# INTRODUCCIÓN:

Mi proyecto trata en añadir una mejora a la plataforma del centro.

Consiste en la creación de una aplicación para que los profesores puedan ver todas las aulas que se pueden reservar, realizar reservas, ver cuándo y a qué hora hay reservas hechas por otros profesores, etc.

## Mi idea inicial:

La primera opción era crear un mapa donde se vieran las salas para reservar, pero después al pensarlo quizá era poco intuitivo, y difícil de ampliar si se hacen cambios en el plano. Así que cambie la idea y decidí mostrar todas las aulas que son reservables y mostrarlas de la mejor forma posible para que fuera fácil su acceso y claro. Una vez dentro del aula que saliera un calendario y las reservas de los diferentes días con las horas etc.

## Material Utilizado:

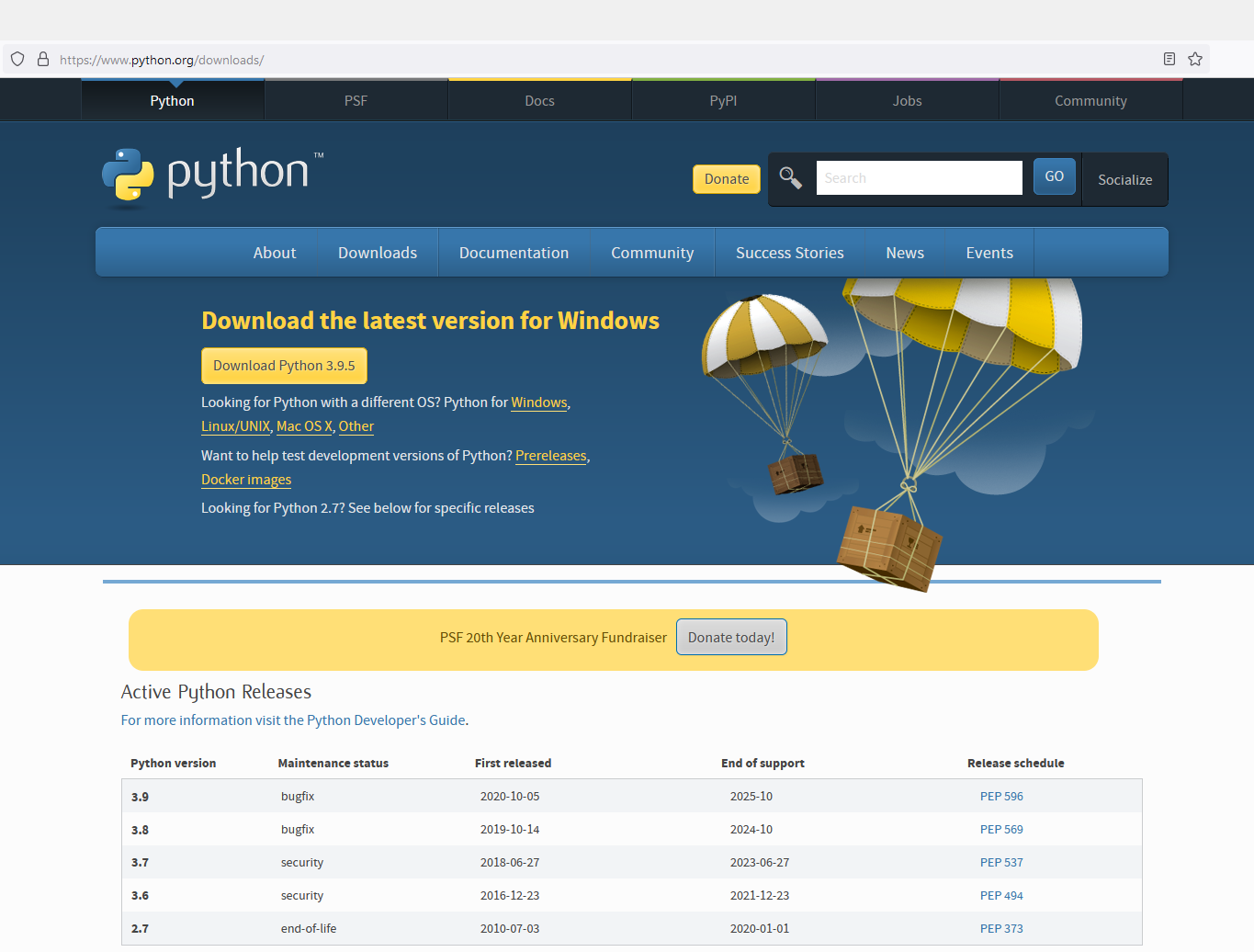
* Framework de servidor **Django** que utilizada como lenguaje **Python**.
* Para la parte de Cliente he utilizado **JavaScript, Ajax, JQuery**, y estuve leyendo en varios sitios que **Django** se puede considerar Framework de Front end.
* En diseño he utilizado librerías para iconos como <https://lordicon.com/icons> y **HTML5** y **CSS**
* **Dockers** para despliegue.

# DESCRIPCIÓN:

El Resultado final de la aplicación consiste en ver todas las clases, hacer una reserva, ver las reservas de un día, ver todas mis reservas en general, poder borrar una reserva hecha, un apartado para administrador, el administrador puede ver todas las reservas de todos los profesores, y se puede filtrar por día, y evento.

# INSTALACION:

Para la instalación necesitaremos tener en nuestra maquina Python 3.9.5 para ellos nos iremos a la página oficial de Python y descargaremos nuestra versión.



Si la versión que apareciera fuera distinta iremos más abajo donde nos saldrán todas las versiones y podremos descargar la 3.9.5

Una vez la hayamos descargado lo abrimos y lo instalamos.

