ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA DEL SOFTWARE

Una herramienta web para la planificación de horarios y de asignaciones de aulas.

A web-based tool for timetable scheduling and classroom allocation.

REALIZADO POR

Jorge García Ruiz

TUTORIZADO POR

Eduardo Guzmán de los Riscos

DEPARTAMENTO

Lenguajes y Ciencias para la Computación

UNIVERSIDAD DE

MÁLAGA, SEPTIEMBRE DE 2015

Fecha defensa:

El Secretario del Tribunal

Resumen:

Este proyecto ha consistido en la realización de una aplicación web para cubrir una necesidad del centro universitario; la necesidad de gestionar los horarios y la asignación de aulas cada inicio de curso de una forma más automática de la que se usa en la actualidad.

El objetivo principal ha sido que la aplicación fuera lo más intuitiva posible y que tenga una facilidad de uso que motivase su utilización. Para ello nos hemos decantado por una serie de características tanto visuales (distribución de la información, códigos de colores), como de uso (sistema de Drag&Drop).

La aplicación tiene tres módulos principales:

Un primer módulo para el manejo de los horarios, el cual nos permite la construcción de un horario para un grupo determinado evitando cualquier tipo de conflicto.

El segundo módulo para la asignación de grupo a aulas, de forma que nos permite tener un control de los espacios del centro.

El tercero y último, con una fuerte relación con el segundo, que en este caso se utiliza para asignar asignaturas optativas a aulas.

Todo esto va acompañado por una variedad de páginas de información y de una vista administrativa para gestionar los datos necesarios para usar la aplicación. Una de las ventajas más importantes que tiene esta aplicación es la automatización a la hora de realizar todas las comprobaciones necesarias para hacer cualquier asignación.

Debido a la envergadura del proyecto, optamos por realizar este proyecto conjuntamente entre tres personas para ser capaces de ofrecer un producto lo más completo posible sin salirnos de los límites de este trabajo de fin de grado.

Palabras claves:

Gestión de horarios, gestión de aulas, gestión de asignaturas, aplicación web, SCRUM, XP (eXtreme Programming), Python, Django, JQuery, Backbone,.

Abstract:

This project has consisted in the development of a web-based application to fulfill a need of the school; to build schedules and to assign classrooms at the beginning of each course, in a more automatic way than the used currently.

Our target were to make the application as intuitive as possible and providing it of a ease way to use, motivating its usage. For all that we have chosen using a serie of features, some of them are visual (distribution of information, color codes) and others regarding usability (Drag&Drop system).

The application has three principal modules:

A first module for the schedule management, which allows us to build a schedule for a particular group avoiding conflicts.

The second one is about the groups and classrooms assignment, which allows us to have certain control over the centre spaces.

And the third, which has a strong relation with the second one, is used to assign optional subjects to classrooms.

All of this is accompanied by a diversity of information pages and an administration view for managing all necessary data for the application.

One of the more important advantages of all this is the automation of the constraints checking every time we want to make an assignment.

Due to the magnitude of the project we chose to make a project team of three members, so we could deliver a finished application without going beyond of the TFG limits.

Keywords:

Schedule manager, classroom manager, optative subject manager, SCRUM, XP (eXtreme Programming), Python, Django, JQuery, JQueryUI, Backbone.

Índice:

· Backbone

· Backbone-Tastypie

· Marionette

· Materialize

Introducción

Estructura de la memoria:

La memoria consta de varias partes. Una primera parte de introducción que permite al lector hacerse una idea general sobre el proyecto. En esta primera parte se describe muy fugazmente qué motivó y con qué objetivo se realizó el proyecto. También se hace un repaso por encima de las diferentes tecnologías que se han utilizado a lo largo del desarrollo. De esta forma cualquier persona que lea este apartado puede decidir si está interesado en aprender más sobre él o no.

La segunda parte entra más en profundidad en el proyecto en sí. Para empezar se realiza una descripción general de todo el proyecto, de forma que quien lea esta parte puede saber lo que va a ver cuando trabaje con la aplicación. En esta parte se explica las distintas funcionalidades que trae y se dan algunos detalles que se consideran importantes, pero no se entra en detalles técnicos.

Tras esto se explica la metodología de desarrollo que se ha seguido durante el transcurso del proyecto y se dan detalles del funcionamiento y de las cosas positivas y negativas que han surgido de su utilización.

El siguiente apartado habla sobre la parte de especificación y análisis. En esta parte se cuenta la parte en la que forma grupal se realiza una formalización del proyecto.

Otro apartado explica la parte de diseño de la aplicación. En concreto el diseño del modelo de datos y del proyecto y su estructura.

Para concluir esta segunda parte se habla sobre todo el desarrollo realizado, explicando los pasos que se han seguido desde el principio de la implementación hasta la finalización de la aplicación.

Por último la tercera parte son las conclusiones del proyecto, una evaluación global de todo el proceso donde se comentan tanto lo que se ha conseguido y los aspectos positivos como las cosas negativas que han podido mejorarse. Para finalizar se analizan algunas posibles ampliaciones que se podrían aplicar a la aplicación en el futuro.

Motivación y objetivos del proyecto:

En la actualidad el proceso que se sigue a la hora de planificar un nuevo curso, es decir la confección de horarios y la asignación de la ocupación de las aulas, se realiza de una forma poco automatizada, de forma que todas las restricciones que es necesario controlar a la hora de realizar estas acciones se controlan de una manera estrictamente manual. Esto resulta una tarea excesivamente pesada.

De ahí surge la necesidad y motivación de una herramienta que automatizara todos estos procesos y facilitara la realización de estas tareas.

El objetivo de este proyecto ha sido el desarrollo de esta herramienta de la cual se pueden distinguir tres módulos principales y una serie de pequeñas funcionalidades y páginas de información que acompañan a estos módulos.

Tecnologías:

En este apartado haremos un recorrido por las diferentes tecnologías que se han utilizado a lo largo de la planificación y desarrollo del proyecto. Están divididas en tres categorías: herramientas, tecnologías de frontend y tecnologías de backend.

- Herramientas y aplicaciones para el desarrollo:
    - VirtualBox:

Es un software de virtualización que utilizado instalar el sistema operativo que se ha utilizado a lo largo del proyecto, Ubuntu.

    - Sistema operativo Ubuntu:

Se ha trabajado con este sistema operativo debido a la buena compatibilidad y facilidad de uso en conjunto con otras de las tecnologías posteriormente descritas.

    - Pip

Pip (Pip Install Package) es un gestor de paquetes que se utiliza para instalar y manejar paquetes escritos en Python. Se ha utilizado fundamentalmente para instalar el framework principal sobre el que se ha trabajado y diferentes librerías que han sido necesarias durante el desarrollo.

    - Virtualenv

Es una herramienta que permite crear entornos aislados de Python. Esto permite que sea posible instalar una aplicación en un entorno virtual propio aislado del resto del entorno. Una ventaja que tiene esto es que evita posibles conflictos entre las dependencias de esta aplicación con los de otra, ya que existe la posibilidad de que diferentes aplicaciones requieran de diferentes versiones de las mismas dependencias.

- VirtualenvWrapper

Es un conjunto de extensiones para Virtualenv. Proporciona una serie de herramientas y utilidades para administrar todos los virtualenvs creados en el sistema. Facilita mucho el manejo de virtualenvs desde la consola.

- Pycharm

Es el entorno de desarrollo (IDE) sobre el que se ha realizado el desarrollo. Se trata de un IDE multiplataforma que se utiliza para programar en Python. Incluye integración con sistemas de control de versiones y soporta desarrollo web con Django. Estos son algunos de los motivos por los que nos decantamos por utilizar esta IDE.

- Git

Es un sistema de control de versiones. Debido a su fácil integración con la IDE y a que todos los integrantes del grupo estábamos familiarizados con su uso nos decidimos por él.

- MagicDraw:

MagicDraw es una herramienta de modelado de software que permite construir una gran variedad de diagramas de diferente tipo que resultan fundamentales a la hora de diseñar y planificar el diseño de una aplicación.

- Asana

Es una herramienta de gestión y organización de equipos. Proporciona una serie de herramientas que facilitan la ejecución de diferentes metodologías de desarrollo.

- Backend:

  - Python

Python es un lenguaje interpretado que utiliza tipado dinámico. Se trata de un lenguaje multiparadigma, esto quiere decir que soporta varios paradigmas de programación como programación orientada a objetos, imperativa y funcional. Una de las filosofías fundamentales de Python es que la sintaxis favorezca un código legible y en ello contribuyen algunas de las características de este. La elección por este lenguaje está basada en estas características ya comentadas y también en el hecho de que como

grupo queríamos aprovechar la oportunidad que nos brindaba este proyecto para aprender a manejar tecnologías que durante el transcurso de la carrera no hemos tenido la suerte de aprender y que forman parte de la actualidad.

- Django

Es un framework de Python que se utiliza para desarrollo web. Django se basa en el patrón de diseño Modelo-Vista-Controlador (MVC) aunque en este caso recibe el nombre Modelo-Plantilla-Vista (MTV como en Model-Template-View) debido a que en el caso de Python la 'C' correspondiente a Controlador es manejada por el mismo framework y realizan una interpretación distinta del patrón de diseño.

Es uno de los frameworks de desarrollo web de código abierto más completos de Python y además posee una amplia biblioteca de librerías y una extensa documentación que facilita en gran medida el trabajo a desarrollar.

- Tastypie

Es una aplicación de Django que permite crear APIs RESTful para poder acceder a los datos de los modelos definidos en la aplicación. Para poder realizar estos accesos es necesario definir recursos que van asociados a estos modelos. También ofrece la posibilidad de crear filtros para poder modificar qué datos y cómo los recibimos cuando accedemos a los recursos.

- xhtml2pdf

Librería escrita en Python que permite transformar documentos HTML en archivos PDFs de forma sencilla y práctica.

- Frontend

- JQuery

JQuery es una librería de JavaScript diseñada para simplificar la interacción con los documentos HTML y los elementos del DOM desde el lado del cliente. También incluye funcionalidades un poco más complejas como la creación de animaciones o la posibilidad de utilizar fácilmente AJAX. Es una de las librerías de JavaScript más populares y más usadas del mundo.

- JQuery UI

Es una librería de interfaces de usuario oficial de JQuery. Esta añade un conjunto de widgets, plugins y efectos visuales que pueden utilizar durante el desarrollo web para crear aplicaciones webs más completas e interactivas.

- JQuery UI Touch Punch

Es una pequeña librería que permite que aprovechar las ventajas que proporciona JQuery UI en dispositivos móviles. Ya que a día de hoy estos dispositivos no detectan bien los eventos de toque lo cual no permite aprovechar JQuery UI a su máximo potencial.

- Underscore

Otra librería JavaScript que proporciona una gran cantidad de funcionalidades. Es una dependencia de Backbone del cual se habla a continuación. Además permite renderizar contenido de manera dinámica en plantillas HTML.

- Backbone

Backbone es de nuevo una librería JavaScript que se basa en el paradigma Modelo-Vista-Presentación (MVP) que es una variación del MVC e integra este mismo en la vista de una aplicación web. Esta librería permite crear modelos que se corresponden a los que existen en la base de datos y colecciones de estos datos para poder trabajar con objetos que adquirimos desde la base de datos de una forma más sencilla.

- Backbone-Tastypie

Backbone-Tastypie como su propio nombre indica es una librería que integra Backbone con Tastypie de forma que sincroniza los modelos de ambas partes. Esta librería sobrescribe varios métodos de Backbone de forma que podamos realizar llamadas a los recursos de Tastypie y encapsular los resultados que obtenemos en los modelos de Backbone. Y de esta forma podemos tener los datos sincronizados entre el servidor y el cliente en todo momento.

- Marionette

En una librería que complementa y trabaja sobre Backbone. Proporciona una variedad de funciones y utilidades para completar aquellas partes en las que Backbone flojea un poco. Permite organizar el código de una forma más intuitiva y construir estructuras más ricas y formadas por pequeños componentes. Es muy potente para renderizar el contenido de forma dinámica en nuestras páginas web.

- Materialize

Es un framework CSS responsive de reciente creación que proporciona un conjunto de componentes y animaciones basados en Material Design de Google.

Desarrollo del proyecto:

Descripción general del proyecto

Tal y como se introdujo al principio, el objetivo de este proyecto es satisfacer una necesidad presente en la Escuela, y para ello se planteó una aplicación web que hiciera más fácil una tarea que en este momento puede llegar a ser bastante tediosa y complicada.

Por ello esta aplicación web debe ser capaz de realizar algunas de esas tareas necesarias cada vez que se realiza la planificación de un nuevo curso académico.

Como ya se ha dicho con anterioridad el proyecto consta de 3 módulos principales para cumplir con este propósito.

