 días, pero la normativa interna de la empresa permite cambiar esto si fuese preciso. El número de turnos seguidos de trabajo entre descansos o vacaciones recibe el nombre de *grupo de trabajo*.
- El número de descansos seguidos que pueden tener oscilan entre 2 y 4 días. El grupo de turnos seguidos de descansos se denomina *grupo de descanso*.
- No se permiten días de descanso o trabajo aislados.
- Los conductores no pueden variar el tipo de turnos realizados durante un grupo de trabajo.
- Los conductores de turnos mixtos después de un periodo de descanso varían su turno respecto al periodo anterior a dicho descanso. Por ejemplo, si un conductor ha tenido turno de mañana, y después un periodo de descanso, entonces, el siguiente grupo de trabajo que le tocará será de tarde (M M M M D D T T T T).
- Ningún conductor puede trabajar más de 4 domingos de forma consecutiva.
- Antes o después de los periodos vacacionales un conductor puede tener cualquier número de días seguidos de trabajo (sin superar 6 días) y cualquier número de días de descanso (oscilando entre 2 y 4 días). Ejemplo: D M V V V V...V D o
T D V V .. V V D T T T T.
- Los conductores tienen un número máximo de días de trabajo al año, dependiendo del tipo de conductor.

2. Para los conductores fijos:

- Todos los conductores fijos (corretornos y tríos):
 - Tienen un máximo de 209 días de trabajo anuales, que pueden ser modificados si se desea.
 - Tienen un número máximo de grupos de trabajo con 6 días.
 - Tienen un total de 30 días de vacaciones naturales al año, divididos en dos quincenas, una de verano situada en los meses de junio, julio y agosto y otra quincena denominada de invierno que abarca el resto de los meses del año.
- Los conductores de tríos:
 - Deben estar un porcentaje de días mínimo cumpliendo la regla del trío, según la cual los conductores de trío deben encontrarse o todos de descanso o todos de vacaciones o de forma que uno esté de turno de mañana, otro de turno de tarde y el tercero de descanso.
 - Los conductores de tríos tienen una línea asignada por lo que tienen que trabajar el mayor número de días posible en esta o estar de refuerzo, es decir, no asignados a ninguna línea.
 - El número de tríos asignados a cada línea es igual al máximo número de vehículos utilizado en esa línea en cualquier día y turno del año. Ejemplo: si la línea 7 utiliza un día del año 20 vehículos en el turno de mañana y los demás días y turnos del año menos de 20, el número de tríos asignado a esa línea será de 20.
 - Todos tienen turnos de trabajo mixtos.

- Los correturnos:
 - No tienen ninguna línea asignada, por lo que no tienen prioridad en la asignación de línea en sus turnos de trabajo.
 - Como hemos mencionado antes, pueden tener turnos de mañana o de tarde, no sólo mixtos.

3. Para los conductores eventuales:

- El número de días de trabajo de estos conductores es principio de 56, pero la normativa de la empresa permite modificar este número si fuese necesario.
- Como los calendarios de los conductores son anuales, los calendarios de los eventuales quedarán representados como vacaciones todos los días en los que no están contratados.
- Estos conductores, a diferencia de los otros, no tienen un número máximo de grupos de trabajo con 6 días seguidos.

2.2.3. Infraestructura del proyecto

Para llevar a cabo este proyecto, hemos contado con el equipo informático necesario (un ordenador I5 de novena generación, 32 Gb de memoria RAM y tarjeta gráfica Nvidia RTX2060 y conexión a internet), también, hemos utilizado programas y lenguajes de programación de libre distribución (que comentaremos en la Sección 2) y los siguientes programas con licencia:

- **Astah professional.** Con este programa se han realizado todos los diagramas de diseño de software (por ejemplo, el modelo de dominio o el diagrama de casos de uso). Véase [10] para más información.
- **Microsoft Project.** Con este programa hemos construido el diagrama de Gantt o diagrama de tareas del proyecto. Véase [11].
- **Balsamiq Wireframes.** Programa que se ha utilizado para realizar los *mock ups* o versiones preliminares de la aplicación.

Por último, necesitábamos elegir un modelo de arquitectura web para desarrollar la aplicación. Las posibles arquitecturas que consideramos inicialmente fueron:

- **P2P.** En este modelo todos los ordenadores (o nodos) pueden modificar su comportamiento como clientes o servidores para intercambiar información según convenga. Además, permite un intercambio directo de información entre nodos interconectados. Véase Kurose y Ross [7].

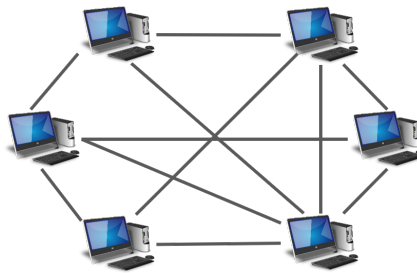


Figura 3: Modelo P2P

- **Cliente–Servidor.** En este modelo, los nodos proveedores de recursos (los servidores) y los nodos demandantes de servicios, no intercambian su función. El cliente tiene que conectarse al servidor (acción que normalmente se realiza a través de la web) para cualquier intercambio de información y se envían los recursos solicitados por el cliente sin hacer ningún tratamiento sobre los mismos (Kurose y Ross [7]).

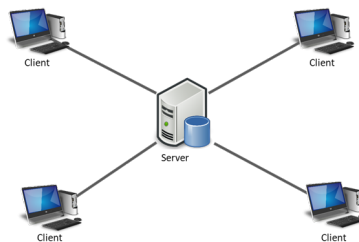


Figura 4: Modelo Cliente–Servidor.

- **Modelo servidor de aplicaciones.** En este modelo los recursos que se van a utilizar no son archivos estáticos, sino que contienen el código que se tiene que ejecutar. Es decir, un servidor web en solitario envía al cliente el recurso solicitado sin modificaciones, mientras que el servidor de aplicaciones lo ejecuta y envía al cliente el resultado a través del servidor web. Por último, cabe destacar que el servidor de aplicaciones, normalmente, está asociado a una base de datos para obtener la información necesaria.

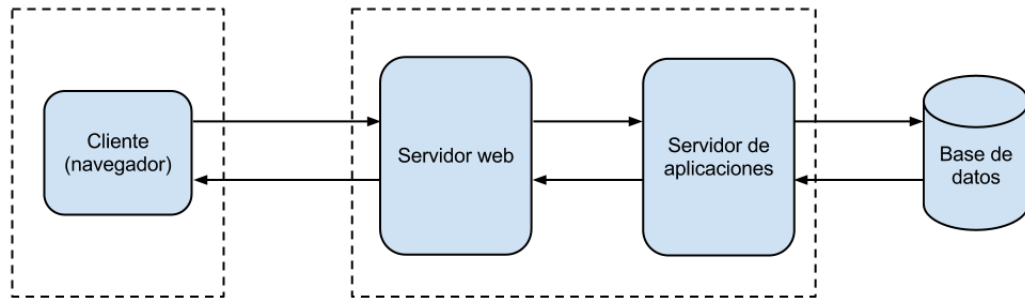


Figura 5: Modelo servidor de aplicaciones.

Como el modelo de arquitectura web debe tener en cuenta el lenguaje de programación utilizado, hemos condicionado esta decisión a la elección del lenguaje, apartado 3.1.

2.2.4. Actividades del proyecto

Hemos considerado que la metodología que mejor se adapta a la realización de este proyecto es el método iterativo incremental (Hughes [6]). Este método se basa en subdividir el proyecto en diversos bloques temporales que se denominan *iteraciones*. En cada iteración se entrega un producto nuevo o mejorado o nuevo y mejorado.

El método se denomina iterativo porque está dividido en varias etapas o pasos de forma que en cada una se añade una mejora tras validarlo con el usuario y con el fin de obtener el mejor producto final. Se denomina incremental ya que en cada paso se le añaden nuevas funcionalidades que antes no tenía.

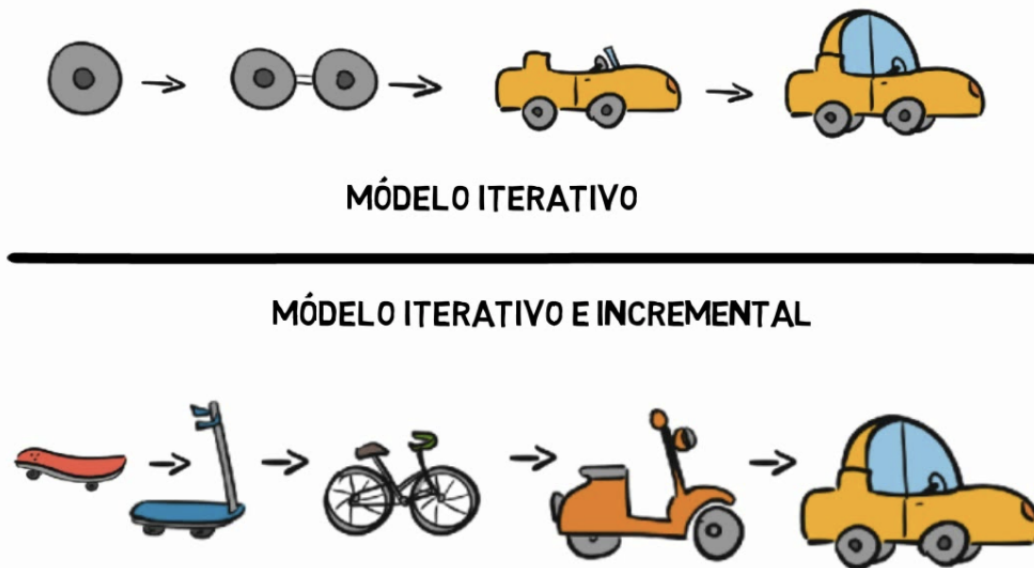


Figura 6: Representación gráfica del método iterativo e incremental.

Habitualmente, con esta metodología, el proyecto se divide en cuatro etapas con distintas funcionalidades. Estas son:

- **Inicio.** En esta etapa se analiza qué se va a utilizar en el proyecto a través del estudio de los objetivos a conseguir, los requisitos, la planificación...
- **Diseño.** En este paso desarrollaremos el funcionamiento del software sin entrar en detalles y se establecerán las características de los recursos utilizados por el sistema. En este momento, se crearán los diagramas de casos de uso y de secuencia para definir la funcionalidad del sistema y la descripción de las tecnologías que utilizará en el mismo.
- **Implementación.** En esta etapa se programa el código de cada caso de uso de la aplicación y se realizan las pruebas o tests necesarios para comprobar el correcto funcionamiento de las distintas funcionalidades de la aplicación.
- **Finalización.** En esta etapa se busca finalizar la aplicación realizando labores de mantenimiento, redacción de los manuales de funcionamiento y mejora final de la aplicación.

Las actividades que se realizarán en cada etapa son las siguientes:

■ **Inicio:**

- Planificación del proyecto.
- Identificación de requisitos.
- Identificación y manejo de riesgos.
- Estudio de las tecnologías.
- Redacción la memoria.

■ **Diseño de software:**

- Modelo de dominio.
- Diagrama e identificación de los casos de uso.
- Diagramas de secuencia.
- Diagrama de datos.
- *Mock ups*.
- Redacción de la memoria.

■ **Implementación:**

- Implementación del algoritmo.
- Implementación de los casos de uso.
- Realización de pruebas.
- Corrección de errores.
- Redacción la memoria.

■ **Finalización:**

- Manuales de uso e instalación.
- Finalización de la memoria.
- Preparación de la presentación.

2.2.5. Esfuerzo de cada etapa

El esfuerzo y/o coste de cada actividad del proyecto se ha medido con las mismas unidades, número de días necesarios para completar cada actividad ya que a priori no existe ningún otro coste económico para las distintas actividades.

Los costes de las cuatro etapas principales llevadas a cabo en este proyecto se repartirán de la siguiente forma:

Etapas	Coste	Fecha inicio	Fecha fin
Inicio	12 días	6/03/22	19/03/22
Diseño de Software	11 días	20/03/22	01/04/22
Implementación	21 días	02/04/22	29/04/22
Finalización	17 días	30/04/22	23/05/22

Tabla 1: Costes de cada etapa.

Como se puede ver en la Tabla 1 el día de inicio del proyecto fue el 6 de marzo y el final del mismo estaba previsto para el 23 de mayo con un total de 61 días de trabajo y un día de descanso semanal.

El esfuerzo de cada una de las actividades de cada etapa se podrá ver en el diagrama de Gantt (7).

2.2.6. Riesgos

Durante los proyectos pueden surgir problemas que irremediablemente hagan que no se puedan cumplir los plazos de entrega e incluso puedan llevar a que no se pueda concluir el mismo. Estos problemas que pueden cambiar un plan inicial se los denomina *riesgos*.

Por esta razón es muy importante tener en cuenta los riesgos que pueden surgir en un proyecto concreto de cara a intentar prevenirlos, y en caso de que ocurran, poder seguir un plan de contingencias para intentar resolverlos o manejarlos de la forma más rápida posible y con el menor daño al plan inicial de trabajo.

Una de las herramientas más utilizadas y eficientes para mitigar los riesgos de un proyecto, y que nosotros hemos utilizado, es su estudio o análisis a lo largo de tres pasos:

1. **Identificación de riesgos.** En esta etapa se identifican y describen las características del máximo número de riesgos que pueden surgir en el proyecto, ya que, cuantos más se identifiquen es menos probable que puedan ocurrir imprevistos.
2. **Análisis de riesgos.** Consiste en determinar tanto la probabilidad de que ocurra un suceso como el daño o perjuicio que pueda generar en el desarrollo del proyecto.

Debido a la dificultad de dar un valor exacto tanto a la probabilidad como al impacto de un riesgo, se suelen usar valores discretos para evaluarlo. En nuestro caso particular hemos utilizado los siguiente valores:

- **Probabilidad:**

1. **Muy alta:** más del 70 % de probabilidades de que ocurra.
2. **Alta:** entre el 50 % y el 70 % de probabilidades de que ocurra.
3. **Media:** entre el 30 % y el 50 % de probabilidades de que ocurra.
4. **Baja:** entre el 10 % y el 29 % de probabilidades de que ocurra.
5. **Muy baja:** menos del 10 % de probabilidades de que ocurra.

- **Impacto:**

1. **Muy alta:** más del 30 % del coste.
2. **alta:** entre el 20 % y el 30 % del coste.
3. **Media:** entre el 10 % y el 19 % del coste.
4. **Baja:** entre el 5 % y el 9 % del coste.
5. **Muy baja:** menos del 5 % del coste.

3. **Gestión de riesgos.** Consiste en evaluar el riesgo potencial y crear planes para minimizar el impacto de los riesgos en el proyecto. Existen dos tipos de planes:

- **Planes de mitigación** que sirven para prevenir los posibles riesgos del proyecto.
- **Planes de contingencia** que sirven para tratar de resolver los problemas derivados por la aparición de un riesgo.

En la Tabla 2 podemos observar la plantilla de registro de riesgos que hemos utilizado en este proyecto.

Número	–
Título	–
Descripción	–
Probabilidad	–
Impacto	–
Evaluación del peligro	–
Plan de mitigación	–
Plan de contingencia	–

Tabla 2: Plantilla de registro de riesgos.

De manera que los riesgos de este proyecto son los siguientes:

Número	1
Título	Enfermedad del desarrollador.
Descripción	El desarrollador no puede continuar la realización del proyecto debido a una enfermedad.
Probabilidad	Baja.
Impacto	Medio.
Plan de mitigación	Seguir los métodos preventivos necesarios con las enfermedades habituales.
Plan de contingencia	Si es posible para el desarrollador, avanzar en las tareas más sencillas o rutinarias, de forma que afecte lo menos posible a los tiempos de entrega.

Tabla 3: Riesgo 1. Enfermedad del desarrollador.

Número	2
Título	Incompatibilidad con otras tareas.
Descripción	Si el desarrollador no puede avanzar durante los días de trabajo asignado debido a tareas externas al proyecto, por ejemplo de asignaturas del 2º cuatrimestre.
Probabilidad	Alta
Impacto	Alto
Plan de mitigación	Tratar de llevar todas las actividades externas al día.
Plan de contingencia	Si es posible para el desarrollador aumentar el tiempo de trabajo o modificar los plazos de entrega de cada una de las siguientes etapas.

Tabla 4: Riesgo 2. Incompatibilidad con otras tareas.

Número	3
Título	Modificación de los requisitos.
Descripción	Si se tienen que modificar levemente algunos de los requisitos dados por peticiones tanto externas como internas.
Probabilidad	Baja.
Impacto	Muy alto.
Plan de mitigación	Planteamiento de los requisitos de una forma exhaustiva y amplia para que los cambios sean mínimos.
Plan de contingencia	Modificar la planificación de la etapa de implementación y final.

Tabla 5: Riesgo 3. Modificación de los requisitos.

Número	4
Título	Problemas con el material de trabajo.
Descripción	Imposibilidad de continuar con el proyecto debido a averías con los equipos informáticos, falta de licencias, problemas eléctricos, etc.
Probabilidad	Muy baja.
Impacto	Muy alto.
Plan de mitigación	Tratar de guardar la información del proyecto en la nube.
Plan de contingencia	Continuar con el desarrollo del proyecto con equipos nuevos o en lugares equipados con lo necesario.

Tabla 6: Riesgo 4. Problemas con el material de trabajo.

Número	5
Título	Retrasos en las tareas.
Descripción	El tiempo empleado en las actividades es mayor que el previsto en la planificación.
Probabilidad	Medio.
Impacto	Medio.
Plan de mitigación	Dejar un margen de error para cada tarea en la planificación.
Plan de contingencia	Reorganizar las tareas planificadas aumentando el tiempo en las que lo necesiten.

Tabla 7: Riesgo 5. Retrasos en las tareas.

Número	6
Título	Errores de la aplicación.
Descripción	Diseñar de forma errónea la aplicación y por lo tanto que aparezcan más errores en la aplicación de los esperados.
Probabilidad	Bajo.
Impacto	Medio.
Plan de mitigación	Tratar de conseguir un buen diseño en las etapas de diseño de software para no tener que modificarse.
Plan de contingencia	Rehacer los diseños erróneos para conseguir el menor número fallos.

Tabla 8: Riesgo 6. Errores de la aplicación.

Número	7
Título	Problemas de comunicación.
Descripción	Problemas en el intercambio de información entre la empresa, el estudiante y el tutor.
Probabilidad	Bajo.
Impacto	Alto.
Plan de mitigación	Tratar de mantener un contacto de forma regular.
Plan de contingencia	Tratar de avanzar en otras partes del proyecto donde no escasee la información, y continuar insistiendo en la comunicación para solucionar los posibles problemas.

Tabla 9: Riesgo 7. Problemas de comunicación.

2.2.7. Diagrama de Gantt

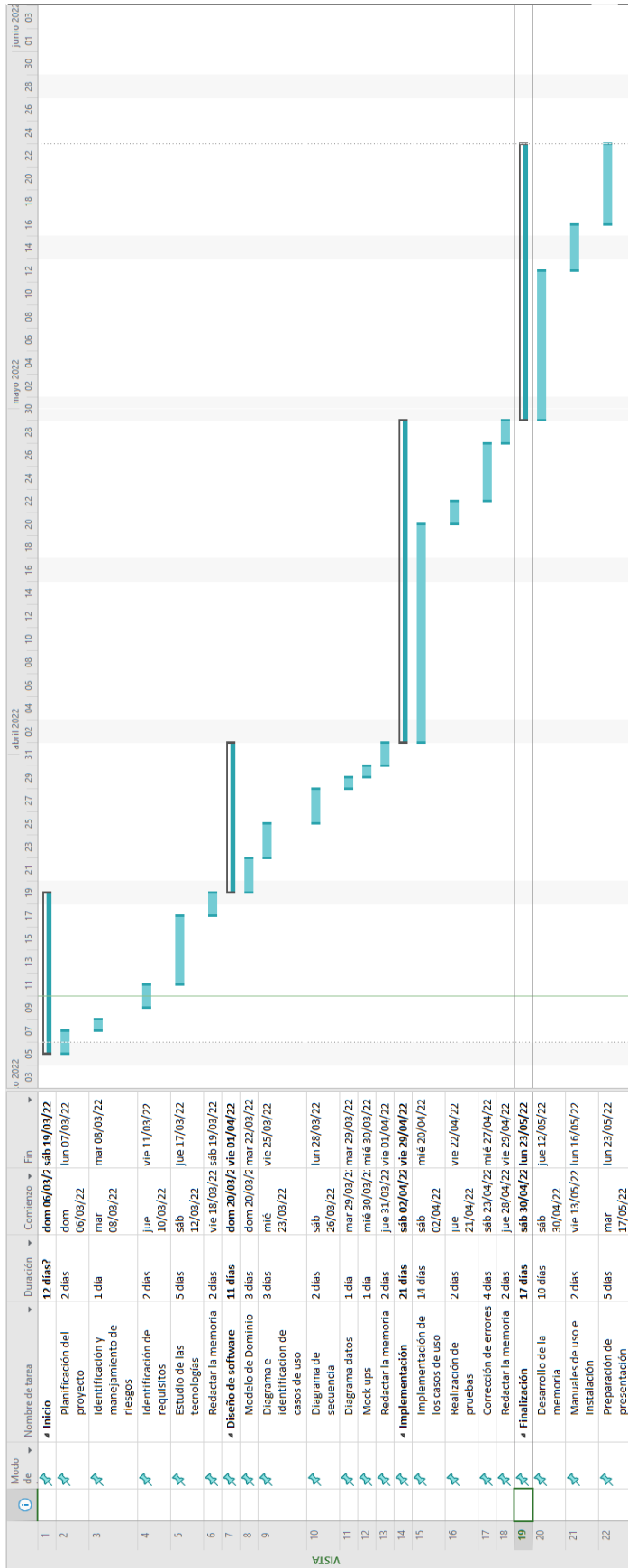


Figura 7: Diagrama de Gantt inicial.

2.3. Seguimiento del proyecto

19 de enero.

Durante la primera etapa de la planificación o inicio todas las tareas exceptuando la identificación de requisitos fueron realizadas en el tiempo previsto. Se produjo un retraso debido al Riesgo 7 (Problemas de comunicación con la empresa) que se salvó realizando boceto de los mismos para poder continuar con las siguientes etapas.

1 de abril.

En esta etapa se habían realizado los distintos diagramas referentes al diseño de software, pero aún se desconocían los requisitos finales, por lo tanto, se mantuvo el boceto elaborado y se continuó completando la memoria y la implementación de la aplicación.

16 de abril.

En esta etapa se logró contactar con la empresa, de manera que se corrigieron los requisitos considerados y los diagramas correspondientes. Además, se diseñó y relleno la base de datos de MySQL. Con ello quedaron afectados en gran medida el riesgo 3 (Modificación de requisitos) y el riesgo 5 (Retrasos en las tareas) y se comenzaron todas las tareas referentes a la implementación de la aplicación.

30 de abril.

Se avanzó en la implementación de la aplicación pero no se consiguió terminar tal y como estaba planificado inicialmente debido al riesgo 2 (Incompatibilidad con otras tareas) puesto que coincidió con trabajos a realizar relacionados con otras asignaturas de la carrera. Además, la elaboración de la aplicación nos llevó más mayor tiempo de lo inicialmente previsto.

24 de mayo.

En este momento, se finalizó la implementación de la aplicación y la corrección de todos los errores y problemas de la misma. Empezamos entonces a redactar la memoria del trabajo, que a estas alturas deberíamos haber tenido terminado pero que debido a nuevos trabajos a realizar en las asignaturas del grado y al comienzo del otro TFG (Trabajo de Fin de Grado) que tenía que hacer (Riesgo 2), a retrasos en las tareas (Riesgo 5) y en menor medida al Riesgo 1 (Enfermedad del desarrollador), se produjo un retraso.

En la Figura 8 se muestra el diagrama de Gantt final con todas las variaciones que se habían producido.

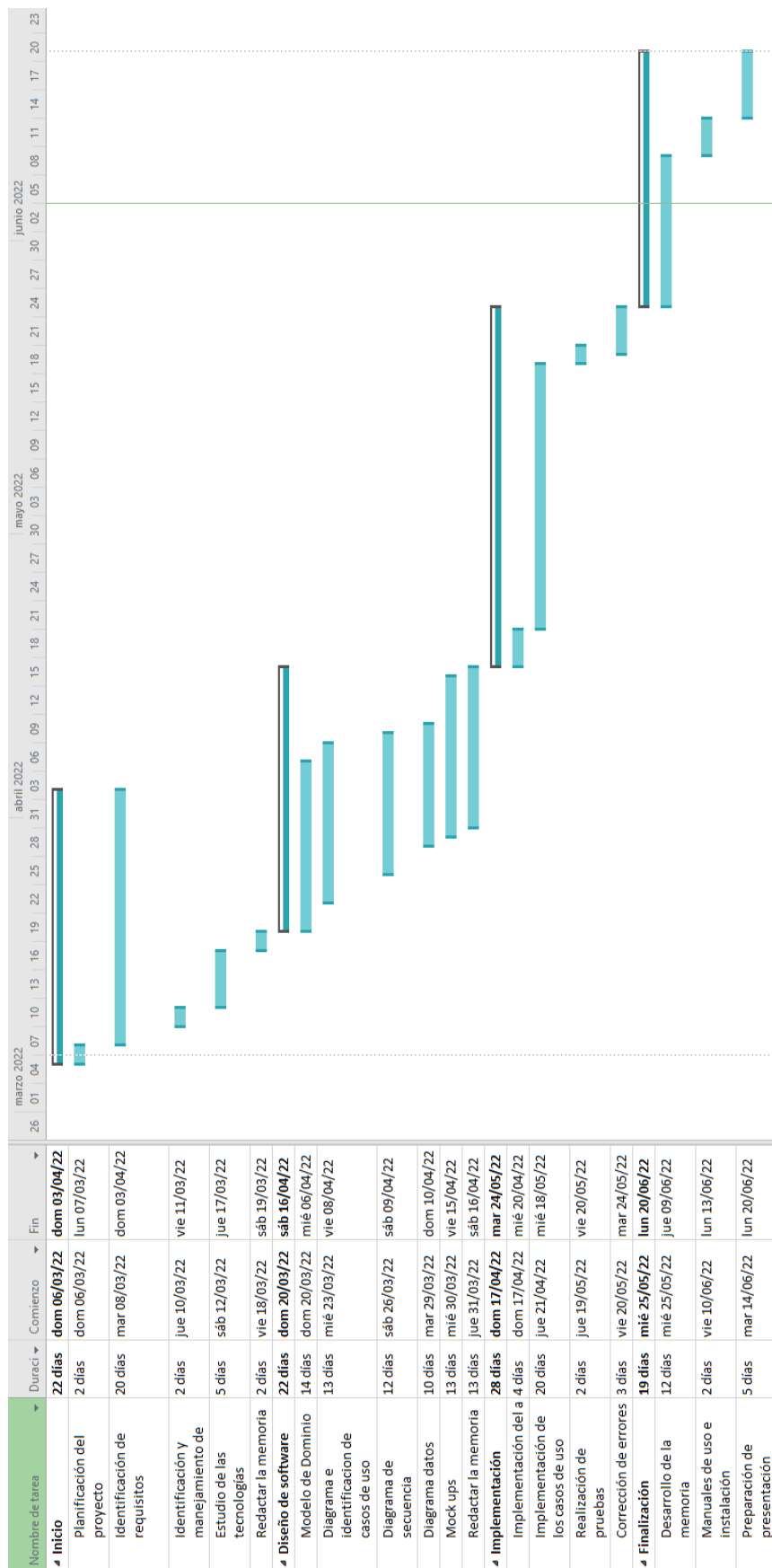


Figura 8: Diagrama de Gantt final.

3. Diseño de Software

En esta sección vamos a exponer todos los aspectos relacionados con el diseño del software, es decir, mostraremos todos los diagramas de diseño necesarios para llevar a cabo el desarrollo de nuestro proyecto, además, seleccionaremos la tecnología para implementarlo, empezando por la elección del lenguaje de programación y, posteriormente, de la arquitectura web.

3.1. Tecnología utilizada

El desarrollo web de esta aplicación se ha realizado utilizando lenguaje **Python** junto con **HTML** y base de datos de **MySQL** proporcionada por **XAMPP**.

El lenguaje de programación que hemos elegido ha sido Python ya que es un lenguaje de programación muy versátil, fácil de usar y que posee una gran cantidad de librerías de gran utilidad para la realización, tanto del algoritmo empleado, como de su visualización en web.

Inicialmente, barajamos la posibilidad de utilizar otros lenguajes de programación como **C++** y **Javascript**. El primero de ellos, se descartó debido a que posee muchas limitaciones para el desarrollo web. En el caso de Javascript la implementación del algoritmo nos suponía un proceso más costoso, razón por la cual decidimos utilizarlo exclusivamente para algunas de las vistas HTML debido a la comodidad que nos ofrecía para la gestión de estas vistas.

Las librerías y paquetes más importantes que hemos utilizado en el desarrollo de la aplicación son:

- **XAMPP**. Este es un paquete de software libre que nos proporciona: **MariaDB** (gestor de bases de datos derivado de MySQL), el servidor web **Apache** e intérpretes de lenguajes como **PHP** y **Perl**.

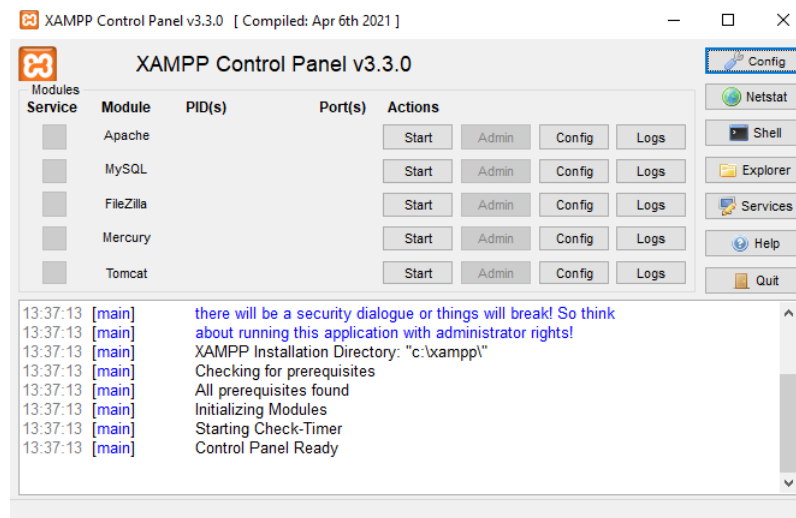


Figura 9: Panel de control de XAMPP.