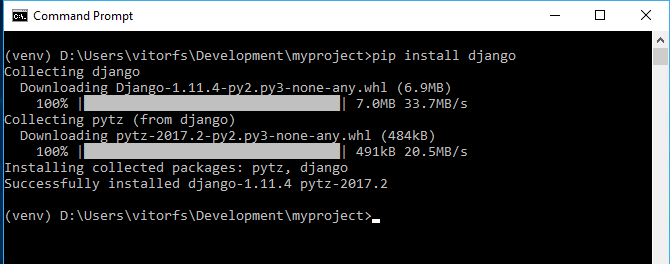
Para ver si lo hemos instalado correctamente abrimos nuestra cmd y escribimos el siguiente comando

**Python -- version**

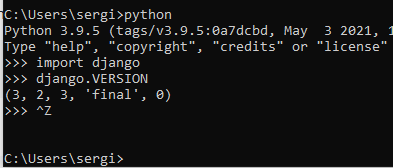


Una vez nos hemos asegurado que Python lo tenemos instalado vamos a instalar Django, para instalar Django es necesario tener antes instalado pip. Esta herramienta en la versión de Python 3.9 ya viene instalada así que no tendremos que preocuparnos, si utilizamos otra versión de Python tendremos que comprobar si tenemos el pip instalado.

Después de asegurarnos que lo tenemos vamos a instalar Django, usamos el comando en nuestra cmd: **py -m pip install Django**

****

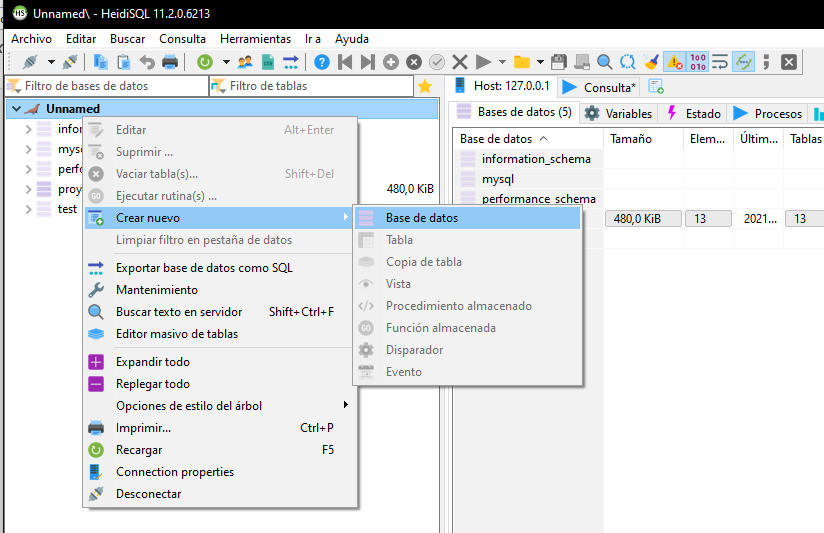
Una vez haya finalizado comprobaremos en nuestra cmd que lo hemos instalado

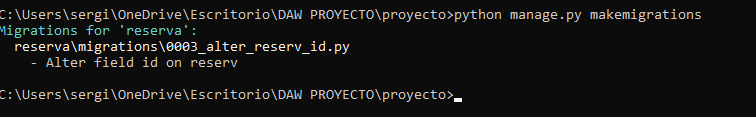


Ahora instalaremos nuestro servicio de base de datos (MySQL) con el comando **pip install mysqlclient**

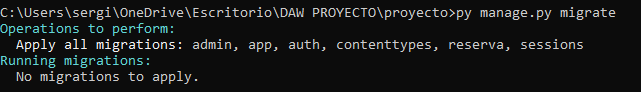
Una vez tengamos todo instalado, tendremos que configurar un archivo llamado settings en el cual le vamos a decir como conectar a nuestra base de datos.

Antes de poder conectar con nuestra base de datos debemos tenerla creada. Debemos de asegurarnos que los puertos que vamos a utilizar estén disponibles ya que si están en uso no nos dejara conectarnos por esos puertos.

En mi caso para control de base de datos he utilizado HeidiSQL, para crear la base de datos tendremos que hacer click derecho crear nuevo, base de datos.

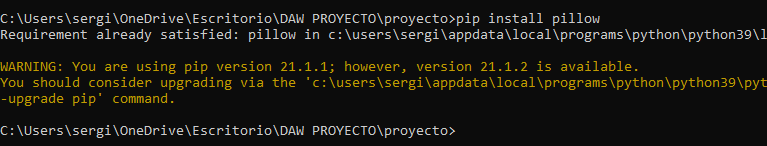
Para que la base de datos coja la información de nuestro Django tendremos que hacer migraciones con **Python manage.py makemigrations**

Y una vez esto tendremos que hacer **Python manage.py migrate** y asi se nos subirá la información a nuestra db



Por último, instalaremos Pillow con **pip install pillow**

Pillow es una herramienta para poder subir y trabajar con imágenes con Django.



Una vez lo tengamos todo crearemos nuestro proyecto Django con

**django-admin.exe startproject nombre\_proyecto**

Una vez hecho podremos empezar a trabajar con nuestro trabajo.

# GUÍA DE ESTILOS Y PROTOTIPADO

## Identidad Visual:

Al comenzar el proyecto tuve claro que esta aplicación se iba a usar sobre todo en los ordenadores del centro, así que enfoqué el diseño para desktop en vez de mobile first.

## Colores seleccionados:

En cuanto al color del diseño estuve pensado si escoger el azul de la Moodle del centro o en usar unos colores rojos ya que son realmente los colores de nuestro centro, al final me decanté por usar unos colores rojizos y corporativos, como podemos ver en el logo son los colores que utilizan.



La tonalidad de colores que he escogido es la misma ya que el centro tiene su página web con estos mismos colores para seguir en la misma línea.