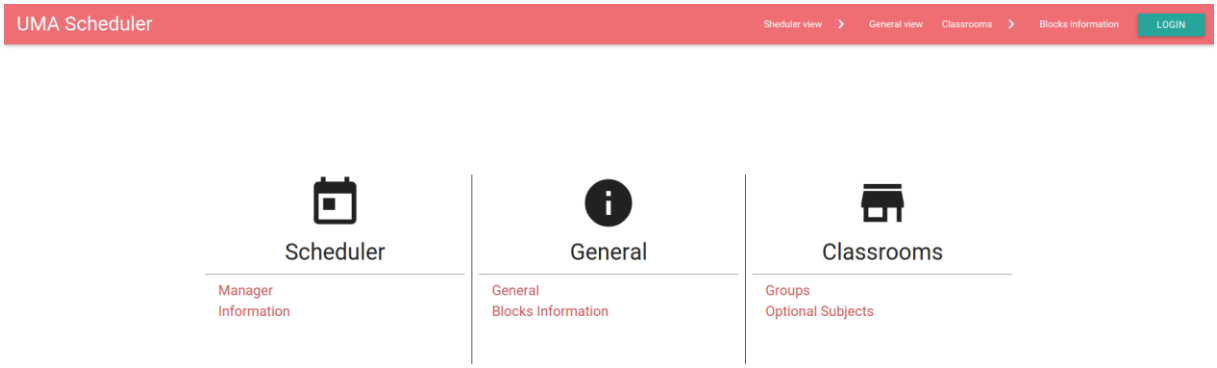

ILUSTRACIÓN 1. PÁGINA DE INICIO

En primer lugar tenemos el módulo para el manejo de horarios. En este módulo será posible crear horarios para los grupos que existen en el sistema. A través de un sistema Drag&Drop podemos ir asignando las diferentes asignaturas a las franjas horarias que creamos convenientes. La aplicación se encargará de realizar las comprobaciones pertinentes entre las que se encuentran el no poder asignar dos veces la misma asignatura al mismo día, o no asignar más franjas de las necesarias a una asignatura, lo cual dependerá del número de créditos de esa asignatura. Ya dependiendo del tipo de asignatura se comprobarán otros tipos de restricciones. Ya que en el caso de tratarse de asignaturas ligadas, compartidas o asignaturas

optativas, su asignación afecta a las asignaturas de otros grupos, por lo que hay que asegurarse de que todas ellas cumplen los requisitos. También seremos capaces de intercambiar asignaturas de forma que las restricciones se cumplan para ambas asignaturas y podremos eliminar asignaciones realizadas previamente. Por último la aplicación nos permite seleccionar cuál de las franjas asignadas a una asignatura es la franja de grupo grande, que se corresponde a la franja en cuyas horas se impartirá la clase teórica de esa asignatura.

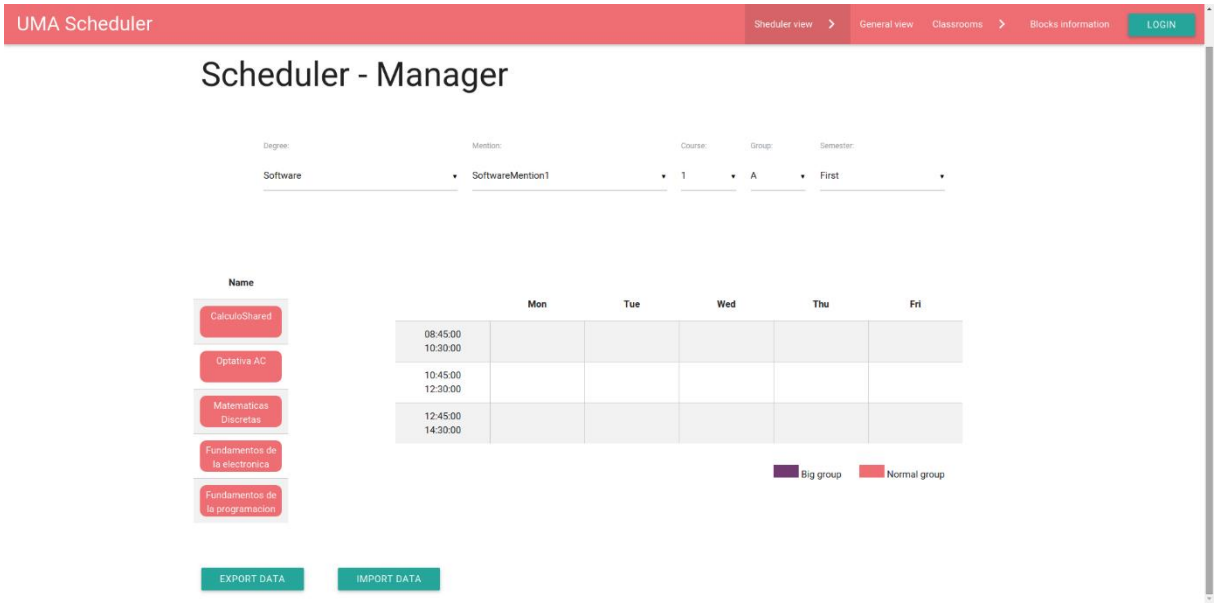

ILUSTRACIÓN 2. PRIMER MÓDULO

En un segundo módulo podremos tener un control sobre los espacios del centro en relación a los grupos, es decir, podremos asignar los distintos grupos a las aulas existentes, teniendo en cuenta tanto la capacidad del aula como la ocupación de la misma. De esta forma evitamos conflictos en los que se imparta más de una asignatura a la vez en el mismo aula. En este caso aunque estemos asignando grupos a aulas también tenemos en cuenta que las aulas pueden tener asignaturas optativas asociadas y evitamos que se generen conflictos también de esta forma. En esta vista nos basamos en un código de colores para transmitir la información necesaria para un uso eficiente del módulo. De esta forma indicamos con un color cuando un aula se encuentra vacía, otro color cuando se encuentra ocupada por otro grupo y un último color para cuando solo está ocupada por otras asignaturas optativas. Igual que en el anterior módulo, utilizamos un sistema Drag&Drop a la hora de realizar las asignaciones. Y ya a nivel interno la aplicación se encarga de comprobar que todas las restricciones son satisfechas antes de consumar la asignación.

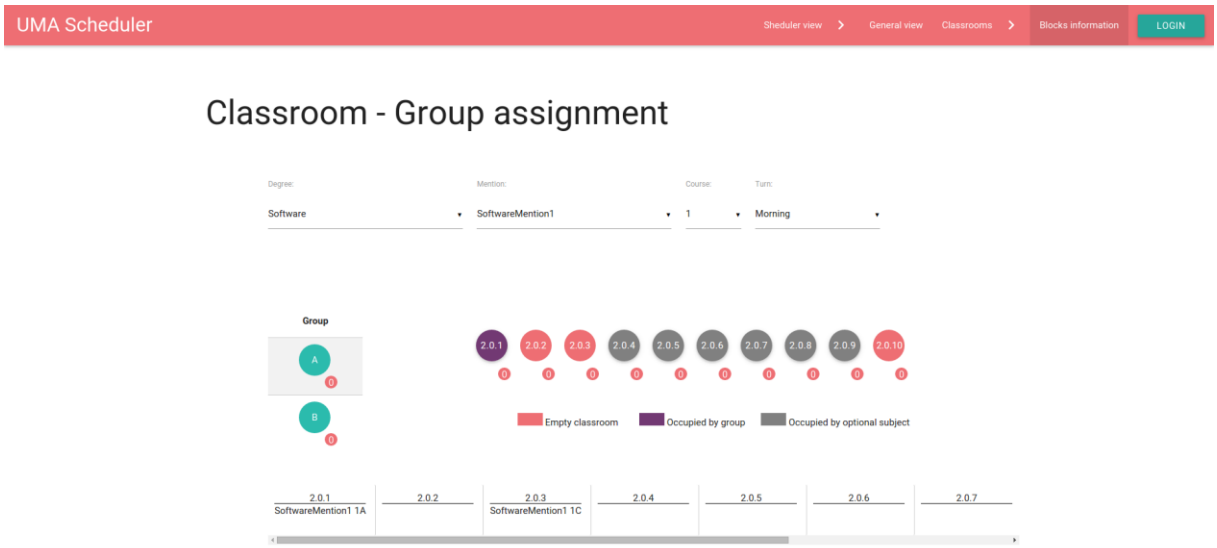

ILUSTRACIÓN 3. SEGUNDO MÓDULO

Por último tenemos un módulo sobre el papel muy similar al segundo, ya que también tiene como propósito principal la ocupación de las aulas, pero en este caso lo que asignamos a estas son las asignaturas optativas. Debido a esto la lógica que funciona por detrás a la hora de realizar las asignaciones es totalmente opuesta al módulo anterior. Pero la vista es fundamentalmente similar, aunque con ciertos detalles necesarios para mostrar la información necesaria, ya que en este caso asignamos las asignaturas optativas que a su vez dependen de los bloques.

ILUSTRACIÓN 4. TERCER MÓDULO

Aparte de estos tres módulos tenemos una serie de vistas de información para complementar a estos módulos.

Una vista de información sobre los horarios, en esta vista se podrá consultar los horarios construidos en el primer módulo comentado anteriormente. A través de los selectores es posible filtrar para ir viendo los horarios de cada grupo dividido por semestres. En esta vista también se incluye una opción para poder exportar el conjunto de todos los horarios a pdf, de forma que se pueda generar un documento con toda la información para poder imprimirla.

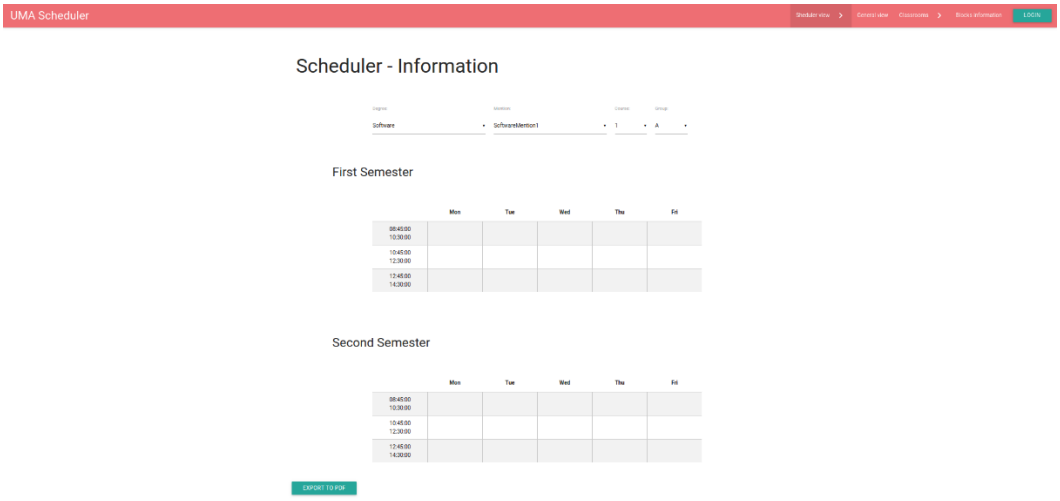

ILUSTRACIÓN 5. INFORMACIÓN SOBRE LOS HORARIOS

Otra vista llamada Visión General, en la cual se puede visualizar la ocupación en todo momento de las aulas. Para ello se tiene la capacidad de filtrar por el semestre, el día de la semana y el turno (mañana o tarde) y se mostrará una tabla en la que podremos ver qué hay en cada aula dividido por horas.

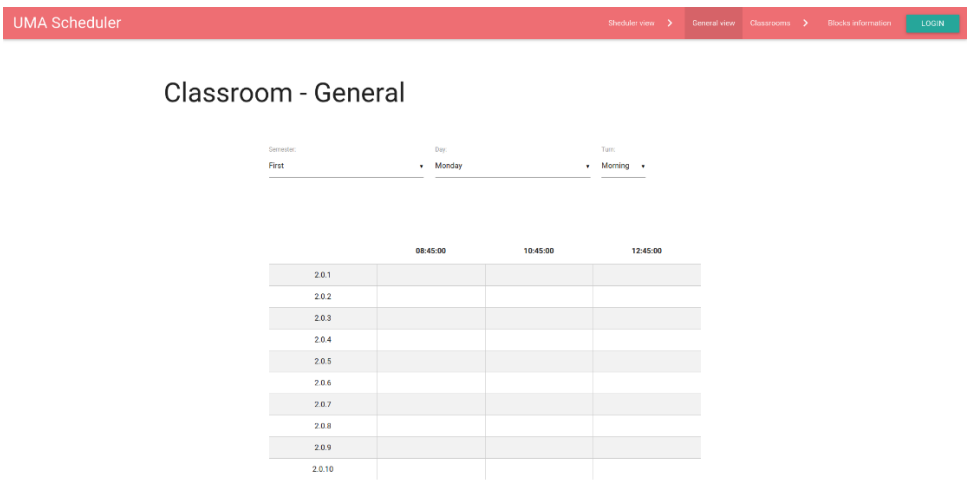

ILUSTRACIÓN 6. VISIÓN GENERAL

Una última vista en la aparecerá la información de los bloques de optativas, de forma que se verá el conjunto de asignaturas optativas que conforman cada bloque además de información correspondiente al bloque en sí mismo.

