DESARROLLO APLICACIONES WEB

PLANTARIUM

EL TEMPLO DE LAS PLANTAS

Alumno: López Lafita, Claudia Y.

Curso: 2º DAW

Curso Académico: 2022/2023

ÍNDICE

INTRODUCCIÓN

Este apartado sirve como una breve introducción de los conceptos que se explican en los sucesivos epígrafes del documento, así como un primer encuentro con este proyecto de final de Ciclo Formativo de Grado Superior en Desarrollo de Aplicaciones Web.

En la actualidad el auge de internet, ha generado nuevas necesidades en la sociedad hoy en día; grandes necesidades que hacen que hoy la mayor parte del mundo posea una conexión a internet de alta velocidad con el cuál se pueden acceder a diferentes e innumerables páginas web. Este avance tecnológico abre puertas para la digitalización de varios servicios tradicionales, así como la mejora continua de los ya existentes.

El objetivo principal de este proyecto es ofrecer una solución al problema de obtener información, cuidados y tratamientos de enfermedades de plantas ornamentales. Que hasta hace relativamente unos años se realizaba de manera tradicional, como la búsqueda en enciclopedias físicas o expertos en la materia (jardineros, tiendas, invernaderos...) así como la necesidad de realizar múltiples búsquedas en medios digitales.

Los visitantes de nuestro servicio serán capaces de informarse a modo de enciclopedia digital de las plantas que tengamos en nuestra base de datos, no solo de información científica como su taxonomía, ecologías, etc.; sino también de precauciones que se han de tener, como es por ejemplo la Lengua de Suegra (*Sansevieria trifasciata*), la cual es tóxica para los gatos. Además también tendrán la posibilidad de ver en cada ficha de cada planta un listado de las floristerías e invernaderos donde se pueden obtener, facilitando así la búsqueda de proveedores.

Los visitantes que se suscriban, ya sea gratuitamente o pagando dispondrán de un “jardín” donde podrán guardar aquellas plantas que tengan o que les interese. Gracias a la suscripción, tendrán el privilegio de tener a su disposición la información de los cuidados específicos para el crecimiento y mantenimiento de la planta; así como un “servicio” de enfermería donde el

usuario podrá seleccionar un síntoma que presente su planta para saber la causa de este así como el tratamiento que necesite para su mejoría.

En resumen, queremos ofrecer un servicio que ayude a los amantes de las plantas en la búsqueda de información sobre aquellas plantas que están y entraran en sus vidas para darles un poco de verde esperanza.

ANÁLISIS DE NECESIDADES DEL SISTEMA

En este apartado se van a especificar las condiciones generales que tiene el proyecto, así como sus capacidades.

DESCRIPCIÓN GENERALES

CAPACIDADES GENERALES

En este punto se definen los objetivos que debe de cumplir el sistema de cara a los usuarios:

- El sistema contiene tres tipos de usuarios: Administrados, Usuario Premium y Usuario General.
- El usuario administrador puede en todo momento gestionar plantas, cuidados, proveedores y síntomas.
- El usuario administrador podrá visualizar una gráficas de estadísticas de las plantas más “añadidas” teniendo en cuenta diferentes aspectos: categoría, por género, por nombre.
- El usuario administrador podrá visualizar estadísticas en relación a los usuarios: nº de suscritos, localidad, tipo de suscripción.
- El usuario administrador podrá visualizar estadísticas de ganancias: mensuales y anuales.
- Los usuarios premium / general pueden añadir y/o eliminar planta de su “jardín”
- El usuario (premium y general) podrá acceder a la información (cuidados, enfermedades, síntomas, tratamientos) de las plantas que tiene añadidas en su jardín, así como en caso de que tuviese peticiones, el estado (en curso, resuelto) y la información de devuelve.
- En todo momento se da la posibilidad de gestionar el cambio de contraseña para cada tipo de usuario final.
- El usuario podrá acceder a la información de los proveedores de las plantas en las que está interesado.

RESTRICCIONES GENERALES

En este punto se van a comentar las restricciones que se darań inicialmente en nuestro sistema necesarias para diseñar el sistema de control de la gestión de nuestra aplicación web:

- El usuario administrador tendrá mayores privilegios que el resto de usuarios, de forma que sea el único capaz de gestionar toda la información de nuestro sistema (plantas, proveedores, enfermedades, síntomas...).
- Las vistas del administrador y la de los usuarios finales serán diferentes de forma que se mantenga la seguridad frente a la aplicación que gestiona todo el sistema que es la del administrador.

CARACTERÍSTICAS DE LOS USUARIOS

En el sistema hay tres tipos de usuarios que son los destinatarios del sistema. El primer tipo es el usuario administrador, el segundo (usuario Premium) y tercero (usuario General) son los destinados a realizar un uso común de la aplicación, pero con ciertas diferencias. Cada uno de ellos tiene un rol y unos privilegios distintos que hacen que el uso que cada usuario hace de la aplicación sea distinto.

Administrador: este usuario es el encargado de gestionar y administrar la aplicación. Puede añadir o eliminar plantas, tratamientos, cuidados.... De esta forma, el rol de este usuario permite gestionar todos los datos que componen el conjunto de información del sistema. Es el usuario con más privilegios capaz de acceder a la aplicación.

Usuario Premium: usuario final que mediante la aplicación puede realizar gestiones ilimitadas: no tendrá límite para añadir plantas a su “jardín” junto con sus cuidados y tratamientos. Así mismo podrá eliminar las plantas de su “jardín” a su antojo. Para poder ser usuario premium se ha de pagar una cuota mensual de 5.99€/mes.

Usuario General: usuario final que mediante la aplicación puede realizar gestiones limitadas: tendrá un límite de 5 plantas para añadir plantas a su "jardín" junto con sus cuidados y tratamientos. Sí podrá eliminar las plantas de su "jardín" a su antojo.

En ambos casos tanto el usuario premium como el usuario general podrán pasar a usuarios premium y viceversa, en este caso el usuario se quedará con las cinco primeras plantas que ha añadido a su "jardín".

REQUISITOS DE USUARIO

El objetivo de este punto es precisar los requisitos de usuario para poder diseñar en futuros apartados el sistema de gestión del control de nuestra wiki. Estos requisitos se dividen en requisitos de capacidad (RC) y requisitos de restricción (RR), los primeros muestran qué debe ser capaz de realizar el sistema y los segundos especifican las restricciones que establecen el cómo se realizan las tareas que fueron indicadas en los primeros.