#BA1C1A #FFFFFF

## Tipografía:

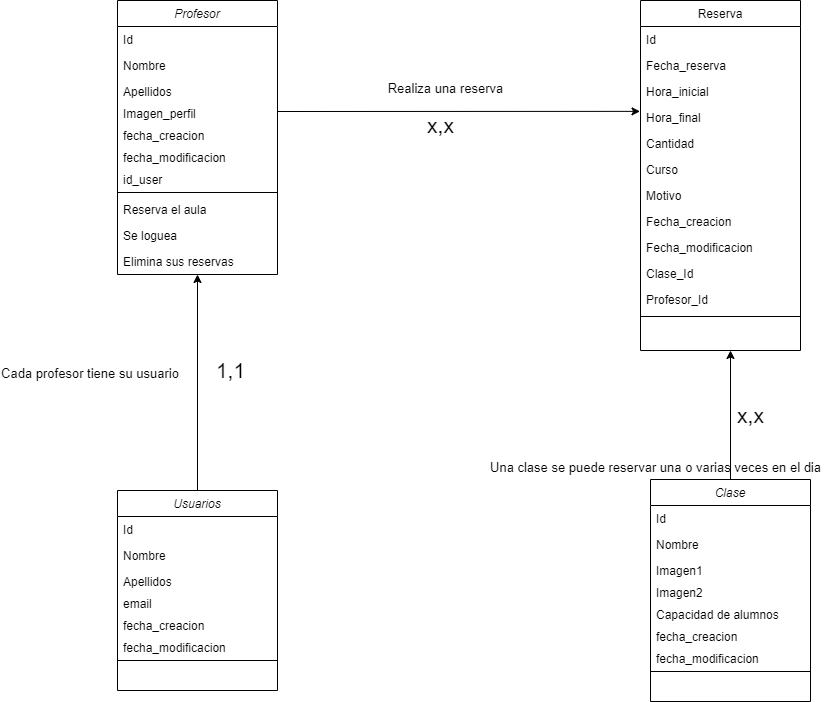
En cuanto a la tipografía al principio decidí utilizar la misma de la plataforma, pero no me convencía ya que parecía que era una web antigua y la cambie por una ROBOTO, así le doy un toque más actual a la web, aparte de esto, roboto es una letra clara, fácil de leer y de entender y para una aplicación que se va a utilizar en un Instituto creo que esto es de las cosas más importantes.

ROBOTO:

ABCČĆDĐEFGHIJKLMNOPQRSŠTUVWXYZŽabcčćdđefghijklmnopqrsštuvwxyzžАБВГҐДЂЕЁЄЖЗЅИІЇЙЈКЛЉМНЊОПРСТЋУЎФХЦЧЏШЩЪЫЬЭЮЯабвгґдђеёєжзѕиіїйјклљмнњопрстћуўфхцчџшщъыьэюяΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩαβγδεζηθικλμνξοπρστυφχψωάΆέΈέΉίϊΐΊόΌύΰϋΎΫΏĂÂÊÔƠƯăâêôơư1234567890‘?’“!” (%) [#]{@}/&\<-+÷×=>®©$€£¥¢:;,.\*

# DISEÑO

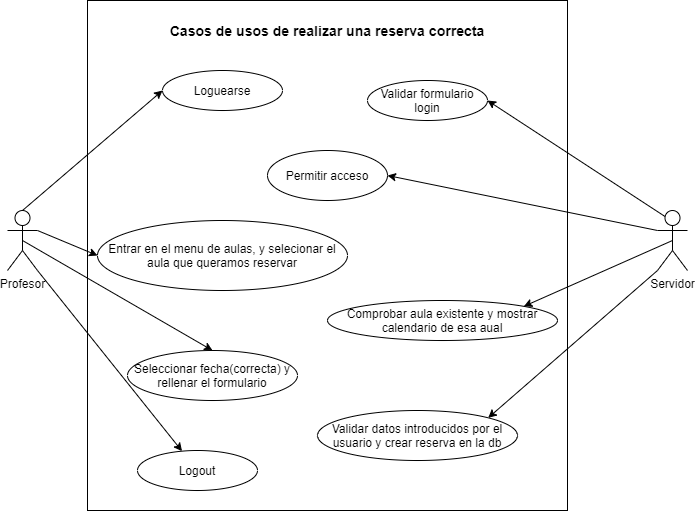
## Diseño UML



Tenemos varias clases o entidades

* **Usuarios**: Los usuarios son los profesores que se loguean y cada usuario va relacionado con un profesor así que tienen una relación de uno a uno.
* **Profesores**: Son los que realmente contienen todos los datos de los profesores, nombre, apellidos, etc… Los profesores son los que pueden hacer una reserva de una clase. Un profesor puede reservar una o varias clases y una clase puede ser reservada por un profesor o por varios lo que significa que entre profesor y clases existe una relación de muchos a muchos. Al haber una relación de muchos a muchos tengo que crear una tabla que se llama reserva para controlar todas las aulas que se reservan y que profesores la reservan.
* **Clase**: Las clases como hemos especificado anteriormente tienen una relación de muchos a muchos y su tabla guarda el nombre, el id, la cantidad de alumnos etc.
* **Reserva**: Esta tabla generada al haber una relación muchos a muchos guardan las fechas de las reservas, las horas, los id de las diferentes tablas etc.

## Casos de uso



Un caso de uso de una reserva correcta entre el usuario y el servidor.

* Primero el usuario entra en la aplicación y lo primero que se encuentra es el login.
* El servidor comprueba que los datos son correctos, si lo son pasa a la siguiente página o si no muestra un error
* El usuario navega por la página hasta que llega a la zona donde se encuentran todas las aulas y selecciona una.
* El servidor comprueba si el aula existe y si no lanza un 404, si la página es correcta manda al usuario a la web de esa aula en concreto y puede realizar una reserva.
* El usuario tendrá que elegir un día y rellenar un formulario con datos correctos para que la petición se tramita exitosamente.
* Por último, el servidor comprueba ese formulario y si es correcto crear una reserva en la base de datos.

# DESARROLLO

Debido a un cambio de planes y falta de tiempo he tenido que trabaja con una secuencia de desarrollo rápida. Antes de empezar a instalar nada tenía que tener claro lo que quería, las tablas de bases de datos que iba a tener, las relaciones que podía existir, así que creé el modelo UML para tenerlo claro. Una vez hecho esto, estuve barajando dos posibles Frameworks para trabajar en la parte de servidor, Flask y Django.

Los dos están escritos en lenguaje Python, lenguaje que habíamos dado el año anterior durante todo el curso y por eso me sentía bastante cómodo con este lenguaje, aparte que es flexible y cómodo a la hora de trabajar.