ILUSTRACIÓN 7. INFORMACIÓN BLOQUES DE OPTATIVA

Para concluir tendrá una vista de administración. Esta vista permitirá crear todos los datos necesarios para poder ejecutar la aplicación de manera correcta. Para ello, a la hora de generar los diferentes datos, se encargará de evitar que existan inconsistencias en ellos, ya que existen una variedad de restricciones a tener en cuenta a la hora de crear cierto tipo de objetos.

Metodología de trabajo

La elección de la metodología de trabajo fue una de las más importantes de la fase inicial del trabajo. Esto se debe a que al tratarse de un proyecto en grupo la forma de organizarse tiene una importancia superior a un proyecto individual.

Finalmente nos decantamos por una metodología ágil como SCRUM en base a experiencias previas vividas durante prácticas realizadas durante la carrera. Y también porque consideramos que tener objetivos claros a corto plazo motiva en gran medida el esfuerzo y permite que el proyecto avance de una manera más fluida. Por ello definimos 'sprints' con una duración de dos semanas ya que la intención era que fuese lo más cortos posibles pero menos tiempo era demasiado escaso.

Debido a circunstancias durante una gran parte del desarrollo uno de los integrantes del grupo estuvo en el extranjero, por ello a la hora de repartir del trabajo se procuró que una de las partes estuviera más aislada y desconectada del resto, de tal forma que se pueden distinguir dos equipos de desarrollo distintos.

A pesar de esto al comienzo de cada sprint se mantenía una reunión, generalmente vía Skype debido a la dificultad que implicaba tener esas reuniones en persona, en la que se hacía una evaluación muy breve del sprint que acababa de terminar y se definían las tareas del nuevo sprint. Luego, durante la semana intermedia se hacía un leve repaso para saber cómo iban avanzando las tareas.

Para gestionar de una forma más eficaz todo este procedimiento se utilizó la herramienta Asana. Esta herramienta permite crear un proyecto e ir añadiéndole tareas con una fecha de inicio y una fecha de fin. De esta manera se puede llevar una monitorización de la evolución de los sprints y del proyecto en general. Para ello Asana proporciona algunas estadísticas muy útiles.

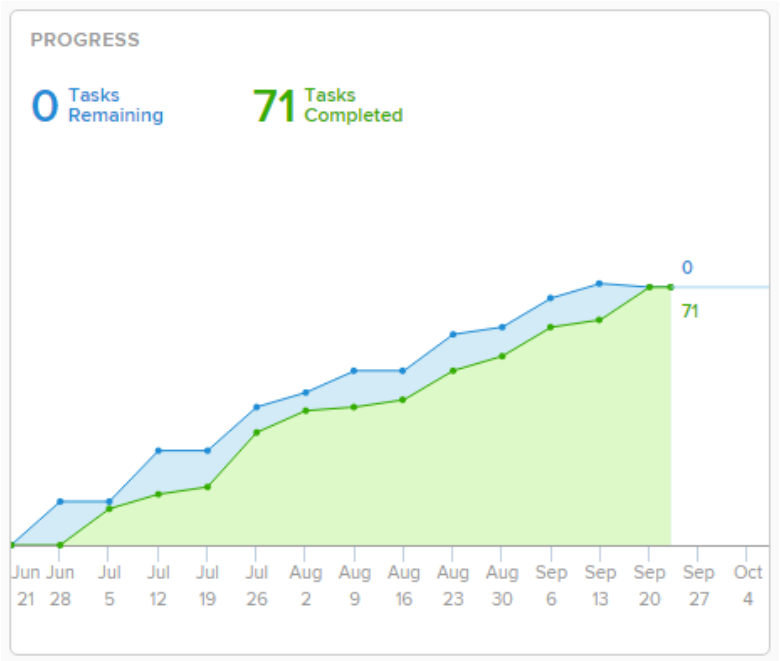

ILUSTRACIÓN 8. GRÁFICO DE SPRINTS DEL PROYECTO

En la imagen se puede observar la evolución durante todo el proceso de desarrollo. En azul se pueden observar las tareas pendientes y en color verde aquellas que ya han sido finalizadas. Lo ideal sería que cada dos semanas la línea azul y la verde intersecaran de forma que simbolizaran los sprints tal y como se observa en el primer sprint, pero debido a una mala estimación de algunas tareas del equipo que se encontraba fuera no es así. Se crearon algunas tareas de envergadura demasiado grande para la duración del sprints y eso se ha ido arrastrando durante la duración del proyecto.

Este seguimiento ha resultado muy útil porque permitía tener una idea global de la velocidad a la que avanzaba el proyecto. Además conforme pasaban los sprints las estimaciones de las tareas se realizaban con más precisión debido a la experiencia obtenida. Aunque por otro lado, el tener sprints en teoría tan cortos a veces producía problemas, porque cualquier imprevisto hacía que se complicara mucho cumplir los plazos de los sprints.

Por otra parte decidimos incluir también características de la metodología eXtreme Programming a nuestra forma de trabajar. Esto es debido a que una de las principales características de esta metodología es el énfasis que pone en la adaptabilidad durante el proceso de desarrollo. Debido a las características del proyecto y las necesidades de esto nos apoyamos en algunas de las características de esta metodología como el desarrollo iterativo e incremental, la refactorización del código o la programación por parejas. En concreto esto último se ha realizado para aquellas partes de la aplicación que poseían una complejidad mayor, de esta forma resultaba más sencillo programar entre dos.

Especificación y análisis del proyecto

A continuación se habla de las primeras fases del proyecto. Durante la cual de forma grupal se realizó toda la fase de análisis y especificación del proyecto.

Lo primero que hubo que hacer fue la definición del proyecto, es decir realizar un análisis de la especificación que nos habían dado y extraer de ella lo que debíamos hacer. Para ello tuvimos varias reuniones tanto entre nosotros como con nuestro tutor, ya que él era el que nos hacía de nexo de unión con el cliente que podía aclararnos todas aquellas dudas que nos surgieron al principio, puesto que consideramos que la especificación inicial que nos fue proporcionada no era la suficientemente clara y descriptiva en alguno puntos.

Tras todas estas reuniones nos hicimos una idea más clara del proyecto que teníamos que llevar a cabo y procedimos a plasmar estas ideas en una lista formalizada de requisitos. A continuación tenemos esos requisitos.

- RF1: El modelo contiene varios grados.
- RF2: Los grados están relacionados con varias menciones.
- RF3: Las menciones están relacionadas con varios cursos.
- RF4: Los cursos están relacionados con uno o más grupos.
- RF5: El modelo contiene varias asignaturas por cada grupo, curso y grado.
- RF6: El modelo contiene asignaturas de tipo "ligada".
- RF7: Las asignaturas "ligadas" puede estar ligada a otras asignaturas del mismo curso en el mismo u otros grados y deben tener el mismo horario.
- RF8: El modelo contiene asignaturas de tipo "compartida".
- RF9: Las asignaturas "compartidas" pueden estar relacionadas con otras asignaturas idénticas a estas de uno o más grupos y titulaciones, compartiendo el horario y el aula en el que se imparten.
- RF10: El modelo contiene varios bloques.
- RF11: Los bloques están compuestos de asignaturas del tipo "optativa común" o del tipo "optativa de mención", no mezclando ambos tipos en un mismo bloque.
- RF12: Las asignaturas de tipo "optativa común" deben ser comunes a varias menciones.
- RF13: Las asignaturas de tipo "optativa de mención" deben ser propias de una mención.
- RF14: Las asignaturas que estén en un mismo bloque deben tener el mismo horario.
- RF15: Las asignaturas de seis créditos tienen asignadas tres franjas horarias en un mismo horario.
- RF16: Las asignaturas de cuatro créditos y medio tienen asignadas dos franjas horarias en un mismo horario.

- RF17: Las asignaturas no podrán tener dos o más franjas asignadas un mismo día de la semana en un mismo horario.
- RF18: Una de las franjas asociada a una asignatura en un mismo horario se podrá marcar como "grupo grande".
- RF19: Es necesaria una página en la que configurar las asignaturas y franjas horarias, eligiendo el horario y asignaturas según el grado, mención, curso, grupo y semestre.
- RF20: Asignar una asignatura a una franja horaria en la página correspondiente al RF19.
- RF21: Desasignar una asignatura de una franja horaria en la página correspondiente al RF19.
- RF22: Intercambiar la posición en el horario de dos asignaturas en la página correspondiente al RF19.
- RF23: Marcar como "grupo grande" una franja horaria con una asignatura asignada en la página correspondiente al RF 19.
- RF24: Es necesario poder exportar/importar un estado de la base de datos para trabajar con varias configuraciones.
- RF25: Es necesaria una página en la que consultar la información configurable en el requisito RF19.
- RF26: Es necesario poder exportar en formato pdf la información representada en el requisito RF25 por cada una de las menciones de la base de datos.
- RF27: Es necesaria una página en la que consultar las asignaturas asignadas a un aula en cada una de las franjas horarias del horario de un semestre de un grupo, curso y mención dados.
- RF28: Es necesaria una página en la que poder configurar las aulas y los grupos.
- RF29: Asignar un grupo a un aula.
- RF30: Deasignar un grupo de un aula.
- RF31: Consultar las relaciones entre aulas y grupos.
- RF32: Es necesaria una página en la que configurar las asignaturas optativas y las aulas.
- RF33: Asignar una asignatura optativa a un aula.
- RF34: Desasignar una asignatura optativa a un aula.
- RF35: Es necesaria una página para consultar la información referente a los bloques (asignaturas contenidas, aula y horario).
- RF36: Es necesario poder exportar en formato pdf la información representada en el requisito RF35.
- RF37: Es necesaria una página que permita la creación de los modelos recogidos en los requisitos RF1, RF2, RF3, RF4, RF5, RF6, RF8, RF10 y RF11.

Después de terminar la definición de los requisitos se procedió a extraer de ellos la lista de casos de uso, en los que se define las diferentes acciones que puede realizar el usuario en el sistema. A continuación se encuentra la lista.

CU01. Autenticar

Resumen: El usuario se autentica en la aplicación.

Actor: Usuario.

Precondición: El usuario debe tener una cuenta registrada en la aplicación.

Escenario:

- El usuario accede a la página de login.
- Introduce su nombre de usuario.
- Introduce su contraseña.
- Hace click en el botón “Login".
- El sistema identifica al usuario.
- El usuario es redireccionado a la página principal.

CU02. Registrar un nuevo Grado

Resumen: El usuario registra un nuevo grado en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir un grado
- Introduce la información necesaria para crear un grado
- Pulsa el botón crear un grado
- El sistema crea el grado correctamente y aparece en la aplicación

CU03. Editar un Grado

Resumen: El usuario edita la información de un grado de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de un grado
- Introduce la información a cambiar en el grado
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU04. Borrar un Grado

Resumen: El usuario borra un grado registrado en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página un grado
- Pulsa el botón borrar
- El sistema borra el grado de la base de datos.

CU05. Registrar una nueva Mención

Resumen: El usuario registra una nueva Mención en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir una mención
- Introduce la información necesaria para crear una mención
- Pulsa el botón crear una mención
- El sistema crea la mención correctamente y aparece en la aplicación

CU06. Editar una Mención

Resumen: El usuario edita la información de una mención de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de una mención
- Introduce la información a cambiar en la mención
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU07. Borrar una Mención

Resumen: El usuario borra una mención registrada en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página una mención
- Pulsa el botón borrar
- El sistema borra la mención de la base de datos.

CU08. Registrar un nuevo Curso

Resumen: El usuario registra un nuevo curso en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir un curso
- Introduce la información necesaria para crear un curso
- Pulsa el botón crear un curso
- El sistema crea el curso correctamente y aparece en la aplicación

CU09. Editar un Curso

Resumen: El usuario edita la información de un curso de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de un curso
- Introduce la información a cambiar en el curso
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU10. Borrar un Curso

Resumen: El usuario borra un curso registrado en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página un curso
- Pulsa el botón borrar
- El sistema borra el curso de la base de datos.

CU11. Registrar un nuevo Grupo

Resumen: El usuario registra un nuevo grupo en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir un grupo
- Introduce la información necesaria para crear un grupo
- Pulsa el botón crear un grupo
- El sistema crea el grupo correctamente y aparece en la aplicación

CU12. Editar un Grupo

Resumen: El usuario edita la información de un grupo de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de un grupo
- Introduce la información a cambiar en el grupo
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU13. Borrar un Grupo

Resumen: El usuario borra un curso registrado en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página un grupo
- Pulsa el botón borrar
- El sistema borra el grupo de la base de datos.

CU14. Registrar una nueva Asignatura

Resumen: El usuario registra una nueva asignatura en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir una asignatura
- Introduce la información necesaria para crear una asignatura
- Pulsa el botón crear una asignatura
- El sistema crea la asignatura correctamente y aparece en la aplicación

CU15. Editar una Asignatura

Resumen: El usuario edita la información de una asignatura de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de una asignatura
- Introduce la información a cambiar en la asignatura
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU16. Borrar una Asignatura

Resumen: El usuario borra una asignatura registrada en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página una asignatura
- Pulsa el botón borrar
- El sistema borra la asignatura de la base de datos.