REQUISITOS DE CAPACIDAD

ID	TÍTULO	DESCRIPCIÓN
RC-1	Acceso a la cuenta	Un usuario puede acceder a su cuenta mediante su 'usuario' y su 'contraseña'.
RC-2	Visualización de información usuario	Un usuario registrado puede visualizar su información en todo momento
RC-3	Modificación de informacion usuario	Un usuario registrado puede modificar su información en cualquier momento.
RC-4	Consulta de planta buscada	Un usuario puede consultar la información de una planta buscada (independientemente del método)
RC-5	Añadir / Eliminar planta de Jardín	Un usuario puede añadir / eliminar una planta a su "jardín"
RC-6	Consulta de Jardín	Un usuario registrado puede visualizar en todo momento su "jardín"

RC-7	Consulta de cuidados de planta determinada	Un usuario registrado puede consultar los cuidados de una planta.
RC-8	Consulta de síntomas de un planta determinada	Un usuario registrado puede consultar los síntomas de una planta.
RC-9	Consulta de enfermedad	Un usuario registrado puede consultar las enfermedades determinadas por un síntoma de una planta.
RC-10	Consulta de tratamiento	Un usuario puede consultar el tratamiento determinado por una enfermedad de una planta.
RC-11	Consulta de proveedores	Un usuario registrado y no registrado puede consultar los proveedores de una planta.
RC-12	Registro de un usuario general	Un usuario puede suscribirse gratuitamente registrándose mediante un formulario
RC-13	Baja de un usuario	Un usuario puede eliminar su cuenta, eliminando sus datos en cualquier momento
RC-14	Modificar suscripción	Un usuario registrado podrá cambiar el tipo de suscripción en cualquier momento, modificándose su jardín convenientemente
RC-15	Añadir una planta	Un usuario administrador puede añadir una planta
RC-16	Eliminar una planta	Un usuario administrador puede eliminar una planta
RC-17	Modificar una planta	Un usuario administrador puede modificar una planta
RC-18	Visualizar proveedor	Un usuario registrado podrá visualizar la información de proveedores en cada planta.
RC-19	Añadir proveedor	Un usuario administrador podrá añadir proveedores a cada planta
RC-20	Modificar proveedor	Un usuario administrador podrá modificar la

		información de un proveedor
RC-21	Eliminar proveedor	Un usuario administrador podrá eliminar de una planta su proveedor (-es)
RC-22	Visualizar estadísticas	Un usuario administrador podrá visualizar gráficas de estadísticas.
RC-23	Distintas vistas para diferentes roles de usuario	Dependiendo del rol de usuario, se accederá a diferentes aplicaciones: La del usuario "cliente" o la del administrador.

REQUISITOS DE RESTRICCIÓN

ID	TÍTULO	DESCRIPCIÓN
RR-1	Uso de servicios por usuarios registrados	Únicamente las personas registradas pueden ser poseedoras de "jardín" y hacer uso de otros servicios.
RR-2	Tiempos de espera mínimos	El sistema debe responder en un tiempo breve a cada petición realizada.
RR-3	Almacenar datos de inicio de sesión	Los datos para el inicio de sesión pueden ser almacenados con el fin de evitar al usuario introducirlos cada vez que quiera acceder.
RR-4	Número máximo de plantas en jardín	Los usuarios generales podrán añadir un máximo de 5 plantas en su jardín.

ALTERNATIVAS EN EL MERCADO

El aumento de la afición de tener plantas en casa ha provocado un considerable incremento de sitios web dedicados a la jardinería, y a dar consejos sobre qué plantas debes tener en casa y cómo debes cuidarlas. En este apartado nos centraremos en indicar las posibles alternativas a nuestro servicio, así como lo que nos diferencia de ellas.

La competencia más fuerte que tendría nuestra aplicación sería Verdeesvida.es; es la página web de la Asociación Española de Centros de Jardinería, en la cual se encontrará un amplio número de consejos sobre el cuidado y mantenimiento de plantas de interior y exterior. Aún cuando se asemeja más a un blog de consejos, se puede encontrar un apartado de fichas de información sobre diferentes especímenes de plantas.

La primera aplicación a la que podemos considerar "competencia" en relación a tratamientos de enfermedades/plagas de plantas es IFAPA Guía Plagas, elaborada por el grupo de Protección Vegetal Sostenible del Centro IFAPA de Almería; donde incluye fotografías para facilitar la identificación de plagas y enemigos naturales en cultivos hortícolas de invernadero.

Una aplicación que nos hace competencia más directamente es Infojardin, es una aplicación bastante completa sobre todo para consultas de árboles y plantas de jardín. Una categoría más extensa y que seguramente se usa más, es la de fichas de plantas ya que, una vez localizada una planta de interés, se encuentra información sobre ella. La otra categoría que más puede interesar sería la de "Plagas y enfermedades", donde agrupadas por categoría se puede encontrar la patología deseada mediante imágenes.

Una vez conocida la competencia ante la que se encontrará nuestro servicio, podemos definir los puntos similares y distintivos que nos destacan. En lo que nos parecemos es que al igual que en las páginas y aplicaciones mencionadas es que dispondremos de información general de la planta a modo de enciclopedia; así como sus cuidados básicos.

Lo que nos diferencia positivamente, es que el usuario podrá ver la información, cuidados, enfermedades y tratamientos de una planta en la misma página muy intuitivamente. Esto permite que el usuario no tenga que realizar búsquedas secundarias o incluso terciarias, para encontrar la enfermedad y posteriormente un tratamiento. Un servicio que también vamos a tener y que no se ha visto ni en las aplicaciones web ni en la de móviles son la lista de proveedores, tiendas físicas, de la localidad donde se encuentre el usuario; este extra hace que el usuario pueda ver donde conseguir las plantas sin tener que realizar otra búsqueda complementaria.

Lo que sí tienen y que tienen un papel importante en el crecimiento de una aplicación, es la disponibilidad de blog donde los usuarios pueden interactuar entre sí, este tipo de servicio da a los propietarios la capacidad de saber las necesidades de sus clientes y las mejoras de su producto. Otro servicio que llama la atención y que ayuda mucho en este sector es la publicación de artículos de consejos, donde se muestra empatía y tips para que el usuario se sienta en una comunidad.

MOTIVACIÓN

El título oficial de “mataplantas” no solo está a la orden del día para aquellas personas que se inician en embellecer su casa con una planta, sino que se extiende a “jardineros” avanzados que se enfrentan a plantas que no conocen. Hay que saber que las plantas ornamentales requieren de determinadas condiciones para su correcto desarrollo. Estás varían dependiendo de la especie, por lo que debemos tenerlo muy en cuenta si queremos que nuestra planta crezca y tenga un aspecto saludable.

Personalmente en más de una ocasión me he visto en la situación en la que doy toda la atención del mundo a una planta, y sin venir a cuento comienza a languidecer. La incertidumbre de no saber qué es lo que he hecho mal es la causa de entrar en una épica búsqueda de información para remediarlo; y digo épica porque la jardinería es más compleja de lo que se imaginan. Y es este el motivo principal por el cual he decidido desarrollar este proyecto.

Lo primero que hace uno es preguntar en la floristería/invernadero de confianza o donde se ha comprado la planta, esto conlleva un desplazamiento y un conocimiento previo de la planta que tenemos en nuestro poder; y aun así podemos encontrarnos diversos contratiempos. Esto nos lleva a la segunda alternativa: búsqueda en internet.