- Flask.** Este es un paquete de software libre de Python que nos proporciona todo lo necesario para crear aplicaciones web de forma rápida utilizando un pequeño número de líneas de código (esta es una de las razones de que sea tan popular). Está basado principalmente en dos proyectos, **WSGI**, que es la principal librería web de este lenguaje para el reenvío de llamadas web y **Jinja2**, que es el encargado de visualizar las plantillas web. Por todo ello, Flask será el motor para crear el servidor web local. Ver [8] para más información.



Figura 10: Logo de Flask y Jinja.

- Numpy.** Es una librería de Python que permite la creación de vectores y matrices de grandes dimensiones y proporciona una gran colección de operaciones para su gestión. Véase [13].

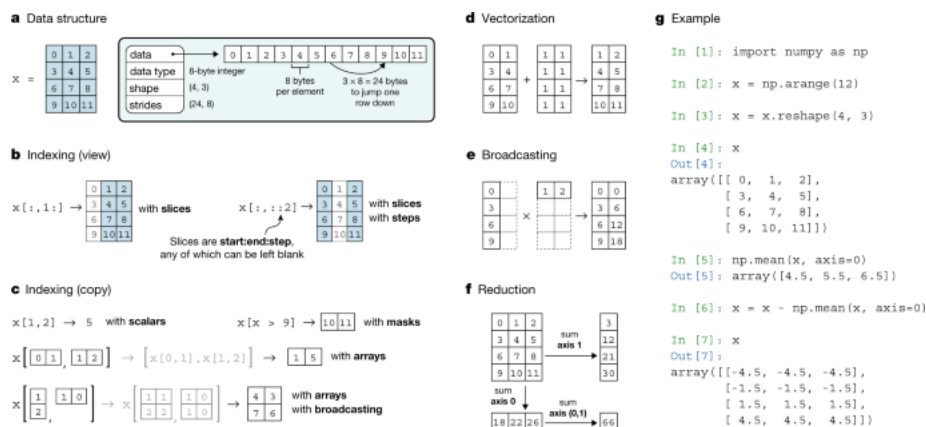


Figura 11: Tablas lógicas de Numpy.

En cuanto a la arquitectura de la aplicación, nos decidimos finalmente por utilizar el modelo Cliente–Servidor donde el servidor contiene la base de datos, la lógica de aplicación, el modelo y las vistas, mientras que el cliente realiza peticiones al servidor.

3.2. Patrones de diseño

En esta sección vamos a comentar los patrones de diseño que se han utilizado para llevar a cabo esta aplicación. Para más información ver los libros de Buschmann *et al.* [4] y Gamma *et al.* [5].

El desarrollo de esta aplicación se ha realizado siguiendo el patrón de diseño **modelo–controlador–templates** junto con **BBDD**, que es una variación del **MVC** (modelo–vista–controlador), muy utilizado en proyectos web en Python. Este modelo está compuesto por cuatro partes principales:

- **Modelo.** En este bloque se almacena la información de las clases que se encuentran en el proyecto, tanto sus funciones como atributos. Las clases pertenecientes al modelo y sus relaciones se podrán ver en el modelo de dominio apartado 3.3. Además, debemos destacar que el modelo solo podrá compartir información y/o ser utilizado desde el controlador (como veremos más adelante).
- **Controlador.** Este bloque se encarga de la gestión y el control del flujo del sistema web. Este último se relaciona con todas las partes del programa (modelo, vista y BBDD), y les comunica cómo deben actuar.
- **Templates.** Este bloque es el que tiene almacenado todas y cada una de las vistas (en este caso en formato html, pues es una aplicación web). También se encarga de capturar todos los eventos producidos por el usuario dentro de ellas y de enviar la información al controlador (si es necesario). No se comunican directamente con el modelo o la BBDD.
- **BBDD.** Este bloque conformado por la base de datos contiene toda la información o datos que se quieran almacenar de forma persistente en el sistema. Será llamado a través del controlador y no del modelo (como suele ocurrir en MVC).

Junto con este patrón de diseño se podrían llegar a usar los siguientes patrones.

El patrón estrategia. Este patrón de diseño sirve para manejar y efectuar otros comportamientos que puede tener una clase de forma más simple, tanto en la modificación de un comportamiento como en la eliminación y adición de nuevos comportamientos en el sistema.

El patrón estado. Este patrón es muy parecido al anterior pero en vez de referirse al comportamiento de un objeto se refiere a distintos estados que puedan tener, por ejemplo, un paquete puede estar vendido o en venta.

El patrón composición. Este patrón de diseño permite que los objetos puedan componerse de varias estructuras como estados o estrategias al mismo tiempo, pero tiene que existir una regla de tratamiento si existe una composición de estas estructuras. Ejemplo, un objeto ticket puede tener un conjunto de descuentos.

El patrón agregación. Este patrón de diseño permite a un objeto estar formado por un conjunto finito de objetos distintos a él y cuyos objetos a su vez tienen sentido como entidad sin su agrupación. Por ejemplo, un objeto motorizado con cuatro ruedas puede ser un coche y si tiene dos ruedas, una moto, pero el propio objeto rueda puede existir.

3.3. Modelo de dominio

El modelo de dominio que hemos utilizado en este proyecto con sus clases, las relaciones entre ellas y sus atributos podemos observarlo en la Figura 12. Tenemos una superclase llamada Usuario de la que heredan Gerente y Conductor, y ambas clases referencian a los distintos tipos de usuarios. También se puede apreciar que dentro de la clase conductor radica toda la complejidad del problema, conteniendo un patrón estrategia de los posibles turnos de los conductores. Además tienen una agregación compuesta por tres conductores (que forman los tríos de conductores) con unas características que siguen la normativa de la empresa en el apartado 2.2.2. Sin embargo, en el modelo la clase gerente es muy simple.

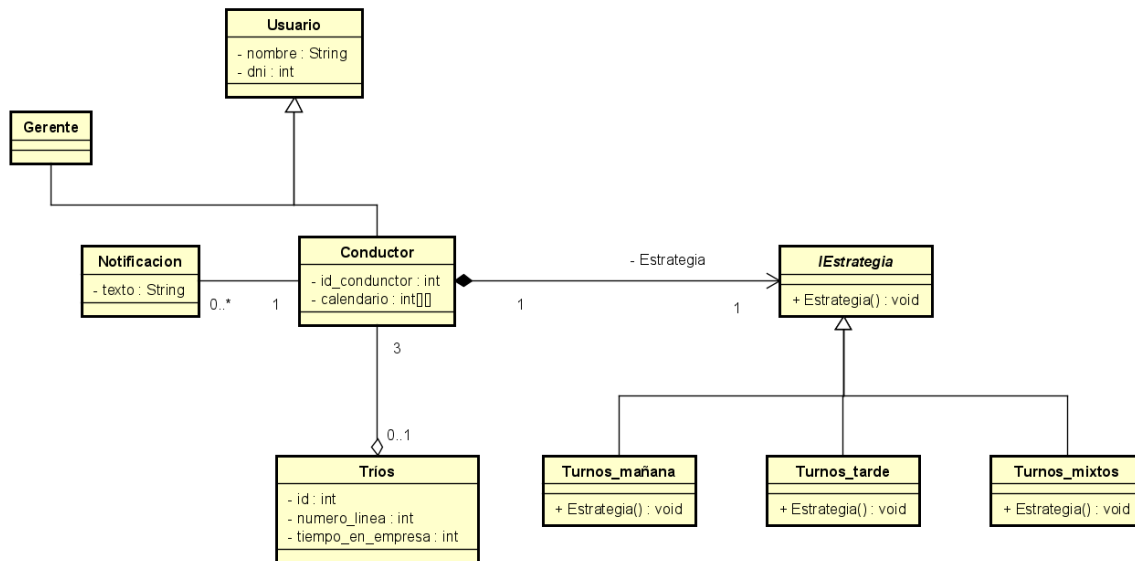


Figura 12: Modelo de dominio.

3.4. Casos de uso

En esta sección se muestran y describen los casos de uso contemplados en el sistema.

3.4.1. Diagrama de casos de uso

La Figura 13 se corresponde con el diagrama de casos de uso y en ella podemos observar un total de 19 casos de uso, donde de forma común se pueden realizar 2, el conductor puede realizar 2 y el gestor 15.

Los casos de uso contemplados son:

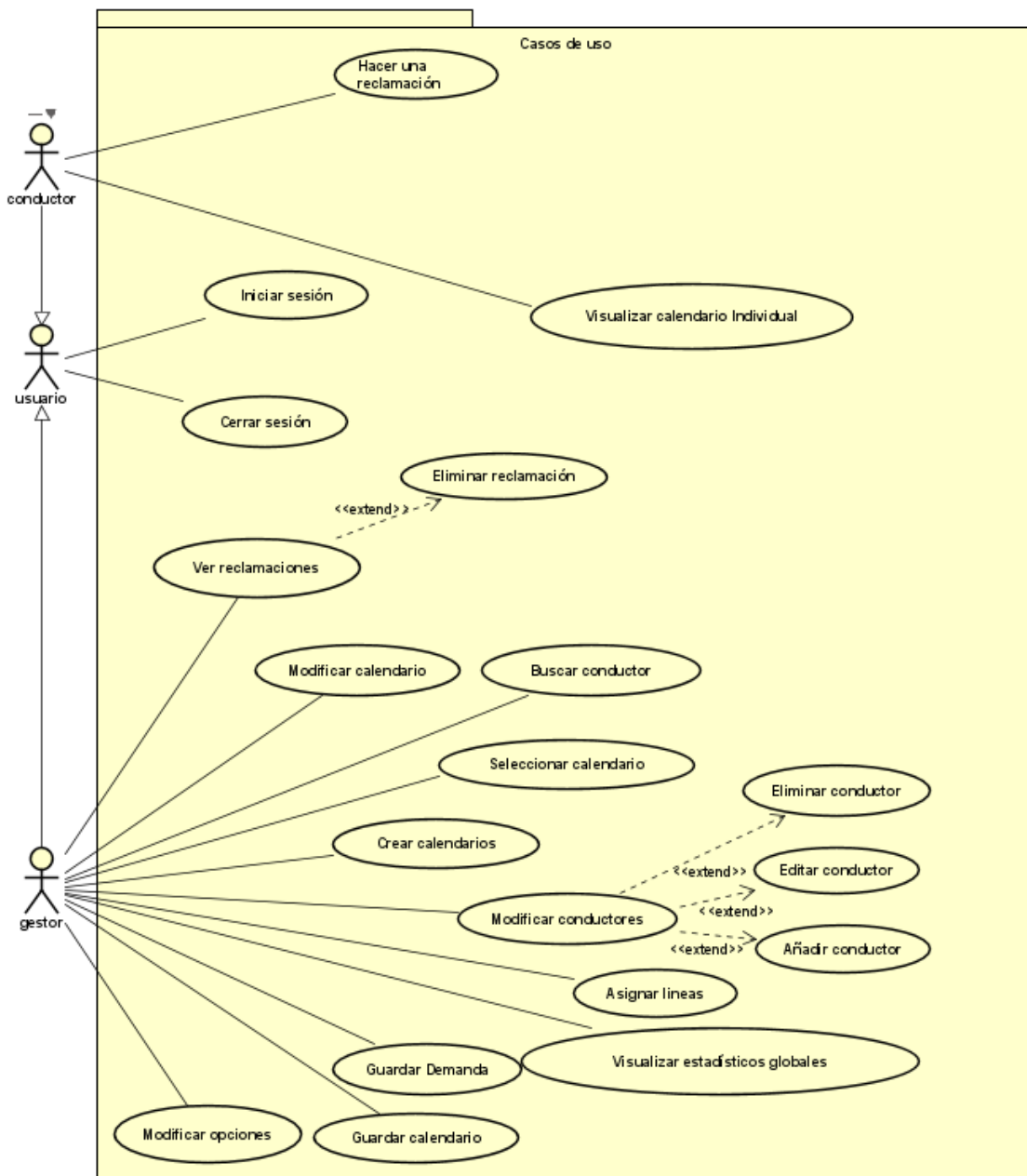


Figura 13: Diagrama de casos de uso.

3.4.2. Descripción de los casos de uso

En esta sección mostramos una descripción más detallada de los casos de uso.

Caso de uso: Iniciar Sesión.

- **Descripción:** iniciar sesión en la web.
- **Actor:** usuario.
- **Precondición:** no se ha realizado ningún inicio de sesión y se ha buscado la página en la web.
- **Flujo normal:**
 1. El usuario introduce su identificador de usuario y la contraseña y pulsa el botón entrar.
 2. El sistema comprueba que ese identificador y la contraseña son correctos.
 3. El sistema comprueba el tipo de usuario, gestor o conductor.
 4. El sistema guarda la sesión de usuario y muestra el menú principal asignado al tipo de usuario.
- **Post condiciones:** el usuario inicia sesión, ya sea como gestor o como conductor.
- **Flujos alternativos:**
 - 2b. El sistema detecta que el identificador y/o contraseña no son correctos.
 - 3b. El sistema muestra que se ha producido un error en el inicio de sesión y vuelve a solicitar los datos de acceso.

Caso de uso: Cerrar sesión.

- **Descripción:** cerrar sesión en la web
- **Actor:** usuario.
- **Precondición:** hay una sesión iniciada y el usuario se encuentra en el menú principal.
- **Flujo normal:**
 1. El usuario pulsa el botón cerrar sesión.
 2. El sistema cierra la sesión del usuario actual.
 3. El sistema muestra el menú de inicio de sesión.
 4. Se cierra la sesión web del usuario.
- **Post condiciones:** No hay.
- **Flujos alternativos:** No hay flujo alternativo.

Caso de uso: Hacer una reclamación.

- **Descripción:** Crear una reclamación en el calendario.
- **Actor:** Conductor.
- **Precondición:** El usuario conductor ha iniciado sesión y se encuentra en el menú principal.
- **Flujo normal:**
 1. El usuario conductor pulsa en crear notificación.
 2. El sistema carga y muestra la vista de creación de la notificación.
 3. El usuario escribe una notificación y pulsa enviar.
 4. El sistema guarda una notificación en la base de datos añadiendo el identificador de usuario, la fecha actual y el contenido de la notificación.
 5. El sistema muestra en la vista de creación de notificación que ha sido enviada la notificación.
- **Post condiciones:** se crea una nueva reclamación en la base de datos con los datos del conductor y la queja que este haya escrito.
- **Flujos alternativos:**
 - 3b. El usuario no escribe texto en la notificación y pulsa enviar.
 - 4b. El sistema muestra en la vista de creación de notificación un error indicando que el campo de texto está vacío.

Caso de uso: Visualizar el calendario individual.

- **Descripción:** mostrar el calendario individual.
- **Actor:** conductor.
- **Precondición:** el usuario conductor acaba de iniciar sesión.
- **Flujo normal:**
 1. El sistema descarga el calendario anual del conductor.
 2. El sistema muestra en la vista del calendario los datos descargados del calendario del conductor, junto con un menú para navegar.
- **Post condiciones:** el conductor visualiza su calendario.
- **Flujos alternativos:** no hay flujo alternativo.

Caso de uso: Buscar conductor.

- **Descripción:** se busca un conductor de un calendario.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de un calendario.

- **Flujo normal:**

1. El gestor pulsa el botón buscar conductor.
2. El sistema descarga la información de los conductores en el calendario visualizado.
3. El sistema muestra en una tabla la información de los conductores pertenecientes al actual calendario.
4. El gestor pulsa visualizar en el conductor que desea ver.
5. El sistema descarga el calendario individual del conductor y calcula algunos datos estadísticos relevantes.
6. El sistema muestra el calendario del conductor seleccionado y algunas estadísticas relevantes.

- **Post condiciones:** el gestor visualiza el calendario y algunas estadísticas del conductor seleccionado.

- **Flujos alternativos:** No hay flujo alternativo.

Caso de uso: Modificar calendario.

- **Descripción:** se quiere modificar unas fechas del calendario.

- **Actor:** gestor.

- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de un calendario.

- **Flujo normal:**

1. El usuario pulsa el botón modificar calendario.
2. El sistema muestra una vista solicitando que añadan las fechas de modificación del calendario.
3. El usuario escribe las fechas que quiere modificar.
4. El sistema comprueba que las fechas son correctas.
5. El sistema pone un mensaje de espera al usuario y llama al algoritmo de creación de calendarios con los datos pertinentes.

- **Post condiciones:** se visualiza el nuevo calendario global con los datos modificados.

- **Flujos alternativos:**

- 4b. El sistema comprueba las fechas y son erróneas.
- 5b. El sistema muestra un mensaje de error.
- 6b. El sistema vuelve a solicitar los datos.

Caso de uso: Seleccionar calendario.

- **Descripción:** se quiere mirar un calendario.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en el menú principal.
- **Flujo normal:**
 1. El usuario pulsa el botón de seleccionar calendario.
 2. El sistema muestra una página solicitando datos acerca del nombre del calendario y la demanda en la base de datos.
 3. El usuario rellena los datos y pulsa continuar.
 4. El sistema comprueba que sean correctos los datos introducidos.
 5. El sistema descarga los datos de la demanda y los calendarios.
 6. El sistema muestra el calendario anual en otra vista junto con un menú de navegación.
- **Post condiciones:** visualiza el calendario global que ha seleccionado.
- **Flujos alternativos:**
 - 4b. El sistema comprueba los datos y son incorrectos.
 - 5b. El sistema muestra en la vista donde solicita los datos un error en la búsqueda de las bases de datos.
 - 6b. El sistema vuelve a solicitar los datos.

Caso de uso: Crear calendarios.

- **Descripción:** se quiere crear un calendario anual.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en el menú principal.
- **Flujo normal:**
 1. El usuario pulsa el botón de crear calendario.
 2. El sistema muestra una página solicitando datos acerca del nombre del ciclo y la demanda en la base de datos y qué tipos de conductores debe tener en cuenta en la creación de calendarios.
 3. El usuario rellena los datos y pulsa continuar.
 4. El sistema comprueba que sean correctos los datos introducidos.
 5. Descarga los datos de la demanda, los ciclos y crea los objetos conductores.
 6. El sistema muestra un mensaje de espera y ejecuta al algoritmo con los datos pertinentes.
 7. El sistema muestra una vista con el calendario anual creado junto a un menú de navegación.
- **Post condiciones:** visualiza el calendario global que ha creado.

- **Flujos alternativos:**

- 4b. El sistema comprueba los datos y son incorrectos.
- 5b. El sistema muestra en la vista donde solicita los datos el error producido.
- 6b. El sistema vuelve a solicitar los datos.

Caso de uso: Asignar líneas.

- **Descripción:** asignar las líneas de bus a los conductores.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de un calendario.
- **Flujo normal:**
 1. El usuario pulsa el botón asignar líneas.
 2. El sistema descarga los datos de la demanda y los calendarios y crea los objetos conductores.
 3. El sistema ejecuta el algoritmo de asignar líneas.
 4. El sistema lanza un mensaje indicando que las líneas han sido asignadas.
- **Post condiciones:** se asignan las líneas de bus a los calendarios de los conductores y visualiza un mensaje para confirmar su asignación.
- **Flujos alternativos:** no hay flujo alternativo.

Caso de uso: Visualizar estadísticos globales.

- **Descripción:** visualización de estadísticas del calendario.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de un calendario.
- **Flujo normal:**
 1. El usuario pulsa el botón visualizar estadísticas globales.
 2. El sistema descarga los datos de la demanda y los calendarios y crea los objetos conductores.
 3. El sistema calcula las estadísticas y las muestra en una vista.
- **Post condiciones:** se visualizan las estadísticas de un calendario seleccionado.
- **Flujos alternativos:** no hay flujo alternativo

Caso de uso: Guardar calendario.

- **Descripción:** guardar calendario actual.
- **Actor:** gestor
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de un calendario.

- **Flujo normal:**

1. El usuario pulsa el botón de guardar.
2. El sistema muestra un menú para preguntar si desea guardar el calendario o la demanda.
3. El usuario pulsa el botón de guardar calendario.
4. El sistema muestra una vista para que el usuario escriba el nombre del guardado en la base de datos.
5. El usuario escribe un nombre de guardado y pulsa el botón guardar.
6. El sistema guarda el calendario anual en la base de datos en una tabla con el nombre especificado.
7. El sistema vuelve a mostrar la vista de la visualización del calendario previamente guardado.

- **Post condiciones:** el usuario guarda el calendario en la base de datos con un nombre concreto.

- **Flujos alternativos:**

- 5b. El sistema tiene un error en el guardado.
- 6b. El sistema muestra un mensaje de error y permanece en esa vista.

Caso de uso: Guardar demanda.

- **Descripción:** guardar la demanda actual.

- **Actor:** gestor

- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de un calendario.

- **Flujo normal:**

1. El usuario pulsa el botón de guardar.
2. El sistema muestra un menú para preguntar si desea guardar el calendario o la demanda.
3. El usuario pulsa el botón de guardar demanda.
4. El sistema muestra una vista para que el usuario escriba el nombre del guardado en la base de datos.
5. El usuario escribe un nombre de guardado y pulsa el botón guardar.
6. El sistema guarda la demanda anual en la base de datos en una tabla con el nombre especificado.
7. El sistema vuelve a mostrar la vista de la visualización del calendario previamente guardado.

- **Post condiciones:** el usuario guarda la demanda en la base de datos con un nombre concreto.

- **Flujos alternativos:**

- 5b. El sistema tiene un error en el guardado.
- 6b. El sistema muestra un mensaje de error y permanece en esa vista.

Caso de uso: Modificar opciones.

- **Descripción:** se quieren modificar las opciones de creación de calendarios.

- **Actor:** gestor.

- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en el menú principal.

- **Flujo normal:**

1. El usuario pulsa el botón de opciones.
2. El sistema descarga el último guardado de opciones realizado.
3. El sistema muestra una página de edición de opciones donde muestra las opciones descargadas previamente.
4. El usuario modifica las opciones deseadas y pulsa el botón de guardar y salir.
5. El sistema guarda las opciones con junto a la fecha de realización.
6. El sistema vuelve a mostrar el menú principal.

- **Post condiciones:** se guardan en la base de datos las nuevas opciones de creación y añade la fecha de realización.

- **Flujos alternativos:** no hay flujo alternativo.

Caso de uso: Ver reclamaciones.

- **Descripción:** ver las reclamaciones realizadas por los conductores.

- **Actor:** gestor.

- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en el menú principal.

- **Flujo normal:**

1. El usuario pulsa en ver reclamaciones.
2. El sistema descarga de la base de datos todas las notificaciones.
3. El sistema muestra una página con las notificaciones.

- **Post condiciones:** se visualizan todas las reclamaciones realizadas junto a su fecha y el conductor que las ha realizado.

- **Flujos alternativos:** no hay flujo alternativo.

Caso de uso: Eliminar reclamación.

- **Descripción:** eliminar una reclamación de un conductor.

- **Actor:** gestor

- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de reclamaciones.

■ Flujo normal:

1. El usuario pulsa en eliminar reclamación.
2. El sistema borra la reclamación señalada previamente de la base de datos.
3. El sistema descarga las notificaciones de la base de datos.
4. El sistema muestra una página visualizando todas las notificaciones.

■ **Post condiciones:** se continúa en la visualización de las reclamaciones pero se ha eliminado la reclamación marcada.

■ **Flujos alternativos:** no hay flujos alternativos.

Caso de uso: Modificar conductores.

■ **Descripción:** modificar los usuarios y conductores de la base de datos.

■ **Actor:** gestor.

■ **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en el menú principal.

■ Flujo normal:

1. El usuario pulsa en modificar conductores.
2. El sistema descarga los datos de los conductores de la base de datos.
3. El sistema muestra la página con los conductores junto con un menú para poder gestionarlos.

■ **Post condiciones:** visualiza todos los conductores de la base de datos y opciones para gestionarlos.

■ **Flujos alternativos:** no hay flujo alternativo.

Caso de uso: Editar conductor.

■ **Descripción:** editar la información conductor.

■ **Actor:** gestor.

■ **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de los conductores.

■ Flujo normal:

1. El usuario pulsa en editar conductor.
2. El sistema descarga los datos del conductor seleccionado de la base de datos.
3. El sistema muestra una página donde se muestran los datos del usuario y es posible su modificación.
4. El usuario rellena los campos del usuario y pulsa guardar.
5. El sistema guarda los datos del usuario editado en la base de datos.
6. El sistema muestra de nuevo la visualización de la página de los conductores.

■ **Post condiciones:** se guardan en la base de datos las modificaciones realizadas al conductor.

- **Flujos alternativos:**

- 5b. El sistema tiene un error al guardar el usuario.
- 6b. El sistema muestra un mensaje de error en la edición del usuario.

Caso de uso: Eliminar conductor.

- **Descripción:** se quiere eliminar un conductor.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de los conductores.
- **Flujo normal:**
 1. El usuario pulsa en eliminar conductor.
 2. El sistema borra al conductor seleccionado de la base de datos.
 3. El sistema muestra de nuevo la visualización de la página de los conductores.
- **Post condiciones:** se elimina el conductor de la base de datos.
- **Flujos alternativos:** no hay flujo alternativo.

Caso de uso: Añadir conductor.

- **Descripción:** se quiere añadir un conductor en la base de datos.
- **Actor:** gestor.
- **Precondición:** el usuario gestor ha iniciado sesión y se encuentra en la visualización de los conductores.
- **Flujo normal:**
 1. El usuario pulsa en añadir conductor.
 2. El sistema muestra un formulario para completar los datos del conductor.
 3. El usuario rellena los datos y pulsa en el botón guardar.
 4. El sistema guarda los datos del usuario creado en la base de datos.
 5. El sistema muestra de nuevo la visualización de la página de los conductores.
- **Post condiciones:** se añade un nuevo conductor en la base de datos.
- **Flujos alternativos:**
 - 4b. El sistema no guarda los datos debido a un error.
 - 5b. El sistema muestra un mensaje de error en la creación del usuario.

3.4.3. Diagramas de secuencia de los casos de uso

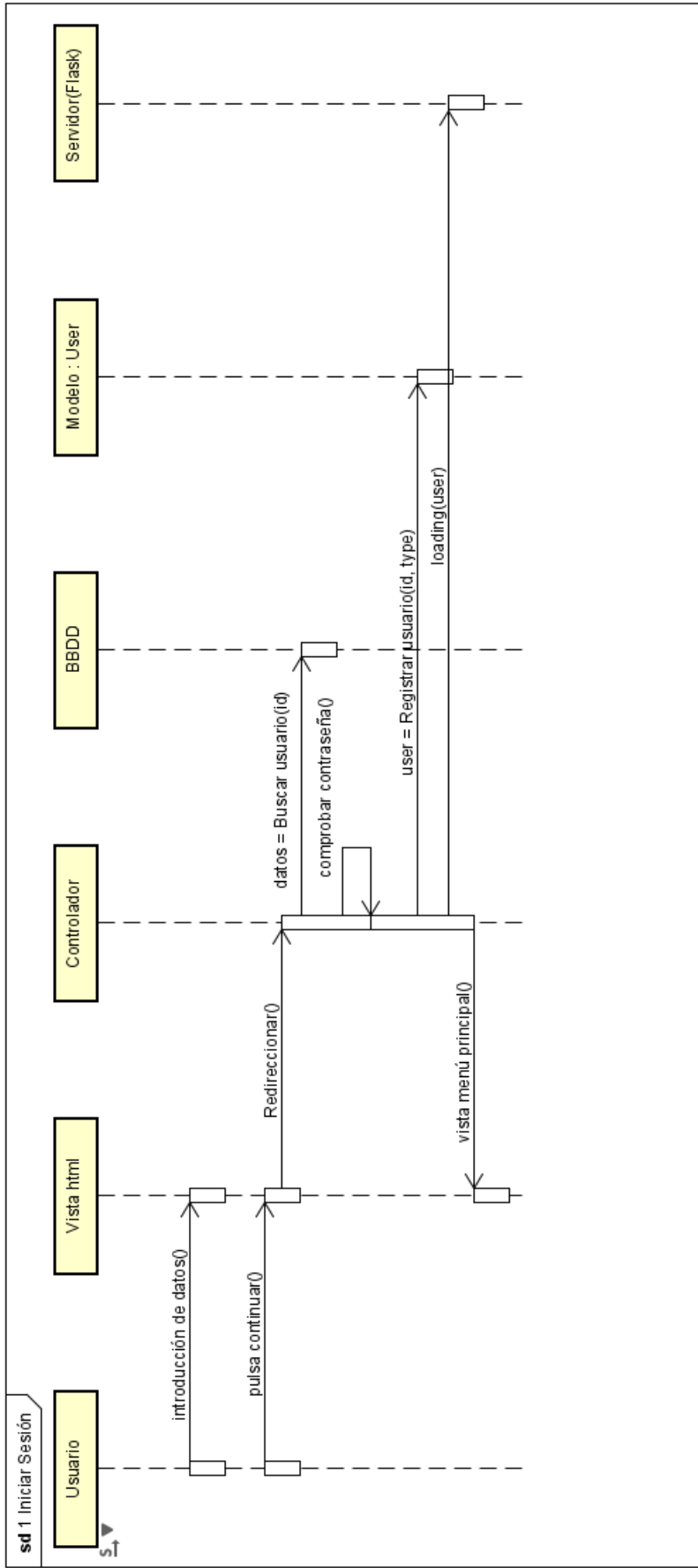


Figura 14: Diagrama de secuencia del inicio de sesión.