CU17. Registrar una nueva Asignatura Ligada

Resumen: El usuario registra una nueva asignatura ligada en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir una asignatura ligada
- Introduce la información necesaria para crear una asignatura ligada

- Pulsa el botón crear una asignatura ligada
- El sistema crea la asignatura ligada correctamente y aparece en la aplicación

CU18. Editar una Asignatura Ligada

Resumen: El usuario edita la información de una asignatura ligada de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de una asignatura ligada
- Introduce la información a cambiar en la asignatura ligada
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU19. Borrar una Asignatura Ligada

Resumen: El usuario borra una asignatura ligada registrada en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página una asignatura ligada
- Pulsa el botón borrar
- El sistema borra la asignatura ligada de la base de datos.

CU20. Registrar una nueva Asignatura Compartida

Resumen: El usuario registra una nueva asignatura compartida en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir una asignatura compartida
- Introduce la información necesaria para crear una asignatura compartida
- Pulsa el botón crear una asignatura compartida
- El sistema crea la asignatura compartida correctamente y aparece en la aplicación

CU21. Editar una Asignatura Compartida

Resumen: El usuario edita la información de una asignatura compartida de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de una asignatura compartida
- Introduce la información a cambiar en la asignatura compartida
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU22. Borrar una Asignatura Compartida

Resumen: El usuario borra una asignatura compartida registrada en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página una asignatura compartida
- Pulsa el botón borrar
- El sistema borra la asignatura compartida de la base de datos.

CU23. Registrar una nueva Asignatura Optativa

Resumen: El usuario registra una nueva asignatura optativa en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir una asignatura optativa
- Introduce la información necesaria para crear una asignatura optativa
- Pulsa el botón crear una asignatura optativa
- El sistema crea la asignatura optativa correctamente y aparece en la aplicación

CU24. Editar una Asignatura Optativa

Resumen: El usuario edita la información de una asignatura optativa de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de una asignatura optativa
- Introduce la información a cambiar en la asignatura optativa
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU25. Borrar una Asignatura Optativa

Resumen: El usuario borra una asignatura optativa registrada en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página una asignatura optativa
- Pulsa el botón borrar

- El sistema borra la asignatura optativa de la base de datos.

CU26. Registrar un nuevo Bloque de Optativas

Resumen: El usuario registra un nuevo bloque de optativas en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir un bloque de optativas
- Introduce la información necesaria para crear un bloque de optativas
- Pulsa el botón crear un bloque de optativas
- El sistema crea el bloque de optativas correctamente y aparece en la aplicación

CU27. Editar un Bloque de Optativas

Resumen: El usuario edita la información de un bloque de optativas de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de un bloque de optativas
- Introduce la información a cambiar en el bloque de optativas
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU28. Borrar un Bloque de Optativas

Resumen: El usuario borra un bloque de optativas registrado en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página un bloque de optativas
- Pulsa el botón borrar
- El sistema borra el bloque de optativas de la base de datos.

CU29. Registrar una nueva Clase

Resumen: El usuario registra una nueva clase en la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de añadir una clase
- Introduce la información necesaria para crear una clase
- Pulsa el botón crear una clase
- El sistema crea la clase correctamente y aparece en la aplicación

CU30. Editar una Clase

Resumen: El usuario edita la información de una clase de la aplicación.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página de editar de una clase
- Introduce la información a cambiar en la clase
- Pulsa el botón de guardar la información
- El sistema guarda la información correctamente.

CU31. Borrar una Clase

Resumen: El usuario borra una clase registrada en el sistema.

Actor: Administrador.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de administrador
- Accede a la página una clase
- Pulsa el botón borrar
- El sistema borra la clase de la base de datos.

CU32. Asignar una asignatura a una franja horaria vacía

Resumen: El usuario asigna una asignatura a una franja horaria vacía de un horario perteneciente a un grupo.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- El usuario arrastra una asignatura a una franja del horario vacía
- El sistema asigna la asignatura a esa franja horaria
- La asignatura aparece en la franja en la cual se ha asignado.

CU33. Desasignar una asignatura a una franja horaria

Resumen: El usuario desasigna una asignatura de una franja horaria de un horario perteneciente a un grupo.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- El usuario hace click en la cruz para desasignar una asignatura
- El sistema desasigna la asignatura de esa franja horaria
- La asignatura ya no aparece en la franja de la cual se ha desasignado.

CU34. Asignar una asignatura a una franja horaria con otra asignatura

Resumen: El usuario reemplaza una asignatura de una franja horaria con otra asignatura.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- El usuario arrastra una asignatura a una franja del horario con otra asignatura
- El sistema asigna la asignatura a esa franja horaria
- El sistema desasigna la asignatura reemplazada de esa franja horaria
- La asignatura aparece en la franja en la cual se ha asignado
- La asignatura reemplazada ya no aparece en la franja en la cual se encontraba.

CU35. Intercambiar las franjas horarias entre dos asignaturas

Resumen: El usuario intercambia la posición de dos asignaturas ya asignadas en el horario.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- El usuario arrastra una asignatura que ya está en el horario a una franja del horario con otra asignatura
- El sistema asigna la asignatura a esa franja horaria
- El sistema asigna la asignatura reemplazada en la franja horaria correspondiente a la asignatura que la ha reemplazado
- La asignatura aparece en la franja en la cual se ha asignado
- La asignatura reemplazada aparece en la franja en la cual se encontraba la asignatura que la ha reemplazado.

CU36. Establecer la franja horaria de una asignatura como “franja de grupo grande”

Resumen: El usuario establece la franja horaria de una asignatura como “franja de grupo grande”.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- Marca el check de una asignatura que se encuentre asignada en el horario
- El sistema marca y muestra la franja en la que se encuentra la asignatura como “franja de grupo grande”

CU37. Desestablecer la franja horaria de una asignatura como “franja de grupo grande”

Resumen: El usuario desmarca la franja horaria de una asignatura como “franja de grupo grande”.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- Desmarca el check de una asignatura que se encuentre asignada en el horario y cuya franja esté marcada como “franja de grupo grande”
- El sistema desmarca la franja en la que se encuentra la asignatura como “franja de grupo grande”.

CU38. Guardar una configuración

Resumen: El usuario guarda todos los datos creados en la base de datos en un archivo para guardar una configuración del sistema.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de gestión de horarios
- Pulsa el botón de exportar la configuración
- Introduce el nombre de la configuración
- Hace click en guardar la configuración
- El sistema guarda los datos de la configuración de la aplicación.

CU40. Cargar una configuración

Resumen: El usuario carga datos creados previamente desde un archivo para restablecer una configuración del sistema.

Actor: Usuario.

Precondición: El usuario debe estar autenticado. Debe existir al menos una configuración creada.

Escenario:

- El usuario accede a la página de gestión de horarios
- Pulsa el botón de importar la configuración
- Escoge que configuración quiere cargar en el sistema
- Hace click en cargar la configuración
- El sistema carga todos los datos de la configuración de la aplicación.

CU41. Consultar los horarios

Resumen: El usuario consulta los horarios que existen en el sistema divididos por grupos.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de información de horarios
- El sistema muestra los horarios pertenecientes a un grupo.

CU42. Exportar horarios a PDF

Resumen: El usuario exporta a un fichero "pdf" la información de los horarios del sistema.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de información de horarios
- Hace click en el botón de exportar a pdf
- El sistema genera un archivo pdf con toda la información de los horarios del sistema.

CU43. Consultar ocupación de aulas por asignaturas.

Resumen: El usuario consulta la ocupación por asignaturas de cada aula por cada franja horaria.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de visión general
- El sistema muestra la ocupación por asignaturas de las clases dividido por horas de un determinado día.

CU44. Asignar un grupo a un aula sin grupo.

Resumen: El usuario asigna un grupo a un aula que no tenga ningún grupo asignado.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página asignación de grupos a aulas
- Arrastra un grupo a un aula sin grupo asignado
- El sistema realiza la asignación
- Muestra el grupo asignado al aula.

CU45. Desasignar un grupo de un aula.

Resumen: El usuario desasigna un grupo de un aula.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página asignación de grupos a aulas
- Hace click en desasignar la asignatura que esté asignada en un grupo
- El sistema realiza la desasignación
- Ya no muestra el grupo asignado al aula.

CU46. Asignar un grupo a un aula con un grupo ya asignado.

Resumen: El usuario asigna un grupo a un aula que tiene un grupo asignado.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página asignación de grupos a aulas
- Arrastra un grupo a un aula con un grupo ya asignado
- El sistema realiza la asignación del grupo arrastrado en la franja
- Desasigna el grupo que estaba asignado en la franja
- Muestra el nuevo grupo asignado al aula.

CU47. Consultar la ocupación por grupos de las aulas

Resumen: El usuario consulta las asignaciones entre grupos y aulas.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página asignación de grupos a aulas
- El sistema muestra una tabla con la ocupación de las aulas por los grupos asignados.

CU48. Asignar una asignatura optativa a un aula.

Resumen: El usuario asigna una asignatura optativa a un aula.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página asignación de asignaturas optativas a
- Escoge un bloque de optativas
- Arrastra una asignatura optativa a un aula.
- El sistema realiza la asignación
- Muestra la asignatura optativa asignada al aula.

CU49. Desasignar una asignatura optativa de un aula.

Resumen: El usuario desasigna un grupo de un aula.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página asignación de grupos a aulas
- Hace click en desasignar la asignatura optativa que esté asignada en un grupo
- El sistema realiza la desasignación
- Ya no muestra la asignatura optativa asignada al aula.

CU50. Consultar bloques de optativas

Resumen: El usuario consulta la información de los bloques de asignaturas optativas.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página información de bloques de optativas
- El sistema muestra los bloques de optativas con su información y sus asignaturas.

CU51. Exportar bloques de optativas a PDF

Resumen: El usuario exporta a un fichero "pdf" la información de los bloques de optativas del sistema.

Actor: Usuario.

Precondición: El usuario debe estar autenticado.

Escenario:

- El usuario accede a la página de información de bloques
- Hace click en el botón de exportar a pdf
- El sistema genera un archivo pdf con toda la información de los bloques de optativas del sistema.

CU52. Desautenticar

Resumen: El usuario se desautenticar en la aplicación.

Actor: Usuario.

Precondición: El usuario debe tener una cuenta registrada en la aplicación y estar autenticado en ella.

Escenario:

- El usuario hace click que en el botón “Logout”
- El sistema desautenticar al usuario de la aplicación
- Redirige a la página de login.

Todos estos casos de uso quedan recogidos en los siguientes diagramas de casos de uso. Estos diagramas están divididos en base a los diferentes módulos o funcionalidades de la aplicación.