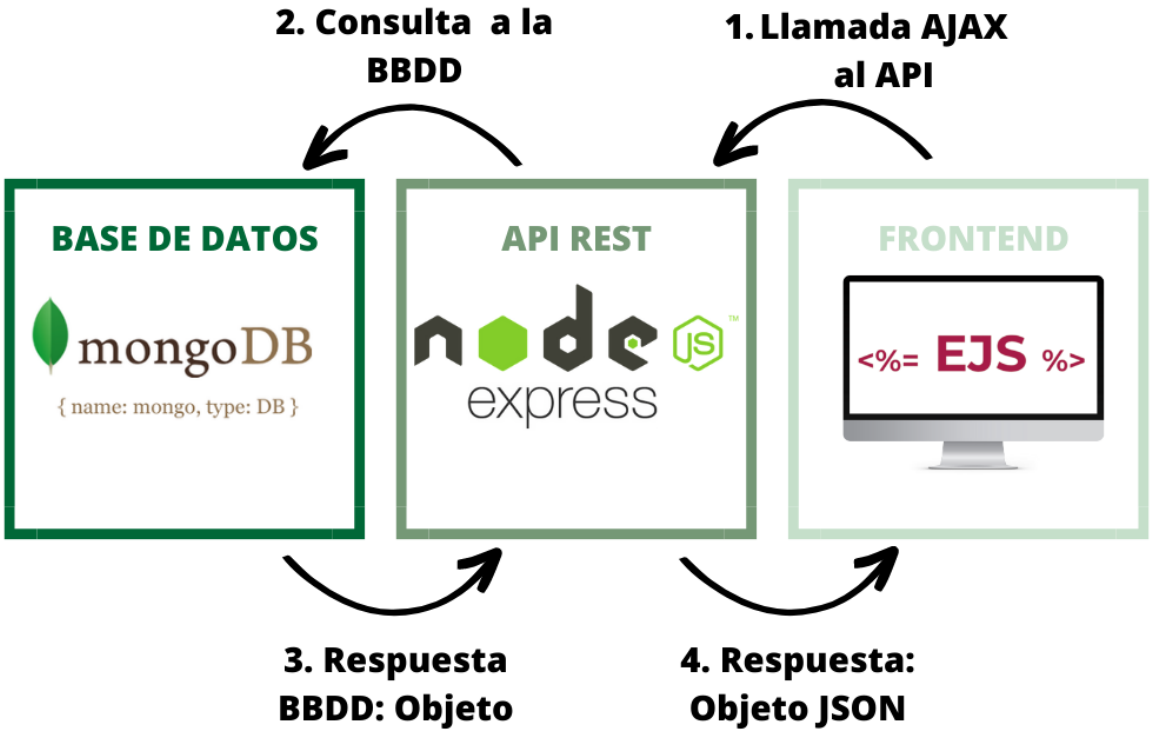
En internet podemos encontrar un sinfín de información sobre nuestra planta, desde blog donde sus dueños publican sus experiencias, hasta páginas de centros oficiales. Pero en este mar de información podemos encontrar datos correctos o incorrectos teniendo que ir navegando por varias páginas hasta encontrar la información que necesitamos.

Pero una cosa es la información sobre nuestra planta y otra es la información sobre la enfermedad y/o tratamiento; como podemos ver en el apartado de alternativas en el mercado, en una misma página no encontramos toda la información que nos interesa, sino que tras una larga búsqueda podemos considerar que tenemos la información necesaria para poder continuar con el cuidado de nuestra planta.

STACK TECNOLÓGICO

En este apartado incluiremos el lenguaje de programación, bases de datos, frameworks, etc. que utilizaremos para desarrollar nuestro servicio web. Así mismo al final se explicará la arquitectura usada para este proyecto.

Elegir el stack tecnológico "adecuado" para nuestra aplicación web es esencial para poder ofrecer un producto de calidad y eficiente. Nuestro stack tecnológico está conformado por varios elementos que permiten que podamos interactuar con nuestra aplicación (Imagen 1).



LENGUAJE DE PROGRAMACIÓN

En el desarrollo de aplicaciones web, el uso de JavaScript es fundamental para integrar elementos interactivos, animaciones y manipulación de datos, entre otros. Este lenguaje de programación interpretado, basado en ECMAScript, permite que nuestra aplicación sea compatible con diferentes navegadores, como Chrome, Firefox y Safari. Además, la popularidad de JavaScript entre los programadores ha generado una amplia comunidad dispuesta a compartir

conocimientos y publicar proyectos en repositorios remotos, lo que nos brinda acceso a información útil.

Una de las ventajas más importantes de JavaScript radica en su capacidad para funcionar en el **lado del cliente,** lo que conlleva beneficios significativos. Entre ellos se encuentran la reducción del consumo de ancho de banda y la aceleración tanto en la ejecución del código del programa como en el rendimiento general del sitio web. Estas características hacen de JavaScript una elección favorable en el desarrollo de aplicaciones web.

LADO BACK-END

Para el correcto funcionamiento de nuestra aplicación web, es necesario contar con una parte backend que se encargue de almacenar y gestionar la información del sistema, además de optimizar aspectos como la seguridad y la privacidad. En este sentido, utilizaremos el framework Express.js para el desarrollo del lado servidor de nuestro proyecto, el cual se ejecuta en el entorno de Node.js. Node.js es un entorno de ejecución que permite correr JavaScript fuera del navegador, basado en eventos asíncronos y en el motor V8 de Google. Esta combinación lo convierte en un entorno altamente **escalable** y **eficiente**, capaz de procesar múltiples conexiones simultáneamente.

El uso de Express.js nos permitirá acelerar el desarrollo de nuestro proyecto al evitar la repetición de código y asegurarnos buenas prácticas y consistencia en el código. Este framework es ampliamente utilizado en el ecosistema de Node.js y presenta ventajas propias del entorno de ejecución. Además, cuenta con un sistema de **enrutamiento** poderoso y robusto incorporado por defecto, así como una amplia gama de middleware que facilitan el desarrollo sin problemas. Estos **middleware** nos permiten interceptar el flujo de la aplicación y realizar diversas acciones durante el proceso de desarrollo.

LADO FRONT-END

El frontend desempeña un papel fundamental al crear la interfaz de un sitio web, abarcando desde su estructura hasta los estilos que incluyen colores, tipografías y secciones. Su correcta implementación garantiza una experiencia positiva para los usuarios y fomenta su compromiso con la aplicación. Los lenguajes de diseño utilizados en el frontend son CSS, HTML y JavaScript, y son responsables de los componentes visibles para los visitantes del sitio.

En nuestro proyecto, optamos por utilizar **EJS** (Embedded JavaScript Templating), uno de los motores de plantillas más populares en JavaScript. Este motor de plantillas nos permitirá combinar plantillas con modelos de datos, lo que nos facilitará la interpolación de datos en código HTML. Entre las características que nos han llevado a elegir esta tecnología se encuentra la similitud del código EJS con **HTML puro**, lo que facilitará su integración con Bootstrap, una biblioteca de diseño multiplataforma de código abierto. Además, los errores de EJS son **fáciles de depurar**, ya que se presentan como excepciones simples de JavaScript, que incluyen números de línea en las plantillas. Por último, valoramos la existencia de una gran comunidad de usuarios activos y el hecho de que la biblioteca se encuentre en desarrollo activo.