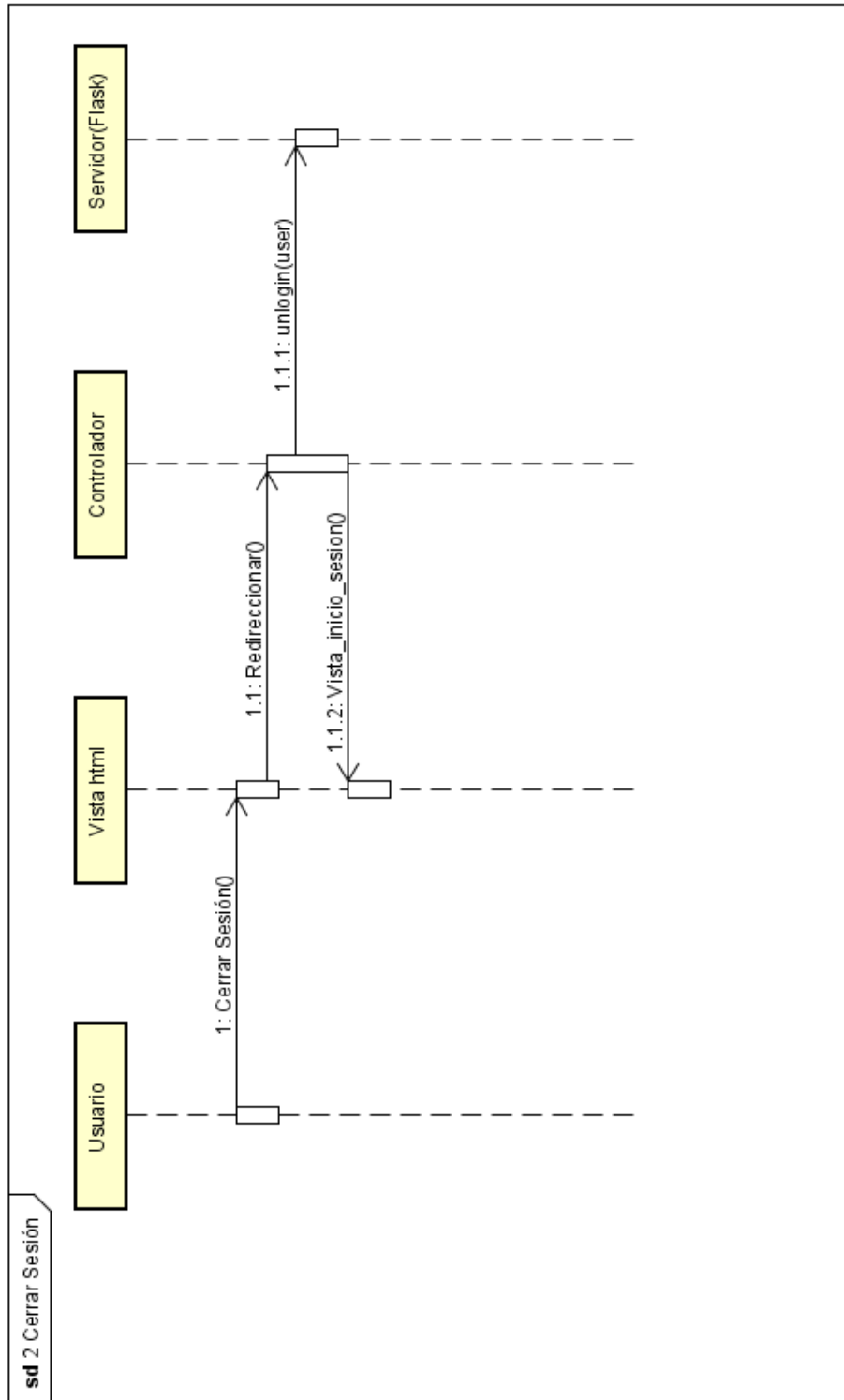


Figura 15: Diagrama de secuencia del cierre de sesión.

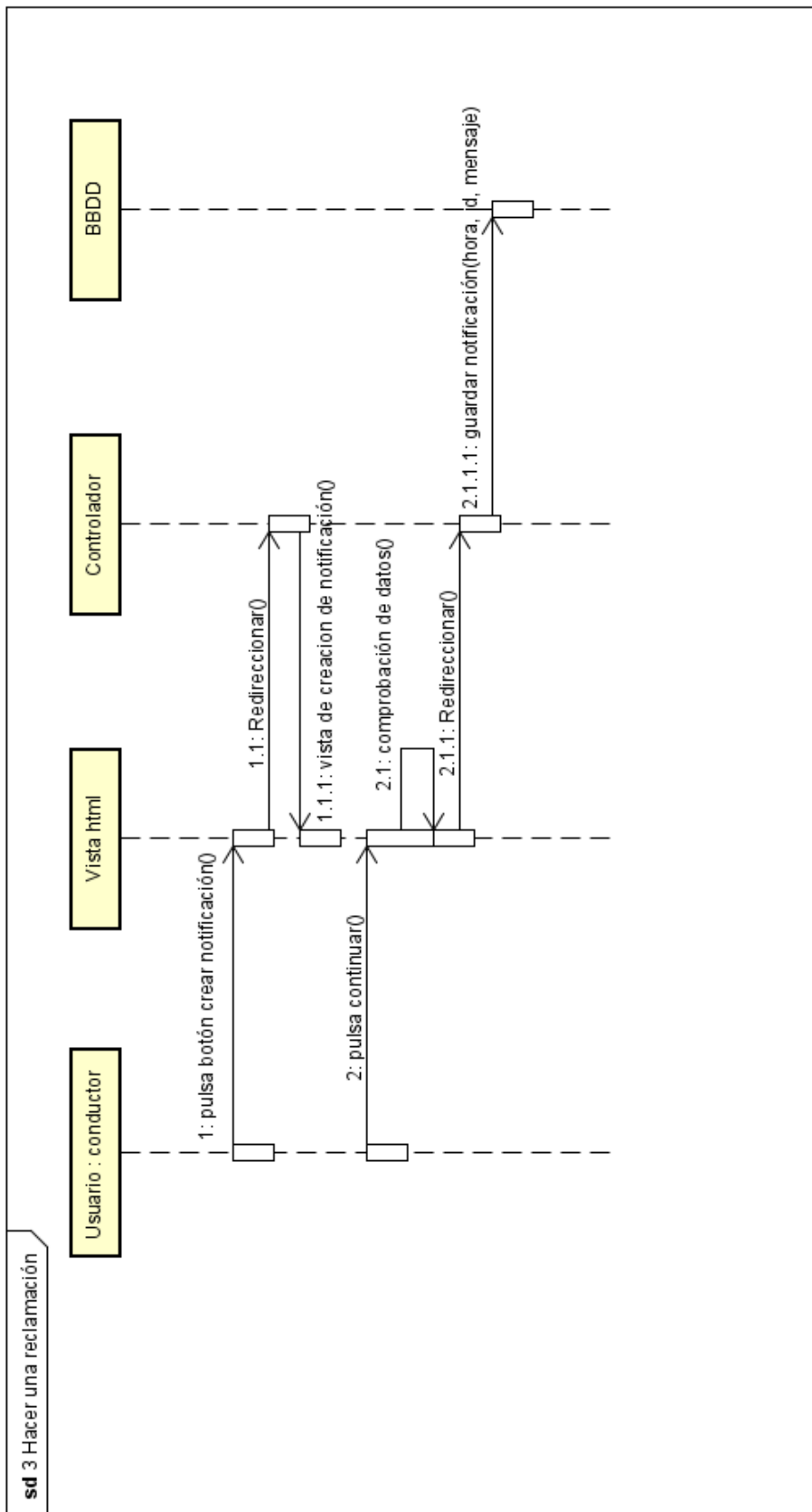


Figura 16: Diagrama de secuencia de crear una reclamación.

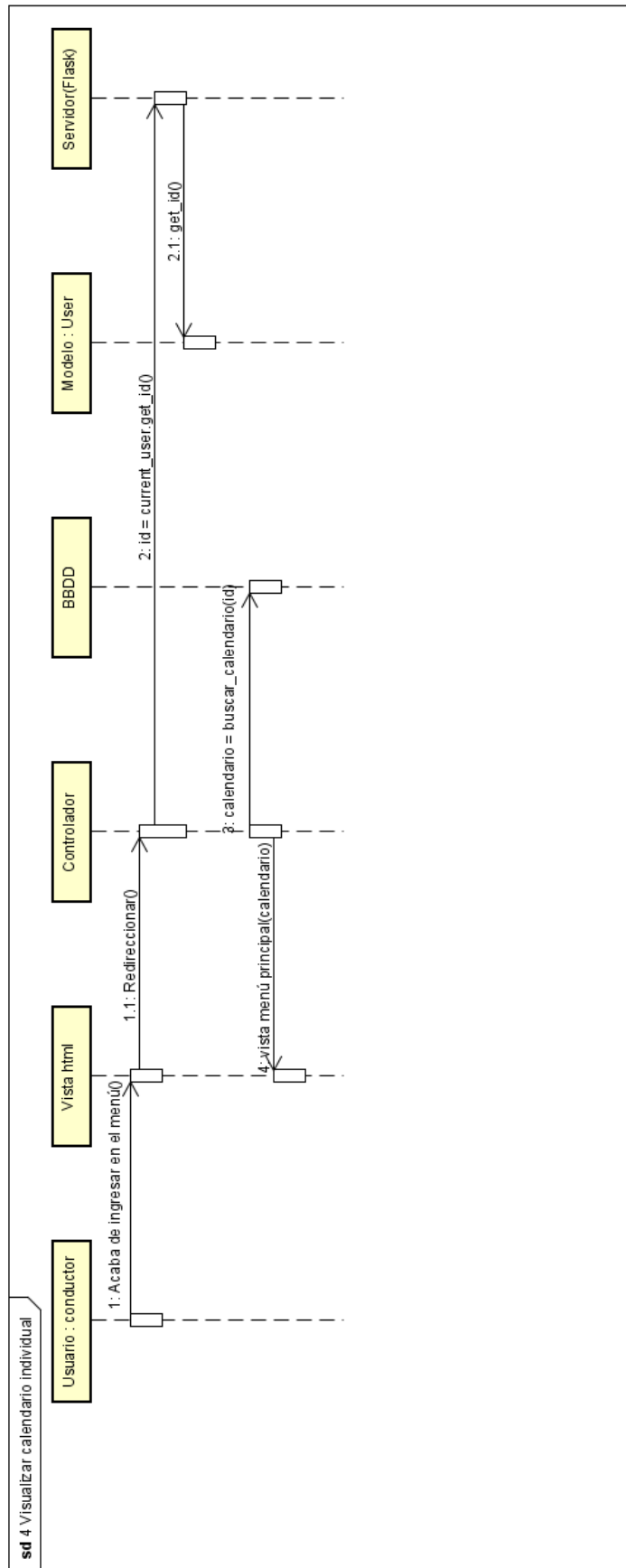


Figura 17: Diagrama de secuencia de la visualización del calendario individual.

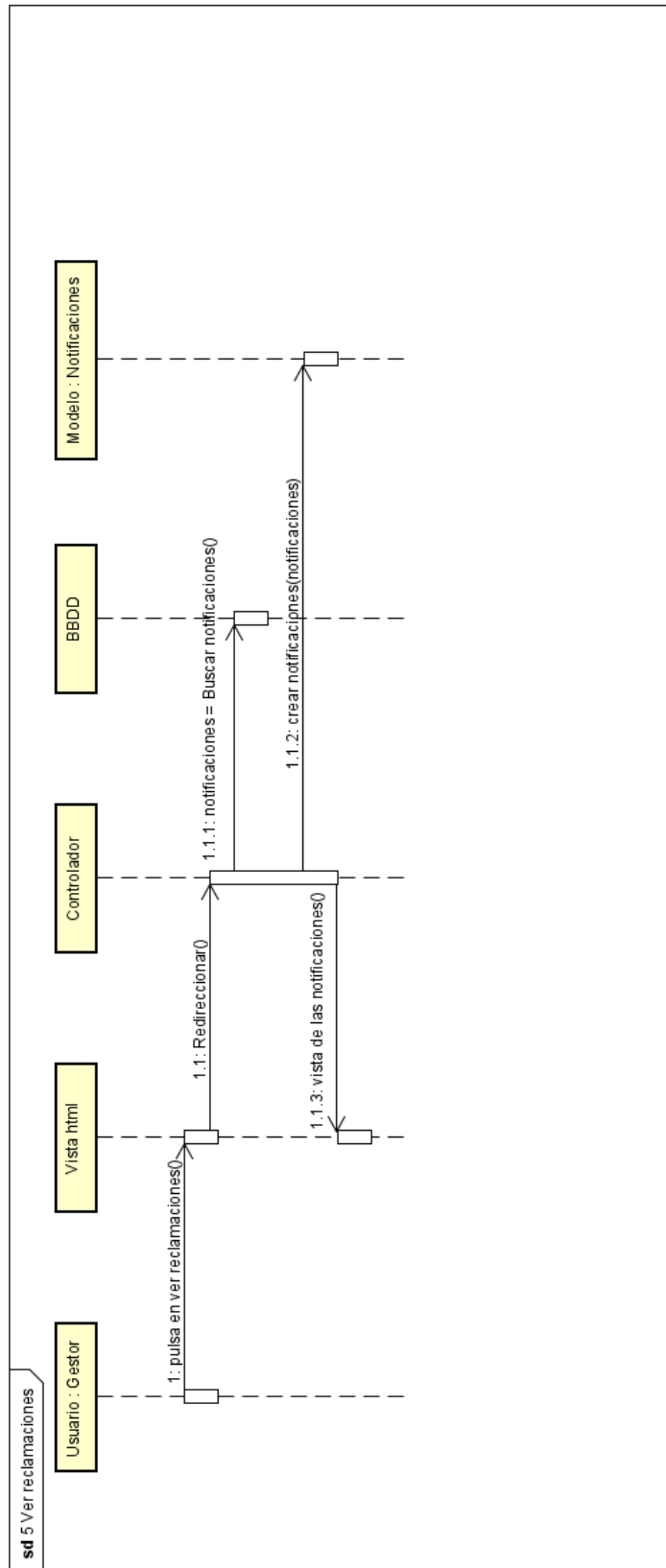


Figura 18: Diagrama de secuencia de la visualización de reclamaciones.

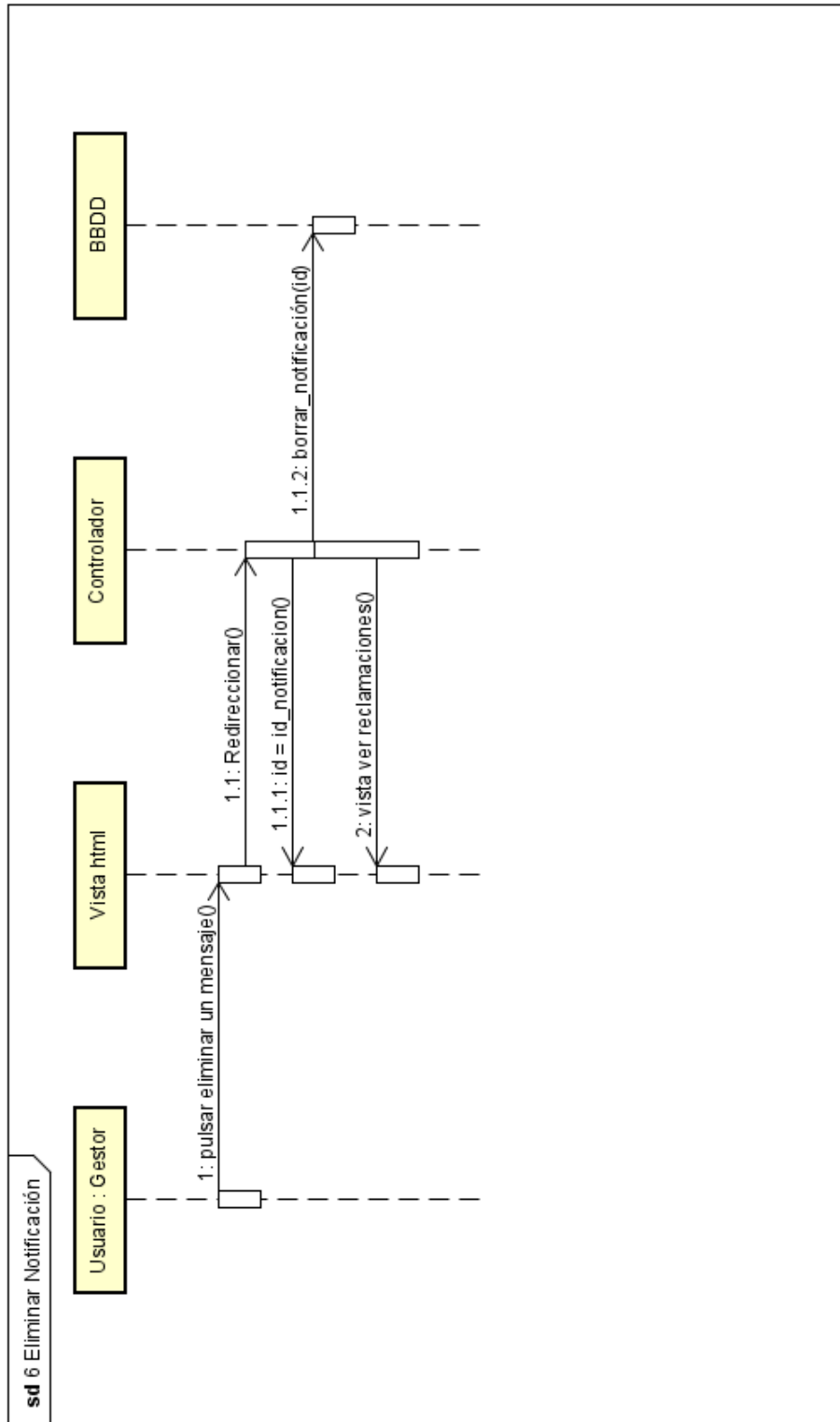


Figura 19: Diagrama de secuencia de eliminación de reclamaciones.

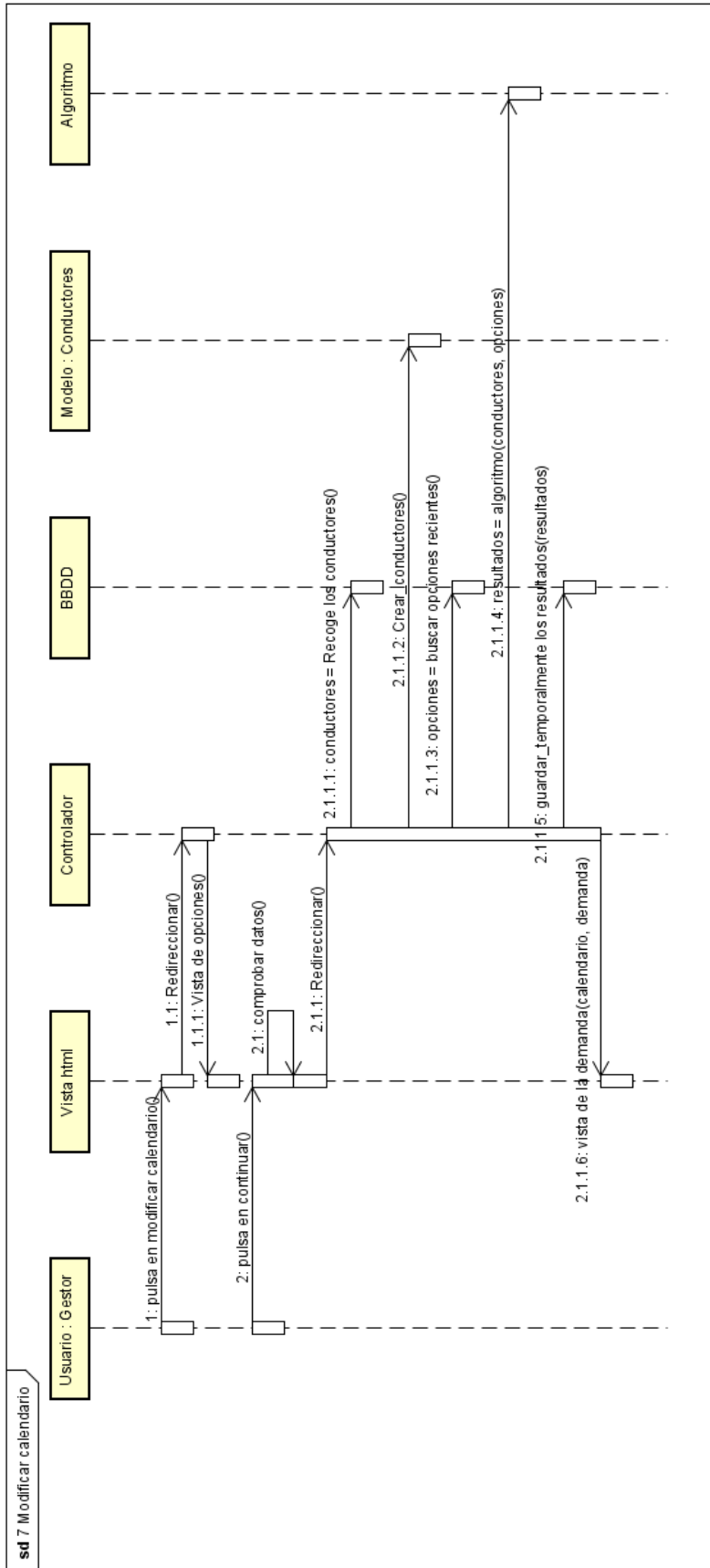


Figura 20: Diagrama de secuencia de la modificación del calendario.

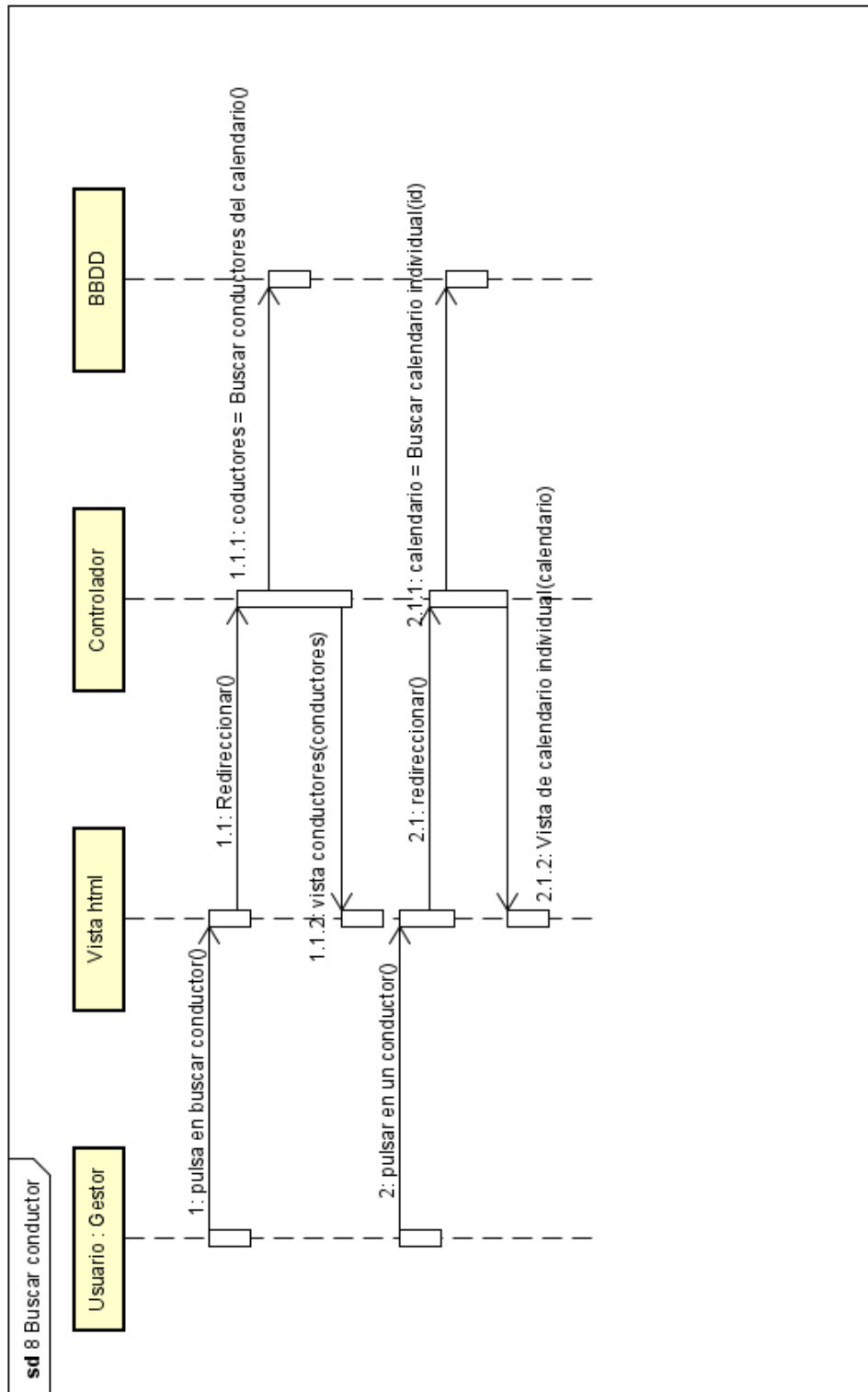


Figura 21: Diagrama de secuencia de la búsqueda de conductores.

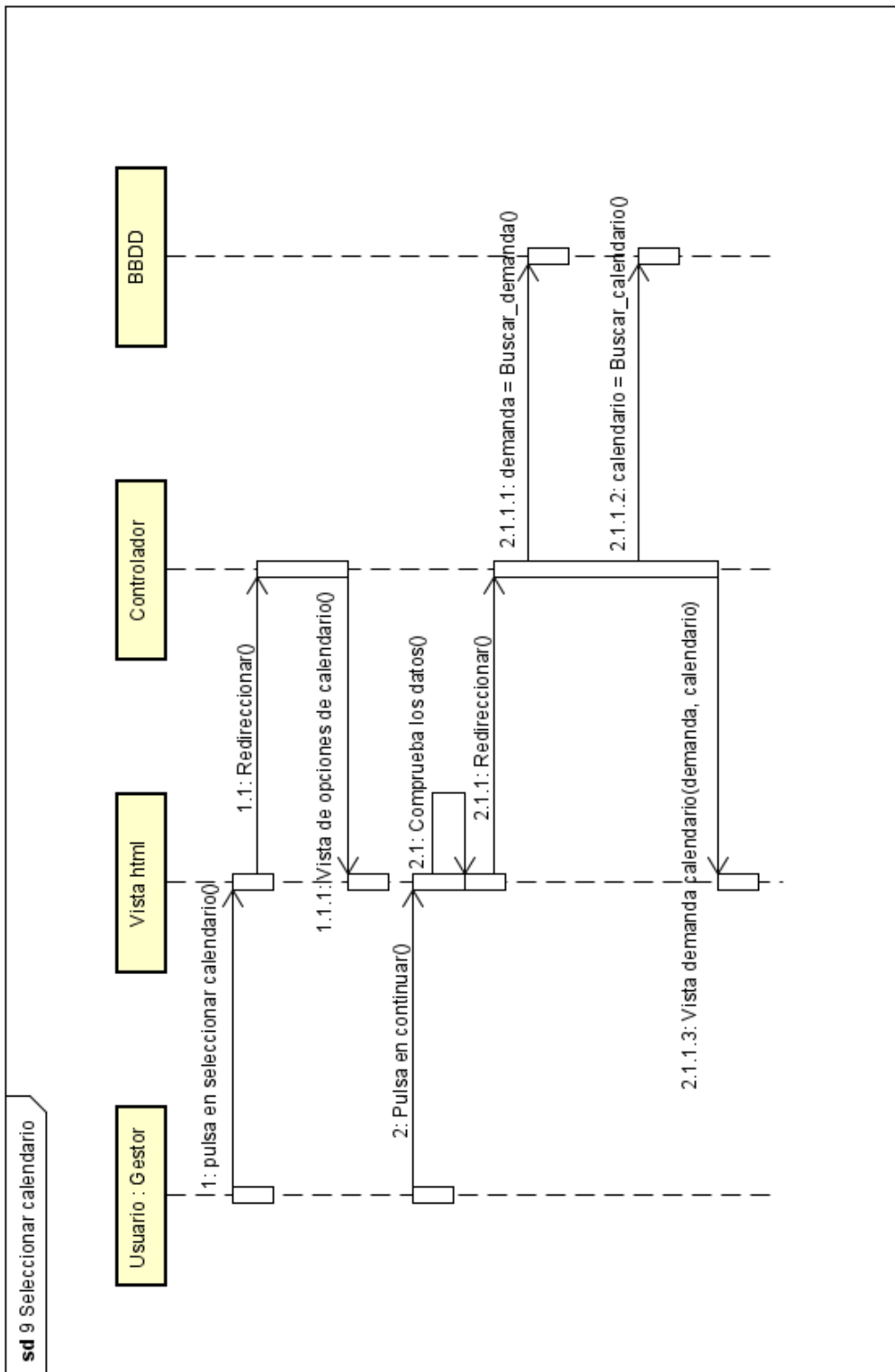


Figura 22: Diagrama de secuencia de la selección de calendario.

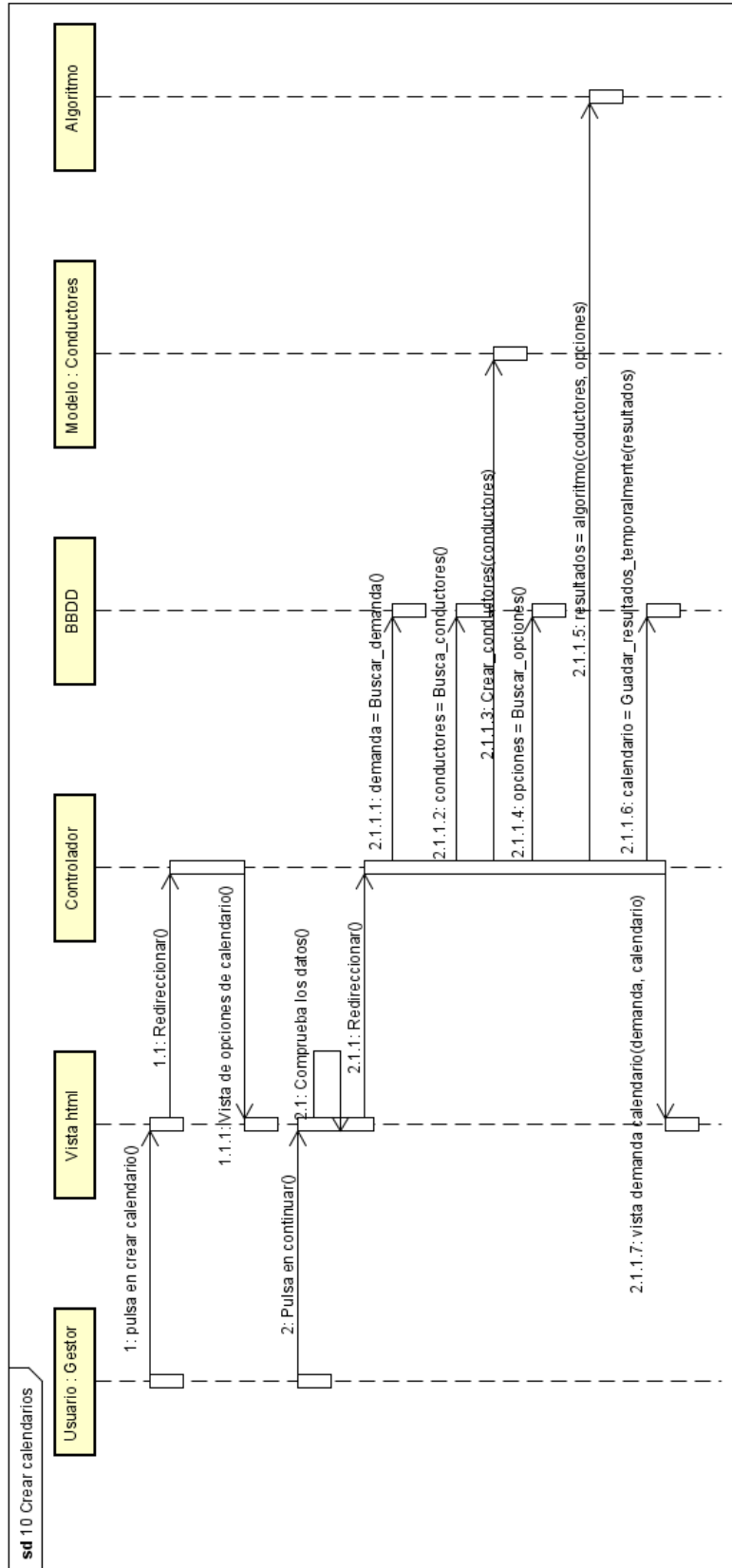


Figura 23: Diagrama de secuencia de la creación de calendarios.