Estuve buscando información sobre cuál de los dos utilizar:

## Django:

Django es lo que se conoce como un framework “full stack” para Python con el cual se pueden abordar todo tipo de proyectos en este lenguaje como el desarrollo web escalable y de alta calidad.

Se trata del entorno de desarrollo de Python más popular, ya que permite un desarrollo ágil al encargarse de manejar los controladores.

Las ventajas que tiene Django son:

* Cuenta con un sistema de autentificación de usuarios.
* Ofrece un gran rendimiento y flexibilidad, pudiendo escalar proyectos de forma sencilla.
* Trabajar bajo un patrón MVC (Modelo Vista Controlador), lo que permite un desarrollo ágil y reutilizable.
* Incorpora una amplia variedad de paquetes de librerías (más de 4000).
* Dispone de una inmensa comunidad de usuarios en internet, su documentación está actualizada y es muy amplia, aparte en la misma documentación tiene un curso para empezar a trabajar con django.
* Cuenta con panel de administración para bases de datos.

## Flask:

Flask es un framework perfecto para personas que quieran hacer una pequeña aplicación y estén empezando ya que es muy flexible. Con Flask se utilizan las líneas de código necesarias para realizar cualquier acción por lo que es mucho más sencillo comprender la estructura de cualquier aplicación o script, y saber qué es lo que realiza.

Ventajas de Flask:

* Se adapta a cada proyecto instalando extensiones específicas para el mismo.
* Incluye servidor web propio para pruebas.
* El diseño minimalista de su estructura le permite ser rápido y con un gran desempeño.
* Cuenta con documentación extensa para el desarrollo de aplicaciones.
* Es muy sencillo de utilizar, por lo que es el indicado para empezar a programar con Python.
* Se integra con otras herramientas para incrementar sus funciones, como Jinja2 (motor de plantillas web) o SQLAlchemy (kit de herramientas SQL de código abierto).

Después de haber visto todas sus ventajas, ver como funcionaban, como se trabajaban con ellas, decidí utilizar django, no solo porque me parecía más cómoda, si no por la gran cantidad de información que había en internet ya que era más conocida que flask, aparte Flask estaba enfocada a hacer pequeñas aplicaciones y a trabajar con API y django para hacer aplicaciones web a un nivel más profesional. Por otro lado, ambos tienen desventajas y las de flask me hicieron elegir definitivamente django. Desventajas de flask como no contener librerías integradas, Se trata de un entorno que genera dificultades a la hora de realizar migraciones o pruebas unitarias. También es una desventaja tener que recurrir a un mapeo de objetos relacionales (ORM) externo para conectar con bases de datos.

Al empezar con Django y ver su forma estructurada de trabajar no me resulto difícil.

## Ejemplo de un recorrido de una llamada MVT.

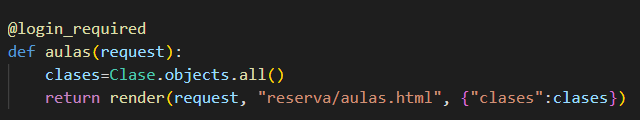
Primero vamos al Template en el cual voy a tener un enlace a la siguiente página. En este ejemplo voy a situarme en la página ramas, que es la página justo después de loguearnos en la aplicación, luego vamos a ir a la parte donde muestro todas las aulas. En el archivo HTML tengo que tener un enlace el cual tenga la ruta que esta está controlada por el archivo url:



En la línea de código de arriba indico en la etiqueta href a la ruta que quiero ir es a aulas, en la imagen de abajo es la configuración para llamar a la vista de aulas que tengo en el archivo urls.



La ruta está definida y llama a la vista de aulas.



En este caso la vista no tiene mucha dificultad, tiene un decorador que indica que el usuario tiene que estar logueado para poder acceder, luego guardo todas las clases para poder trabajar con la información dentro del template, por último, en el “return” digo que me lleve a la siguiente página con la lista de clases para usarlo posteriormente.

Si en la página tuviera que controlar datos, validación o etc, se configura en la parte de la vista ya que estas funcionan como controladores.

## Dificultades:

Durante el trabajo me he encontrado varias dificultades, una de ellas fue a la hora de hacer el calendario, estuve mirando por internet ya que había muchos hechos y plugins que te ayudaban a hacerlo, pero al final viendo varios ejemplos conseguí hacer el mío propio.

Otra dificultad que tuve fue a la hora de hacer las reservas y comprobar cuando una reserva ya esté en una hora determinada.

Intenté desde el lado del servidor guardar las aulas que tenía reservadas para cuando coincidiera con la hora no dejara realizar la reserva, pero no es tan fácil ya que si una reserva dura varias horas en medio hay horas sueltas. Para resolver este problema desde el lado del cliente no dejo seleccionar las horas que estén reservadas así si el cliente no rellenar el formulario dará un error y si lo rellena tendrá que seleccionar las horas que haya disponibles.

A la hora de implementar Ajax tuve algunos problemas ya que la petición no se realizaba correctamente al principio, después de ver un par de tutoriales conseguí hacer varias peticiones de esta forma.

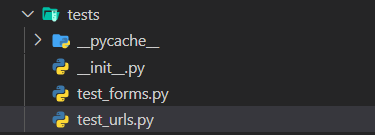
## Control de versiones:

Para el control de versiones, he ido subiendo varias veces el proyecto a github, Normalmente lo he ido subiendo después de haber conseguido un objetivo o después de un día entero trabajando, aparte he subido otros archivos por si modificaba el original tener una copia en GitHub segura.

# PRUEBAS

## Pruebas de servidor

He realizado varios test unitarios dentro del proyecto para comprobar que funciona, he hecho test a las Urls, a los modelos….

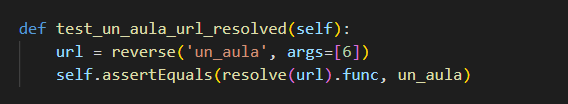
Para hacer los test he creado una carpeta y dentro los diferentes test que puedo hacer, y el archivo \_\_init\_\_.py el archivo init es como un enrutador o un lanzador para poder llamar a los test y que se ejecuten, una vez hemos creados nuestros archivos vamos a configurarlos por dentro.