BASE DE DATOS

La base de datos es crucial para almacenar y gestionar los datos en nuestro servicio, permitiendo adaptar y optimizar nuestra aplicación. Para este propósito, hemos elegido MongoDB, un sistema de base de datos NoSQL. Los principales motivos de su elección son su **flexibilidad**, ya que no sigue un esquema rígido y permite cambios en la estructura de datos con el tiempo, y su **escalabilidad**, tanto vertical como horizontal, lo que mejora el rendimiento de la aplicación. La excelente **documentación** oficial de MongoDB también nos brinda recursos para mejorar las operaciones y su uso.

Además de las razones técnicas mencionadas, hay otras razones por las cuales hemos elegido MongoDB. Es un software de **código abierto** respaldado por una comunidad activa, lo que asegura su constante evolución. Dado que estamos desarrollando nuestro proyecto con JavaScript y Node.js, MongoDB se integra de manera natural. Además, es una base de datos fácil de aprender y una herramienta gratuita, lo que nos permite incorporarlo al proyecto sin incurrir en costos de licencia.

MODELO ARQUITECTÓNICO DEL SOFTWARE

La arquitectura monolítica, utilizada en las primeras aplicaciones, sigue teniendo ventajas importantes a pesar del desarrollo de alternativas más complejas. Una de estas ventajas es la centralización del desarrollo, lo que facilita la localización del código fuente en un único lugar. Esto proporciona simplicidad en la aplicación desde el punto de vista de la lógica empresarial, ya que todos los datos necesarios para nuevas características se encuentran en un mismo paquete. Además, el despliegue y la ejecución son simples, sin necesidad de herramientas sofisticadas de automatización.

BASE DE DATOS

En este apartado nos centraremos en la definición de premisas que nos permitirán diseñar la base de datos; así mismos se mostrará el modelo entidad relación y finalmente el modelado de datos.

DISEÑO BASE DE DATOS

Los datos son el activo más importante de nuestra aplicación y una base de datos bien diseñada influye de forma directa en la eficiencia que obtendremos a la hora de almacenar, recuperar y analizar dichos datos. Por lo tanto, el diseño de base de datos es un proceso fundamental.

En este subapartado realizaremos una descripción a alto nivel del contenido de nuestra base de datos, independientemente del sistema de gestión de base de datos que usaremos. Nuestra base de datos tiene una totalidad de 10 entidades y para establecer la "relación" entre ellas se han definido las siguientes premisas:

- Un **usuario** realiza varios pagos o ningún **pago**.
- Un **pago** es realizado por un único **usuario**.
- Un **pago** determina una única **suscripción**.
- Una **suscripción** está determinada por un único **pago**.
- Una **suscripción** permite un único **jardin**
- Un **jardín** es permitido una única **suscripción**
- Un **jardín** puede no contener ninguna planta o contener varias **plantas**.
- Una **planta** puede no estar en ningún jardín o estar en varios **jardines**.
- Una **planta** tiene un **cuidado** único.
- Un **cuidado** pertenece a una única **planta**.
- Un **planta** puede presentar uno a varios **síntomas**
- Un **síntoma** puede estar presente en una o varias **plantas**
- Un **síntoma** es consecuencia de una única **enfermedad**
- Una **enfermedad** causa un único **síntoma**.
- Una **enfermedad** es curado por un único **tratamiento**

- Un **tratamiento** cura una única **enfermedad**.
- Una **planta** es oferta por uno o varios **proveedores**
- Un **proveedor** puede ofertar una o varias **plantas**

MODELO ENTIDAD-RELACIÓN

Una vez definidas las premisas que nos permiten establecer la relación entre entidades, debemos indicar los atributos que presenta cada una de ellas; siendo los atributos las características que definen o identifican a una entidad. Quedando el diagrama Entidad-Relación (Imagen 1) de la siguiente manera:

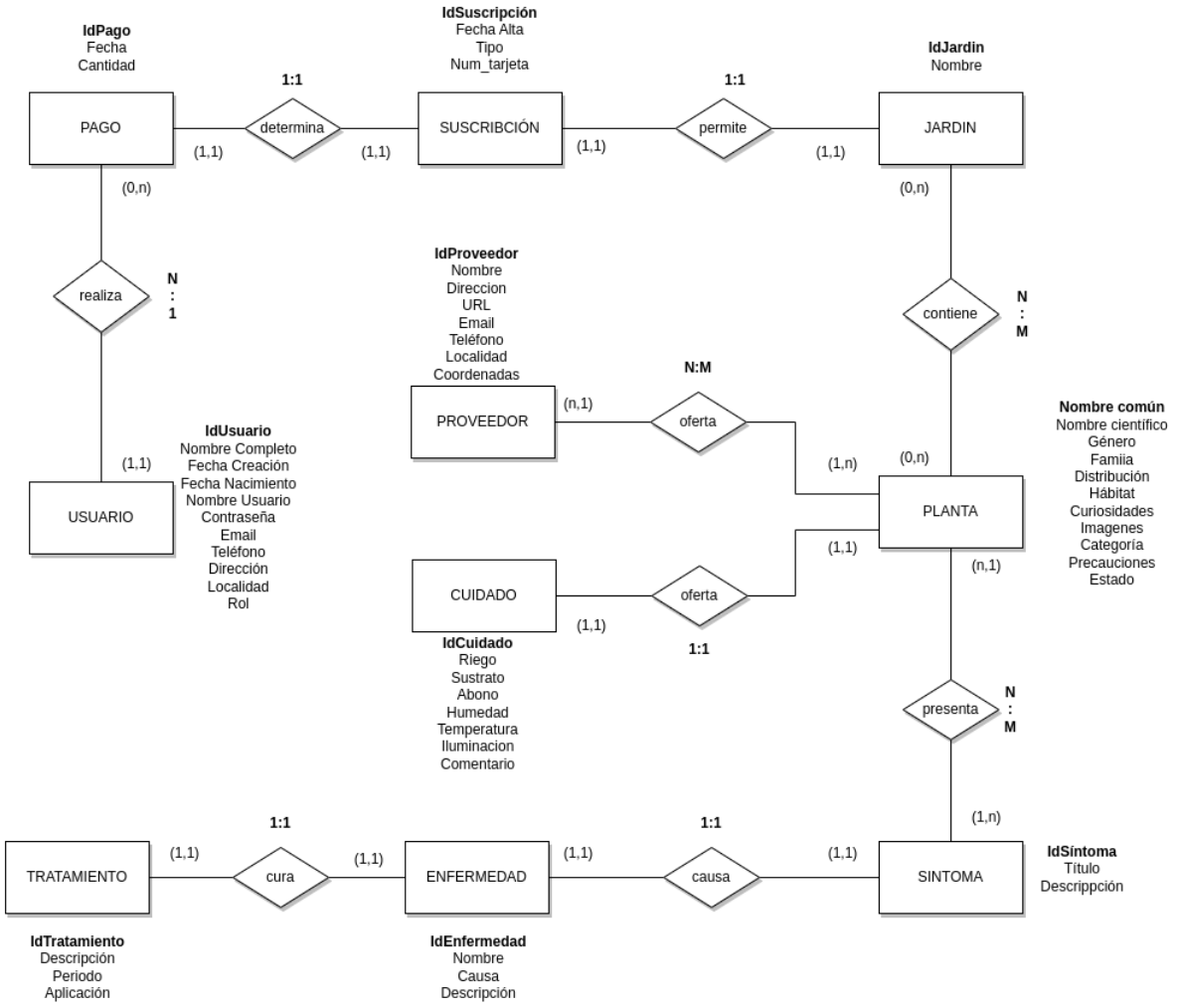


Imagen 1. Diagrama Entidad-Relación. Representación gráfica de la base de datos de Plantarium generada mediante el programa Diagrams. Autoría: Claudia Y. López Lafita.