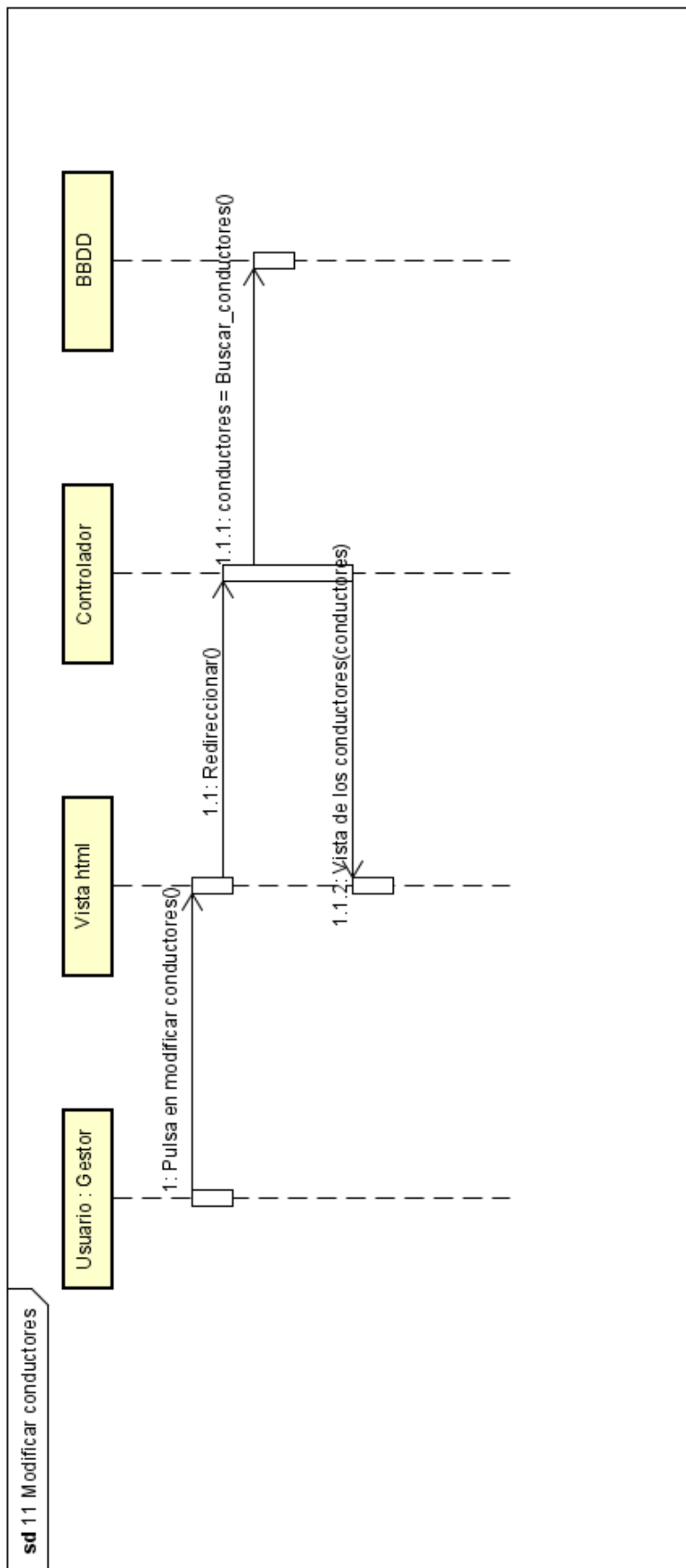


Figura 24: Diagrama de secuencia de la modificación de los conductores.

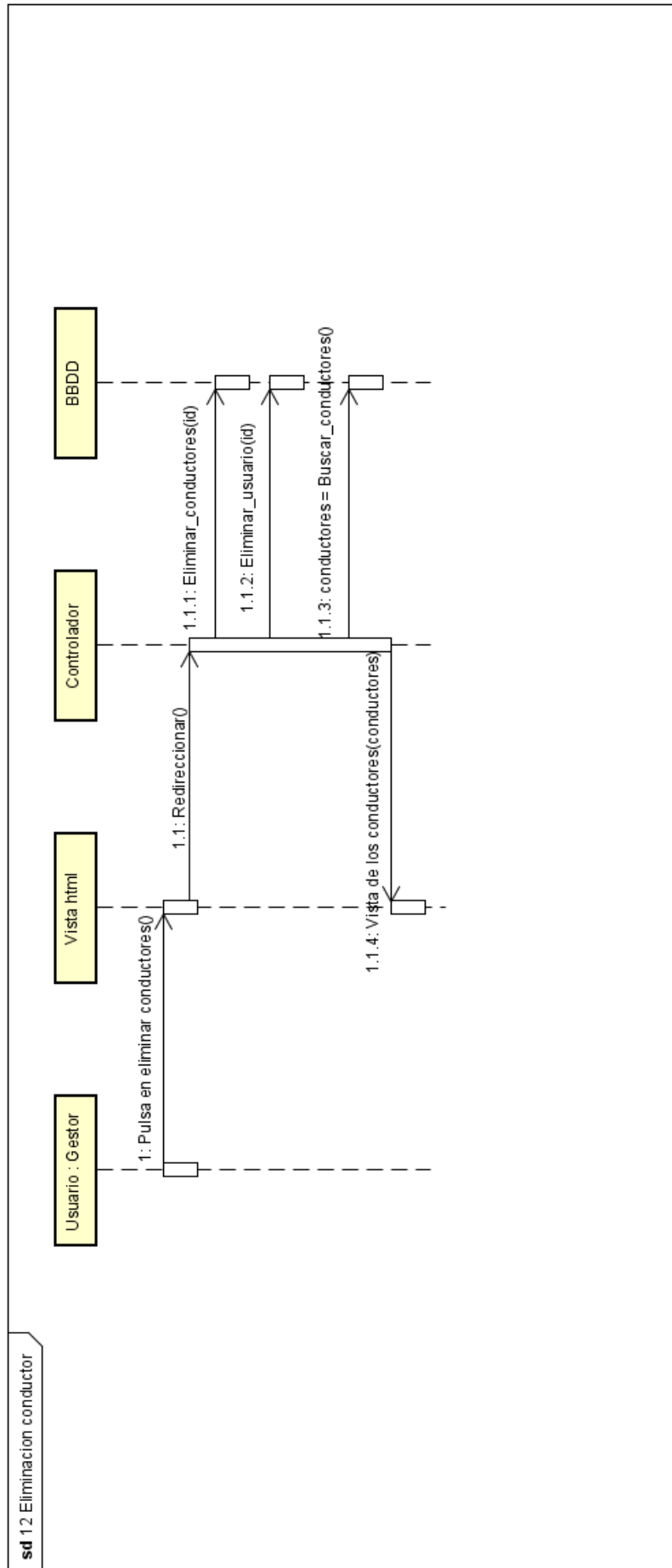


Figura 25: Diagrama de secuencia de la eliminación de conductores.

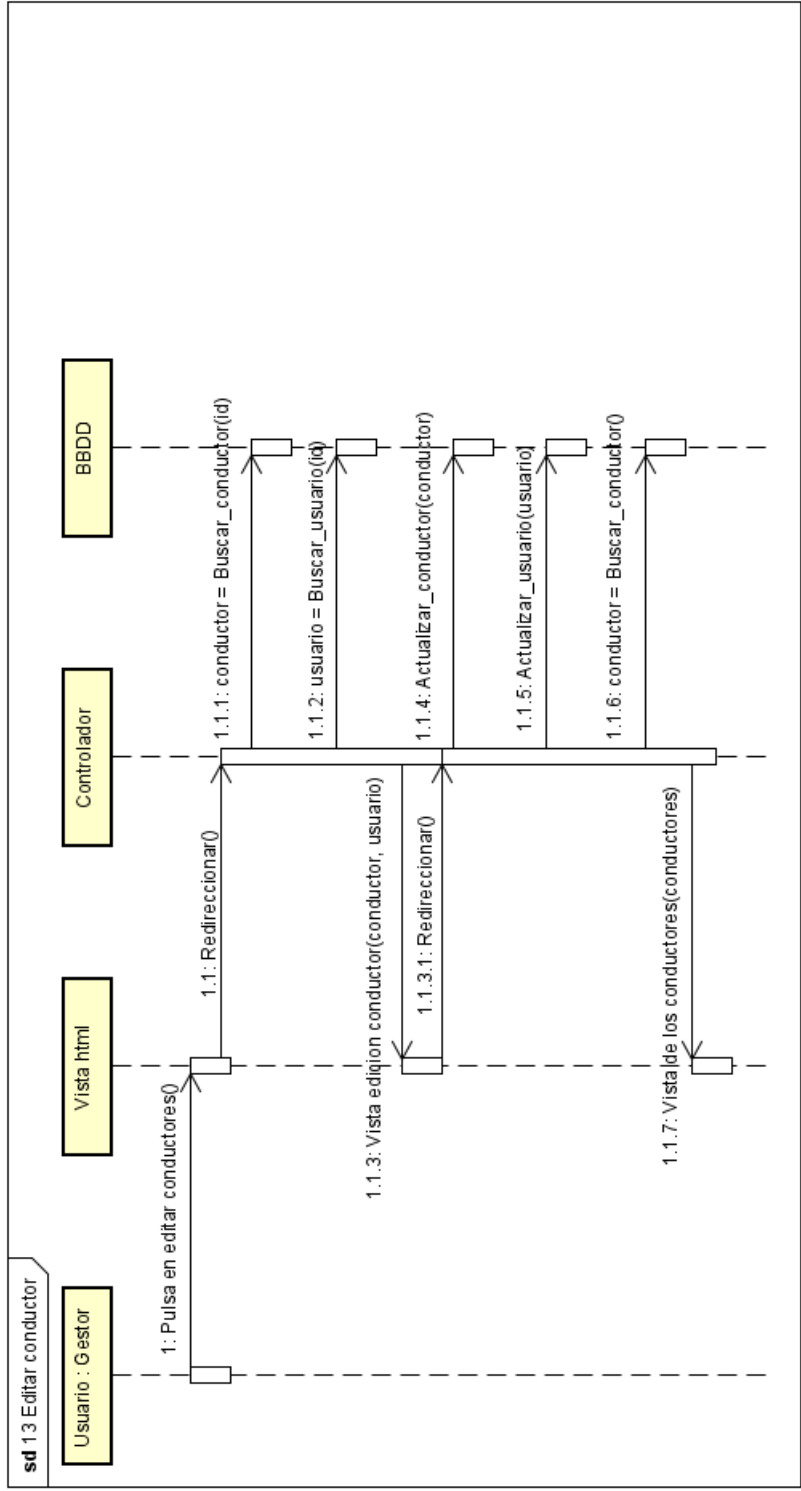


Figura 26: Diagrama de secuencia de la edición de conductores.

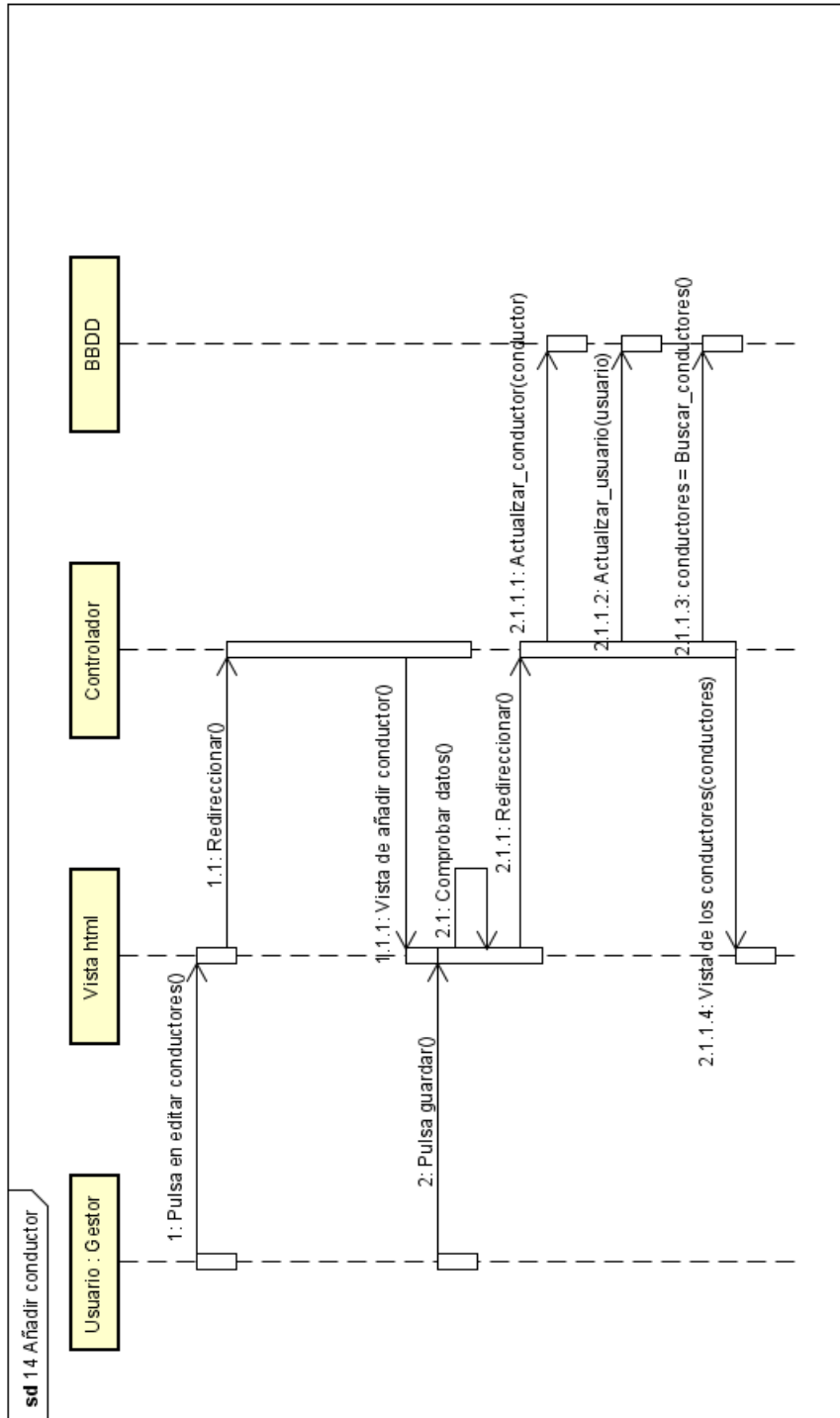


Figura 27: Diagrama de secuencia de la adición de conductores.

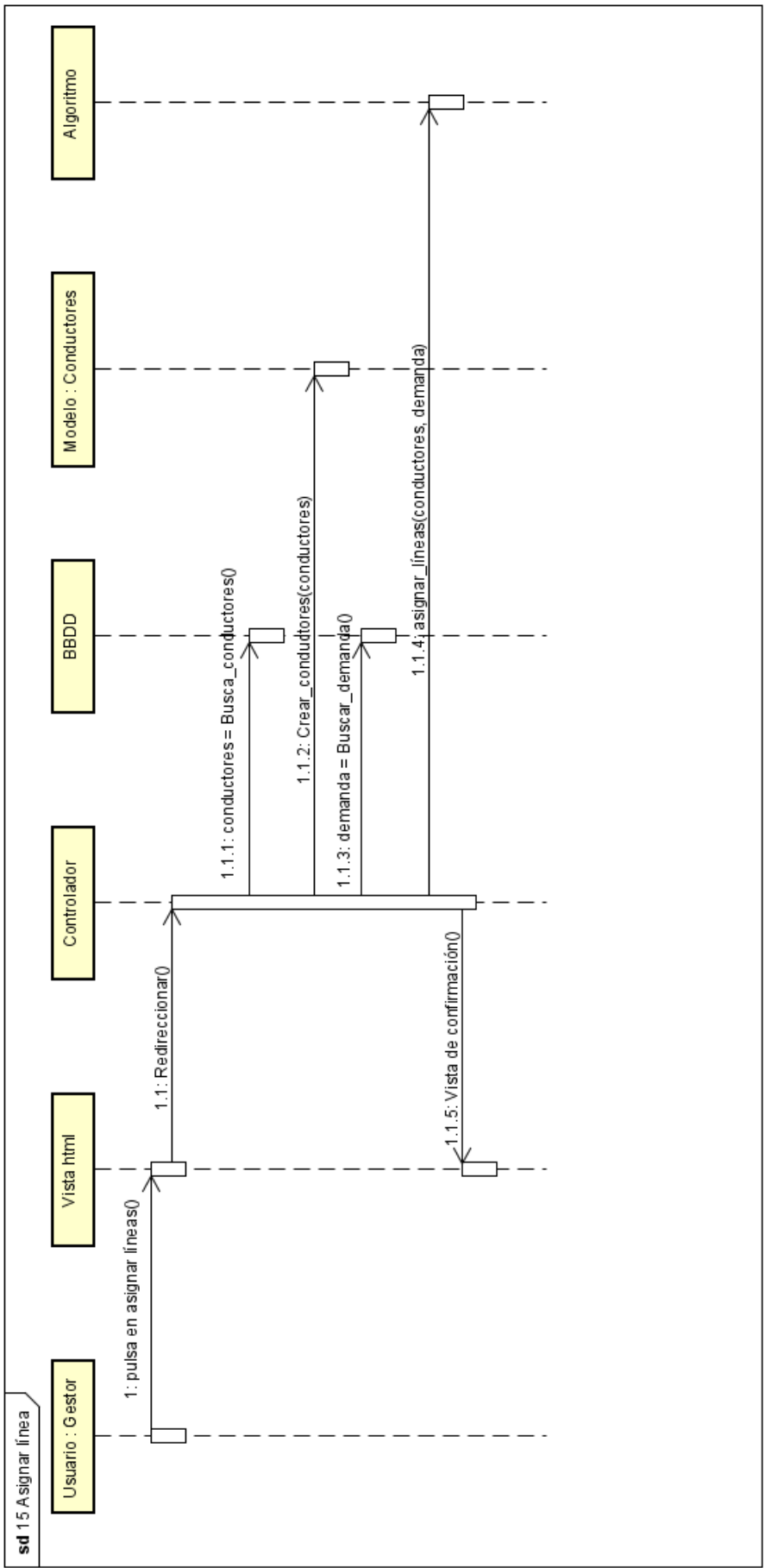


Figura 28: Diagrama de secuencia de la asignación de líneas.

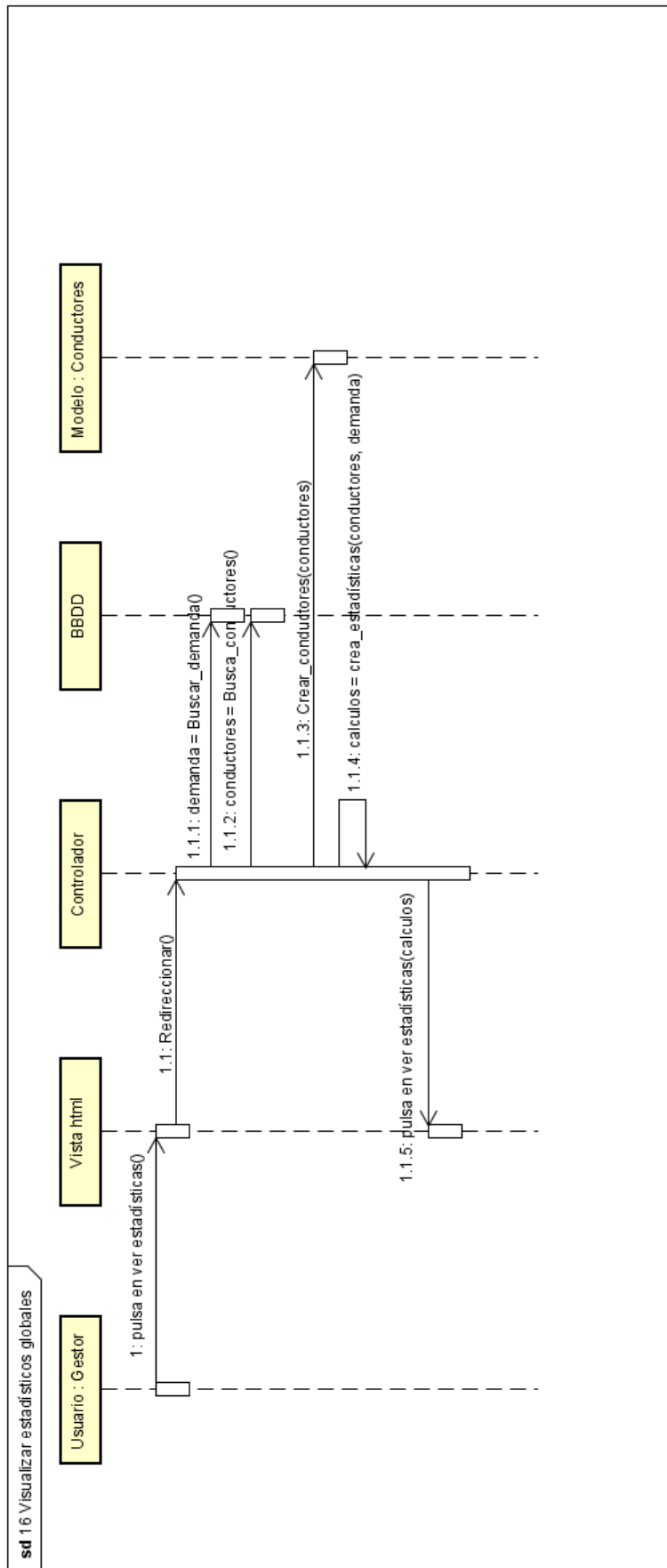


Figura 29: Diagrama de secuencia de la visualización de estadísticas globales del calendario.

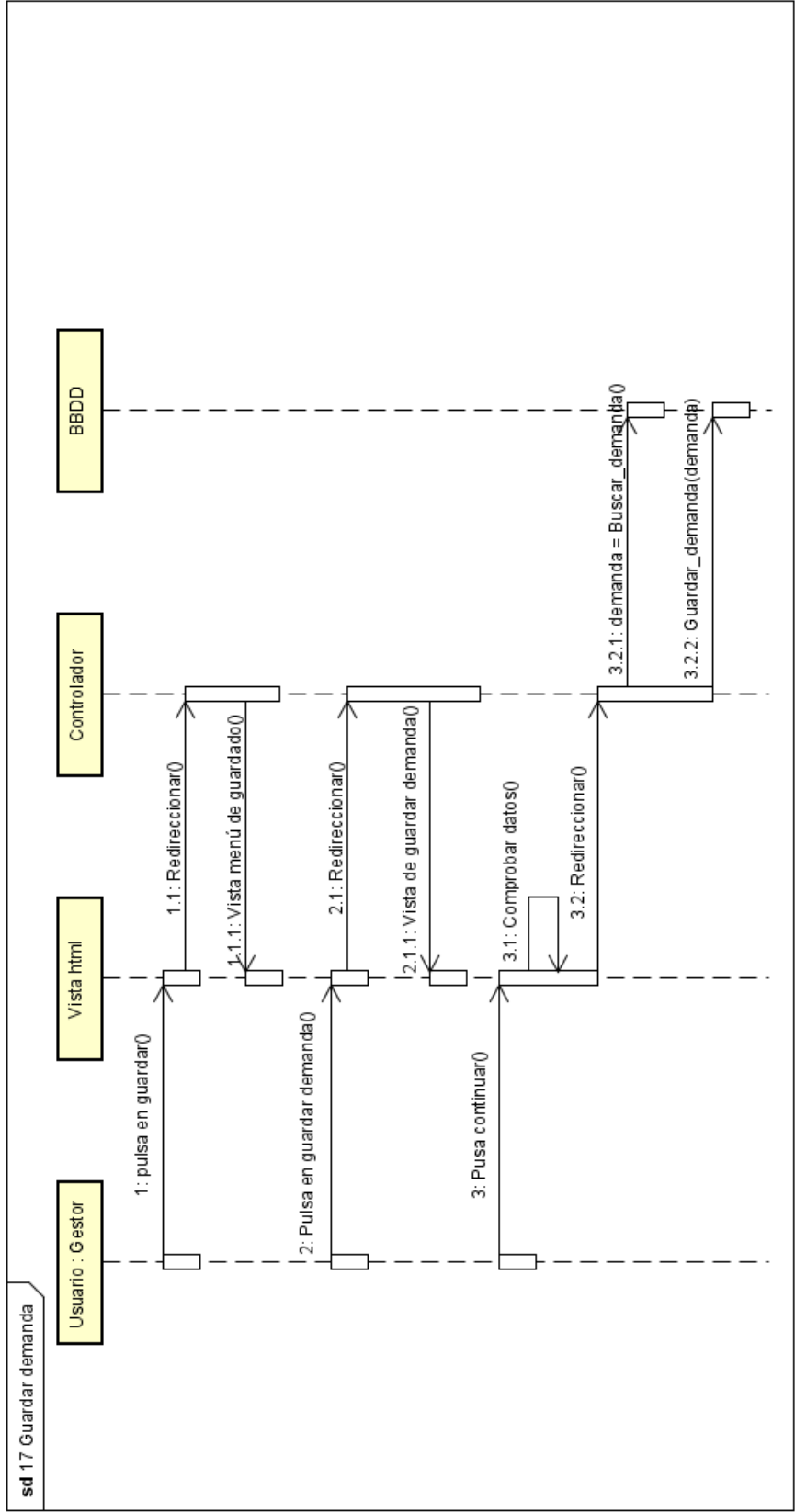


Figura 30: Diagrama de secuencia de guardar la demanda del calendario.

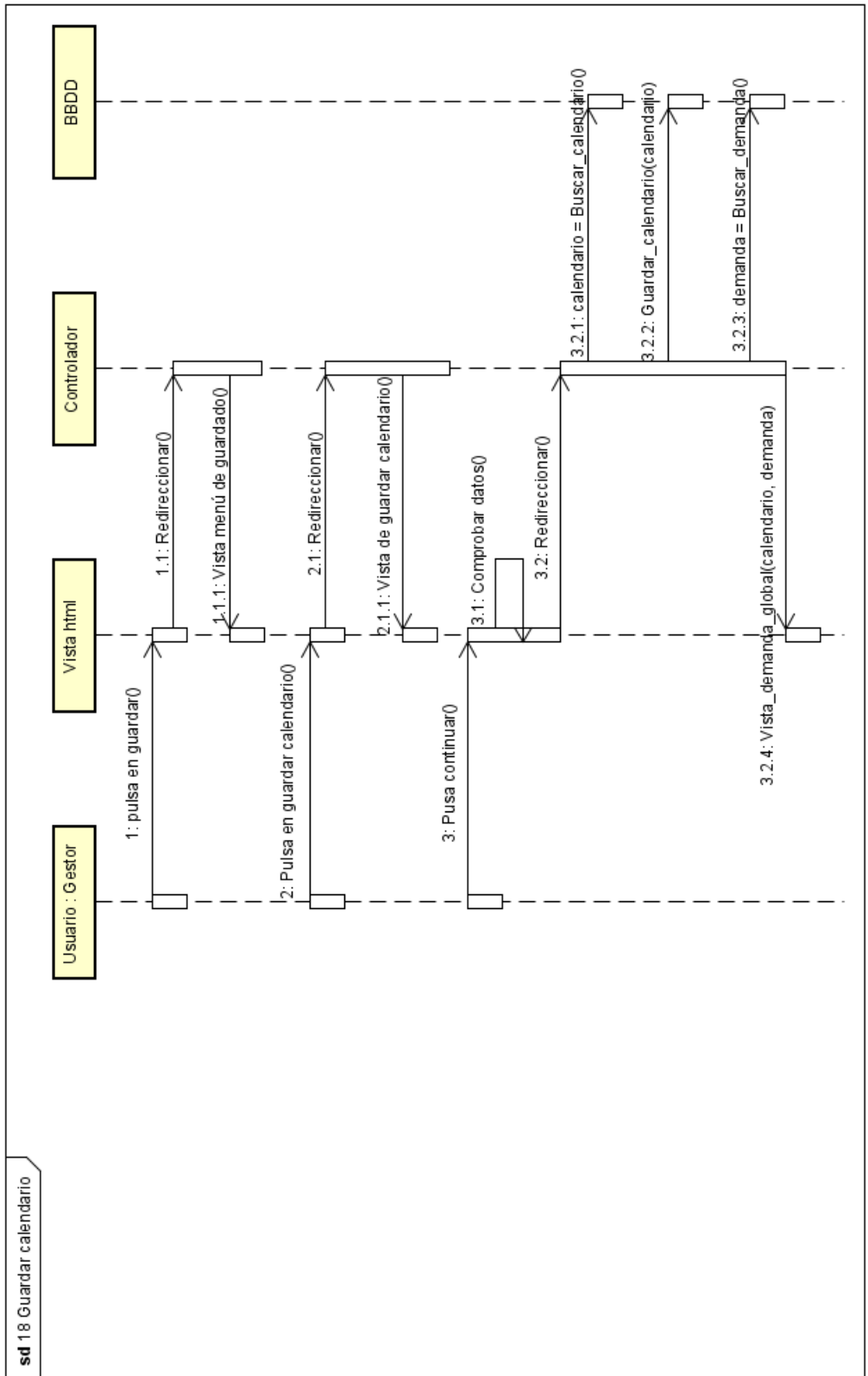


Figura 31: Diagrama de secuencia de guardar los turnos de trabajo del calendario.