Entro en el test de las urls:

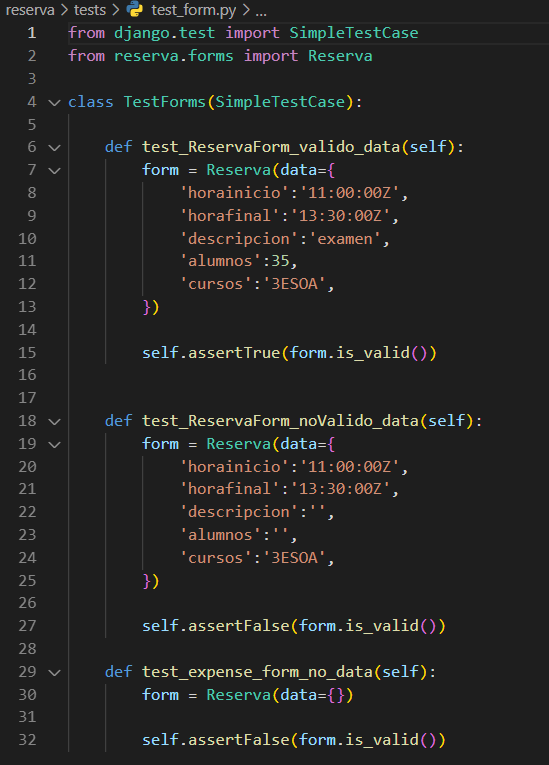


Como la aplicación la tengo dividida en dos pequeñas partes no tengo todas las Url en el mismo, cada app del proyecto tiene su propia carpeta de test.

Por ejemplo, si tenemos que hacer un test al que hay que pasarle un argumento como el ID de la clase se pondría así:

Como vemos le estamos pasando el argumento el número 6 que es el Id de una clase de prueba.

Como dije Antes se pueden hacer diferentes tipos de test ahora voy a mostrar los test que he hecho a los formularios que he creado con Django.

Dentro de los test del formulario he creado test con todos los datos bien puesto (un caso de test valido), y también he creado test a los que le faltaban un campo, o sin información (test no validos).

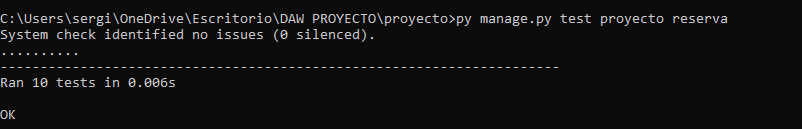
El primero contiene todos los datos lo cual es exitoso.

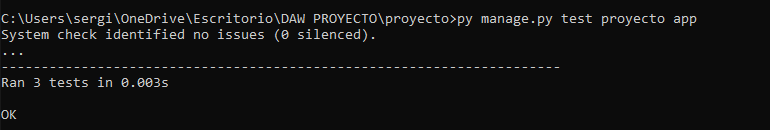
El segundo le falta el número de alumnos y el ultimo test no le paso datos directamente.

A la hora de probar los test nos iremos a nuestra cmd y escribiremos:

**Python manage.py proyecto reserva**

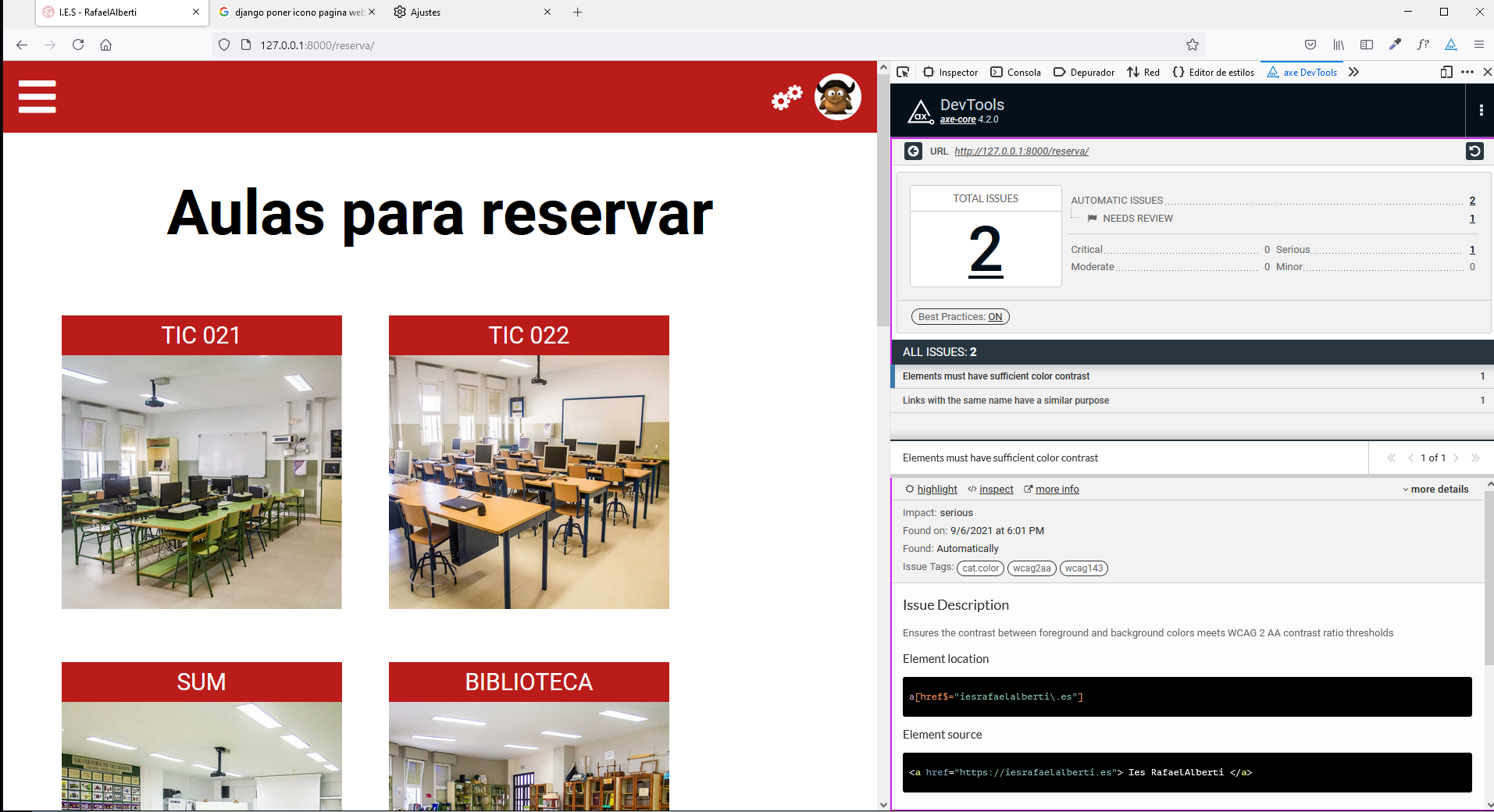
En mi caso proyecto es el nombre de mi proyecto de Django y reserva es el nombre del módulo al cual le voy a hacer los test, como tengo dos tendré que hacer reserva y luego app