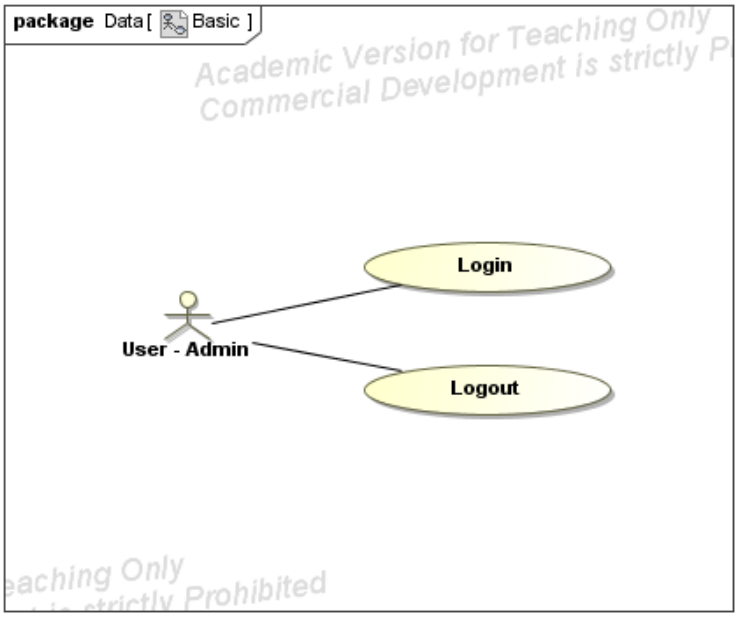

ILUSTRACIÓN 9. CASOS DE USO BÁSICOS

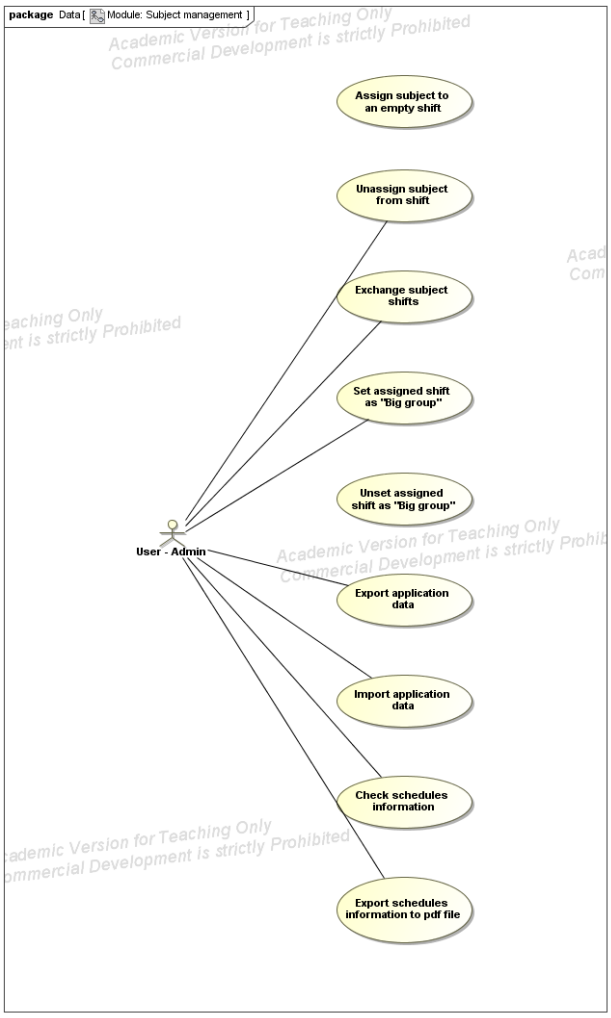

ILUSTRACIÓN 10. CASOS DE USO: MÓDULO DE GESTIÓN DE ASIGNATURAS

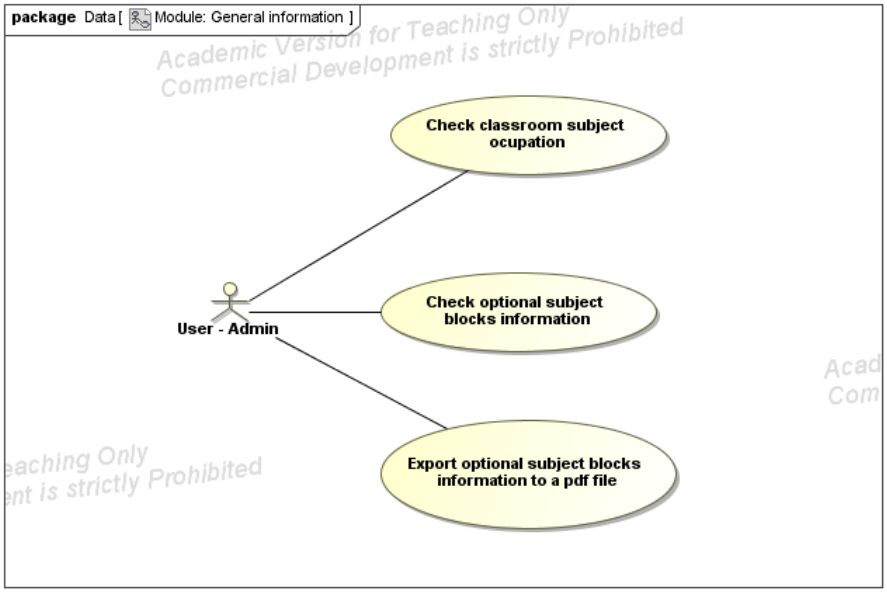

ILUSTRACIÓN 11. CASOS DE USO: MÓDULO DE VISIÓN GENERAL

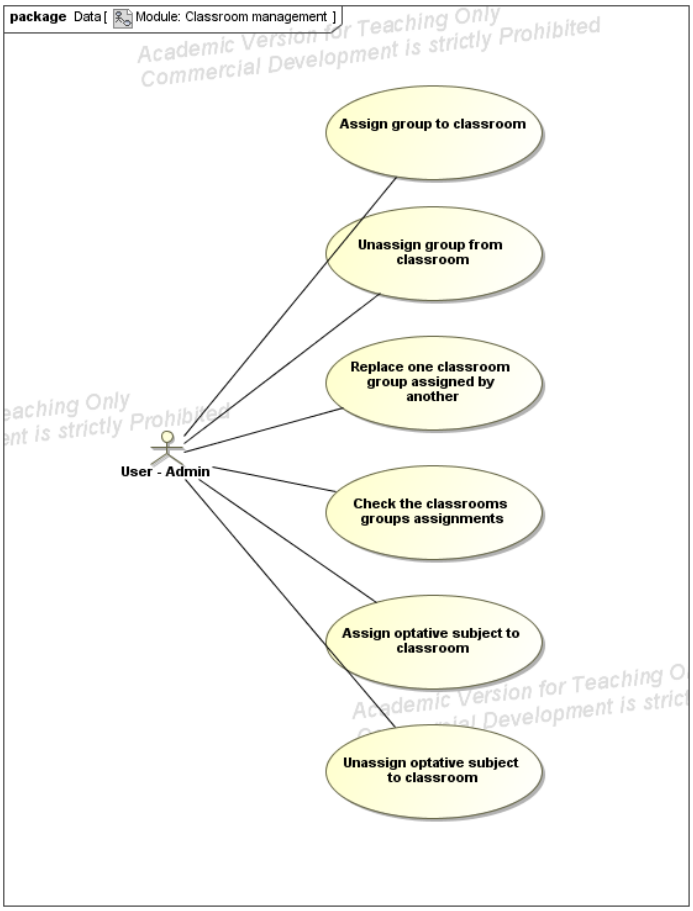

ILUSTRACIÓN 12. CASOS DE USO: MÓDULO DE GESTIÓN DE AULAS

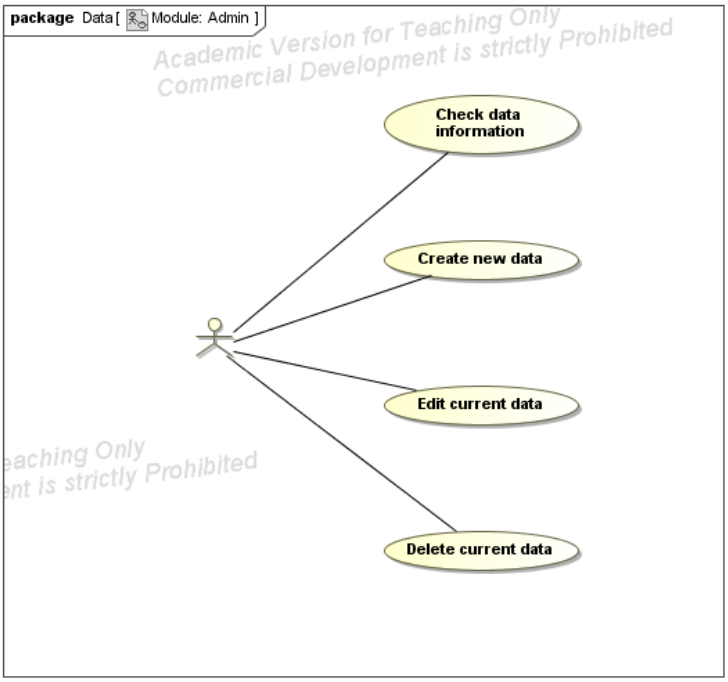

ILUSTRACIÓN 13. CASOS DE USO: MÓDULO DE ADMINISTRACIÓN

Diseño del proyecto

Una vez que ya estaban definidos los requisitos, con sus correspondientes casos de uso y sus diagramas, el siguiente paso lógico era definir el modelo de datos que utilizaríamos posteriormente en el desarrollo, para ello utilizamos un diagrama de clases. Esta no fue una tarea fácil, ya que como ya se ha comentado con anterioridad la definición de algunos conceptos en la especificación. Por ello durante las etapas iniciales del desarrollo este modelo cambió en más de una ocasión. En la siguiente imagen se puede ver cuál fue el primer modelo que se creó.

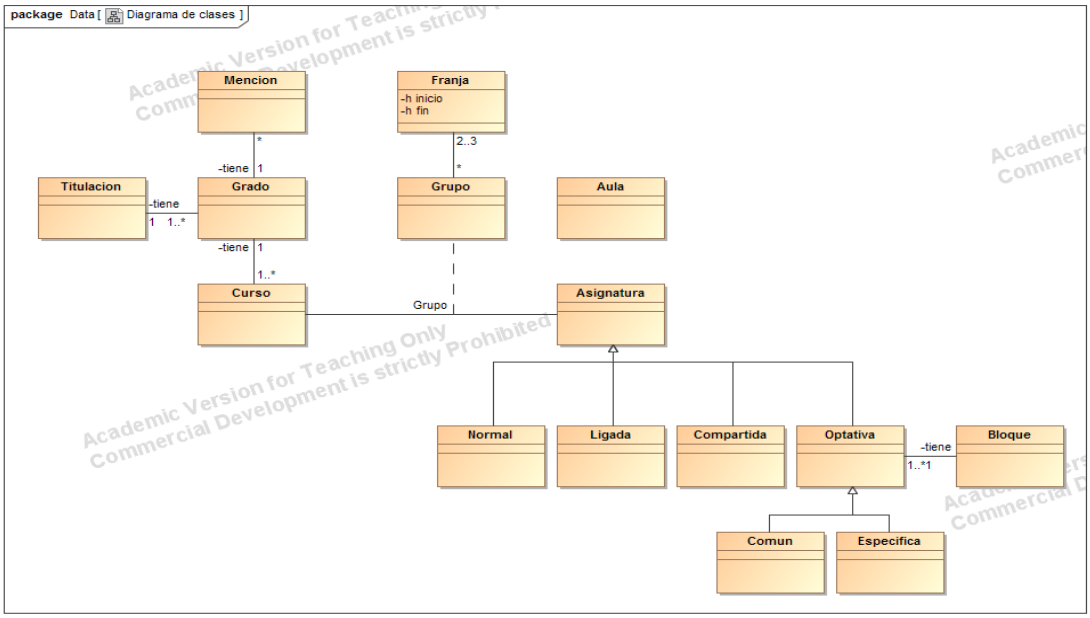

ILUSTRACIÓN 14. PRIMERA VERSIÓN DEL DIAGRAMA DE CLASES

Esta primera versión sufrió infinidad de cambios en las etapas iniciales del proyecto. Incluso en etapas ya más avanzadas de este sufrió pequeñas modificaciones para adaptarse a ciertas situaciones que no habíamos definido bien al inicio o incluso para añadir algunas pequeñas funcionalidades que surgieron a la largo de todo el proceso.

Finalmente tras muchos cambios y reuniones para exponer las diferentes opiniones de los integrantes respecto a la definición del modelo se llegó a la versión definitiva que forma parte del proyecto.

Conceptualmente el modelo de datos puede no ser el más correcto que podría haberse hecho, pero tras considerar otras cuestiones como la funcionalidad final de la aplicación optamos por esta versión.

Por ejemplo, una de las cosas que más quebraderos de cabeza supuso fue la definición de un bloque de optativas. En nuestro diagrama este bloque de optativas está definido como una herencia desde la clase Asignatura, lo cual a nivel conceptual no tiene demasiado sentido. Pero a nivel de la aplicación no hay que olvidar el comportamiento que estos bloques tienen, ya que a la hora de crear los horarios,

aunque un bloque esté formado por un conjunto de asignaturas optativas, este se comporta como una única asignatura a la hora de ser asignado a una franja horaria. Por ello, al tener un comportamiento a las asignaturas compartidas y las ligadas, al igual que ellas los bloques en nuestro modelo son objetos que heredan de asignaturas.

En la siguiente imagen podemos observar esta versión definitiva del modelo de datos en forma de diagrama de clases.

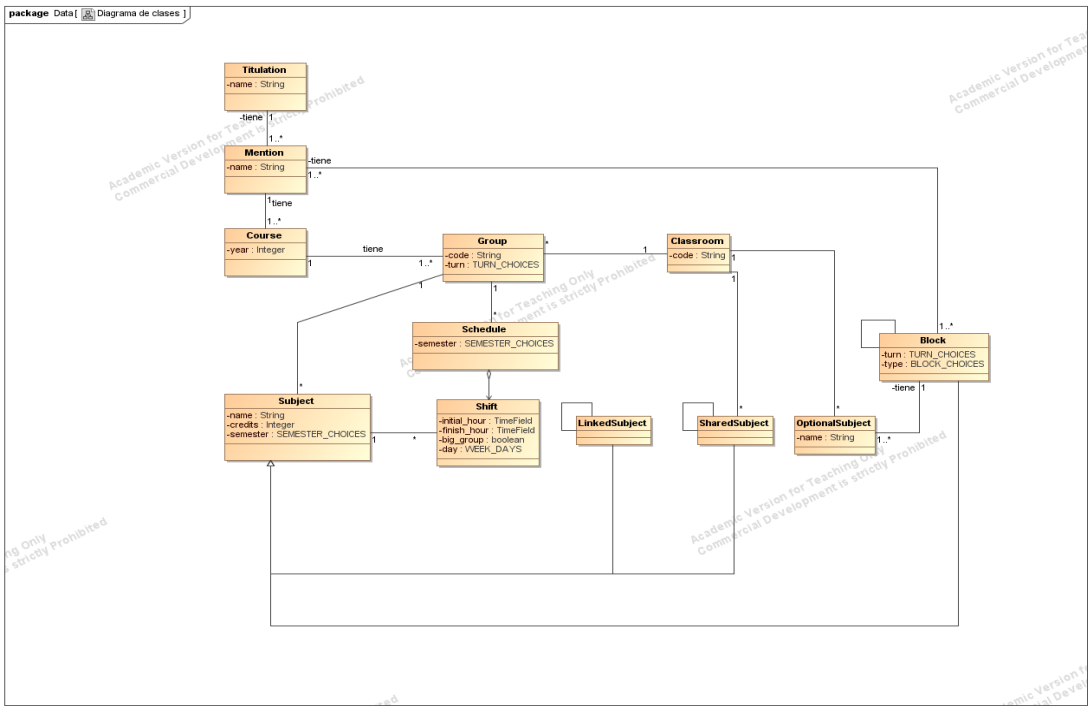

ILUSTRACIÓN 15. ÚLTIMA VERSIÓN DEL DIAGRAMA DE CLASES

Tras esto ya se iba acercando el momento de empezar a desarrollar. Pero antes de eso era necesario hacer una primera división del trabajo. Para ello haciendo una estimación lo más aproximada posible de las tareas que podían surgir del proyecto, dividimos el trabajo en tres posibles partes y las repartimos, de forma que el equipo de desarrollo que se encontraba aquí tuviera dos partes y el equipo que se encontraba fuera tuviera una tercera parte, intentando que no estuviera demasiado cohesionada con el resto para evitar posibles conflictos. Ya que debido a las dificultades existentes entre la comunicación entre ambos equipos debido a la distancia, era preferible no encontrarnos con este tipo de conflictos.