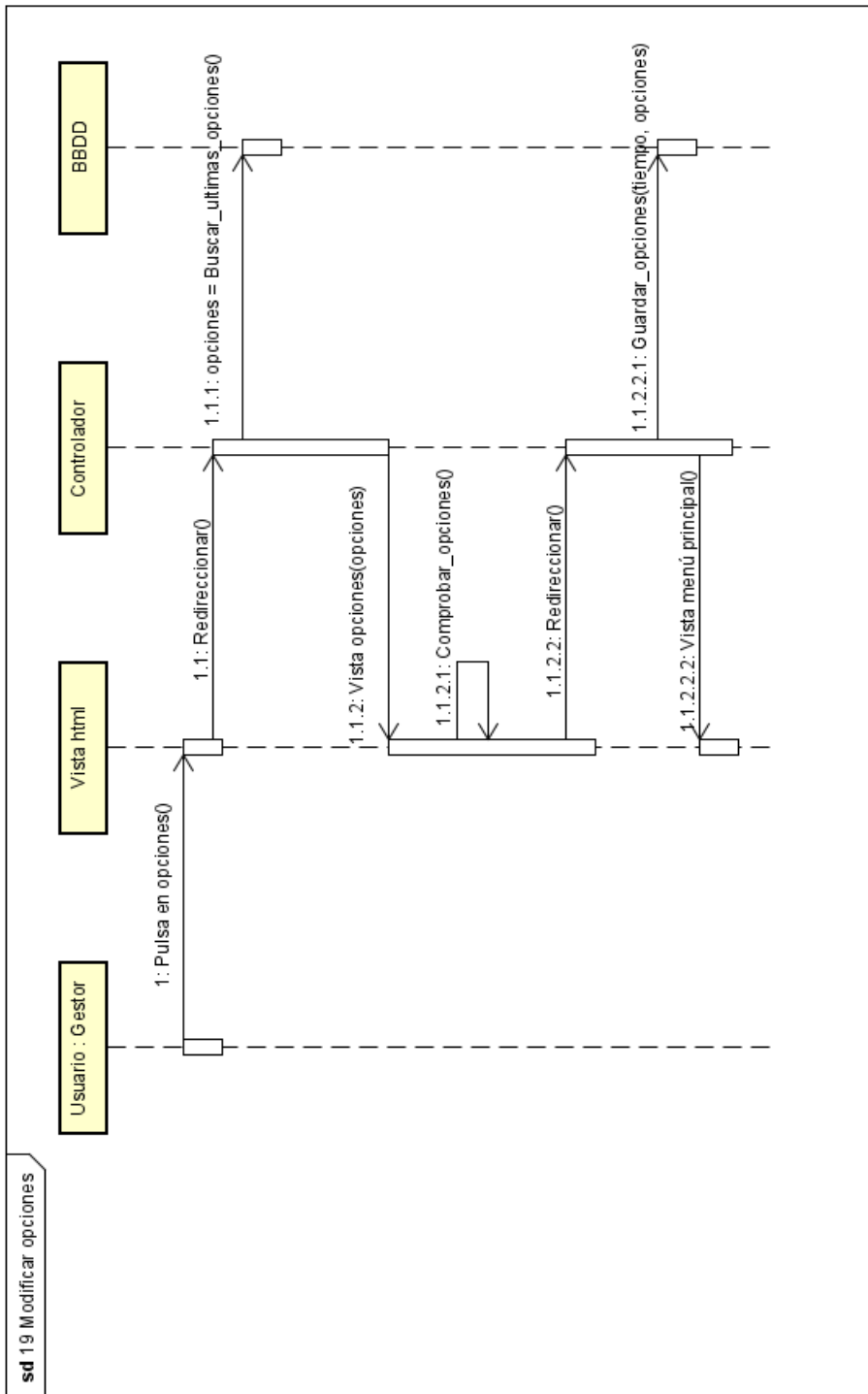


Figura 32: Diagrama de secuencia de la modificación de opciones de creación.

3.5. Diagrama de datos

En la Figura 33 podemos observar las tablas y sus relaciones dentro de la base de datos. Tenemos también otra tabla, llamada *opciones*, que no hemos mostrado ya que no tiene una relación directa con cualquier otra tabla de este diagrama.

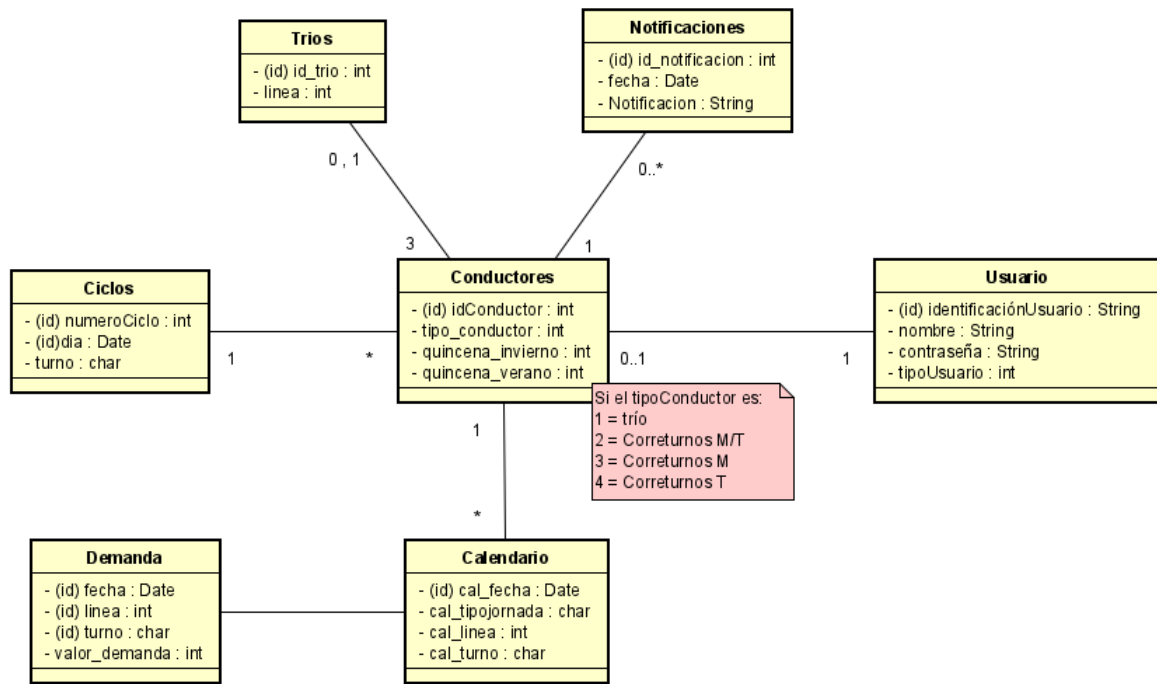


Figura 33: Diagrama de datos.

La tabla *opciones* contiene, como su propio nombre indica, las opciones de creación de los calendarios, además de un histórico de las mismas y con los siguientes atributos que son tipo INT, salvo la *fecha_ajuste* que es un DOUBLE:

1. *fecha_ajuste* DOUBLE
2. *porcentaje_trio* 3. *numero_6_dias_trabajo* 4. *minimos_dias_trabajo_seguido*
5. *maximos_dias_trabajo_seguido* 6. *minimos_dias_descanso_seguido*
7. *maximos_dias_descanso_seguido* 8. *numero_maximo_trabajo*
9. *minimos_dias_trabajo_seguido_eventuales* 10. *maximos_dias_trabajo_seguido_eventuales*
11. *minimos_dias_descanso_seguido_eventuales* 12. *maximos_dias_descanso_seguido_eventuales*
13. *numero_maximo_trabajo_eventuales*

Cada uno de los valores enteros de la tabla *opciones* es una opción de creación modificable.

3.6. Diagrama de paquetes

En la Figura 34 se muestra la división por paquetes de la aplicación. Se pueden observar, no sólo la estructura de la aplicación, sino también las asociaciones y relaciones entre las distintas partes de la misma.

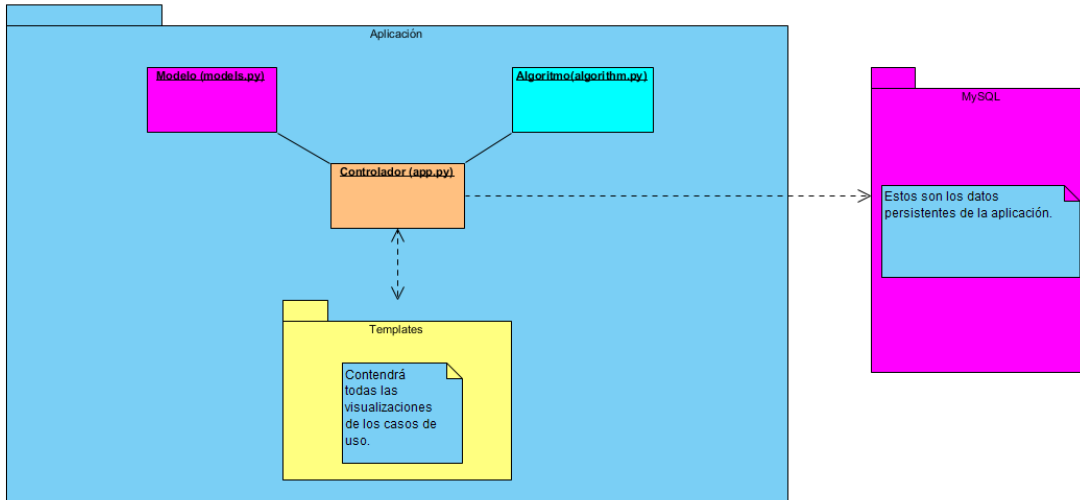


Figura 34: Diagrama de paquetes de la aplicación.

Debemos destacar que el modelo constará de un solo archivo de extensión `.py` con todas las clases del modelo de dominio en él, al archivo que hemos denominado `algoritmo.py` contiene a su vez otros dos algoritmos. Uno de ellos dedicado a la creación de calendarios anuales y el otro a la asignación de líneas a los conductores. En la sección 4) explicaremos esta última parte con más detalle. Ambos archivos se asocian con el controlador, pero no directamente entre ellos. Por otro lado, tenemos una carpeta llamada `templates` que contiene todas las vistas de la aplicación y una lógica básica de control de las mismas. También se dispone de unos datos persistentes situados en el MySQL de la aplicación, cuyo acceso y gestión se realizará a través del controlador. Por último, cabe destacar que tenemos un controlador sobrecargado ya que todas las partes del programa se relacionan gracias a él.

3.7. Bocetos de la aplicación

En esta sección mostraremos las distintas imágenes previas de cada página de la aplicación.

Para este apartado dividiremos los ejemplos de las distintas pantallas según los tipos de usuarios a los que se les mostrarán.

PARA TODOS LOS USUARIOS

El menú de inicio de sesión (Figura 35) será lo primero que muestre nuestra página para que un usuario pueda poner su acreditación e identificarse en nuestro sistema.



The image shows a wireframe of a web browser window. The title bar reads "A Web Page". The address bar contains "https://". The main content area displays the text "INICIO DE SESIÓN" in bold. Below this, there are two input fields: "Usuario:" followed by a text box, and "Contraseña" followed by a text box. Below the password field is a button labeled "Entrar" with a right-pointing arrow. The browser window has standard navigation icons (back, forward, stop, home) and a search icon in the top left, and a refresh icon in the top right. A scrollbar is visible in the bottom right corner.

Figura 35: Menú de inicio de sesión.

Si se produce un error en los datos ingresados se mostrará la pantalla de la Figura (36) donde podemos ver un mensaje de error en el que se especificará la razón del error.



Figura 36: Error en el menú de inicio de sesión.

Estas son todas las pantallas que podrán compartir todos los usuarios. Una vez que el inicio de sesión tiene éxito, los conductores y el gestor, tienen distintas visualizaciones que describiremos seguidamente.

PARA LOS CONDUCTORES DE LA EMPRESA

Nada más iniciar sesión (Figura 37) cualquier conductor podrá visualizar su calendario de trabajo anual. En la parte superior de este, estarán las estadísticas generales de su calendario: número de días trabajados, domingos trabajados...

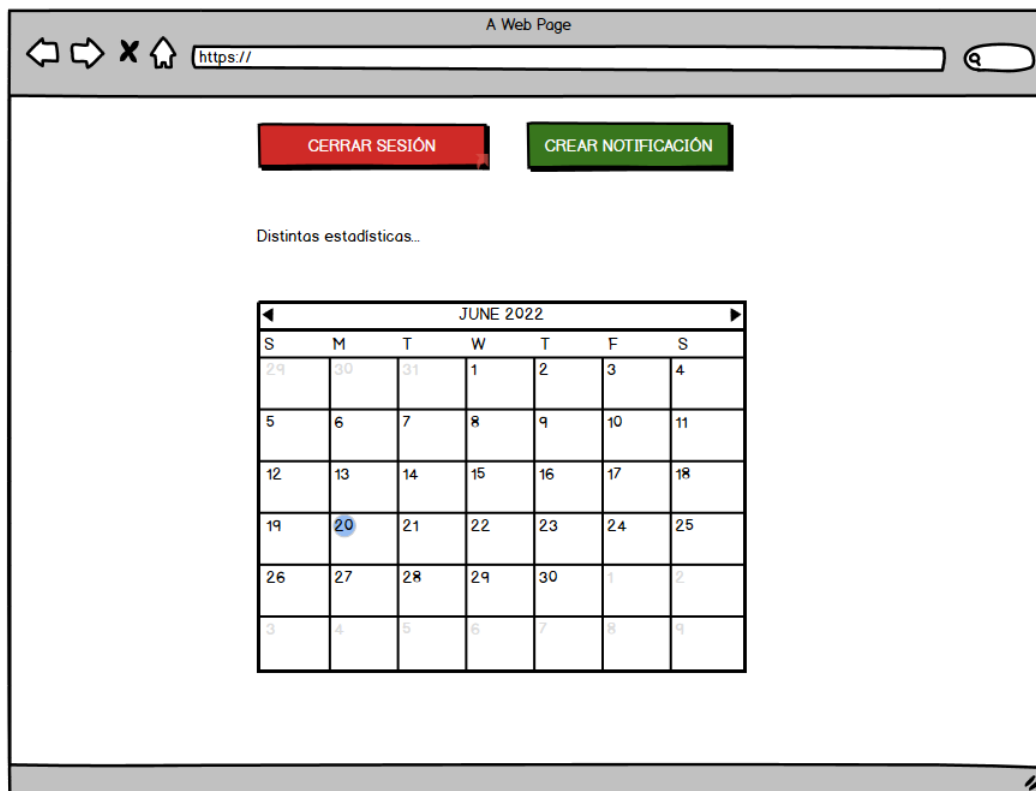


Figura 37: Menú principal de los conductores.

Además, en la parte superior de la página podrá cerrar su sesión y crear una notificación. Si cierra sesión, volverá al menú de inicio y por otro lado, si quiere, puede crear una notificación que se mostrará tal y como aparece en la Figura 38 donde el conductor puede escribir en la parte central el mensaje que desea exponer y cuando esté listo puede pulsar enviar. Si el mensaje se envía con éxito, Figura 39, se mostrará una confirmación y se mantendrá en la pestaña de crear notificaciones para poder redactar otra notificación si así lo desea. Si no ha tenido éxito el envío (Figura 40) también se mostrará un mensaje comentando el error producido.

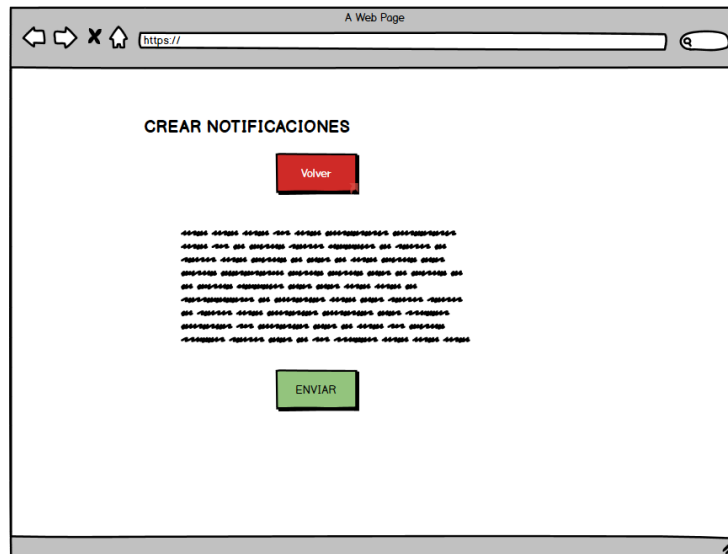


Figura 38: Pantalla de creación de una notificación.

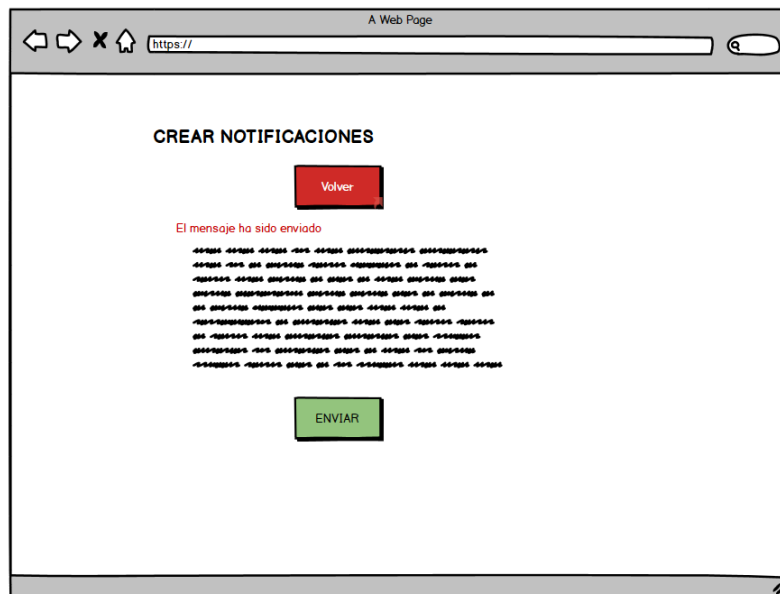


Figura 39: Pantalla al enviar con éxito una notificación.

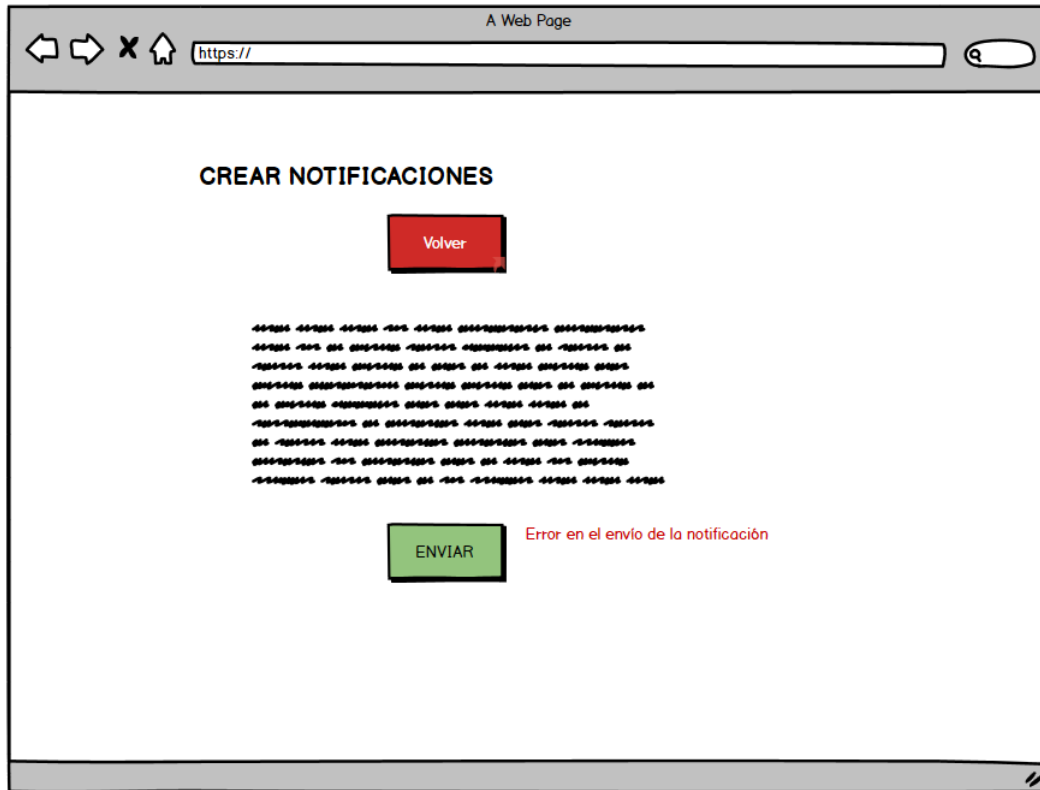


Figura 40: Pantalla al enviar sin éxito una notificación.

Si el conductor quiere salir de este menú de notificaciones tan solo tendrá que pulsar *volver* y será reenviado al menú anterior.

PANTALLAS PARA LOS GESTORES DE LA EMPRESA

Al iniciar sesión un gestor en el menú principal, se le mostrará una pantalla como la de la Figura 41. El menú dispone de los principales casos de uso que puede realizar. En la parte inferior puede cerrar su sesión, y en el centro, ligeramente a la izquierda navegar por las distintas opciones como: ver notificaciones, crear un calendario anual nuevo, seleccionar un calendario para visualizarlo, modificar los conductores y usuarios presentes en la empresa, y por último, modificar las opciones del algoritmo de creación de calendarios.

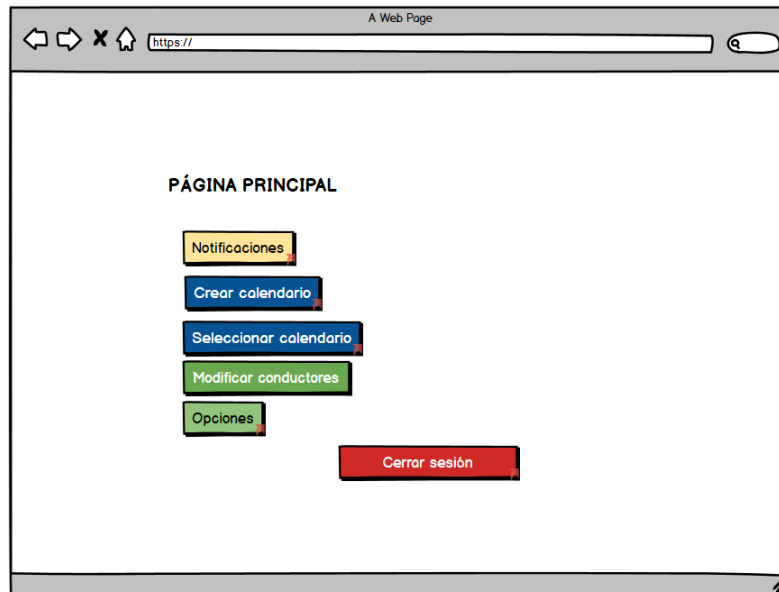


Figura 41: Menú principal de los gestores de la empresa.

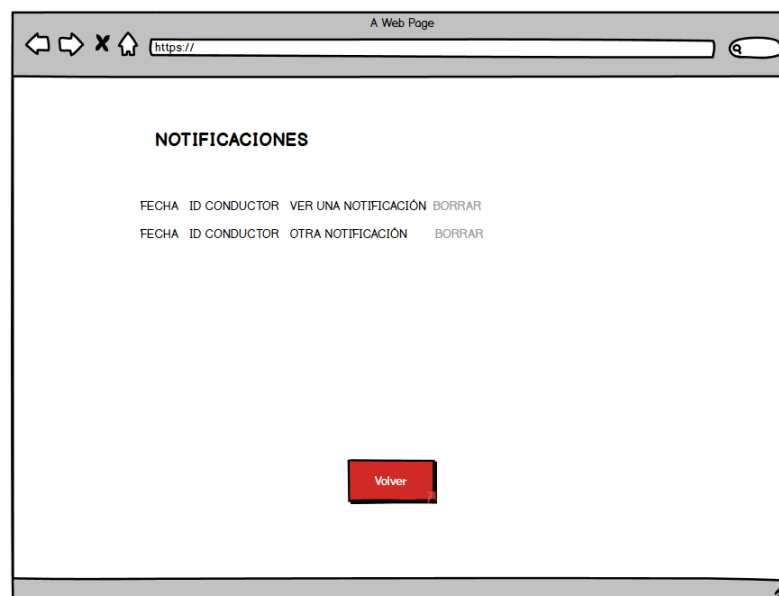
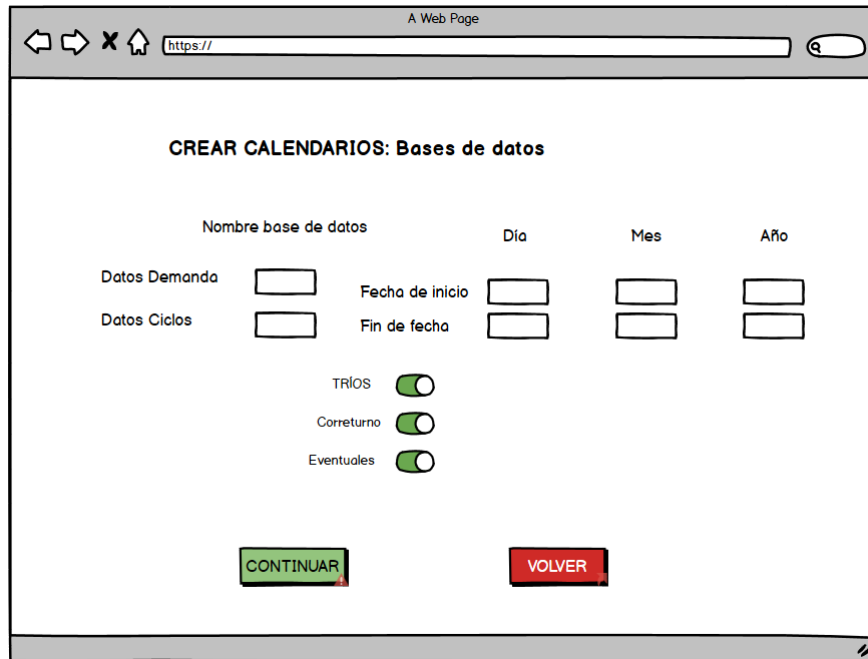


Figura 42: Página donde se muestran las notificaciones realizadas por los conductores.

Si el gestor pulsa el botón notificaciones visualizará una pantalla como la de la Figura 42 que tiene formato de tabla y podrá ver, además del mensaje, la información acerca de qué conductor ha enviado ese mensaje y en qué fecha realizó dicho envío. También podrá borrar la notificación si esta es antigua o ya no es necesaria. Por último, en la parte inferior de la página, tendrá el botón de volver, tras cuya pulsación hará que el gestor vuelva al menú principal.

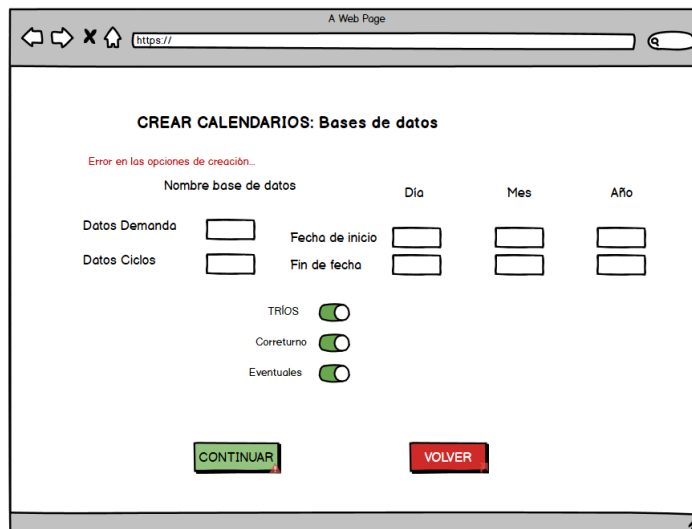
Si el gestor en el menú principal pulsa en crear calendario se mostrará la una pantalla como la de la Figura 43. En esta página, el gestor escribirá todos los datos del calendario anual, el nombre de la tabla de la demanda, el nombre de la tabla de los ciclos base utilizados, los intervalos de tiempo donde se pueden modificar los ciclos y por último la selección de los tipos de conductores utilizados en su creación.



The screenshot shows a web browser window with the address bar containing "https://". The page title is "A Web Page". The main content area is titled "CREAR CALENDARIOS: Bases de datos". Below the title, there are several input fields and toggle switches. The fields are arranged in a grid-like structure. The first row has "Nombre base de datos" followed by three empty input boxes. The second row has "Fecha de inicio" followed by three empty input boxes. The third row has "Fin de fecha" followed by three empty input boxes. Below these fields are three toggle switches labeled "TRÍOS", "Corretorno", and "Eventuales", all of which are currently turned on. At the bottom of the form, there are two buttons: "CONTINUAR" (green) and "VOLVER" (red).

Figura 43: Página donde se muestran los datos de creación de los calendarios.

Si hubiera algún error en el ingreso de los de los datos se mostrará una pantalla como la de la Figura 44.



The screenshot shows the same web browser window as Figure 43. The page title is "A Web Page". The main content area is titled "CREAR CALENDARIOS: Bases de datos". Below the title, there is a red error message: "Error en las opciones de creación...". The form fields and toggle switches are the same as in Figure 43. At the bottom of the form, there are two buttons: "CONTINUAR" (green) and "VOLVER" (red).

Figura 44: Página donde hay un error en los datos de creación de los calendarios.

Si los datos son correctos se ejecutará el algoritmo de creación de calendarios y se mostrará una página tal y como aparece en la Figura 45. En esta pantalla mostrará el calendario anual junto a las faltas y sobras de demanda del calendario obtenido. Además se mostrará, un menú de gestión del calendario. Las opciones del menú son: buscar un conductor, ver estadísticas del calendario, volver al menú, modificar calendario, asignar líneas y guardar calendario.



Nombre base de datos:

Fecha de inicio: Dia: Mes: Año:

Datos Ciclos: Fin de fecha:

TRIOS:

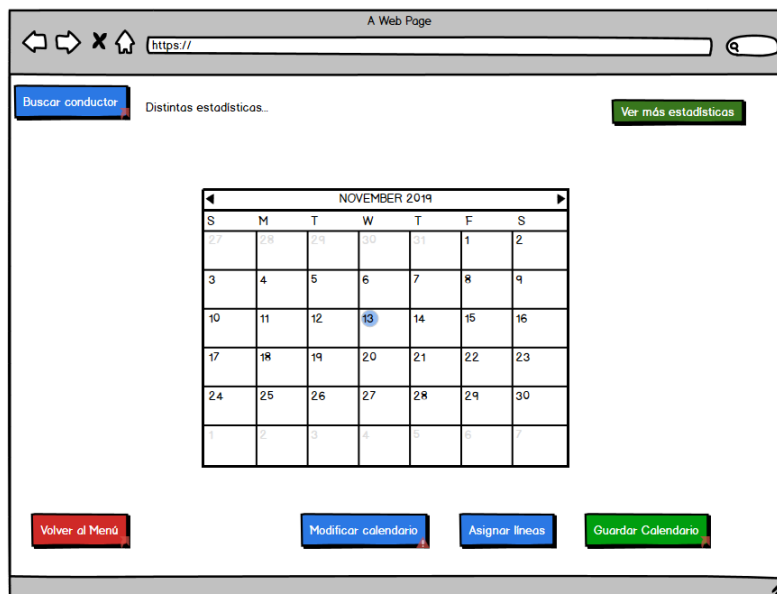
Correturno:

Eventuales:

ALGORITMO EN EJECUCIÓN

Figura 45: Página donde se está ejecutando el algoritmo de creación de los calendarios.