En la base de datos de la figura anterior, hay que tener en cuenta que cada cambio que se realiza es de forma automatizada, debido a que es el servicio

web el que la gestiona, es decir, mayormente no hay ningún tipo de interacción humana con la base de datos sino, que mediante procedimientos del servicio web, se pueden desde añadir, modificar y/o borrar información a las distintas "tablas".

MODELADO DE DATOS

Una vez tenemos el diagrama entidad-relación con los atributos de nuestra entidad, y sabiendo que para el almacenamiento y tratamiento de datos en nuestro proyecto será en MongoDB, podemos modelar nuestras entidades más específicamente, indicando que tipo de dato será cada uno de los atributos. Así mismo al ser una base de datos no relacional explicaremos que se va a usar, si embeber datos o referenciarlos en cada relación.

Debemos de tener claro, que cuando hablamos de embeber, queremos decir que un documento puede contener a otros. Mientras que al hablar de referencias simplemente se mostrará en un documento el identificador del otro documento con el que tiene relación. Teniendo en cuenta los conceptos anteriores a continuación se definirán las entidades como documentos, para tener claro cómo quedará cada uno en la base de datos; por ello tanto las entidades como los atributos se han pasado a inglés, pués así como se guardarán en la base de datos, en contraposición a la representación gráfica anteriormente mostrada; la cuál está en español para que sea más cómoda de entender.

USER:

username - type: String /* Nick de usuario, será el identificador único */
password - type: String /* Contraseña del usuario, obligatorio se guardará en la base de datos de manera encriptada*/
creationdate - type: Date /* Fecha de alta del usuario, será por defecto la fecha actual y de tipo Date*/
fullName - type: String /* Nombre completo, campo obligatorio*/
photo - type: String /* Ruta donde se encontrará la foto del usuario*/
email - type: String /*Correo electrónico del usuario único*/

birthdate - type: Date /* Fecha de alta del usuario */

address - type: String /*Dirección del usuario */

locality - type: String /* Localidad actual del usuario */

phone - type: number /* Teléfono del usuario */

dni - type: String /* Documento de identidad del usuario */

numCard - type: String /* Tarjeta de crédito del usuario */

role - type: String /* Rol del usuario, que solo podrá tener dos valores dependiendo de cómo interactúa con la aplicación; será por defecto suscriptor.*/

payments - type: ObjectId /* Array de identificadores de cada pago que realice el usuario, por defecto estará vacío */

subscription - type: ObjectId /* Array de identificadores de la suscripción del usuario */

PAY:

codPay - type:String /* Código de pago */

date - type: Date /* Fecha cobro del pago de la suscripción premium */

amount - type: Number /* Cantidad a cobrar al usuario con suscripción premium, por defecto será 5.95€ */

subscription - type: ObjectId /* Identificador de la suscripción a la que está asociado el pago */

user - type: ObjectId /* Identificador del usuario que realiza el pago */

SUBSCRIPTION:

codSubscription - type: String /* Código de la suscripción; se creará aleatoriamente cuando se dé de alta.*/

date - type: Date /* Fecha de suscripción que coincide principalmente con la fecha de alta del usuario, pero que puede cambiar si este cambia el tipo.*/

type - type: String /* Tipo de suscripción que por defecto tendrá el valor de general. Enumerado con dos opciones: General y Premium*/

payments - type: ObjectId /*Array con los identificadores de los pagos relacionados con la suscripción premium*/

garden - type: ObjectId /* Identificador del jardín que se genera cuando el usuario se suscribe */

user - type: ObjectId /* Identificador del usuario al que pertenece la suscripción*/

GARDEN:

codGarden - type: String /* Código único, generado aleatoriamente */

name - type: String /*Nombre del jardín creado automáticamente con el nick de usuario */

subscription - type: ObjectId /* Identificador de la suscripción a la que está relacionado el jardín*/

plants - type: ObjectId /*Array de los identificadores de las plantas que el usuario irá añadiendo*/

PLANT:

codPlant - type: String /*Código generado automáticamente */

sciName - type: String /* Nombre científico de la planta */

comName - type: String /* Nombre común de la planta */

genus - type: String /* Género de la planta */

family - type: String /* Familia de la planta */

distribution - type: String /* Distribución de la planta, enumerado de dos valores: cosmopolita o endémico */

habitat - type: String /* Campo que informa del hábitat de la planta */

description - type: String /*Breve descripción de la planta y que irá en la tarjeta. */

curiosities - type: String /* Campo que informa de curiosidades de la planta*/

precautions - type: String /* Campo donde se indicarán precauciones que se ha de tener con esta planta. */

categories - type: String /* Array donde se guardaran las categorías que se le asigne a la planta */

images - type: String /* Array con las rutas donde se localizan las imágenes de las plantas */

attendance - type: ObjectId /* Referencia del cuidado de la planta*/

gardens - type: ObjectId /* Array de los identificadores de los jardines que contienen la planta */

suppliers - type: ObjectId /* Array de identificadores de los proveedores que proveen la planta */

symptoms - type: ObjectId /* Array de identificadores de los síntomas registrados de la planta */

ATTENDANCE:

codAttendace - type: String /* Código generado aleatoriamente */

water - type: String /* Frecuencia de riego, enumerado de tres posibles opciones: [‘Poco frecuente’, ‘Frecuente’, ‘Muy Frecuente’] */

soil - type: String /* Tipo de sustrato que necesita la planta */

compost - type: String /* Tipo de abono complementario que necesita la planta */

moisture - type: String /* Humedad a la cual se debe de mantener la planta */

temperature - type: String /* Rango de temperatura a la cual se debe de mantener la planta */

lightning - type: String /* Grado de iluminación que debe de tener la planta para un desarrollo, enumerado de tres opciones: [‘Alta’, ‘Moderada’, ‘Baja’] */

comment - type: String /* Campo donde se especificarán los cuidados especiales, si es necesario, de las plantas.*/

plants - type: ObjectId /* Identificador de la planta a la que corresponde el cuidado*/