## Pruebas de Diseño

Aparte de pruebas de código, he hecho pruebas de diseño con la extensión AXE. Estas pruebas las he realizado en Firefox



Casi todos los errores que me ha dado han sido por contrastes de colores, y porque los botones que tengo un icono me dicen que tienen que tener texto dentro. Como el botón de la papelera o el del ojo del login.

# DESPLIEGUE

Para el despliegue de la web he usado un contenedor Docker. He montado la aplicación, pero la base de datos la he tenido que poner de forma local ya que a la hora de conectar con los datos he tenido varios problemas.

# MANUAL

Al acceder a la aplicación web lo primero que nos mostrará será la página de inicio de sesión del usuario. En la cual podremos nuestras credenciales



Una vez entremos nos mostrará dos opciones, reservar un aula o temas de convivencia, este manual se centra en la reserva de aulas con lo cual hacemos clic en nuestro apartado.



Al entrar en la zona de Reservar un aula nos mostrará todas las aulas que se suelen reservar en el colegio.



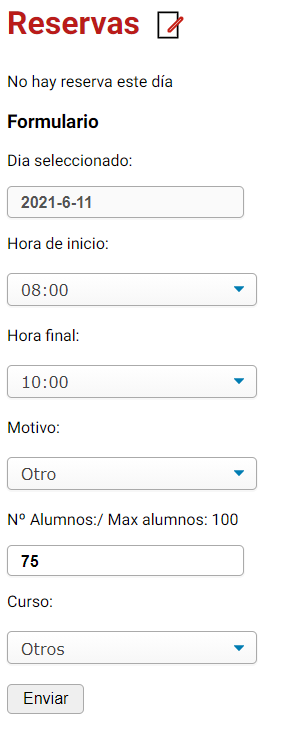
Para reservar un aula, tendremos que seleccionar la clase que queremos reservar, en este ejemplo elegiremos la sala de usos múltiples (SUM)



Dentro nos encontraremos este calendario en el cual se pueden seleccionar todos los días laborales, ya que un domingo o un sábado no se pueden reservar las clases.

Al seleccionar un día nos mostrará que eventos hay ese día y a su vez un formulario en el cual podemos hacer una reserva si queremos.