Cuando el algoritmo finalice se visualizará una pantalla como en la Figura 46

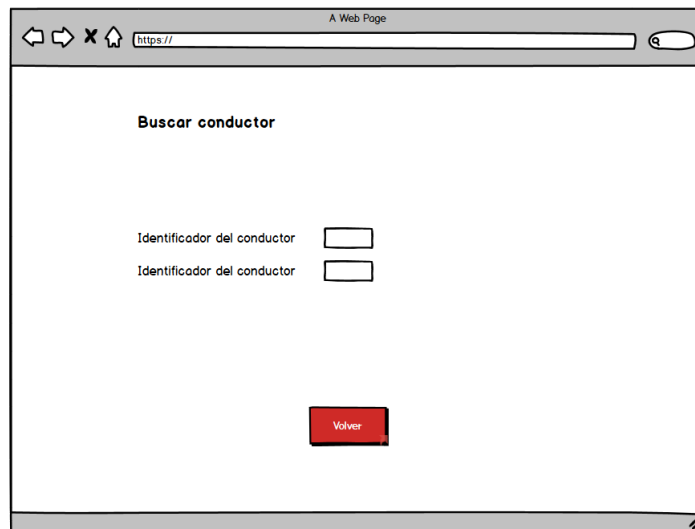


Buscar conductor Distintas estadísticas...

NOVEMBER 2019						
S	M	T	W	T	F	S
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7

Figura 46: Página de visualización de la demanda del calendario.

Si el usuario pulsa el botón *Buscar conductor*, se le abrirá un menú con los conductores que intervienen en el calendario (Figura 47) donde se mostrará una tabla con la lista de los conductores implicados en el calendario, donde el gestor podrá pulsar en el botón cercano al conductor para seleccionarlo (mostrando una vista como en la Figura 48 muy parecida al menú principal del conductor, con su calendario y algunos estadísticos de este como podemos ver en la Figura 49), o pulsar el botón para *Volver*.



Buscar conductor

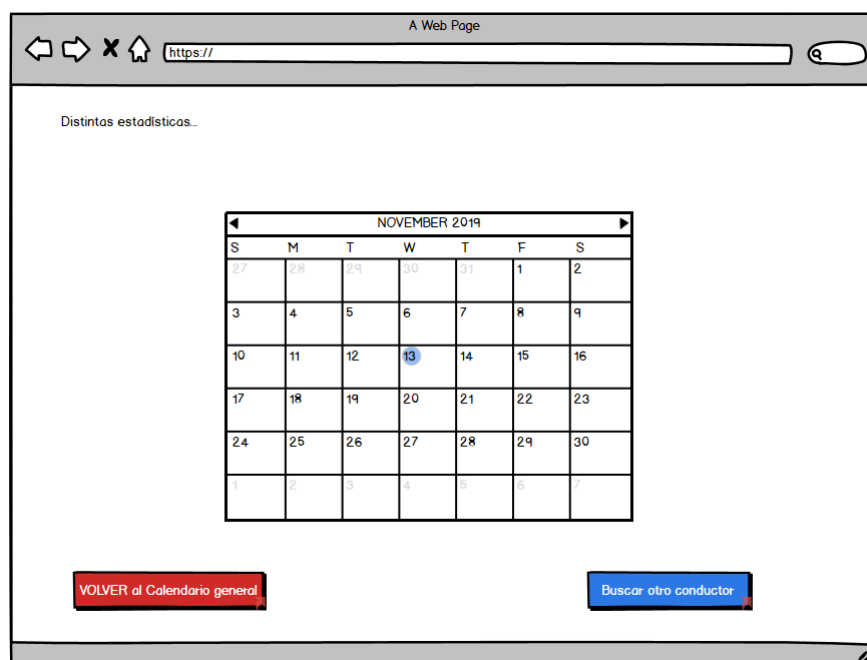
Identificador del conductor

Identificador del conductor

Volver

Figura 47: Página de Búsqueda de un conductor.

En la Figura 48 vemos el gestor pulsa en buscar otro conductor volverá a la visualización de la tabla de conductores y si pulsa el botón de volver al calendario general volverá a la visualización de la demanda con su menú.



Distintas estadísticas...

NOVEMBER 2019						
S	M	T	W	T	F	S
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

VOLVER al Calendario general

Buscar otro conductor

Figura 48: Página de visualización del calendario del conductor.

Si el gestor en la visualización del calendario pulsa en ver estadísticas, le saldrá una página que contendrá una tabla con los datos estadísticos del calendario generado junto con un botón para poder volver a la página de visualización de la demanda global del calendario.



Figura 49: Página de visualización de estadísticas del calendario.

En la Figura 50 podemos ver la pantalla que se visualizará si el gestor pulsa en modificar calendario donde añadirá la fechas que desea que sean modificadas por el algoritmo y podrá pulsar ejecutar. Si quiere volver pulsará el botón de volver.

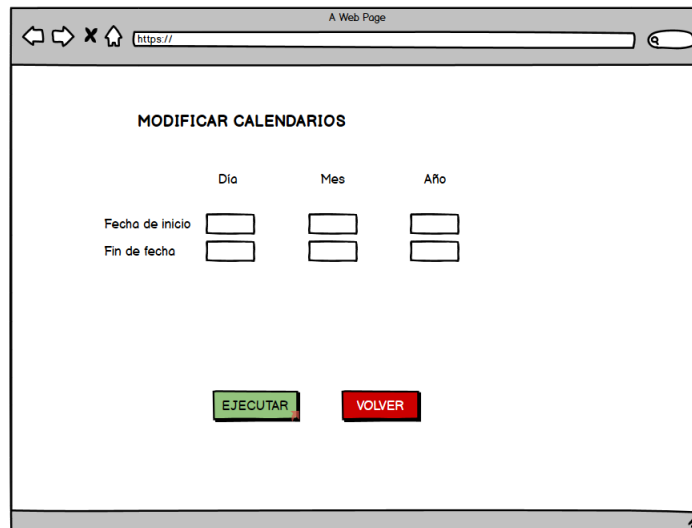


Figura 50: Página de modificación del calendario.

La Figura 51 indica que se ha producido un error en la selección de las fechas del calendario y pedirá que se reescriban.

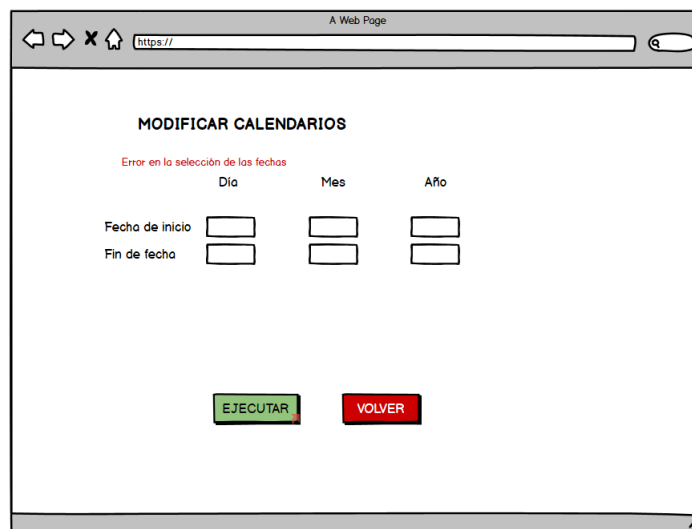


Figura 51: Página de modificación del calendario con error en los datos.

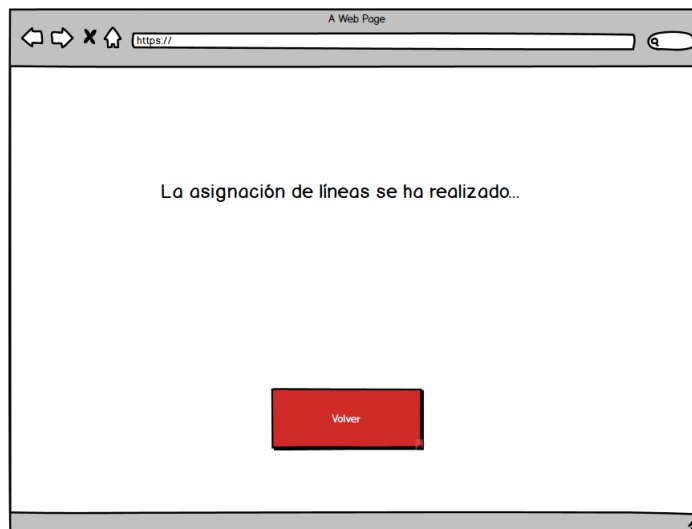
La Figura 52 ejecutará el algoritmo tras comprobar que los datos de las fechas introducidas son correctos. Si tiene éxito, mostrará la nueva visualización del calendario.



The screenshot shows a web browser window with the address bar containing 'https://'. The page title is 'A Web Page'. The main content area is titled 'MODIFICAR CALENDARIOS'. It features three columns of date selection fields labeled 'Dia', 'Mes', and 'Año'. Below these are two rows of input boxes: 'Fecha de inicio' and 'Fin de fecha'. In the center, the text 'ALGORITMO EJECUTANDOSE' is displayed in red. At the bottom, there are two buttons: a green 'EJECUTAR' button and a red 'VOLVER' button.

Figura 52: Página de modificación del calendario con el algoritmo en ejecución.

Si el gestor pulsa en asignar líneas en la página de visualización de la demanda global, el sistema mostrará esta página cuando haya completado su asignación:



The screenshot shows a web browser window with the address bar containing 'https://'. The page title is 'A Web Page'. The main content area displays the message 'La asignación de líneas se ha realizado..' in the center. Below the message is a red button labeled 'Volver'.

Figura 53: Página de confirmación de asignación de las líneas.

Esta mostrará un mensaje de finalización y tendrá el botón volver con el que el gestor puede volver a la anterior página.

Si el gestor pulsa en guardar calendario en la página de visualización de la demanda global, el sistema mostrará el siguiente menú de la Figura 54.

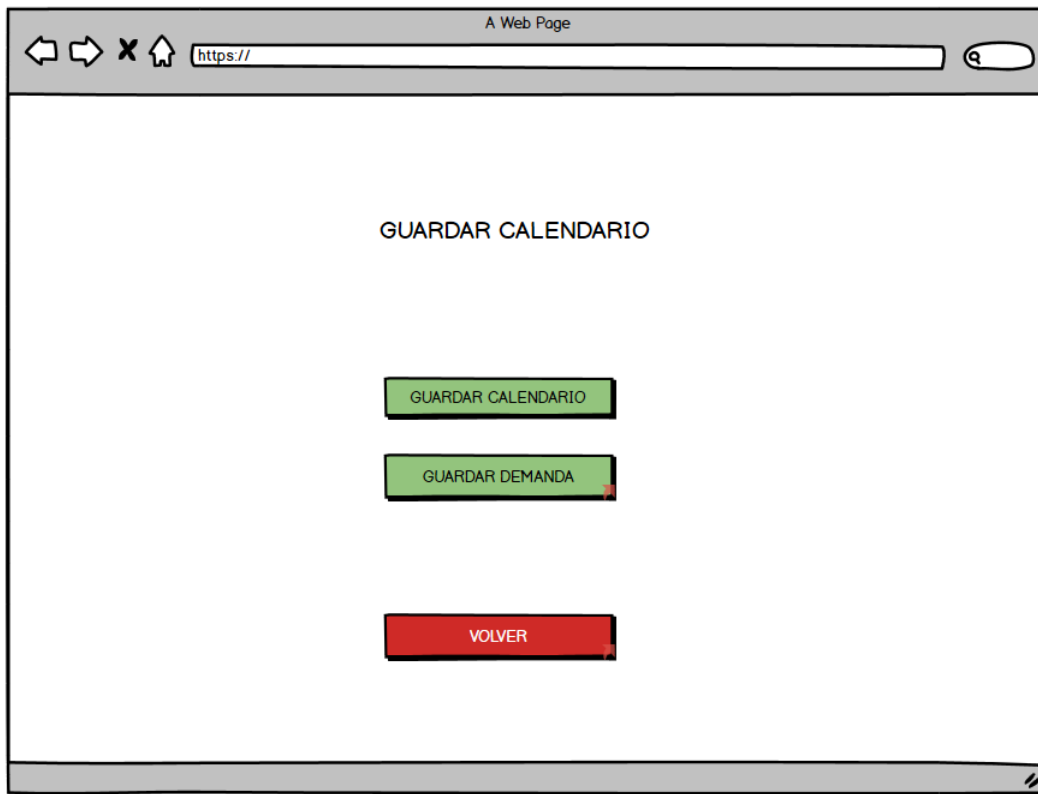
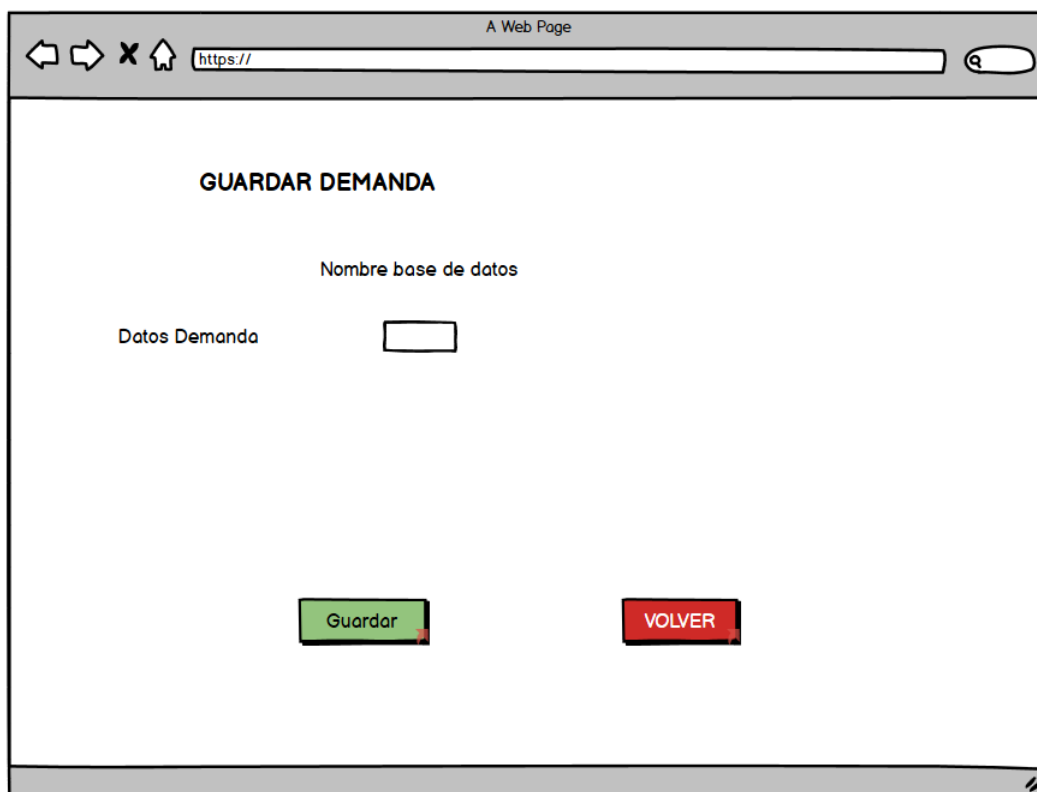


Figura 54: Página de menú de guardado.

En esta página el gestor podrá seleccionar que desea guardar o si desea volver a la página de visualización de la demanda.

Las páginas de guardado de demanda y calendario son muy parecidas por lo que solo se mostrará una de ejemplo (Figura 55).



The image shows a web browser window titled "A Web Page" with a search bar containing "https://". The main content area displays the following elements:

- GUARDAR DEMANDA** (Save Demand)
- Nombre base de datos (Database name)
- Datos Demanda (Demand Data) with an adjacent text input field.
- Guardar (Save) button (green)
- VOLVER (Return) button (red)

Figura 55: Página de ejemplo de guardado.

En esta página el gestor podrá escribir el nombre de la tabla y pulsar guardar. En este caso, se volverá a la visualización de la demanda global. Si el usuario pulsa volver irá al menú de guardado.

Si el gestor está en el menú principal de la aplicación y pulsa en seleccionar calendario verá un menú en el que escribirá el nombre de la base de datos del calendario y la demanda y a su vez tendrá un botón para volver al menú principal y otro para continuar en la visualización del calendario seleccionado como hemos visto en la Figura 46. Esta pantalla de selección del calendario se verá como en la Figura 56.

	Host	Usuario	Contraseña	Base de Datos	Nombre base de datos
Datos Demanda	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Datos Calendario	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Figura 56: Página de selección de calendario.

En caso de que se produzca algún error en la selección del calendario se mostrará un mensaje con dicho error (Figura 57).

Error en la seleccion de las tablas de los calendarios

	Host	Usuario	Contraseña	Base de Datos	Nombre base de datos
Datos Demanda	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Datos Calendario	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Figura 57: Página de error en la selección de calendario.

Si el gestor en el menú principal pulsa en modificar conductores, verá una pantalla de gestión de conductores (Figura 58).

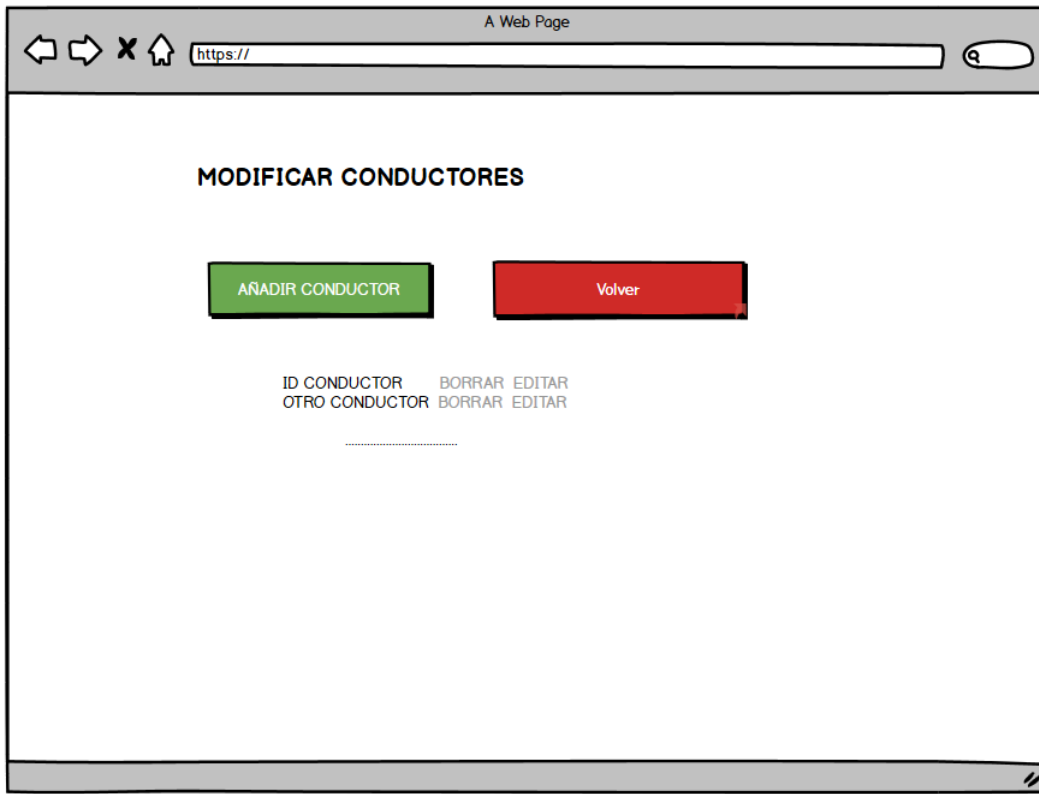


Figura 58: Página de modificación de conductores.

Si el usuario pulsa volver, regresará al menú principal, si pulsa añadir, le saldrá un menú para cumplimentar los datos del conductor, si pulsa en borrar, eliminará dicho conductor y volverá a mostrar esta página y si pulsa editar se podrán modificar los datos del conductor por otros nuevos que sean válidos.

Las visualizaciones de crear y editar conductores (Figura 59) serán muy parecidas, donde ambas tendrán un botón para guardar y salir y un botón para salir sin guardar. Además de un formulario donde podrá escribir las características del conductor.

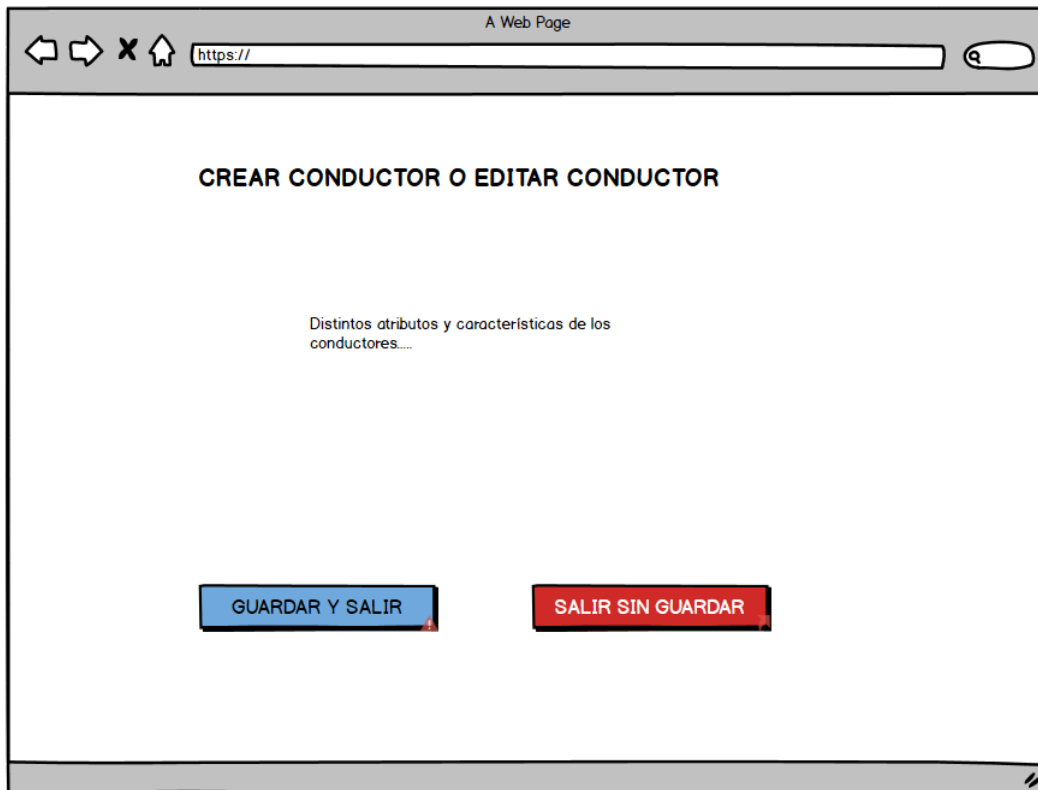


Figura 59: Página de creación o edición de conductores.

Si el gestor está en el menú principal y pulsa opciones, le saldrá un formulario con todas las características actuales de la creación de calendarios anuales de los conductores y podrá modificarlas según sus necesidades.

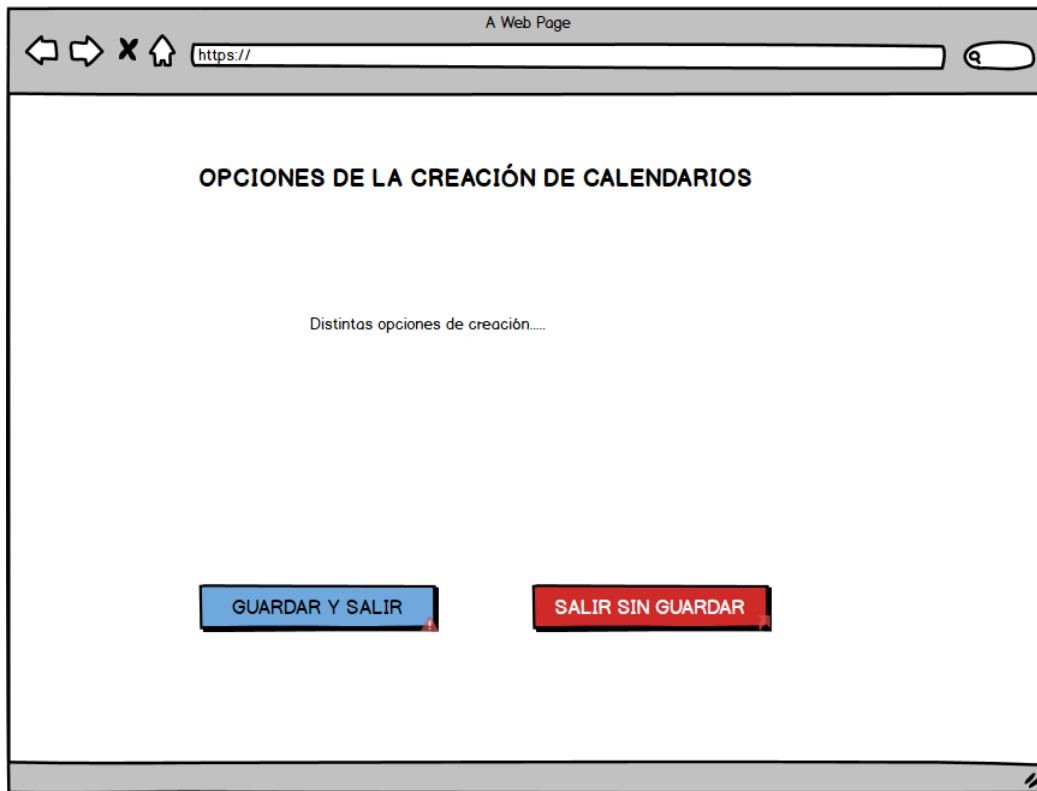


Figura 60: Página de edición de opciones del algoritmo.

Como se puede ver, tendrá al igual que la edición de los conductores, una salida guardando las opciones y otra sin guardarlo, regresando después al menú principal.

4. Diseño del algoritmo

En esta sección estudiaremos como está diseñado el algoritmo y su evolución a lo largo del tiempo.

Además, debemos destacar, que el algoritmo se divide en dos procesos muy marcados, el primero es la asignación de líneas anuales de autobús y, el segundo, es la creación de los calendarios anuales.

4.1. Algoritmo de asignación de líneas

El algoritmo de asignación de líneas anuales es el más simple y el que primero se diseñó. Este busca asignar en los turnos de los conductores las líneas de autobús teniendo en cuenta sus preferencias particulares. Bien asignándoles una línea de trabajo, o como apoyo, asignándoles una línea no existente que se denota como línea 0.

Inicialmente el algoritmo constaba de tres etapas:

- La primera buscaba asignar a *los conductores de trío* cumpliendo la regla del trío (vista en la normativa 2.2.2), es decir, cuando los turnos de los conductores sean de la forma, uno de mañana, otro de tarde y el último de descanso, se les asignará su línea, siempre que esta, esté disponible.
- La segunda etapa buscaba asignar el resto de las líneas con demanda no cubierta a cualquier conductor con disponibilidad laboral ese día y turno.
- La tercera y última etapa, asignaba la línea 0 a todos aquellos conductores que tuvieran disponibilidad de trabajar, pero no existía demanda en ninguna línea para ese día y turno.

Este algoritmo tenía dos problemas. El primero, era que a muchos conductores de trío se les asignaba una línea a la que no pertenecían. El segundo problema, estaba relacionado con el anterior, y era que los conductores del tipo correturnos tenían las mismas asignaciones, e incluso más, a líneas que los conductores de tríos. Lo adecuado es que los conductores correturnos sean seleccionados como recambio si hay conductores de trío pertenecientes a la línea que han sido asignados ese día y turno a otra línea diferente o han sido seleccionados como recambio.

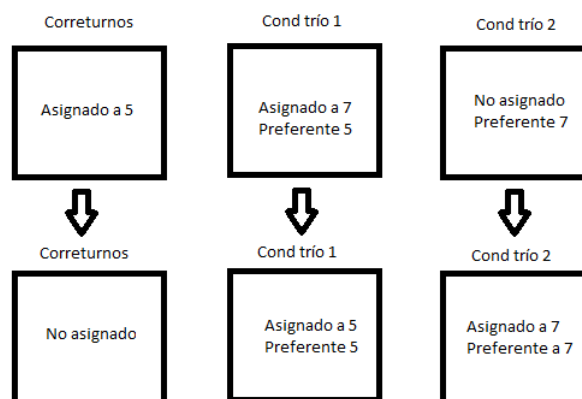


Figura 61: Ejemplo de fallo 2 en la asignación de líneas.

Como ejemplo la Figura 61 donde un conductor correturnos está asignado a la línea 5, un miembro de trío, cuya línea preferente es la 5, está asignado en la línea 7 y un tercer conductor de trío, en apoyo, cuya línea preferente es la 7. Deberían tener la siguiente asignación: el correturnos, de descanso y los conductores de trío, en las líneas de su preferencia.

En el diseño final se añadieron dos etapas más:

- La primera etapa añadida se situaba después de la primera de la versión inicial del algoritmo. Su función era la de asignar a los conductores de tríos sin línea fija, las líneas de su preferencia, siempre y cuando, la demanda de esa línea, en ese día y turno no esté cubierta.
- La segunda etapa añadida, se sitúa a continuación de la descrita previamente. Se buscaba asignar línea a los correturnos y eventuales, sin asignaciones, una línea cuya demanda no se hubiera cubierto, ya que en este momento, no existen conductores de trío que no se hayan asignado a su línea preferente cuando esta tiene demanda sin cubrir.

El resto de etapas del algoritmo son las mismas que el algoritmo inicial, con un total cinco etapas distintas entre sí.