SYMPTOM:

codSymptom - type: String /* Código generado automáticamente */

title - type: String /* “Título” o nombre del síntoma*/

description - type: String /* Campo donde se describe el síntoma */

disease - type: Object /* Documento embebido en síntoma, que corresponde con la enfermedad que causa el síntoma */

plants - type: ObjectId /* Array de identificadores tiene este síntoma*/

DISEASE:

codDisease - type: String /* Código generado automáticamente */

name - type: String /* "Título" o nombre de la enfermedad */

cause - type: String /* Campo que indica la causa de la enfermedad, enumerado de dos valores: por cuidados o patógenos */

description - type: String /* Campo donde se describe la enfermedad */

treatment - type: Object /* Documento embebido en enfermedad, que corresponde con el tratamiento */

TREATMENT:

codTreatment - type: String /* Código generado automáticamente */

period - type: String /* Campo donde se especifica el periodo de tiempo durante el cual se ha de realizar el tratamiento */

application - type: String /* Campo donde se especifica el método de aplicación del tratamiento*/

comment - type: String /* Campo donde se describe cualquier peculiaridad.*/

SUPPLIER:

codSupplier - type: String /* Código generado automáticamente. */

name - type: String /* Nombre empresarial del proveedor */

address - type: String /* Dirección del proveedores */

email - type: String /* Correo electrónico de el proveedor */

url - type: String /* Url de la página web del proveedor */

phone - type: Number /* Número de contacto del proveedor */

locality - type: String /* Localidad donde se ubica el proveedor */

ubicación - type: String /* Coordenadas del proveedor obligatorias */

plants - type: ObjectId /* Array identificadores de las plantas que el proveedor oferta*/

En la mayoría de las relaciones se ha escogido la referencia por encima de embeber; esto se debe a que si realizamos muchas inserciones y sobre todo

actualizaciones, será conveniente usar referencias, mejorando así el rendimiento. En contrapartida, la relación Symptom-Diseases y la relación entre Disease-Treatment son relaciones 1:1 y cuyos datos estarán más sometidos a la lectura de datos. El hecho de que toda la información a recuperar esté almacenada mediante documentos embebidos, favorece que el rendimiento de lectura sea muy alto, ya que sólo se hace un acceso a la base de datos.

PROTOTIPADO

Una interfaz web es un sistema visual que permite a los usuarios acceder a los contenidos de la web utilizando elementos gráficos. El objetivo principal al diseñar una interfaz web es asegurar que los usuarios puedan acceder a los contenidos de manera rápida y fácil. Para lograrlo, los diseños de las interfaces deben destacar en dos aspectos clave: simplicidad y coherencia.

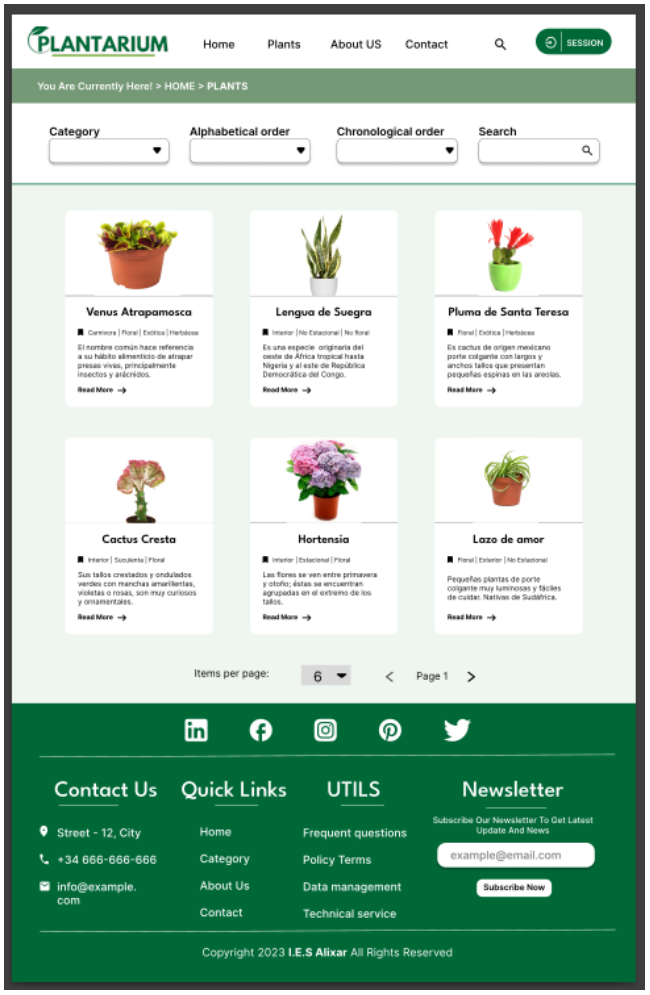
Para el correcto diseño se debe recopilar información precisa sobre las necesidades y objetivos tanto del cliente (proveedor) como del usuario. En el caso del proveedor, se pueden obtener fácilmente mediante entrevistas y reuniones con los responsables del sitio. Sin embargo, obtener información del usuario puede resultar más desafiante pero también más importante. Es crucial comprender las necesidades del usuario, sus objetivos, comportamiento, acciones, contexto de uso, así como el impacto en la interacción, experiencia y conocimientos previos del usuario.

Después de conocer las necesidad y consejos para diseñar la interfaz y los elementos esenciales para planificar correctamente nuestro proyecto web, ahora nos enfocaremos en el desarrollo del "prototipo". Para eso usaremos la herramienta Figma, donde se crearán los MockUp (maquetas).

Un mockup es una representación gráfica completa que combina un wireframe (esquema base) con elementos visuales y gráficos para crear un modelo a escala del producto. Su objetivo es demostrar y probar el diseño, mostrando tanto la apariencia visual como los fundamentos de la funcionalidad. Aunque los mockups son estáticos, proporcionan una idea clara de cómo se verá y funcionará el producto final. Incluyen detalles como colores, tipografía y elementos más detallados como contenidos, paleta de colores, declaraciones CSS, dimensiones e iconografía.