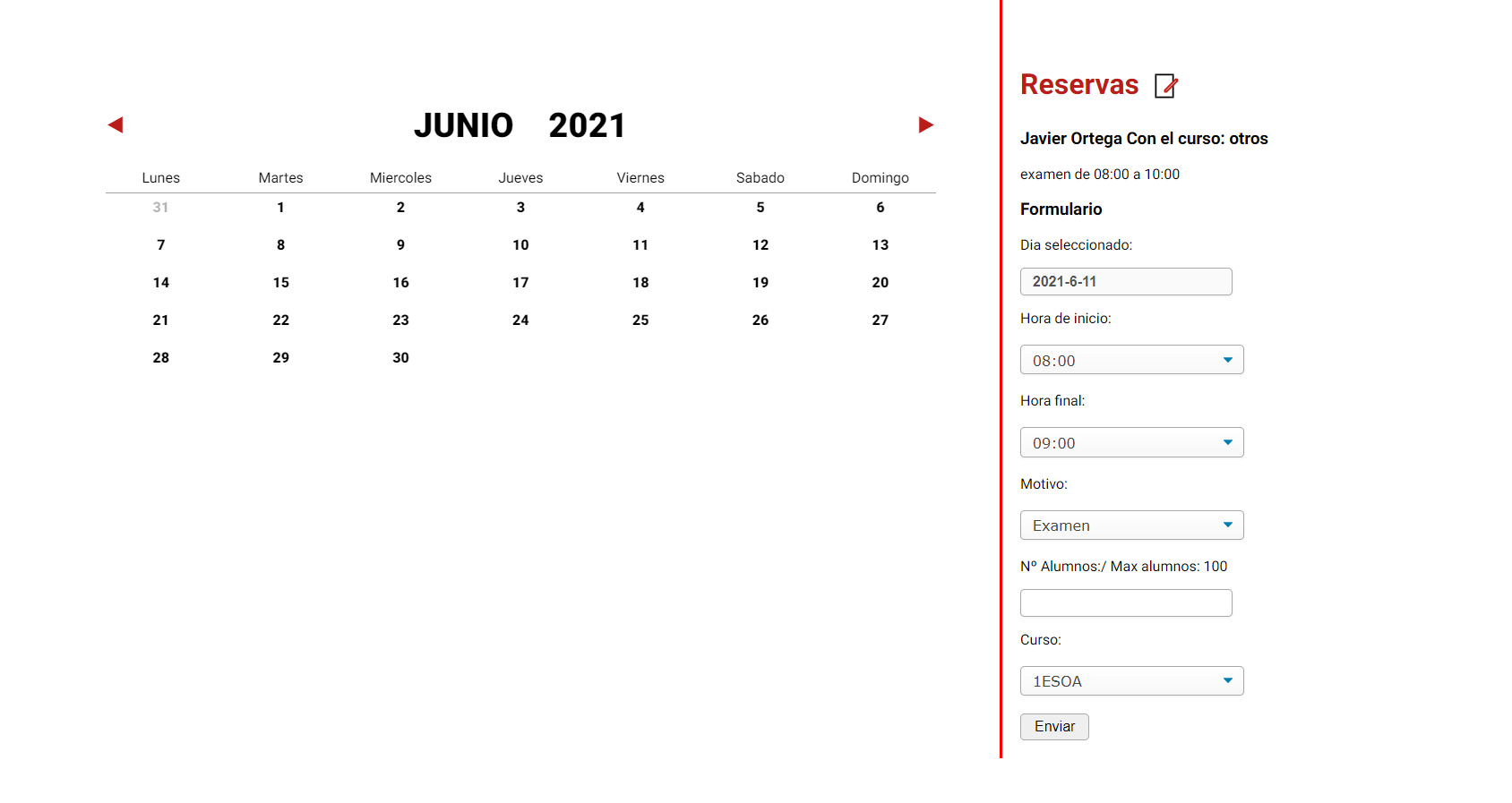


En el formulario podemos ver que para el día 11 de junio no hay reservas hechas así que vamos a crear una nosotros.

* Deberemos de rellenar el formulario adecuadamente para poder realizar la reserva con éxito.
* Deberemos de elegir la hora de inicio menor a la hora final.
* Si la hora de inicio es la misma a la hora final tambien dará un error.
* Tendremos que seleccionar un día que no haya pasado, ya que si la fecha ha pasado no tiene mucho sentido realizar una reserva.
* El numero de alumnos tiene que ser menor al numero maximo que soporta una clase.
* Una vez hayamos rellenado todos los campos adecuadamente la reserva se realizará, si por lo que sea hemos rellenado mal el formulario la web nos devolverá el siguiente error

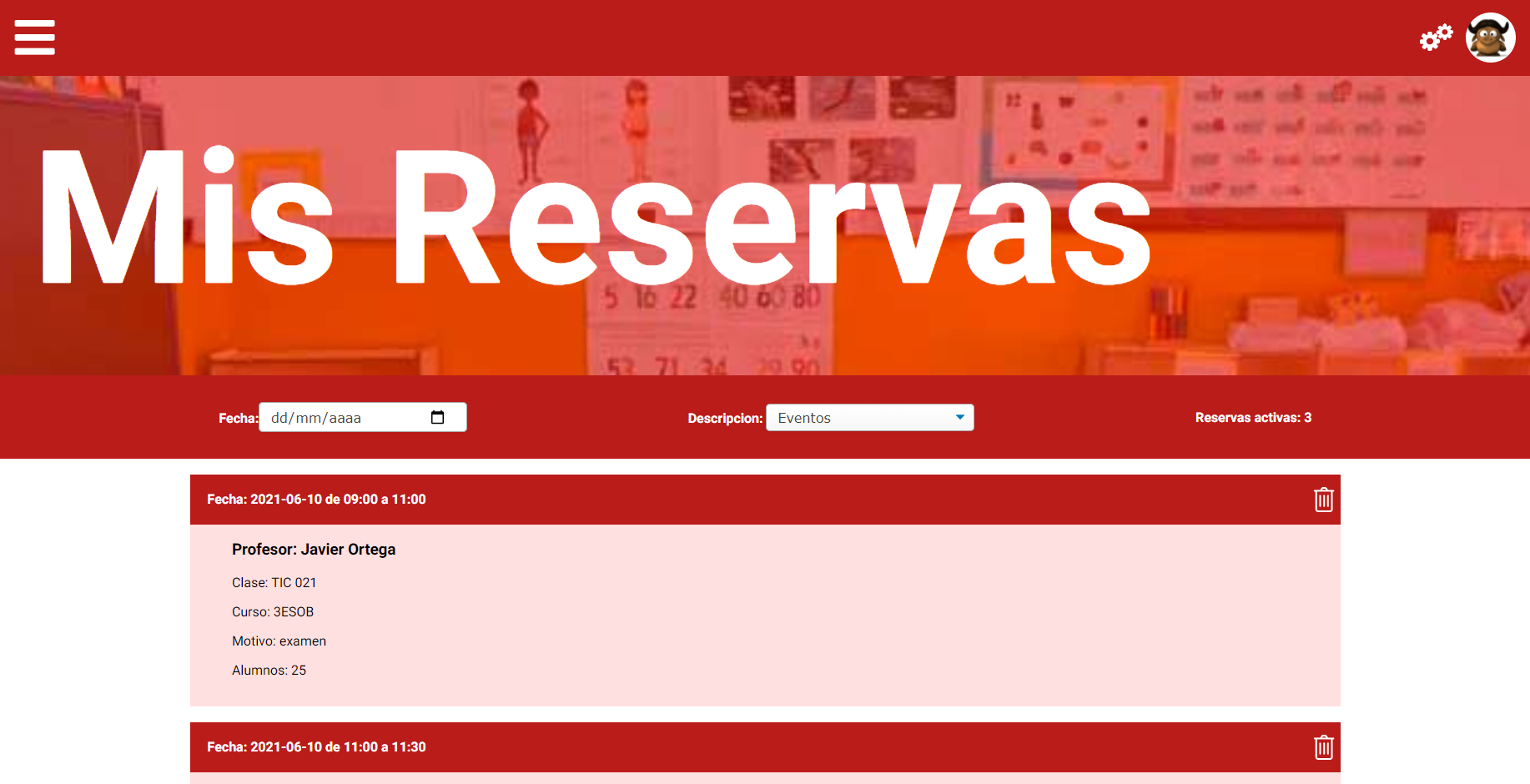


Y volveremos a poder realizar el formulario por ultimo si pinchamos en una clase que ya hay reservas hecha la pagina nos lo mostrará de la siguiente forma



Por ultimo hablar de la existencia de dos tipos de menus diferentes en la esquina superior derecha tendremos la opcion de cerrar sesion y la opcion de poder ir a ver las aulas que tengo reservadas

Si le damos a la Parte de Mis reservas:



Nos mostrará todas las reservas de un profesor desde el dia actual hacia delante, aunque podremos filtrar por dia en concreto y por tipo de evento.