También para facilitar el trabajo de ambos equipos se aprovechó de las herramientas que teníamos a disposición y utilizando el sistema de control de versiones se crearon dos ramas para que cada equipo trabajara en una y evitar conflictos. Ya una vez que el equipo estuvo reunido en las etapas finales se procedió a hacer un “Merge” o mezcla de ambas ramas resultando en el proyecto definitivo.

Una vez definidas todas estas cosas solo faltaban unos pasos más antes de que cada uno pudiera dedicarse a su parte.

Creamos el proyecto básico de Django sobre el que íbamos a trabajar y le añadimos las aplicaciones necesarias que eran necesarias para utilizar las tecnologías que habíamos decidido usar.

A la hora de definir el proyecto, diseñamos un diagrama de componentes en el cual se puede observar la estructura general del proyecto.

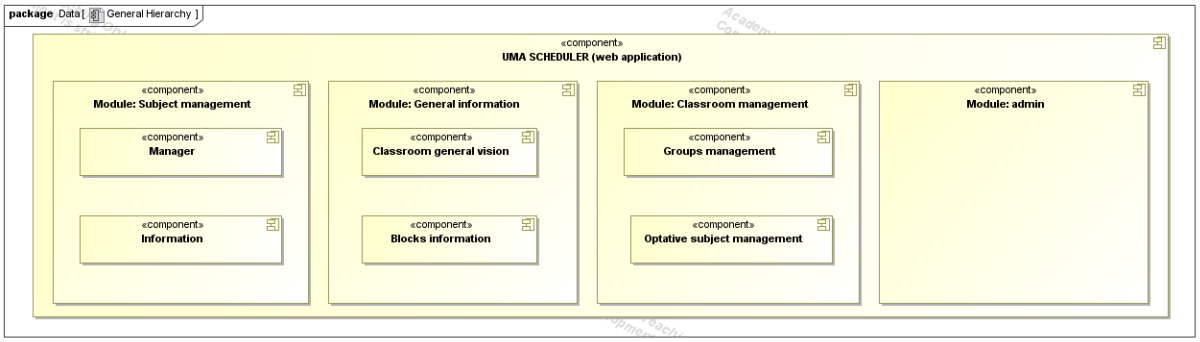

ILUSTRACIÓN 16. DIAGRAMA DE COMPONENTES

De esta forma la estructura de paquetes del proyecto quedó como se muestra en la siguiente ilustración.

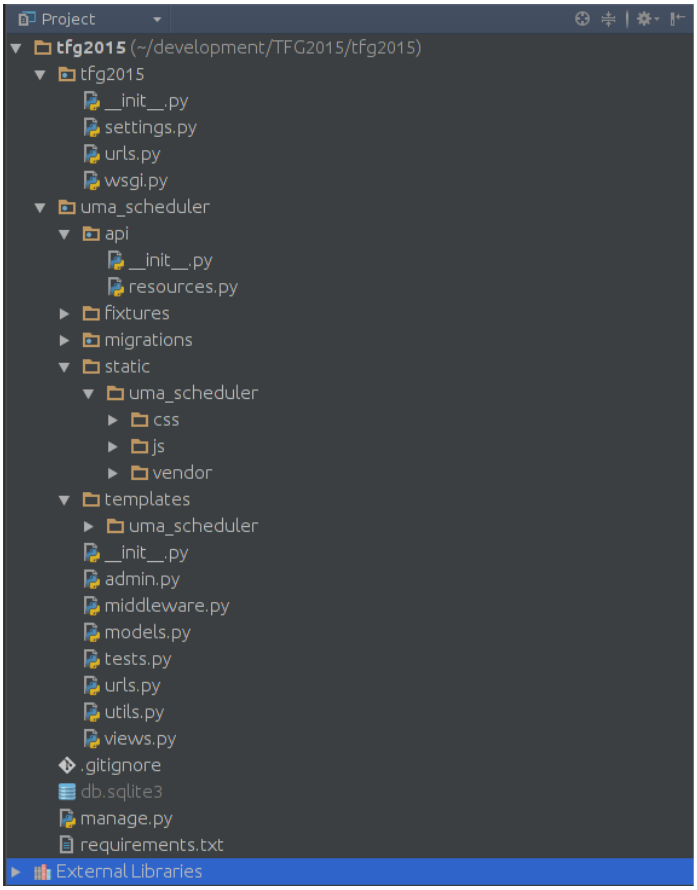

ILUSTRACIÓN 17. ESTRUCTURA DE PAQUETES DE LA APLICACIÓN

Una vez el proyecto estaba creado se inició el repositorio de Git en BitButcket, se subió el proyecto a este y se crearon las ramas sobre las que se iba a trabajar.

Tras toda la selección de las tecnologías y diseño del proyecto quedó con la siguiente arquitectura que se puede observar en la imagen.

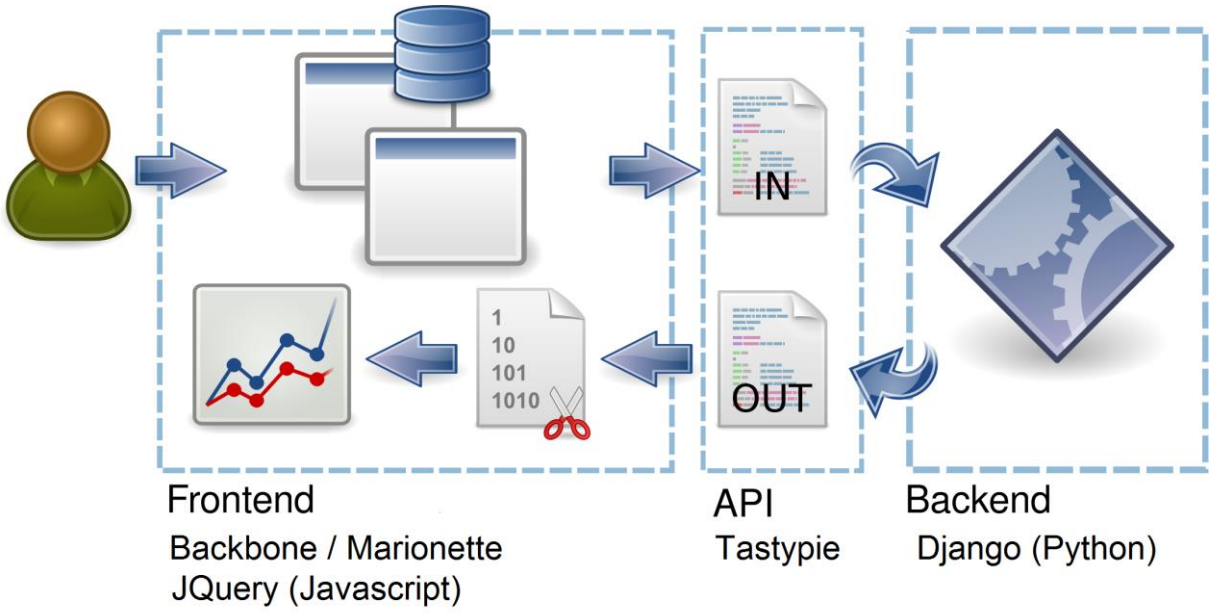

ILUSTRACIÓN 18. DIAGRAMA DE ARQUITECTURA DE LA APLICACIÓN

El siguiente paso en común fue la definición de los recursos de Tastypie para poder acceder a los datos a través de una API RESTful durante todo el desarrollo. Como esto era algo que necesitaríamos los tres integrantes del grupo, se realizó al principio del desarrollo para que todos lo tuviéramos a nuestra disposición. Aunque luego a lo largo del desarrollo cada uno introdujo pequeñas modificaciones para que se adaptara a sus necesidades.

Implementación de la funcionalidad asignada

Una vez acabó toda la fase de diseño llegó el momento de empezar a implementar la aplicación. En esta parte cada uno de los integrantes del grupo se encargó de un conjunto de función

alidades del sistema. Por ello en este apartado se explican aquellas partes que yo implementé completamente o en las cuales colaboré en gran medida.

· Archivo HTML base:

En este archivo colaboré con la importación de algunas de las librerías necesarias para el proyecto. En concreto la importación de la librería JQuery UI Touch Punch, la cual permite utilizar los eventos de toque de JQuery UI en dispositivos móviles y tablets.

· Selectores:

En varias de las vistas de la aplicación es necesario filtrar la información que aparece en función de ciertos datos, como pueden ser la Titulación, la Mención, el Curso, el Grupo, el Semestre, o incluso el día y el turno. Para ello me encargué un sistema de selectores que permitieran decidir qué información es la que se utiliza para filtrar los datos que vemos en cada momento.

La principal complicación que tenía esto es que algunos de los selectores dependían de otros, en el sentido de que por ejemplo las opciones que aparecen en el selector correspondiente a las menciones dependen de la opción que haya seleccionada en el de las titulaciones.

Por ello utilizando las herramientas proporcionadas por Marionette construí un sistema de eventos en el cual cada vez que se seleccionaba un dato en un selector, se disparará un evento el cual hacía que la información de los selectores que dependían de él se actualizara y la información que aparece en la página se actualizara también de forma dinámica. Así la información que uno ve en la página es siempre consistente y no da lugar a confusión.

· Drag&Drop

En este apartado me encargué de construir un prototipo funcional de un sistema Drag&Drop utilizando JQuery UI, que permitía inserta y eliminar objetos de una tabla similar a la que se encuentra en la aplicación, de forma externa, con un HTML y un CSS generado por cuenta propia. Luego más adelante otro compañero se encargó de adaptar este prototipo a nuestra aplicación de forma que se integrase perfectamente con Marionette y con la parte visual de nuestra aplicación.

· Página principal

En esta vista complementé los estilos CSS para darle un visionado más atractivo a la página. Me encargué de añadir los iconos de las diferentes secciones de esta vista, para una identificación más rápida y visual de estas.

· Tablas de horarios y asignaturas:

En esta vista he participado en la construcción del esqueleto HTML en la que posteriormente se inserta el contenido de manera dinámica a través de plantillas HTML encapsuladas en scripts

Estos datos se rellenan de forma dinámica a través de las vistas de Marionette. En Marionette se construye un LayoutView que es el que le da la estructura general a los datos. Este LayoutView está formado por tres regiones: una es la de los selectores que permiten el filtrado de datos, y las otras dos son CompositeViews, uno de ellos es el que pertenece a la tabla de asignaturas y el otro es el de la tabla del horario. Con sus respectivos 'children' que participan en la correcta renderización de los datos.

Fue necesario también la construcción de los modelos Backbone para encapsular los datos necesarios que se obtienen a partir de Tastypie y se usan para rellenar las vistas.

Una vez construido todo esto, a la hora de representar estos datos a través de las plantillas HTML, se usó la librería Underscore para acceder a los atributos que se muestran finalmente en la página.

· Asignación e intercambio de asignaturas normales, ligadas, compartidas y optativas:

Esta es una de las funcionalidades más complejas del proyecto por ello fue uno de los lugares donde se hizo uso de la Programación por Parejas de eXtreme Programming para asegurar la correcta implementación de esta.

Se puede dividir en dos partes, por un lado está el lado del frontend y por otro el backend. En la parte del frontend, como ya se ha indicado anteriormente, me encargué de construir un prototipo del sistema Drag&Drop utilizando funciones de JQuery y JQuery UI, que permitían coger elementos de una tabla y arrastrarlos a otra en la cual se depositaban, o el poder intercambiar las asignaturas dentro de la tabla.