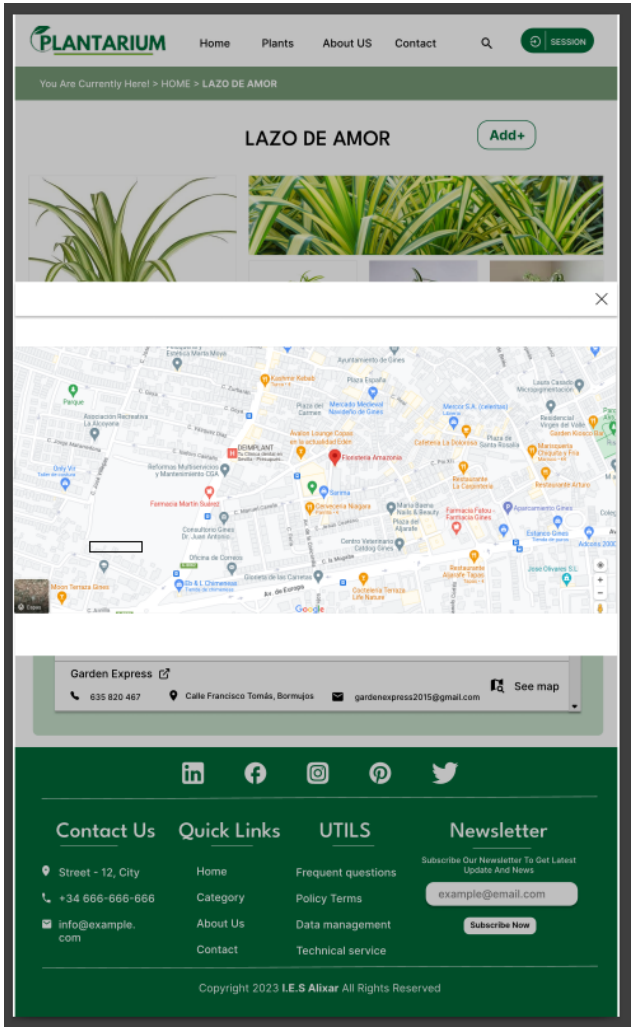
A Continuación se muestran los Mockup más significativos; para la visualización de todos los MockUp vaya a ANEXOS.

Página de listado de plantas

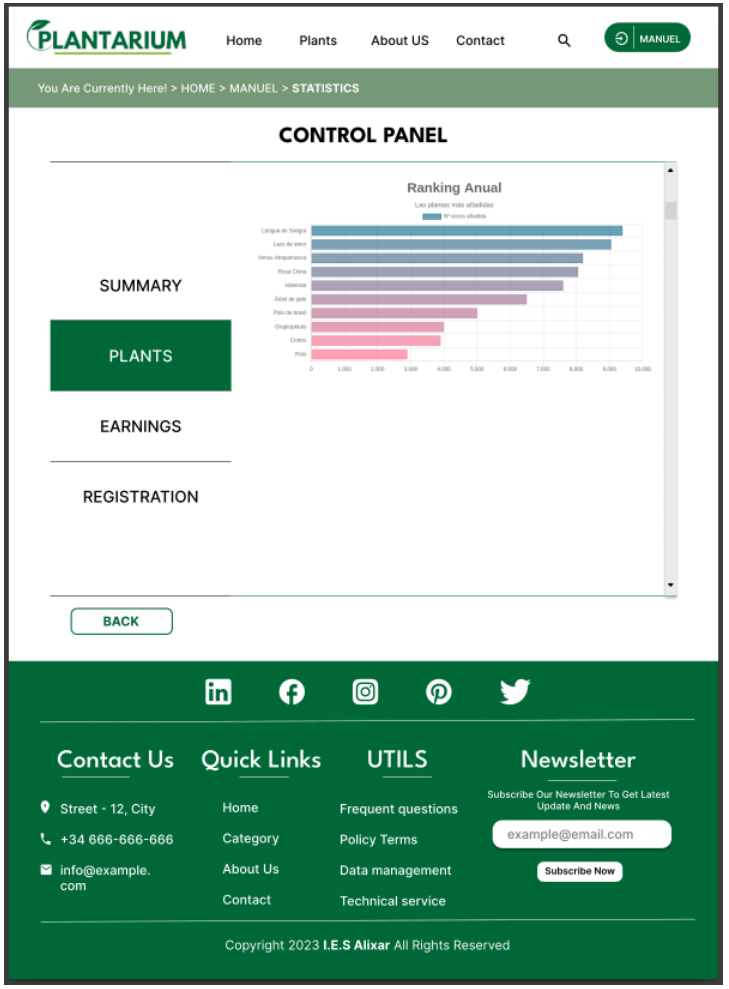


Perfil de un Planta

Mapa de localización de Proveedor



Gráfica top 10 de plantas



SERVICIOS REST

REST, abreviatura de Representational State Transfer, se describe como un protocolo basado en HTTP para intercambiar y manipular datos en los servicios de Internet. Mientras que una API REST podemos definirla como una librería de servicios o recurso REST (funcionalidades) que satisface diferentes necesidades de intercambio y manipulación de datos a través de Internet. El acceso a los recursos se realiza a través de URLs que identifican las solicitudes del servidor las cuales son transmitidas mediante el protocolo HTTP.

En HTTP podemos encontrar diferentes tipos de solicitudes, las cuales se corresponden con operaciones CRUD. Después de que se realicen las solicitudes al servidor, se espera que este envíe una respuesta de vuelta al cliente. De esta forma tendremos:

- **GET:** Utilizado para acceder a los recursos a través de un endpoint, lo que permite leerlos y/o consultarlos. Es similar a la operación de lectura (READ) en el modelo CRUD.
- **POST:** Permite crear un nuevo recurso utilizando la información proporcionada por el cliente. Es similar a la operación de creación (CREATE) en el modelo CRUD.
- **PUT:** Se utiliza para actualizar un recurso utilizando la información proporcionada por el cliente. Es similar a la operación de actualización (UPDATE) en el modelo CRUD.
- **DELETE:** Accede a un recurso a través de su punto final (endpoint) para eliminarlo. Es equivalente a la operación de eliminación (DELETE) en el modelo CRUD.

Teniendo en cuenta lo mencionado anteriormente, en este apartado nos centraremos en definir por modelo de datos las rutas que dirigen las solicitudes entrantes de la API a los recursos de backend.

Cabe destacar que aunque disponemos de un total de 8 entidades con las cuales maniobrar, solo trabajaremos directamente con 3: Usuario, Planta, Proveedor.

USER		
Ruta	**Método**	**Descripción**
/users	GET	Devuelve todos los usuarios
/users	POST	Registra un nuevo usuario a la BBDD
/users/update	POST	Actualiza los datos de un usuario registrado
/users/delete	POST	Elimina un usuario registrado en la BBDD
/users/reset_password	POST	Modifica la contraseña del usuario registrado
/signin	POST	Login de usuario registrado
/back	POST	Permite la vuelta atrás

PLANT		
Ruta	**Método**	**Descripción**
/plants	GET	Devuelve todas las plantas para visualización en la página de las tarjetas.
/planst/lister-plants /planst/plantas /planst/list	GET	Devuelve el listado de todas las plantas-
/plants/plantas/:id /plants/list/:id	GET	Captura toda la información de una planta pasada por parámetro.
/plants	POST	Insertar una nueva planta a la BBDD
/plants/update	POST	Actualiza una planta existente en la BBDD
/plants/delete	POST	Elimina una planta existente en la BBDD
/plants/fiter	POST	Devuelta la planta o listado de plantas que cumplan con los requisitos indicados en el filtro: categoría, regex, orden alfabético
/plants/fiter-admin	POST	Devuelta la planta o listado de plantas que cumplan con los requisitos indicados en el filtro: regex, orden alfabético