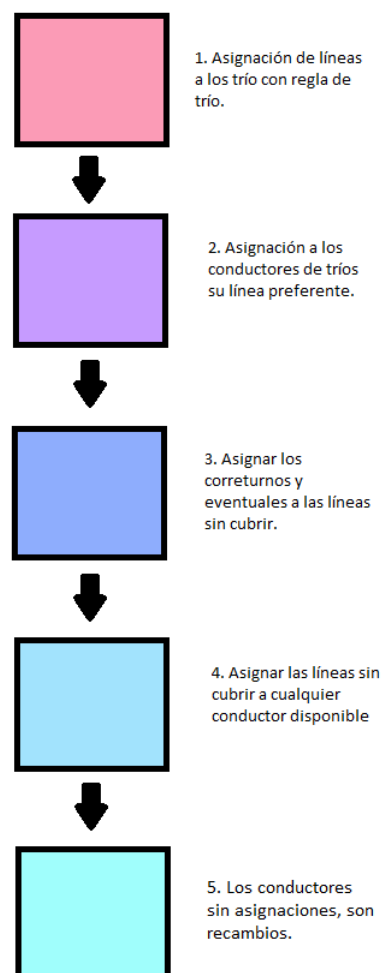


Figura 62: Etapas del algoritmo de asignación de líneas.

4.2. Algoritmo de creación de calendarios anuales

El algoritmo de creación de calendarios anuales es el calendario más complejo y lo desarrollamos a lo largo de dos etapas. Hemos utilizado los conocimientos heurísticos adquiridos en las asignaturas del grado de Estadística que involucraban temas de investigación operativa. Una heurística es un procedimiento para buscar soluciones óptimas, o al menos adecuadas, en un tiempo razonable.

Estos algoritmos buscan crear calendarios anuales que cubran toda la demanda y cuya modificación se realiza en un intervalo de tiempo marcado por el usuario. En la primera etapa, se buscó crear un algoritmo que no tuviera en cuenta ciclos de trabajo prefijados por la empresa y que tratase de forma individualmente a todos los conductores.

En este algoritmo podemos distinguir tres etapas muy marcadas.

- La primera etapa buscaba cubrir la máxima demanda posible teniendo en cuenta las normas relacionadas con el número de días de trabajo y descansos seguidos que pueden realizar. En esta etapa, se asigna a cada conductor unos turnos y días de trabajo, inicialmente coherentes, donde todos los tríos comienzan cumpliendo la regla de trío y se le añade a cada conductor las quincenas de vacaciones correspondientes. Hecho esto, el algoritmo buscaba minimizar la falta de demanda de todos los días del año para esto se siguió el siguiente pseudocódigo:

```
for d in range(días){
    while demanda[d] > 0:
        dias_trabajo = 400
        for cond in matriz_conductores:
            if cond.puede_trabajar:
                if cond.dias_trabajo < dias_trabajo:
```

Se tienen en cuenta esta y otras condiciones de selección, como domingos trabajados, número de turnos con un grupo de seis días seguido de trabajo, etc.

```
        cond.asignar_trabajo
        demanda[d] -= 1
        if demanda[d] ≤ 0 or dias_trabajo == 400:
            break }
```

- En la segunda etapa, tras asignar todo el trabajo a los conductores y con la mínima demanda restante, se procede a ajustar los días de trabajo máximos anuales de los conductores, esto se hace teniendo en cuenta los siguientes casos:
 - **Si un conductor tiene más número de días de trabajo que el valor objetivo** se eliminan aquellos días cuya demanda esté cubierta, priorizando los días con mayores excesos de conductores o con menores faltas.
 - **Si un conductor tiene menor número de días de trabajo que el valor objetivo** se añaden más días de trabajo, priorizando los días con mayor número de demanda o con menor número de conductores sobrantes.
- En la tercera y última etapa de este primer algoritmo, y tras asignar el número de días objetivo a los conductores, se lleva a cabo *un proceso de mejora mediante intercambios*. En esta etapa, a cada conductor se le intercambian días de trabajo y descansos, buscando minimizar los excesos y faltas de conductores a lo largo de todos los días del año teniendo en cuenta sus preferencias.

Las ventajas de este segundo algoritmo son:

1. La existencia de muy pocos días con faltas de personal, que oscilaban desde 0 a 3.
2. La reducción de faltas y excesos diarios producidos.
3. El tiempo de ejecución incluyendo todos los conductores era muy bajo, de menos de una décima de segundo.

Los problemas que tenía este algoritmo eran los siguientes:

1. No se tenía en cuenta la regla de trío, resultando un porcentaje de cumplimiento muy bajo.
2. No tenía en cuenta las preferencias de líneas de los tríos en la creación de los calendarios.
3. No tenía en cuenta que algunos conductores sólo realizaban turnos de un tipo (mañana o tarde y no ambos).
4. No se tenían en cuenta los ciclos de trabajo de la empresa, por lo que los calendarios eran muy distintos a estos ciclos y era preferible una mayor similitud.
5. No tenía en cuenta los intervalos de tiempo marcados por el usuario.

Para solventar estos problemas, hicimos algunas modificaciones en el algoritmo de creación de calendarios y añadimos dos nuevas etapas en el algoritmo.

- Las nuevas etapas añadidas se sitúan al principio del algoritmo y son las siguientes:
 1. Una primera etapa que busca añadir trabajos a los conductores de trío para que cumplan la regla de trío la mayor cantidad de días posibles, siempre y cuando haya demanda sin cubrir y se les pueda asignar trabajo.
 2. Una segunda etapa, que implementamos a continuación de la descrita antes, y que de nuevo, sólo afecta a los conductores de trío. Se busca dar a estos conductores el número de días de trabajo que indica la normativa, mediante una ponderación que conduzca a un equilibrio entre el excedente en la demanda y el cumplimiento de la regla de trío.
- Las modificaciones que se introdujeron respecto al algoritmo anterior son:
 - Partimos de un calendario inicial prediseñado con los ciclos de trabajo de cada conductor.
 - Las modificaciones introducidas por el proceso de mejora mediante intercambios, tienen en cuenta la regla de tríos y de domingos consecutivos de trabajo que indica la normativa.
 - Además, al proceso previo de mejora mediante intercambios, se le añade la posibilidad de intercambiar dos días consecutivos de trabajo, con dos días de descansos (estos últimos consecutivos o no).
 - Los conductores eventuales y los correturnos son tratados de forma similar. La única diferencia entre ambos es que los segundos trabajan todo el año, pero no tienen preferencia por líneas, sólo cubren demanda sobrante.
 - Se añade la existencia de conductores que sólo pueden realizar un tipo de turno de trabajo.

Los beneficios obtenidos con este algoritmo final son los siguientes:

- Se cumple en gran medida toda la normativa de la empresa.
- Tiene en cuenta los ciclos de trabajo de los conductores de forma que el calendario se ajusta mejor.
- Se añade una mejora en un intervalo concreto de tiempo.
- Los días con demanda sin cubrir suelen requerir de un número pequeño de conductores.

Los problemas de este algoritmo son los siguientes:

- El tiempo de ejecución del algoritmo aumenta hasta 30 segundos debido a que se realizan más comprobaciones en cada etapa.
- Existe un número mucho mayor de días sin toda la demanda cubierta, de aproximadamente la mitad de días del año.

5. Implementación de la aplicación

Es preciso destacar que Flask lleva de forma automática toda la complejidad referente al acceso y envío de mensajes entre el cliente y el servidor.

Las distintas etapas por las que ha pasado la aplicación en el proceso de implementación son:

5.1. Creación del modelo, algoritmo y la base de datos

En esta etapa se crearon todos los archivos relacionados con el modelo de dominio, los algoritmos (tanto el de creación de calendarios anuales como el de asignación de líneas) y la creación de las tablas en la BBDD y la introducción de los datos en ellas.

5.2. Casos de uso (Iniciar y cerrar sesión)

En la siguiente iteración se comenzó la inclusión de los casos de uso. Lo primero que se realizó, fue el inicio y cierre de sesión.

Imágenes de la aplicación:

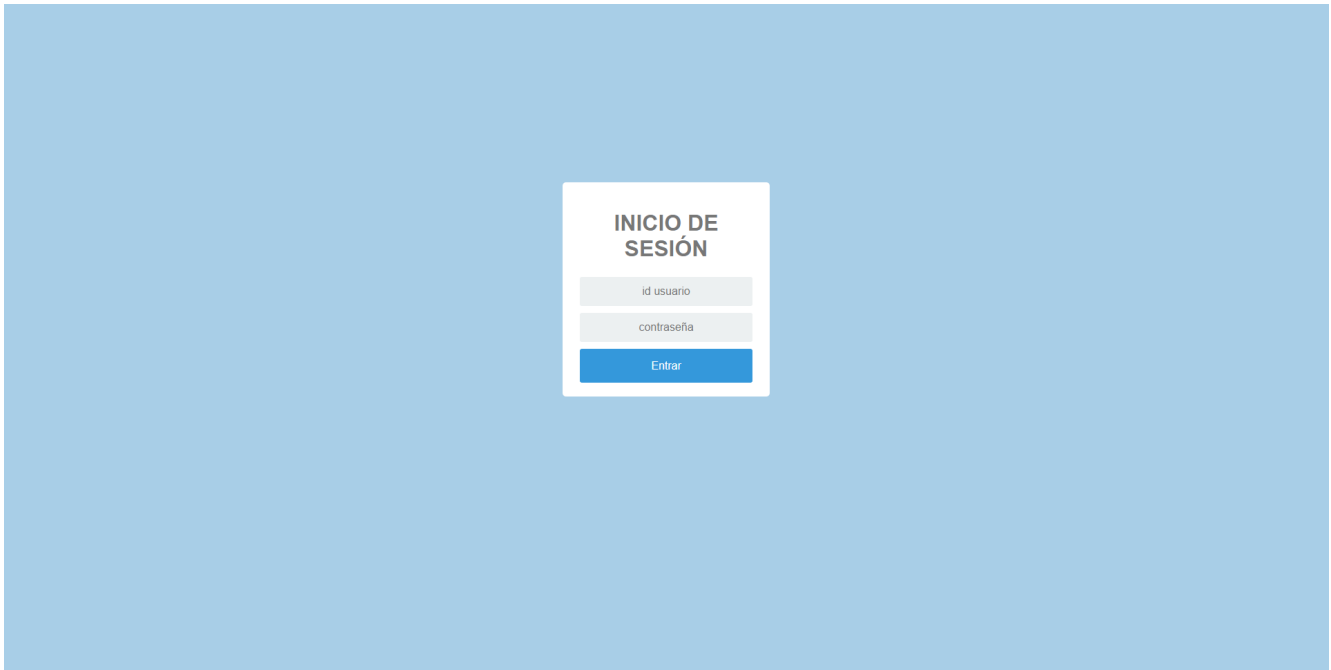


Figura 63: Inicio sesión.

5.3. Casos de uso del gestor

En la tercera iteración se realizaron todos los casos de uso del gestor.

Imágenes de la aplicación:

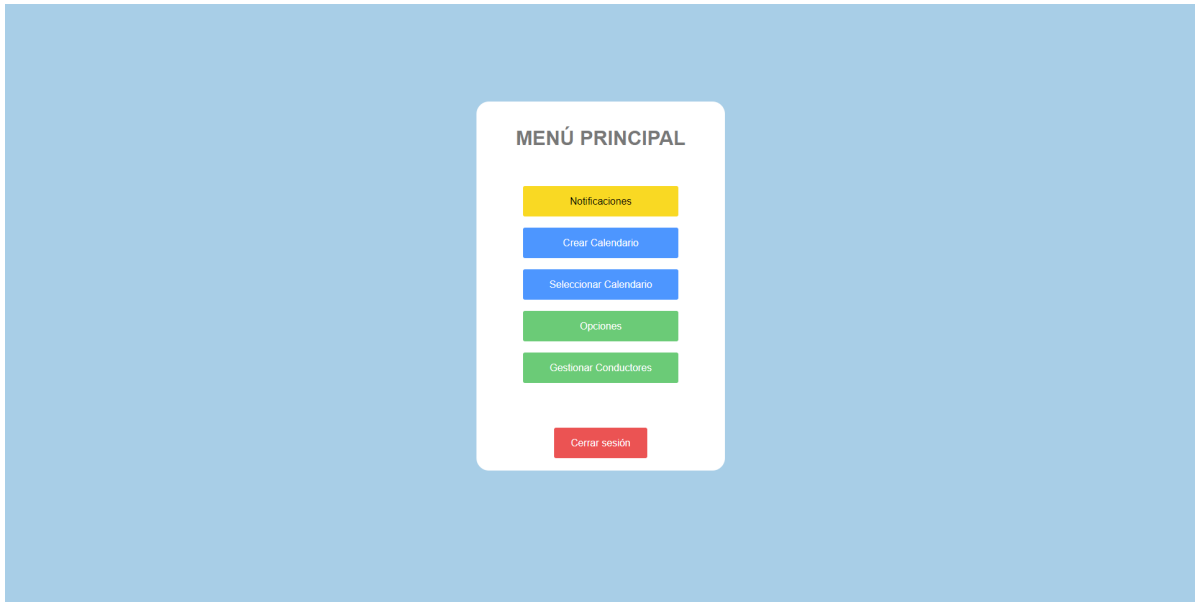


Figura 64: Pantalla principal.



Figura 65: Ver notificaciones.



Figura 66: Opciones de creación de calendario.

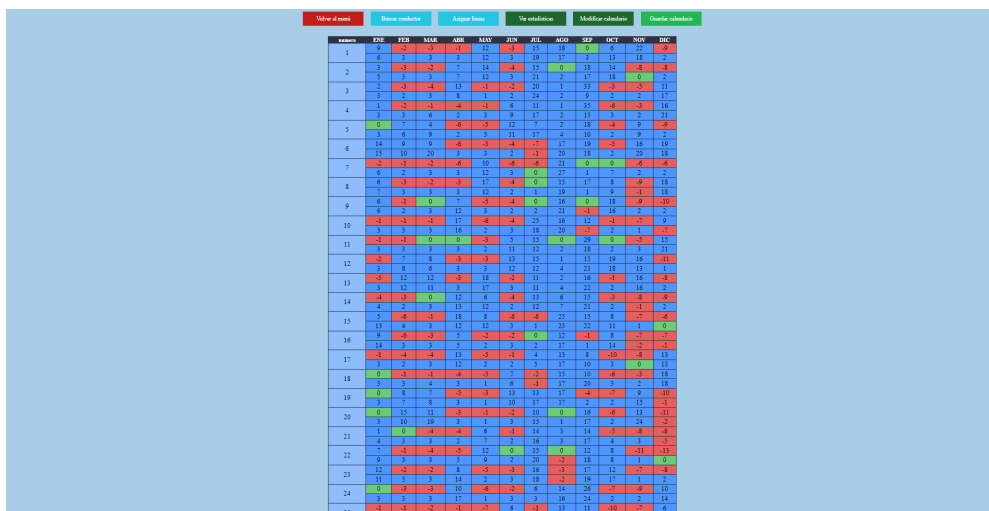


Figura 67: Visualización de la demanda global.

A screenshot of a web application interface. At the top center, there is a red button labeled 'Volver'. Below it is a table with two columns: 'Identificador' and 'Aerónomas'. The table contains 48 rows, each with a number from 1 to 48 in the first column and the text 'Ver calendario' in the second column. The table is styled with alternating light blue and white rows.

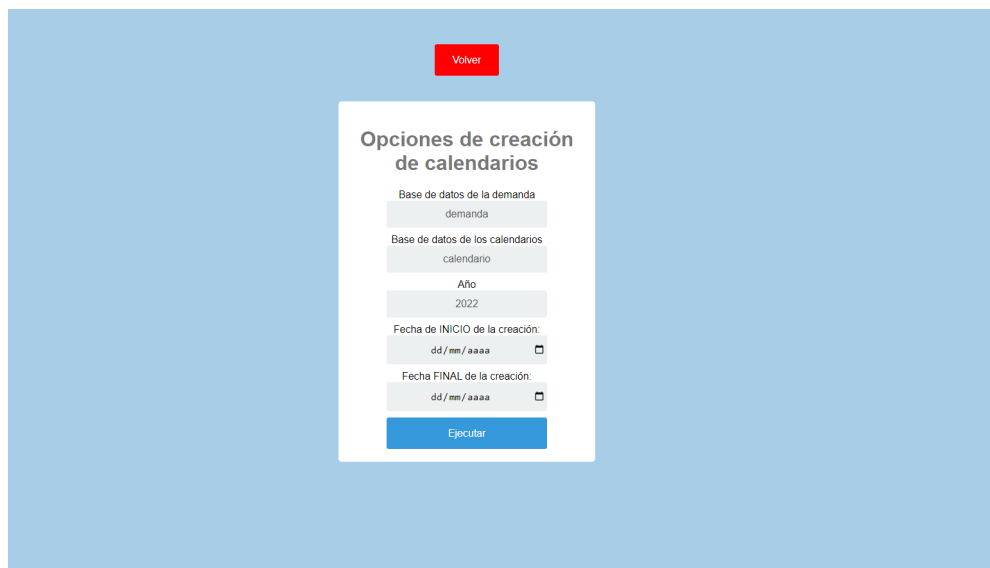
Identificador	Aerónomas
1	Ver calendario
2	Ver calendario
3	Ver calendario
4	Ver calendario
5	Ver calendario
6	Ver calendario
7	Ver calendario
8	Ver calendario
9	Ver calendario
10	Ver calendario
11	Ver calendario
12	Ver calendario
13	Ver calendario
14	Ver calendario
15	Ver calendario
16	Ver calendario
17	Ver calendario
18	Ver calendario
19	Ver calendario
20	Ver calendario
21	Ver calendario
22	Ver calendario
23	Ver calendario
24	Ver calendario
25	Ver calendario
26	Ver calendario
27	Ver calendario
28	Ver calendario
29	Ver calendario
30	Ver calendario
31	Ver calendario
32	Ver calendario
33	Ver calendario
34	Ver calendario
35	Ver calendario
36	Ver calendario
37	Ver calendario
38	Ver calendario
39	Ver calendario
40	Ver calendario
41	Ver calendario
42	Ver calendario
43	Ver calendario
44	Ver calendario
45	Ver calendario
46	Ver calendario
47	Ver calendario
48	Ver calendario

Figura 68: Buscar un conductor del calendario.

A screenshot of a web application interface. At the top center, there is a red button labeled 'Volver'. Below it is a section titled 'ESTADÍSTICAS' containing a table with three columns: 'Días no cubiertos', 'Porcentaje cumplimiento regla tríos', and 'Porcentaje de tríos en su línea'. The table shows the following values: 180.0, 0.8821917808219182, and 0.8821917808219182 respectively.

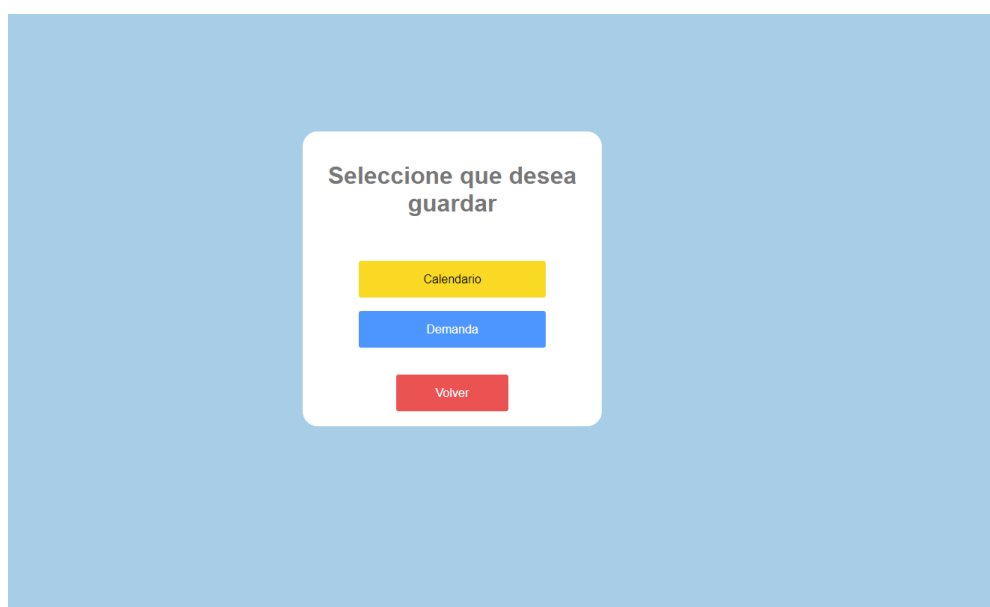
Días no cubiertos	Porcentaje cumplimiento regla tríos	Porcentaje de tríos en su línea
180.0	0.8821917808219182	0.8821917808219182

Figura 69: Ver estadísticas del calendario.



The screenshot shows a web form titled "Opciones de creación de calendarios" centered on a light blue background. At the top of the form is a red button labeled "Volver". Below the title, there are several input fields: "Base de datos de la demanda" with the value "demanda", "Base de datos de los calendarios" with the value "calendario", "Año" with the value "2022", "Fecha de INICIO de la creación:" with a placeholder "dd/mm/aaaa" and a calendar icon, and "Fecha FINAL de la creación:" with a placeholder "dd/mm/aaaa" and a calendar icon. At the bottom of the form is a blue button labeled "Ejecutar".

Figura 70: Modificar el calendario.



The screenshot shows a web form titled "Seleccione que desea guardar" centered on a light blue background. The form contains three buttons: a yellow button labeled "Calendario", a blue button labeled "Demanda", and a red button labeled "Volver".

Figura 71: Menú de guardado.

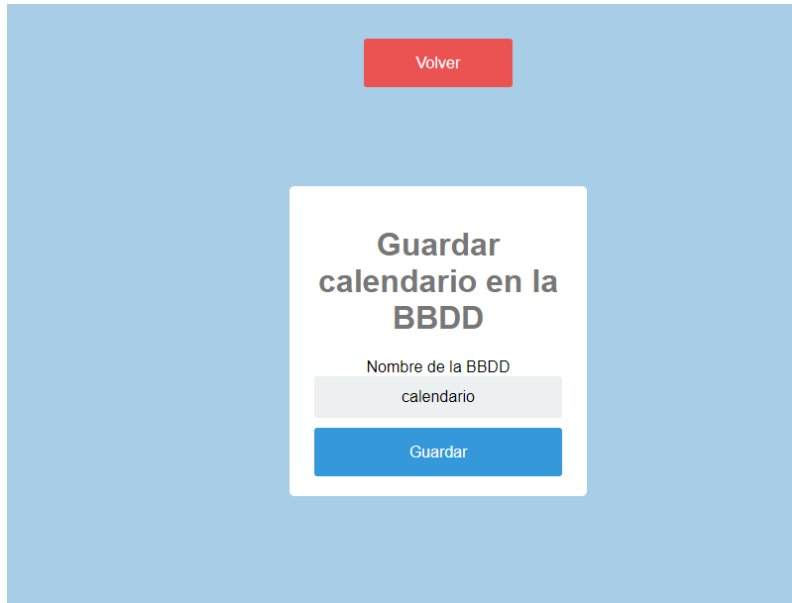


Figura 72: Guardar el calendario de turnos.

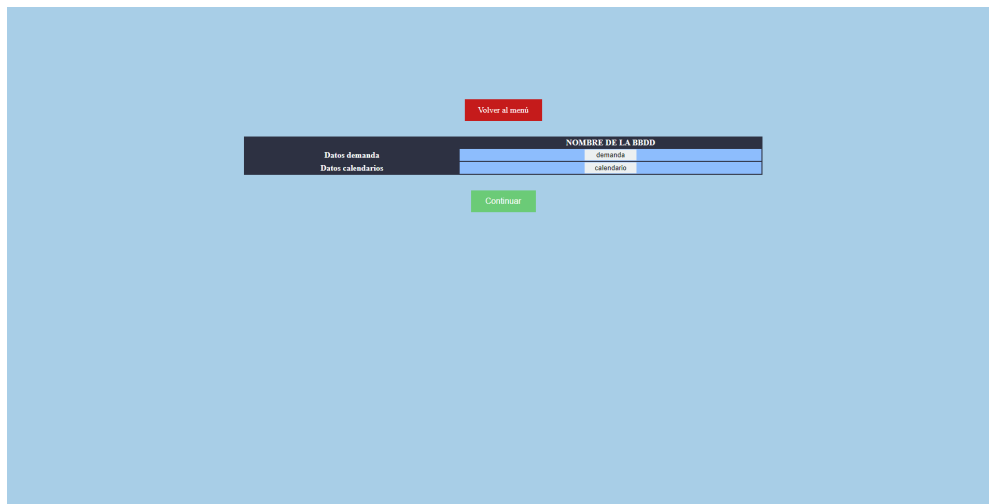


Figura 73: Seleccionar un calendario.

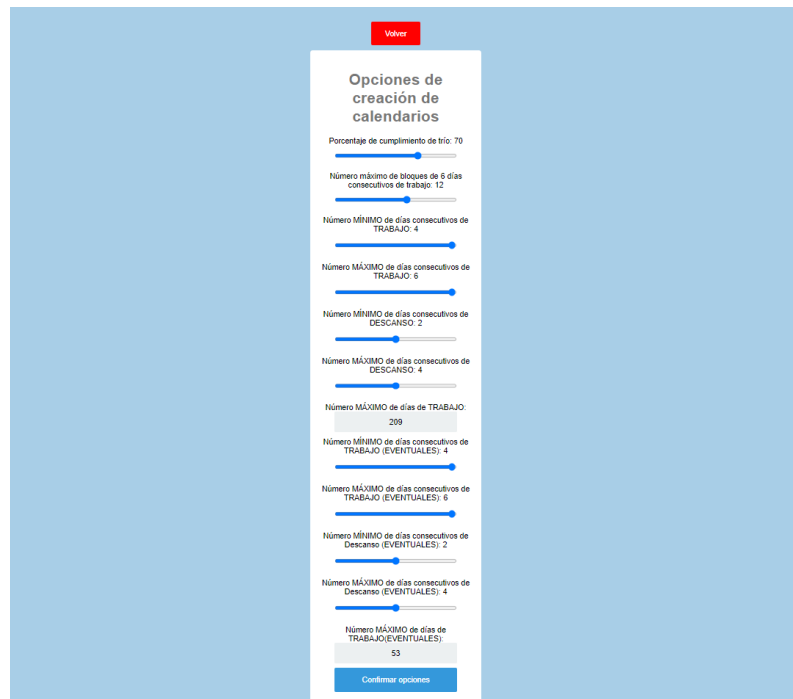


Figura 74: Opciones de creación de los calendarios.

Identificador	Identificador de Trió	Número ciclo	Tipo de conductor	Vacaciones invierno	Vacaciones verano	Acciones
1	66	25	1	18	14	editar, borrar
2	31	32	1	24	16	editar, borrar
3	19	21	1	11	13	editar, borrar
4	73	33	1	3	15	editar, borrar
5	32	18	1	6	15	editar, borrar
6	0	1	4	3	14	editar, borrar
7	0	2	4	1	16	editar, borrar
8	0	3	4	3	13	editar, borrar
9	31	31	1	24	16	editar, borrar
10	25	28	1	9	13	editar, borrar
11	49	35	1	9	13	editar, borrar
12	0	4	2	10	14	editar, borrar
13	0	5	4	10	14	editar, borrar
14	66	26	1	18	14	editar, borrar
15	67	8	1	2	13	editar, borrar
16	19	20	1	11	13	editar, borrar
17	0	6	4	1	13	editar, borrar
18	0	7	2	11	14	editar, borrar
19	0	8	3	9	15	editar, borrar
20	0	9	2	21	14	editar, borrar
21	37	3	1	23	14	editar, borrar
22	36	22	1	5	15	editar, borrar
23	0	10	2	8	16	editar, borrar
24	72	23	1	19	13	editar, borrar
25	23	1	1	4	16	editar, borrar
26	57	1	1	24	13	editar, borrar
27	0	11	3	7	13	editar, borrar
28	0	12	3	24	16	editar, borrar
29	12	16	1	24	16	editar, borrar
30	0	13	2	23	16	editar, borrar
31	2	26	1	1	16	editar, borrar

Figura 75: Modificación de conductores.

The image shows a web form for creating a new driver and user. At the top center, there is a red button labeled 'Volver'. Below it, the form title is 'Crear nuevo conductor y usuario:'. The form contains several input fields with the following labels and values:

- Id conductor: 1
- Nombre: conductor
- Contraseña: 1
- Id_trio: 1
- Ciclo(hasta 33 ciclos): 2
- Tipo Conductor: 3
(1 = trio, 2 = Corretornos M/T, 3 = Corretornos M, 4 = Corretornos T, 5 = eventuales)
- Vacaciones Invierno (No puede contener las semanas 13-16): 1
- Vacaciones Verano (Debe contener las semanas 13-16): 13

At the bottom of the form is a blue button labeled 'Confirmar'.

Figura 76: Añadir un nuevo conductor.

5.4. Casos de uso del conductor

En la cuarta iteración, se realizaron los casos de uso del conductor. Este paso, fue más rápido y sencillo que el anterior, debido a que tenía menos casos de uso y, uno de ellos, no era más que una variación de otro perteneciente al gestor.

Imágenes de la aplicación:

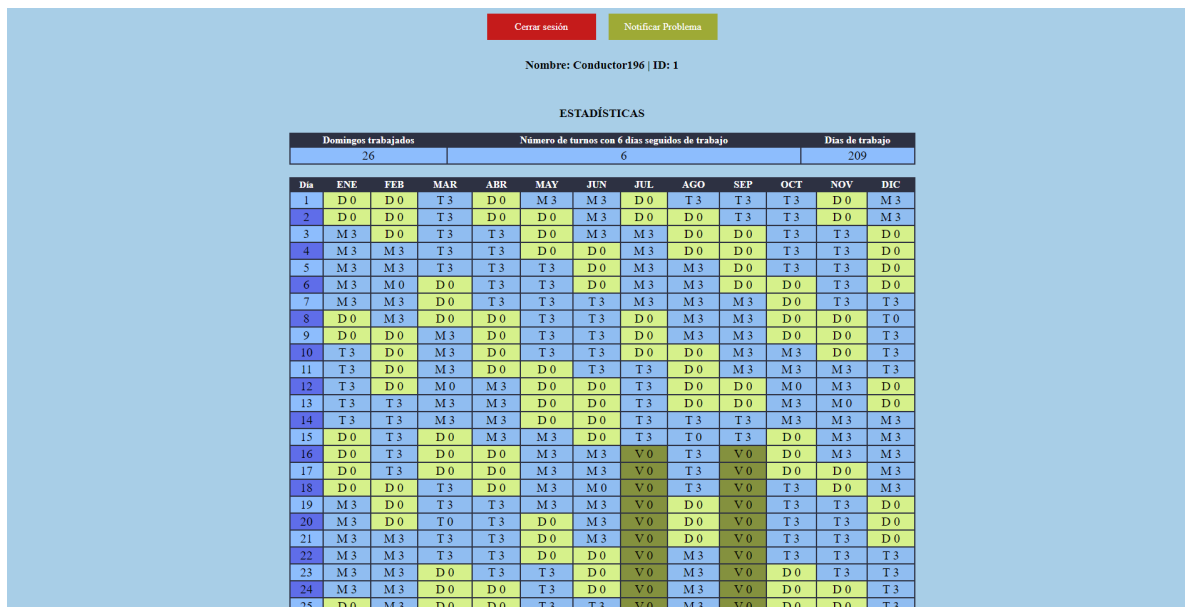


Figura 77: Pantalla principal del conductor, donde se puede ver el calendario individual del conductor.