Luego otro de mis compañeros se encargó de integrar este prototipo con las vistas de Marionette, de forma que los ItemView que forman cada tabla, se comportaran igual que los elementos del prototipo pero cambiando algunos detalles de visualización.

Cuando se suelta alguna asignatura dentro de la tabla antes de realizar la asignación definitiva, es necesario realizar una serie de comprobaciones asegurando que esta no incumple ninguna de las restricciones de la funcionalidad.

Por ello, para realizar estas comprobaciones, que se hacen en el backend, debido a que no son comprobaciones simples, se realiza una llamada AJAX a través de las funciones que JQuery proporciona para ello. Esta llamada AJAX invoca a la vista correspondiente de Django que se encarga de gestionar todo el proceso tanto de comprobación de restricciones y de asignación de la asignatura en caso de cumplirse estas.

A la hora de asignar una asignatura hay que tener en cuenta unas restricciones básicas. Estas dos restricciones básicas son el hecho de no poder asignar dos veces una asignatura a un mismo día o, dependiendo del número de créditos de la asignatura, no poder asignar más de un número de veces la asignatura en el horario correspondiente. En nuestro caso 3 veces si es de 6 créditos o 2 veces si es de 4.5 créditos. Esto puede parecer bastante sencillo en primera instancia, pero ya entran en juego otros factores, siendo estos el hecho de que contamos con varios tipos de asignaturas. Estas asignaturas son las ligadas, las compartidas y también se incluyen los bloques de optativas. Y estas además tienen unas características particulares.

La peculiaridad de estas asignaturas es el hecho de que sus horarios van ligados a otras, las cuales varían dependiendo del tipo que estemos tratando. Esto ya influye en el hecho de que al realizar comprobaciones sobre una de estas asignaturas a su vez tenemos que realizar las misma comprobaciones sobre todas aquellas con las que está relacionada, porque los horarios de todas deben cambiar a la vez. Esto puede afectar a otras asignaturas que estén situadas en alguna de aquellas franjas en las que estamos realizando la asignación de estas asignaturas relacionadas con la primera en la que realizamos el cambio. Estas asignaturas que se pueden ver afectadas a su vez pueden tratarse de asignaturas ligadas, compartidas o bloques de optativas con sus respectivas relaciones.

Esto se complica incluso más cuando estamos hablando de intercambio de asignaturas, ya que hay que tener en cuenta por un lado la asignatura que estamos asignando, y la que como consecuencia se asigna donde estaba la primera. Esto eleva incluso más la cantidad de comprobaciones a realizar.

Cuando uno intenta seguir el alcance de estas comprobaciones resulta bastante complicado debido a que es posible llegar a niveles de comprobaciones de bastante profundidad si se dan las circunstancias apropiadas. Por ello para esta parte de la aplicación optamos por realizar unos diagramas de secuencia que nos ayudaran a visualizar más fácilmente el proceso a seguir.

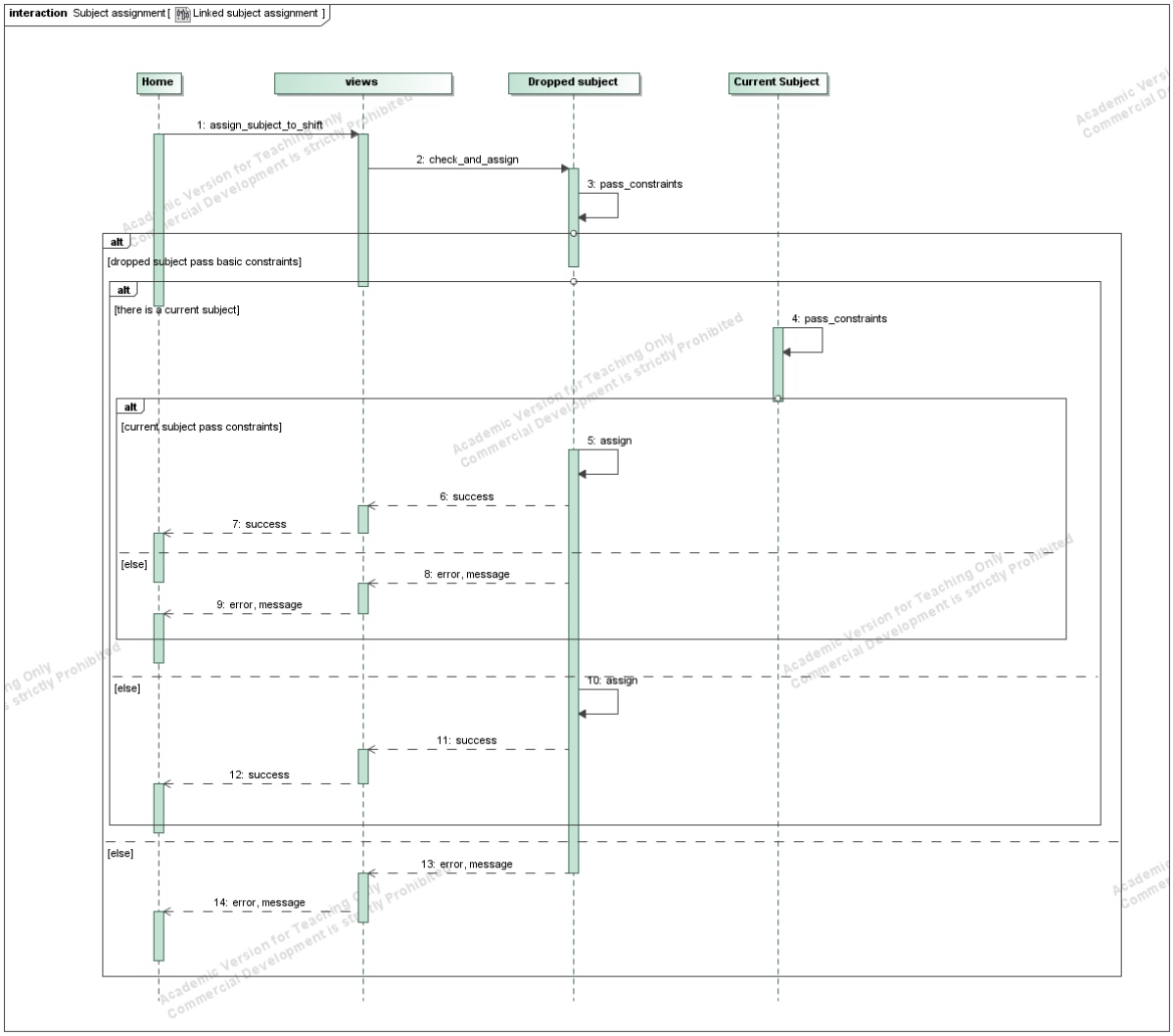

ILUSTRACIÓN 19. ASIGNAR ASIGNATURA A FRANJA HORARIA

En este primer diagrama podemos observar el nivel superior de esta funcionalidad. Se ve cómo se llama al método *check_and_assign*. Este método llama, para la asignatura asignada, al método *pass_constraints*, que comprueba si esa asignatura cumple todas las restricciones, esto incluye a todas con las cuales está relacionada. En caso de cumplirlas se comprueba si se trata de un intercambio, en cuyo caso también se hacer pasar el método *pass_constraints* a la asignatura que va a ser intercambiada. En caso de las comprobaciones que se han realizado den un resultado satisfactorio se hacen las asignaciones correspondientes.

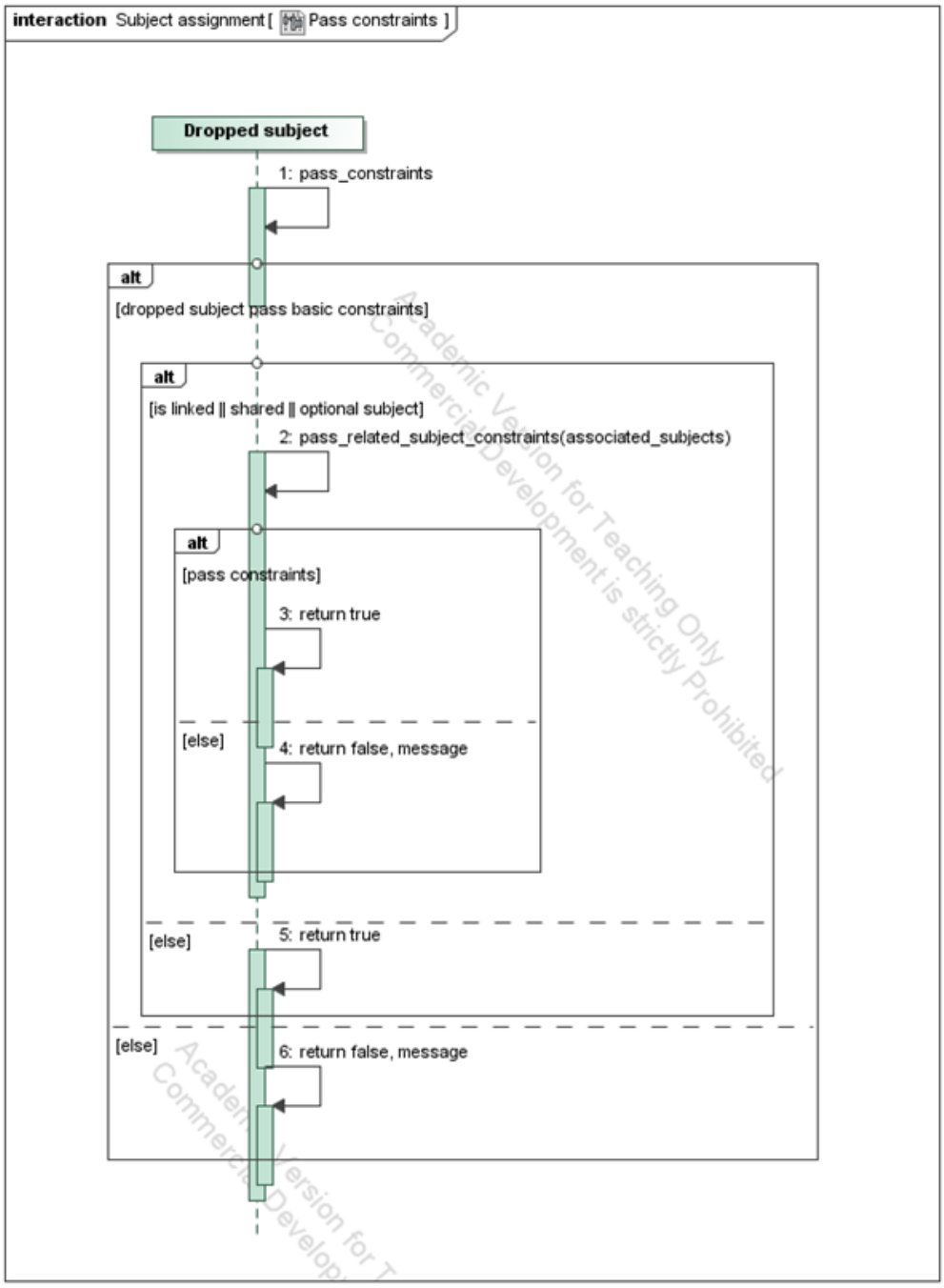

ILUSTRACIÓN 20. MÉTODO PASS_CONSTRAINTS

Si vamos un nivel más allá, es también necesario analizar en más profundidad el método pass_constraints que podemos ver en la imagen anterior.

Este método primero comprueba si la asignatura que lo ha llamado cumple las restricciones básicas, de las cuales se ha hablado con anterioridad.

Después se encarga de comprobar qué tipo de asignatura es, ya que puede tratarse tanto de una asignatura ligada, una asignatura compartida o de un bloque de optativas. También puede darse el caso de que se trate de una asignatura normal, en cuyo caso simplemente se devuelve el resultado de las restricciones básicas.

Luego, ya sabiendo el tipo de asignatura, si no se trata de una normal, se llama al método correspondiente a las comprobaciones particulares de cada tipo de asignatura, y ya dependiendo del resultado de este método se realizará la asignación o no.

La siguiente imagen es el diagrama de los que serían estos métodos.