Hay dos tipos de roles, el usuario normal que son todos los profesores y el administrador y la diferencia que hay entre uno u otro es que el administrador puede ver todas las reservas de todos los profesores, miesntras que el profesor solo puede ver las suyas

Si nos fijamos en la foto del perfil del usuario al lado hay unos engranajes eso quiere decir que es administrador.

Podremos eliminar tambien nuestras reservas si nos fijamos en cada reserva en la esquina superior derecha hay un icono de una papelera, pues desde ahí podremos eliminar la reserva.

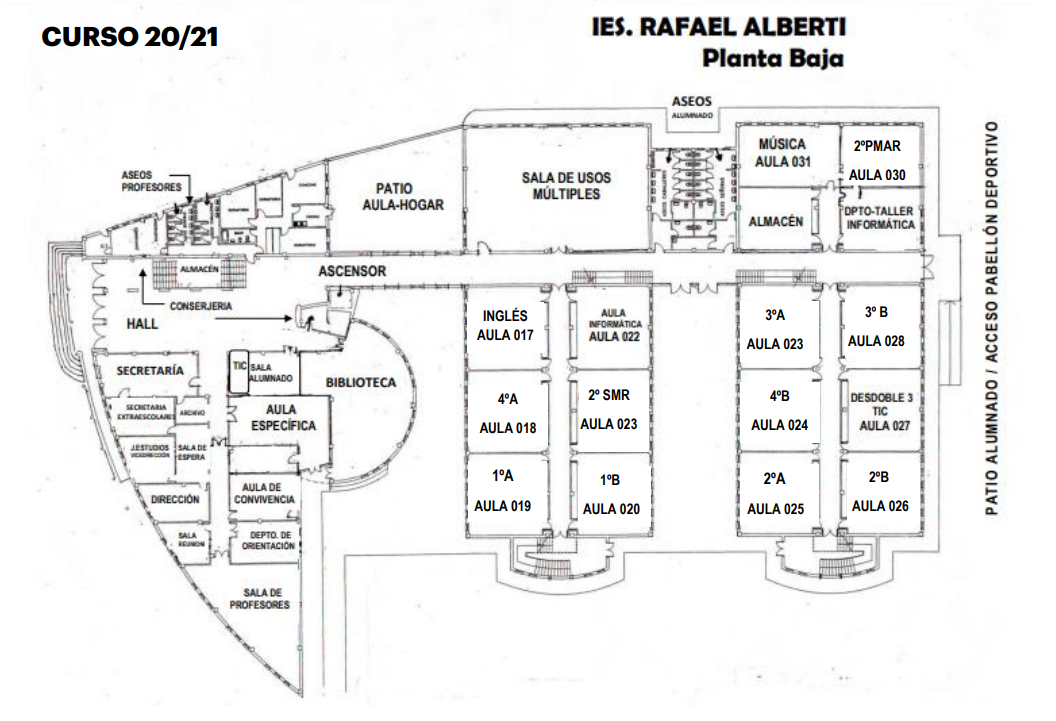
Por ultimo en la esquina superior izquierda tendremos otro menu en el cual podremos acceder a el si pinchamos encima de las tres barritas si hacemos clic en Reserva de aulas nos volverá a mandar al menú donde aparecen todas las aulas para poder realizar otra reserva de la clase que queramos.

# CONCLUSIONES

La idea principal se parecía al resultado final ya que el tema no ha variado mucho, pero si ha variado mucho la forma con la que iba a representar las clases y las ideas que tenía iniciales con el resultado conseguido.

La primera idea que tuve fue hacer un mapa, y que en el mapa se vieran todas las clases que se puedan reservar, pero después de pensarlo un par de veces creo que era complicado de realizar y poco intuitivo, ya que no todas las aulas se pueden reservar, después si se quieren meter más aulas es difícil de ampliar, y menos si hablamos de que tenemos que cambiar el mapa del centro.

Otra desventaja es que el centro tiene dos plantas y al principio no sabía si algunas iban a estar arriba otras abajo en fin que esta idea la descarté.



Un plano del mapa

Mi segunda idea fue el mostrar todas las aulas como están ahora, creo que con sus fotos y el nombre de cada zona cualquier profesor sabe a que aula se está refiriendo, aparte de que si queremos meter aulas nuevas solo con meter una entrada en la base de datos ya estarían introducidas.

Por otra parte, sabia al 100% que un calendario tenía que utilizar ya que se tiene que ver reflejada la fecha en la cual vamos a hacer nuestra reserva, al principio pensé en un calendario global en la que se pudiera ver que aulas estaban reservadas y a qué horas, pero al tener 10 aulas se me iban a pisar muchas de estas horas con las diferentes aulas y quizá podría ser un agobio al tener tanta información a la vez. Así que decidí mostrar dentro de cada aula sus eventos y en vez de en el calendario fuera de este con todos los datos.

Por último, como una especie de historial de quien ha reservado aulas o para asegurar que aulas he reservado añadí el apartado “Mis reservas” en el cual podemos ver todas las reservas que tangamos.

Mejoras futuras añadirle el otro apartado de convivencia para poder tener una aplicación con una mayor funcionalidad.

Añadir al calendario una función para poder reservar todos los días a una misma hora durante x días del año.

# ÍNDICE DE TABLAS E IMÁGENES

# BIBLIOGRAFÍA

Para aprender a manejar Django he visto el curso de [píldoras informáticas](https://www.youtube.com/watch?v=7XO1AzwkPPE&list=PLU8oAlHdN5BmfvwxFO7HdPciOCmmYneAB&index=1) de YouTube de Django La [documentación](https://docs.djangoproject.com/en/3.2/) de Django Y [stackoverflow](https://es.stackoverflow.com)

También he utilizado W3school

Documentación de Docker