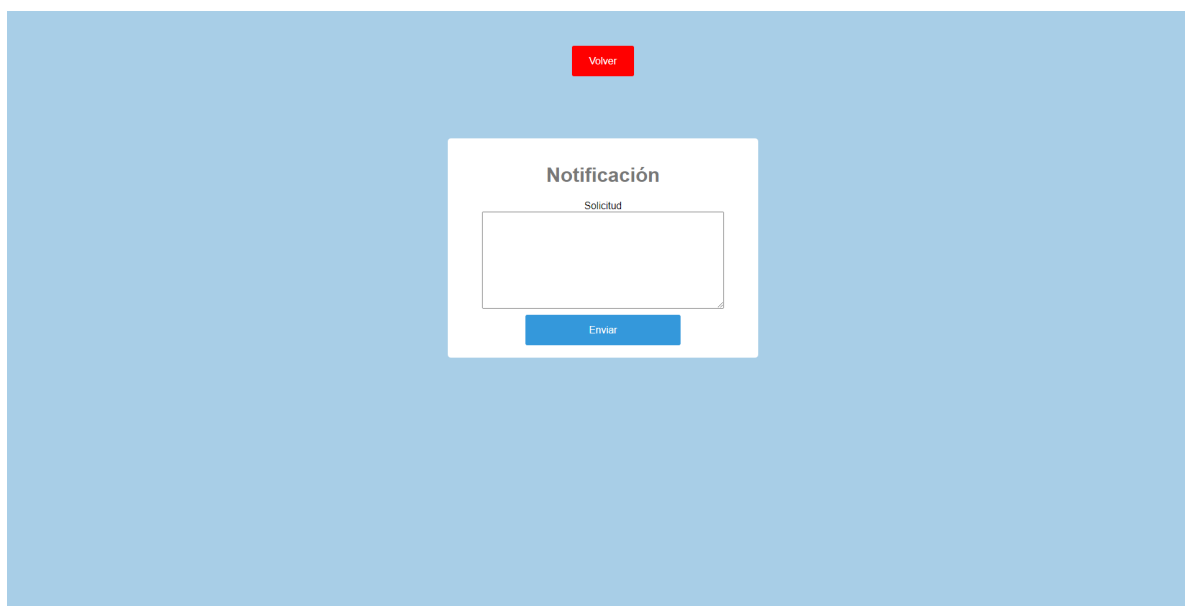


Figura 78: Pantalla de creación de notificaciones por parte del conductor.

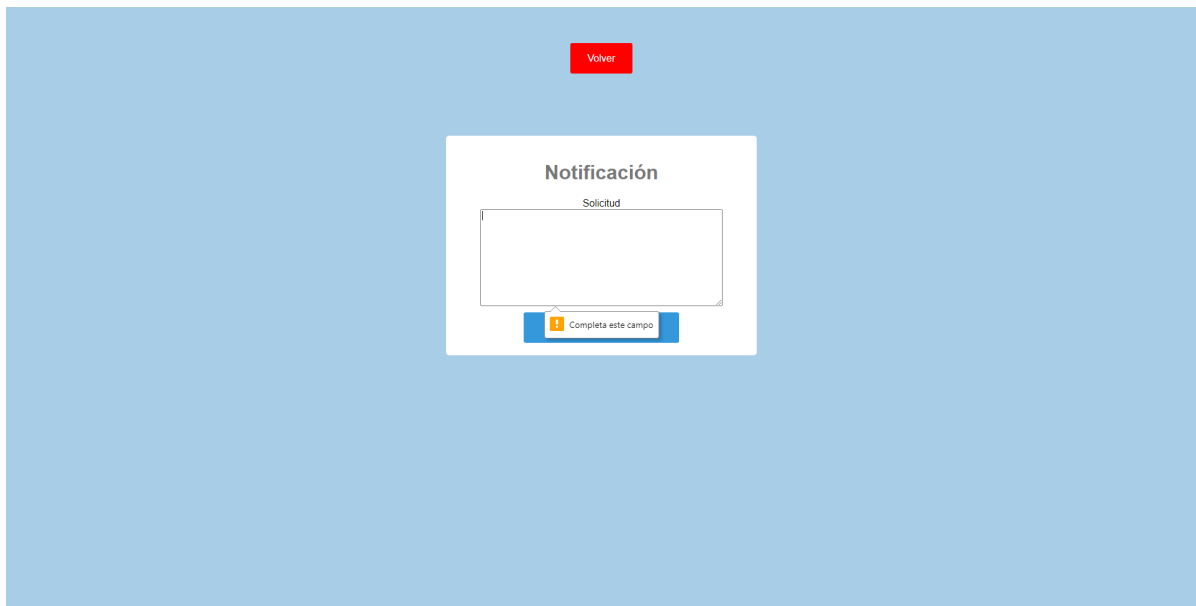


Figura 79: Pantalla de creación de notificaciones por parte del conductor cuando hay un error.

6. Test realizados

En este apartado comentaremos todas las pruebas realizadas con la aplicación en cada una de las fases de implementación.

6.1. Test realizados en la Fase 1 de la implementación

– **En el modelo:**

1. Creación de los distintos objetos de cada clase del modelo con el fin de que comprobar si estaba bien construida su estructura. Fueron un éxito.
2. Ejecución de las funciones de las clases con el fin de comprobar que las funcionalidades de cada clase eran las deseadas. Fueron un éxito.

– **En los algoritmos:**

1. En el algoritmo de la asignación de línea se comprobó que tal asignación fuera correcta, pero se encontró un error en el flujo, debido a la falta de un bucle para recorrer todos los conductores de trío en su primera fase. Una vez corregido el problema, la prueba fue un éxito.
2. En el algoritmo de creación de calendarios también se comprobó que sus distintas etapas funcionaran correctamente. Se produjeron los errores siguientes:
 - El primer error fue un fallo general debido a que el algoritmo en ninguna de sus etapas realizaba los cambios en el intervalo de tiempo agregado.
 - El segundo error se produjo al recorrer los conductores de trío en las primeras etapas del algoritmo, al igual que pasaba con el algoritmo de asignación de línea.
 - El tercer y último error se debía a la incorrecta asignación de los días de trabajo objetivo de cada conductor, ocasionado por la incorrecta salida del bucle de adición de días.

Una vez corregidos estos problemas, todas las pruebas realizadas en el algoritmo de creación de calendarios fueron un éxito.

- En la BBDD se comprobó que los atributos de las distintas tablas fueran los requeridos, y que al poblarla de datos, las instancias creadas fueran las deseadas. No se produjeron problemas. Por lo que las pruebas fueron un éxito.

6.2. Test realizados en la Fase 2 de la implementación

Para el caso de uso **Inicio de sesión** se realizaron cuatro pruebas:

- Iniciar sesión con un usuario gestor y mostrar un menú primitivo de gestor.
- Iniciar sesión con un usuario conductor y mostrar un menú primitivo de conductor.
- Escribir un usuario no existente. En este caso, la aplicación lanzaba un mensaje de error (como era de esperar).
- Escribir una contraseña incorrecta. La aplicación lanzaba el correspondiente mensaje de error.

Se produjo el siguiente error: no se guardaba la sesión de usuario. Debido a esto, tuvimos que realizar pequeñas modificaciones en la clase usuario del modelo para que se almacenaran. Tras estos cambios, todas las pruebas tuvieron éxito.

6.3. Test realizados en la Fase 3 de la implementación

Se realizaron multitud de test en los cuales se comprobó el funcionamiento de cada caso de uso del usuario gestor, tanto su flujo correcto como el alternativo (explicados en la sección 3.4.2).

En este grupo de pruebas se encontraron los siguientes problemas:

- La recogida de la fecha de creación de calendarios en los casos de uso: modificar calendario y crear calendario.
- La selección de los conductores presentes en la creación del calendario. Este problema se detectó en el caso de uso de crear calendario.
- En la introducción de las opciones seleccionadas en el algoritmo de creación de calendarios anuales en los casos de uso crear calendarios y modificar calendarios, no se recogían las últimas opciones guardadas por el usuario, sino las más antiguas.
- En el caso de uso de añadir conductor, se podía introducir un identificador ya existente, debido a que no se comprobaba en la base de datos si ya existía el identificador del nuevo conductor que se pretendía añadir.
- La inclusión de fechas, tanto en el caso de uso crear calendario, como en modificar calendario, no se comprobaba que las fechas seleccionadas fueran del mismo año y distintas entre sí.
- La posibilidad de visualizar un aviso durante la ejecución del algoritmo de creación de calendarios, tanto en el caso de uso de crear calendario, como en modificar calendarios. Se solucionó introduciendo líneas de código en *JavaScript* que gestionasen el flujo de sus vistas.

Como se puede apreciar, la mayor parte de los problemas se localizaron en los casos de uso *Crear calendarios* y *Modificar calendarios*. Una vez resueltos todos estos problemas, los test realizados en los casos de uso del gestor fueron un éxito.

6.4. Test realizados en la Fase 4 de la implementación

Las pruebas realizadas en esta fase son similares a las de la iteración anterior, es decir, comprobamos el funcionamiento correcto y el alternativo de los casos de uso, pero esta vez el correspondiente a los conductores. Tan solo se produjo un error en el caso de uso *Crear notificación*, que no guardaba en la base de datos la notificación realizada por el usuario. Una vez corregido este problema, todas las pruebas fueron un éxito.

7. Manuales

En esta sección veremos los distintos manuales de uso y de instalación en el servidor.

7.1. Manual de uso del Gestor

El acceso a la web se realiza escribiendo en el navegador la dirección dada por el servidor. Una vez dentro de la web, deberá introducir el usuario y la contraseña.

Las acciones que se pueden realizar son:

* **Modificar o ver las opciones de creación.**

1. Deberá encontrarse en el menú principal.
2. A continuación pulsará "Opciones" y las modificará a su gusto.
3. Si desea guardar las opciones deberá pulsar "confirmar", si desea desechar las opciones deberá pulsar "Volver".

* **Modificar conductores.**

1. Deberá encontrarse en el menú principal.
2. A continuación pulsará en "Gestionar conductores".
3. Se abrirá una tabla con todos los conductores en la base de datos, en ella podrá editar, borrar o añadir nuevos conductores.

* **Ver notificaciones.**

1. Deberá encontrarse en el menú principal.
2. A continuación pulsará en "Notificaciones".
3. Una vez hecho esto podrá ver las notificaciones de los conductores y borrar algunas.

* **Generar un calendario.**

1. Deberá encontrarse en el menú principal.
2. A continuación deberá pulsar "Crear calendario".
3. Se abrirá un menú de opciones que deberá rellenar con la información que desee.
4. Una vez rellenados los datos pulsará en "Continuar" y esperará a que se cree el calendario.



Figura 80: Pantalla de ejecución del algoritmo.

* **Seleccionar un calendario.**

1. Deberá encontrarse en el menú principal.
2. A continuación deberá pulsar en "Seleccionar calendario".
3. Se abrirá un menú que deberá rellenar con la información del calendario que desee.
4. Por último pulsar "Continuar" y esperar a que el calendario se muestre.

* **Asignar líneas.**

1. Deberá tener un calendario seleccionado o creado.
2. Pulsará "Asignar líneas" en el menú superior.
3. Esperará a que concluya la asignación y salga un mensaje para indicar que ha terminado.



Figura 81: Pantalla de asignación de líneas.

* Ver calendario de un conductor individual.

1. Deberá tener un calendario seleccionado o creado.
2. A continuación deberá pulsar "Buscar conductor".
3. Se abrirá una tabla con los conductores en el calendario y pulsará "Ver calendario" en el conductor deseado.
4. Esperar y se mostrará el calendario del conductor seleccionado.

Nombre: Conductor217 | ID: 4

ESTADÍSTICAS

Día	Número de turnos con 6 días seguidos de trabajo												
	ENE	FEB	MAR	ABR	MAY	JUN	JUL	AGO	SEP	OCT	NOV	DIC	
1	D0	V0	M2	D0	T0	T2	D0	V0	M2	D0	D0	T2	
2	D0	V0	M2	D0	T2	T2	D0	V0	M2	D0	D0	T2	
3	T2	V0	M2	D0	T2	T2	D0	V0	M2	M2	D0	D0	
4	T2	V0	M0	M2	D0	T2	T2	V0	D0	M2	M2	D0	
5	T2	V0	M2	M2	D0	D0	T2	V0	D0	M2	M2	D0	
6	T2	V0	D0	M2	M2	D0	D0	T2	V0	D0	M2	M0	D0
7	T2	V0	D0	M2	M2	D0	T2	V0	T2	M2	M2	M2	
8	D0	V0	D0	M2	M2	M2	T2	V0	T2	D0	M2	M2	
9	D0	V0	D0	D0	M2	M2	D0	V0	T2	D0	D0	M2	
10	D0	V0	T2	D0	M2	M2	D0	V0	T2	T2	D0	M2	
11	M2	V0	T2	D0	M2	M0	D0	V0	T2	T2	D0	M2	
12	M2	V0	T2	T2	D0	M2	M2	V0	D0	T2	T2	D0	
13	M2	V0	T2	T2	D0	M2	M2	V0	D0	T2	T2	D0	
14	M2	V0	T2	T0	D0	D0	M2	V0	D0	T2	T0	D0	
15	M2	M2	D0	T2	T2	D0	M2	V0	D0	D0	T2	T2	
16	D0	M2	D0	T2	T2	D0	M2	M2	M2	D0	T2	T2	
17	D0	M2	D0	D0	T2	T2	D0	M2	M0	D0	D0	T0	
18	D0	M2	M2	D0	T2	T2	D0	M2	M0	M2	D0	T2	
19	T2	D0	M2	D0	T2	T0	T2	D0	M2	M2	D0	T2	
20	T2	D0	M0	M2	D0	T2	T2	D0	M2	M2	D0	D0	
21	T2	T2	M2	M2	D0	T2	T2	T2	D0	M2	M2	D0	
22	T0	T2	M2	M2	D0	D0	T2	T2	D0	M2	M0	D0	
23	T2	T2	D0	M2	M2	D0	T2	T2	D0	D0	M2	M2	
24	T2	T2	D0	M2	M2	D0	D0	T2	T2	D0	M2	M2	
25	D0	T2	D0	D0	M2	D0	D0	T2	T2	D0	M2	M0	
26	D0	D0	D0	D0	M2	M2	D0	T2	T2	T2	D0	M2	
27	D0	D0	T2	D0	M2	M2	D0	D0	T2	T2	D0	M2	
28	M2	D0	T2	T2	D0	M2	M2	D0	T2	T2	T2	D0	
29	M2	D0	T2	T2	D0	M2	M2	D0	D0	T0	T2	D0	
30	M0	D0	T2	T2	D0	M2	M2	M2	D0	T2	T2	D0	
31	V0	D0	T2	D0	T2	D0	M2	M2	D0	T2	D0	D0	

Figura 82: Pantalla de visualización de los calendarios individuales.

* **Modificar unas fechas concretas de un calendario.**

1. Deberá tener un calendario seleccionado o creado.
2. Pulsará "Modificar calendario" en el menú superior.
3. A continuación añadirá las fechas que desee modificar y pulsará "continuar".
4. Esperará a que el algoritmo haya terminado y se realizarán los cambios pertinentes a los días asignados.

* **Guardar datos.**

1. Deberá tener un calendario seleccionado o creado.
2. Pulsará "Guardar" en el menú superior.
3. Se le abrirá un menú para que elija si desea guardar la demanda o el calendario, en el que seleccionará lo que desee.
4. Se abrirá una página para que escriba el nombre con el que lo quiere guardar en la BBDD y pulsará "Continuar".
5. Esperará y cuando se guarde en la base de datos volverá a la visualización global de la demanda.



The screenshot shows a mobile application interface with a light blue background. At the top center, there is a red rectangular button with the text "Volver" in white. Below this, centered, is a white rectangular dialog box. Inside the dialog box, the text "Guardar Demanda restante en la BBDD" is displayed in a bold, dark grey font. Below the title, there is a smaller line of text "Nombre de la BBDD" followed by a light grey text input field containing the text "nueva_demanda". At the bottom of the dialog box, there is a blue rectangular button with the text "Guardar" in white.

Figura 83: Pantalla de Guardado de la demanda.

7.2. Manual de uso del Conductor

El acceso a la web se realiza escribiendo en el navegador la dirección dada por el servidor. Una vez dentro de la web deberá introducir su usuario y contraseña.

* **Ver el calendario anual.**

1. Deberá encontrarse en la página principal del sistema. En ella podrá ver su calendario y algunas estadísticas generales del calendario.

* **Escribir una reclamación o una notificación.**

1. Deberá encontrarse en la página principal del sistema.
2. En ella pulsará en el menú superior el botón de notificar problema.
3. Escribirá lo que desea notificar y pulsará "Enviar". Si se ha enviado con éxito recibirá una notificación. En caso contrario un aviso de error.

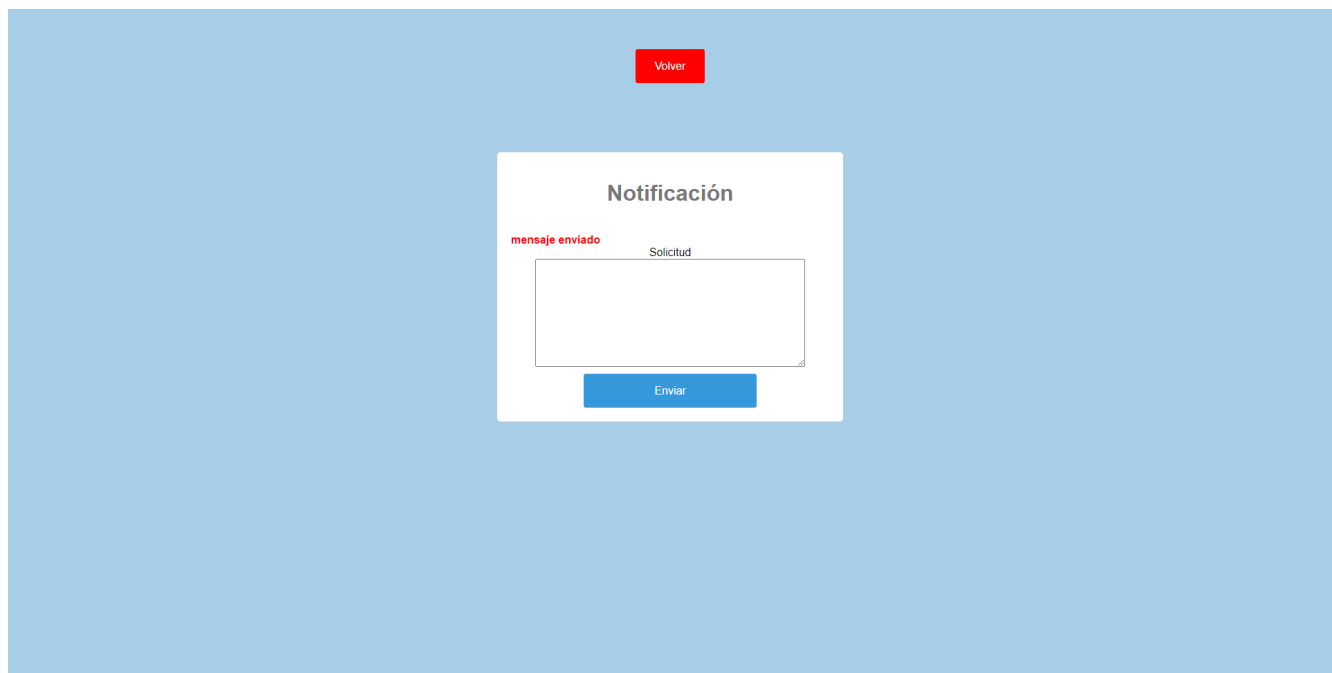


Figura 84: Pantalla de creación de notificaciones por parte del conductor cuando se envía un mensaje con éxito.

7.3. Manual de Instalación

Esta aplicación ha sido desarrollada para ser ejecutada en Windows, pero debería poder hacerse en sistemas Linux.

7.3.1. Paso 1: Instalación de Python

El primer paso de la instalación es descargar Python. La versión que necesitamos es la 3.10.5.

Para Windows:

Se puede descargar e instalar siguiendo los pasos de la siguiente página web:

<https://python.uptodown.com/windows/descargar>.

Para Linux:

Se puede instalar python desde la página web:

<https://www.python.org/downloads/source/>.

También se puede hacer de una forma más sencilla a partir de una terminal con el comando: `apt install python3.10 -y`.

Para comprobar que python está instalado, escribir `python` en la CMD en Windows o en la terminal en el caso de Linux. Si tras esto, observamos el menú de python, la instalación ha sido un éxito. En otro caso, aun no se encuentra instalado.

```
C:\Users\VersusPC>python
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figura 85: Salida de python.

7.3.2. Paso 2: Instalación de las librerías necesarias

Se instalarán las librerías necesarias desde el CMD, en Windows, o con una terminal, en linux. En primer lugar:

Ejecutar el siguiente comando en el CMD de Windows:

```
python get-pip.py.
```

Ejecutar el siguiente comando en la terminal de Linux:

```
sudo apt-get install python3-pip.
```

Después reiniciar el ordenador y se iniciará el proceso de instalación de las librerías.

- Librería para la lectura de la base de datos:

```
pip install mysql-connector-python
```

- Librería con la lógica de vectores y matrices:

```
pip install numpy
```

- Librería para desarrollar el servidor:

```
pip install flask
```

- Librería de gestión de usuarios:

```
pip install flask-login
```

- Librería para leer los ficheros html:

```
pip install jinja2
```

- Para lectura de ficheros excel, bloc de notas...:

```
pip install pandas
```

- Librería de útiles para la conexión WSGI:

```
pip install werkzeug
```

- Librería para el manejo de fechas:

```
pip install datetime
```

Al finalizar la instalación de las librerías, reiniciar el equipo y continuar con el siguiente paso.

7.3.3. Paso 3: Instalación y preparación de la Base de datos

Para instalar y preparar la base de datos necesitará ir a la página

<https://www.apachefriends.org/es/download.html> y seguir las instrucciones para instalar Xampp.

Una vez instalado, ejecute dicha aplicación y saldrá su menú de arranque (como vimos en la figura 9). Se iniciará Apache y MySQL pulsando el botón start en ambos.

Añadimos una base de datos buscando en el buscador la dirección

<http://localhost/phpmyadmin/> y a continuación clicar en Nueva.

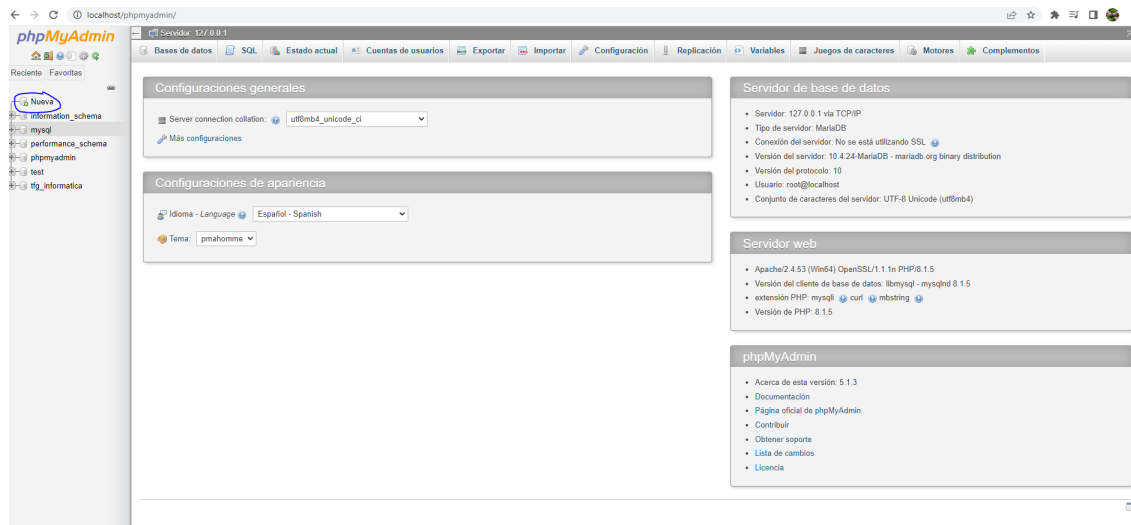


Figura 86: Crear base de datos paso 1.

Por último, teclear el nombre TFG_Informatica y pulsar “Crear”.

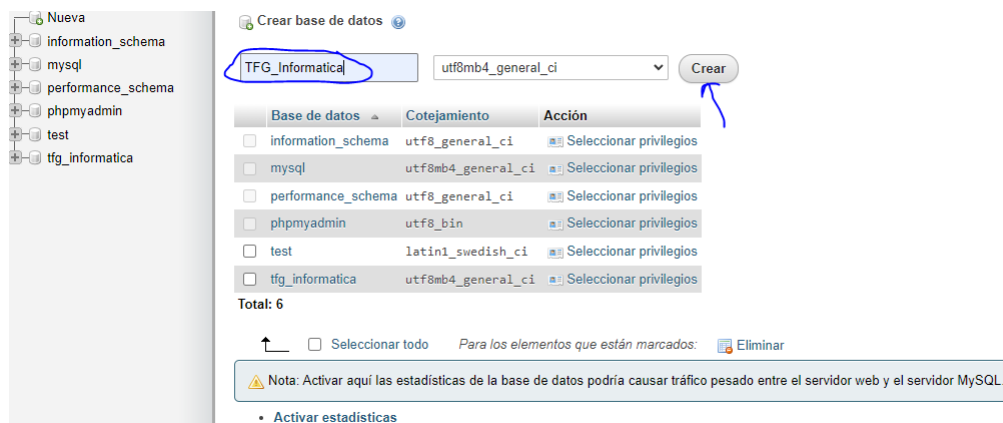


Figura 87: Crear base de datos paso 2.

A continuación situar el terminal en la carpeta 'programa' (se puede mover a través de las carpetas con cd) y ejecutar el comando:

```
python ./crear_BBDD/guardar_BBDD.py
```

Con esto tendremos creadas las tablas de la base de datos y la propia base de datos.

7.3.4. Paso 4: Activar el servidor

Situados en la carpeta de 'programa' ejecutamos cd ../aplicacion para movernos a esa carpeta. Los comandos para iniciar el servidor web son:

```
set FLASK_APP=app python -m flask run --host=dirección para crear el acceso
```

Un ejemplo de host es el 0.0.0.0 pero puede ser cualquier otro.

Para acceder a través de la web a nuestro servidor sólo se debe añadir la segunda dirección localizada en Running on ... Debe ser similar a "http://192.168.1.74:5000".

8. Conclusiones y líneas futuras

8.1. Conclusiones

El trabajo se ha desarrollado de forma satisfactoria, completando el objetivo principal de desarrollar una aplicación que se encargue de gestionar los turnos de trabajo de los conductores de la plantilla. Gracias a esta aplicación, la empresa AUVASA podrá elaborar y controlar de forma cómoda y rápida los calendarios. Por ello, su necesidad de generar calendarios anuales queda satisfecha.

Por otro lado, la adaptación del algoritmo básico con el que contábamos, ha sido nuevamente un éxito, pues obtiene resultados lo suficientemente adecuados como para tenerlos en cuenta.

Los objetivos alcanzados a nivel personal son los siguientes.

- Los conocimientos de aplicaciones web se han ampliado en gran medida. Aunque con Flask y Python se eliminan la mayor parte de las complejidades, se ha estudiado la posibilidad de llevarlo a cabo con otros lenguajes y por lo consiguiente su estudio web para realizarlo en los mismos.
- El conocimiento de los lenguajes elegidos se ha ampliado enormemente, tanto en HTML, como en la librería Flask de Python.
- Se ha profundizado de manera sustancial en la aplicación y estudio de los conocimientos para la creación de turnos de trabajo.
- Por último, destacar también el crecimiento experimentado en la gestión de proyectos, tanto en la experiencia, como en la gestión de imprevistos.

El proyecto ha resultado exitoso, pues se han cumplido todos los objetivos tanto en el ámbito personal como a nivel de la implementación de la aplicación.

8.2. Líneas futuras

Entre las mejoras futuras que se pueden llevar a cabo en la aplicación podemos citar:

- La posibilidad de que el gestor realice una modificación de forma manual.
- La inclusión de chat entre conductores.
- La utilización en el algoritmo heurísticas de simulated annealing (enfriamiento simulado) u otras.
- Ampliar la gestión de los datos almacenados de cada usuario.

Bibliografía

- [1] A. Beer, J. Gärtner, N. Musliu, W. Schafhauser. An AI-based break-scheduling system for supervisory personnel. *Intelligent System*, IEEE **25** (2), 60–73, 2010.
- [2] N. Musliu, A. Schaerf, W. Slany. Local search for shift design. *European Journal of Operational Research*, **153**, 51–64, 2004.
- [3] M. Widl, N. Musliu. The break scheduling problem: complexity results and practical algorithms. *Memetic Computing*, **6** (2), 97–112, 2014.
- [4] F. Buschmann , R. Meunier, H. Rohnert, P. Sommerlad, M. Stal. Pattern-Oriented Software Architecture Volume 1: A System of Patterns. Wiley, ISBN 978-0471958697, 1996
- [5] E. Gamma, R. Helm, R. Johnson, J.M. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Prentice Hall. ISBN-13 978-0201633610, 1994
- [6] B. Hughes, M. Cotterell. Software Project Management. McGraw-Hill Publishing Co. ISBN 978-0070473850, 2002
- [7] J.F. Kurose, K.W. Ross. Redes de computadoras: Un enfoque descendente. Pearson. ISBN 978-8478291199, 2010.
- [8] Guía de Flask. 2010. Última consulta: 7 Junio 2022: <https://flask.palletsprojects.com/en/2.1.x/>
- [9] Tutorial html. 1999-2022. Última consulta: 10 Junio 2022: <https://www.w3schools.com/html/>
- [10] Guía usuario de Astah. 2022. Última consulta: 6 Mayo 2022: <https://astah.net/support/astah-pro/user-guide/>
- [11] Guía básica para la administración de proyectos. 2022. Última consulta: 30 Abril 2022:
<https://support.microsoft.com/es-es/office/gu%C3%ADa-b%C3%A9sica-para-la-administraci%C3%B3n-de-proyectos-ad8c7625-fa14-4e36-9a83-c6af33>
- [12] Códice Gestión de la información. 2022. Última consulta: 28 Abril 2022: <http://codicegestion.com/>
- [13] NumPy user guide. 2008–2022. Última consulta: 31 Mayo 2022: <https://numpy.org/doc/stable/user/index.html>