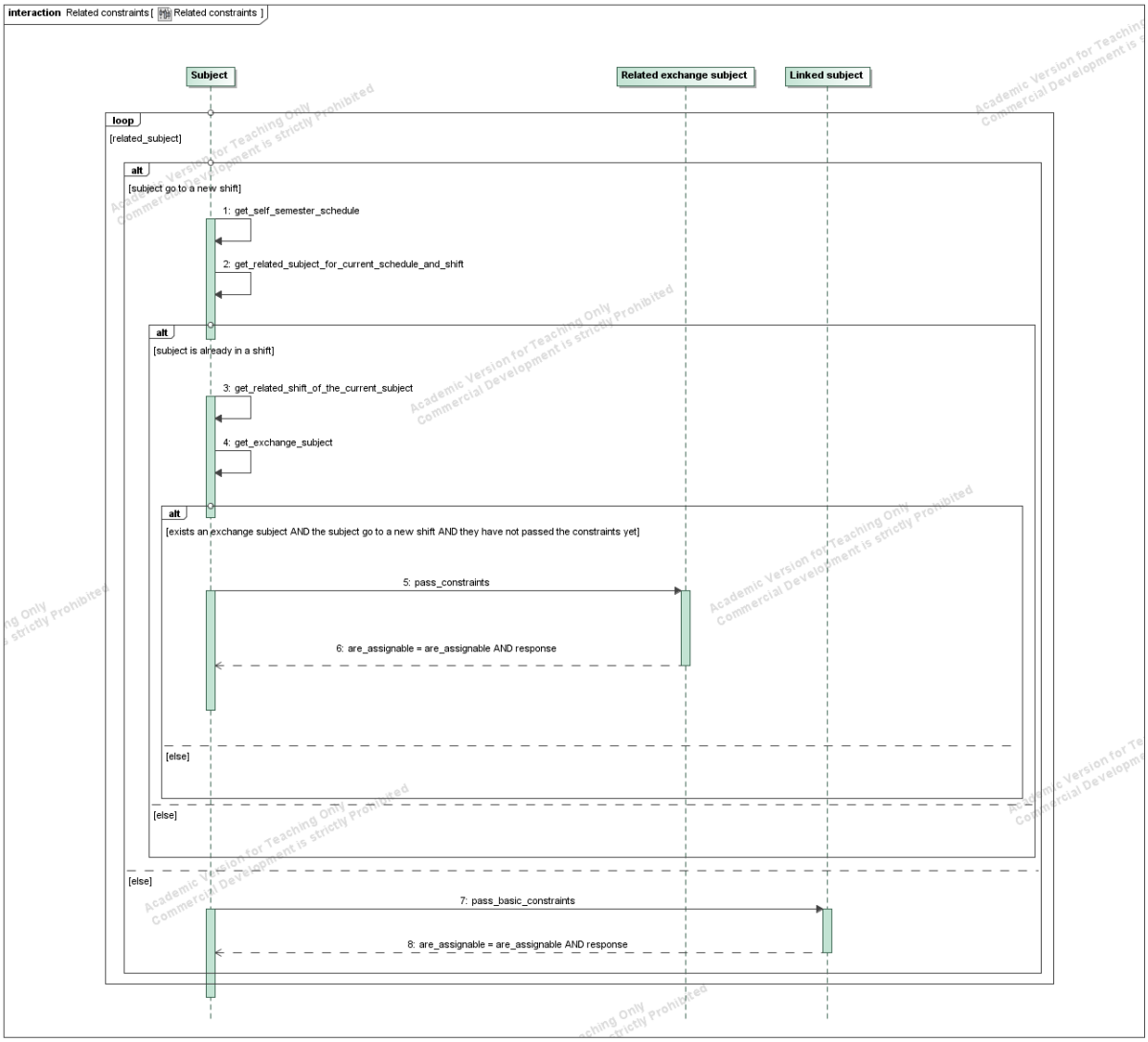

ILUSTRACIÓN 21. CONSTRAINTS PARTICULARES PARA CADA TIPO DE ASIGNATURA

Aunque en realidad contamos con 3 métodos distintos, la realidad es que la estructura general es muy similar, lo único distinto son pequeños detalles a la hora de acceder a los datos que necesitamos.

La clave de estos métodos está en la llamada recursiva que se realiza para cada una de las asignaturas relacionadas solo en aquellos casos en los que es necesario. Es muy importante el hecho de que se tiene un control sobre aquellas asignaturas que ya han pasado las restricciones, porque debido al factor de recursividad de estas funciones existía la posibilidad de entrar en bucles infinitos.

Al final de todas estas comprobaciones si el resultado es positivo se realizará la asignación.

· Desasignar asignatura de una franja

En esta parte participé sobre todo en la elaboración del backend que, aunque en un principio pueda dar una sensación similar en complejidad al anterior, en la práctica resulta mucho más sencillo.

En realidad al no ser necesario realizar ningunas comprobaciones básicas, lo único necesario es obtener aquellas asignaturas que están relacionadas con aquella de la que queremos eliminar la asignación y desasignarlas todas.

· Página de información de los horarios

Esta página contiene información relativa a la de los horarios, por lo que se reutiliza algo de código respecto a esta.

En esta página construí el esqueleto HTML con sus respectivas plantillas para rellenarlas a partir de las vistas de Marionette. Al final se construyen dos tablas de horarios para cada semestre prácticamente idénticas a la tabla de los horarios del módulo de creación de horarios.

Al igual que en aquel módulo tenemos los selectores para filtrar la información, así que tuve que construir la estructura para que cuando cambiaran estos selectores se actualizara la información de la página.

· Página de visión general

En esencia también es una página de información por lo que no presenta una funcionalidad compleja, solo que a la hora de recuperar los datos puede ser un poco menos sencillo que en otros caso.

En esta vista podemos ver las asignaturas que ocupan cada clase en cada hora existente. Pero aunque es trivial adquirir las asignaturas a partir de los horarios y los grupos a los que pertenecen, aquí entran en juega los bloques de optativa. Estos bloques están formados por varias asignaturas optativas y cada una de ellas de imparte en un aula diferente. Por ello a la hora de recuperar estos datos es necesario recibir unas estructuras más complejas formadas por asignaturas y asignaturas optativas, que en nuestro modelo de datos no son dos tipos de entidades similares.

Aparte de esto, al igual que en el resto de páginas fue necesario construir el esqueleto HTML, las plantillas encapsuladas en scripts, los modelos Backbone y las vistas

Marionette, para hacer que la información se pueda actualizar dinámicamente a través de los selectores.

·Página de gestión de aulas y grupos

En esta página también podemos dividir el trabajo realizado en dos partes: el frontend y el backend.

En el frontend por una parte añadí ciertos detalles como un color extra para identificar aquellas clases que solo se encontraban ocupadas por asignaturas optativas. Ya que para lo que queríamos realizar es distinto el caso en el que la clase está ocupada por un grupo o por asignaturas optativas. También me encargué de que se actualizara bien la información cada vez que se realizaba una asignación o se quitaba una asignatura de una clase.

Por otra parte también añadí una tabla que permitía tener una referencia más clara de la ocupación de las aulas sin tener que depender de la información proveída por el códigos de colores o los Toast que proporcionan información sobre qué está ocupando cada aula.

En el backend me encargué de realizar las comprobaciones pertinentes a la hora de asignar un grupo a una clase. Para ello en primer lugar es necesario comprobar si el grupo que se quiere asignar cabe en la clase, ya que cada grupo tiene un tamaño y cada clase una capacidad máxima. Tras esto hay que comprobar si existen asignaturas optativas asignadas a esta misma clase y en caso de que existan, comprobar que no producen conflictos con el grupo en cuestión. Es decir solo en el caso de que la asignatura optativa pertenezca a un bloque que esté en ese grupo, se puede realizar la asignación.

En este caso, como la complejidad era menor, no hubo necesidad de realizar ningún diagrama de secuencia para apoyar al desarrollo.

Conclusiones

Objetivos cumplidos

Haciendo una evaluación general del proyecto una vez acabado considero que se han cumplido de la gran mayoría de objetivos planteados inicialmente.

Tras un arduo trabajo, se ha conseguido desarrollar una aplicación web que permite manejar los horarios de un centro y también la gestión de espacios del mismo a la hora de planificar el curso a principios de cada año lectivo.

Es verdad que por el camino se han podido ir quedando ciertos detalles planteados inicialmente. Por ejemplo, el hecho de hacer que la aplicación fuera totalmente 'responsive'. Aunque en un principio era esta la idea, a lo largo del desarrollo nos dimos cuenta de que no tenía sentido utilizar la aplicación en dispositivos de pequeño tamaño como puede ser un móvil, porque debido a la forma en la que está estructurada la funcionalidad no resultaría práctica ni fácil de usar. Por ello la aplicación no está preparada para un correcto visualizado en estos dispositivos. Si bien es cierto que la aplicación sí está preparada para que funcionalmente no presente ningún tipo de problemas en dispositivos móviles, por lo que la aplicación se puede utilizar perfectamente en una tablet o dispositivos de mayor tamaño a un móvil.

Por otro lado también se puede hablar sobre la metodología de trabajo que se ha usado en general. Este proyecto era de una envergadura mayor a la que una sola persona podría afrontar dentro de los límites de este trabajo, por ello nos embarcamos en este proyecto como un grupo de tres personas porque se nos dio la oportunidad para ello.

Esto ha tenido sus puntos positivos y sus puntos negativos. A la hora de tomar decisiones importantes respecto del proyecto el tener una diversidad de opiniones y un intercambio de ideas entre los miembros del grupo enriquece las decisiones. La calidad final del proyecto se ve aumentada. El trabajar en grupo también permite que uno pueda apoyarse en los demás en aquellos momentos en los que se encuentre con problemas durante el desarrollo, ya que los conocimientos que posee cada uno de los integrantes se complementan con los conocimientos de los demás.

Pero no todo son beneficios, en particular en las fases iniciales del proyecto la necesidad de coordinar a tres personas para realizar un trabajo común ralentiza un montón el arranque del trabajo. Ya que es más difícil organizarse entre tres personas con situaciones distintas a organizarme uno mismo sin depender de nadie. En este caso concreto, uno de los integrantes del grupo estuvo durante todo el curso laboral cursando una beca Erasmus en el extranjero, lo que dificultó todavía más si cabe esta tarea de organización.

Al formar parte de un grupo otro problema que se ha detectado en las fases iniciales del proyecto es que la sensación de responsabilidad está más dispersa, de forma que cuesta más implicarse durante esta etapa. Aunque este último problema se va resolviendo a medida que cada uno empieza a tener su trabajo de forma más individual y diferenciado.

La conclusión a la que yo llego es que a pesar de que tiene una serie de puntos negativos, ha sido una experiencia positiva, ya que además de tener sus pros, hoy en día es necesario ser capaz de trabajar y organizarse en grupos, porque es lo que uno luego va a encontrarse en la realidad del día a día.

En definitiva, si tuviera que valorar de manera global lo que ha sido todo el desarrollo del proyecto, ciertamente le daría una evaluación positiva. Es cierto que ha habido momentos mejores y otros peores, pero eso es parte de lo que significa la profesión que hemos elegido para nuestras vidas. Por ello el haber podido experimentar una situación tan real durante los meses que ha durado el trabajo es una experiencia irremplazable que cada uno de nosotros nos llevamos y que seguro nos servirá de ahora en adelante.

Posibles ampliaciones

La aplicación tal y como se encuentra ahora mismo tiene una gran variedad de posibilidades de ampliación. Por una parte, es posible mejorar los módulos ya existentes. En el primer módulo de creación de horarios, debido a las relaciones que existen entre los distintos tipos de acciones y las dependencias que esto crea entre los horarios de estas asignaturas, cuando se realiza una asignación o un intercambio, a veces es necesario hacer comprobaciones a muchos niveles de profundidad para evitar inconsistencias en las relaciones. Tal y como está implementado ahora, cuando detecta que se va a producir una inconsistencia, no se realiza la asignación o intercambio y se muestra un error poco descriptivo sobre cuál ha sido ese problema. Una posible ampliación sería modificar este código para que devolviera códigos de error descriptivos que permitieran localizar dónde se está produciendo la inconsistencia que no nos permite realizar la acción que queremos.

Siguiendo en la misma línea de los códigos de error, en el segundo módulo se encuentra una situación similar, en la que se podría refinar los códigos de error para poder identificar más fácilmente el motivo que nos impide la asignación.

Continuando por otra línea totalmente diferente, es posible mejorar el funcionamiento de la aplicación. Pensando en un caso práctico, aunque una asignatura tenga asignada varios huecos horarios, luego es posible que alguno de ellos no se use nunca o que nunca se utilice el aula que tiene asignada ese grupo para recibir las clases. Por lo tanto sería una buena idea tener la posibilidad de contemplar esos casos para que,

al tener el conocimiento de que esas aulas están vacías a esas horas, se pueda asignar otras asignaturas o actividades.

También se podrían añadir módulos totalmente nuevos a la aplicación. Un ejemplo sería un módulo para gestionar el uso de los laboratorios para clases prácticas, de forma que podría asignarse varios de los bloques horarios de una asignatura para impartirse en un lugar distinto al que tiene asignado el grupo en cuestión. Y conectando con esto último, también la posibilidad de reservar laboratorios para ocasiones especiales asegurándose de la disponibilidad de este en el momento en el que se quiera utilizar.

En definitiva, se pueden pensar una variedad de opciones de ampliación para la aplicación si centramos nuestra línea de pensamiento en los usos que esta puede tener en un caso real del centro.

Referencias

https://www.python.org/

https://www.djangoproject.com/

http://underscorejs.org/

http://backbonejs.org/

http://marionettejs.com/

https://django-tastypie.readthedocs.org/en/latest/

http://blog.mathandpencil.com/using-django-tastypie-to-create-RESTful-APIs/

http://materializecss.com/

https://jquery.com/

https://jqueryui.com/

http://www.nomagic.com/products/magicdraw.html/

https://balsamiq.com/products/mockups/