SUPPLIER		
Ruta	**Método**	**Descripción**
/suppliers/list	GET	Devuelve todos los proveedores que se mostrarán en la página de proveedores solo visible para administrador
/suppliers/lister	GET	Devuelve todos los proveedores para mostrar en las páginas donde se indique.
/suppliers	POST	Insertar una nueva planta a la BBDD
/suppliers/update	POST	Actualiza los datos de un proveedor existente.
/suppliers/delete	POST	Elimina los datos de un proveedor existente.
/suppliers/filter	GET	Devuelve un único o un listado de proveedores que cumplan con los requisitos indicados en el filtro: regex, orden alfabético

DESPLIEGUE

Cuando se termina de desarrollar una aplicación web, no tiene sentido que se quede únicamente en nuestro ordenador. El objetivo final de haber "gestado" una aplicación web es publicarla para que cualquier potencial usuario la use. Para que esto ocurra, debemos realizar lo que se conoce como despliegue de aplicaciones web. En otras palabras, es necesario configurar y ejecutar la aplicación en una máquina que tenga suficientes recursos para que cualquier persona pueda acceder a ella a través de una URL. Esto implica transferir todo el trabajo realizado por el desarrollador a dicha máquina, de modo que pueda estar disponible en línea para los usuarios..

El despliegue de aplicaciones implica trabajar en el código de la aplicación y realizar configuraciones para que esté disponible en una máquina accesible desde cualquier lugar. Esto se puede lograr utilizando un servidor físico conectado a Internet en nuestra "oficina", o mediante soluciones en la nube como AWS de Amazon o Azure de Microsoft. El despliegue también varía según el lenguaje de programación utilizado, requiriendo diferentes herramientas en cada caso. En los siguientes subapartados se dará a conocer dónde y pasos que vamos a seguir para realizar el despliegue de *Plantarium*.

RENDER

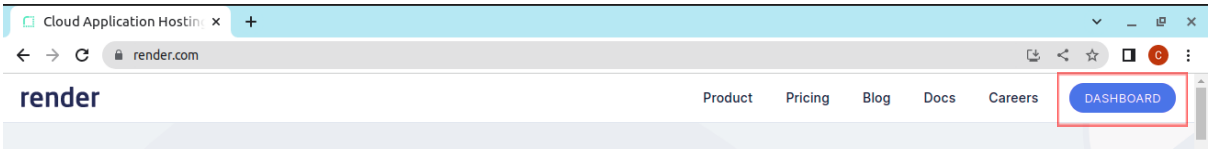
El servicio usado para realizar el despliegue de nuestra aplicación web es Reder.com, la cual cuenta con numerosas ventajas. Simplifica la implementación con una interfaz intuitiva y fácil de usar, así como brindar soporte para una amplia variedad de lenguajes y frameworks. Destaca por su escalabilidad automática, permitiendo que nuestras aplicaciones puedan manejar un elevado pico de tráfico sin complicaciones ni configuraciones adicionales. Además de garantizar una alta disponibilidad mediante enrutamiento inteligente y duplicación automática de instancias. Además, se integra sin problemas con servicios como bases de datos y almacenamiento en la nube, brindando una solución completa para tu aplicación web.

MANUAL DE DESPLIEGUE

Para realizar el despliegue, hemos necesitado acceder al siguiente enlace (https://render.com/) y seguir los siguientes pasos.

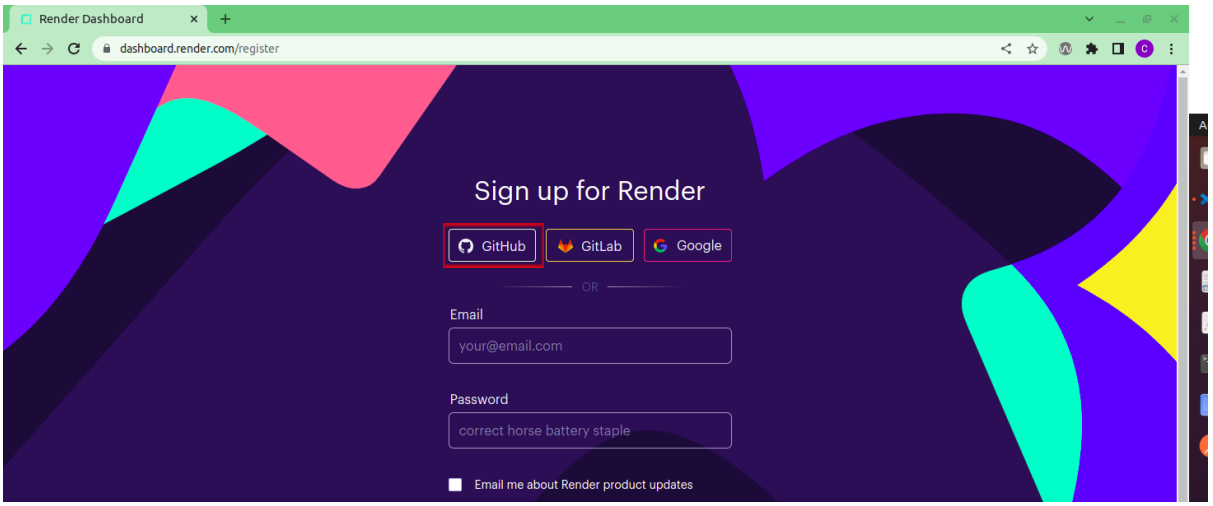
PASO 1 - Creamos nuestro Dashboard

Una vez estemos en la página principal de Render, debemos de cliquear en el botón de Dashboard

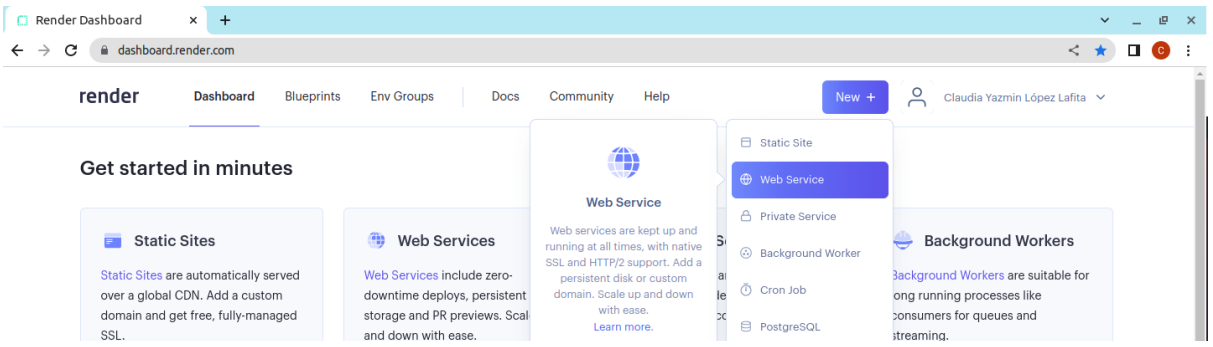


PASO 2 - Vincular nuestra plataforma.

Es necesario que creemos nuestra dashboard vinculando directamente en con la plataforma donde tengamos el repositorio dado que le sea más fácil acceder a él. En nuestro caso lo vinculamos con GitHub.

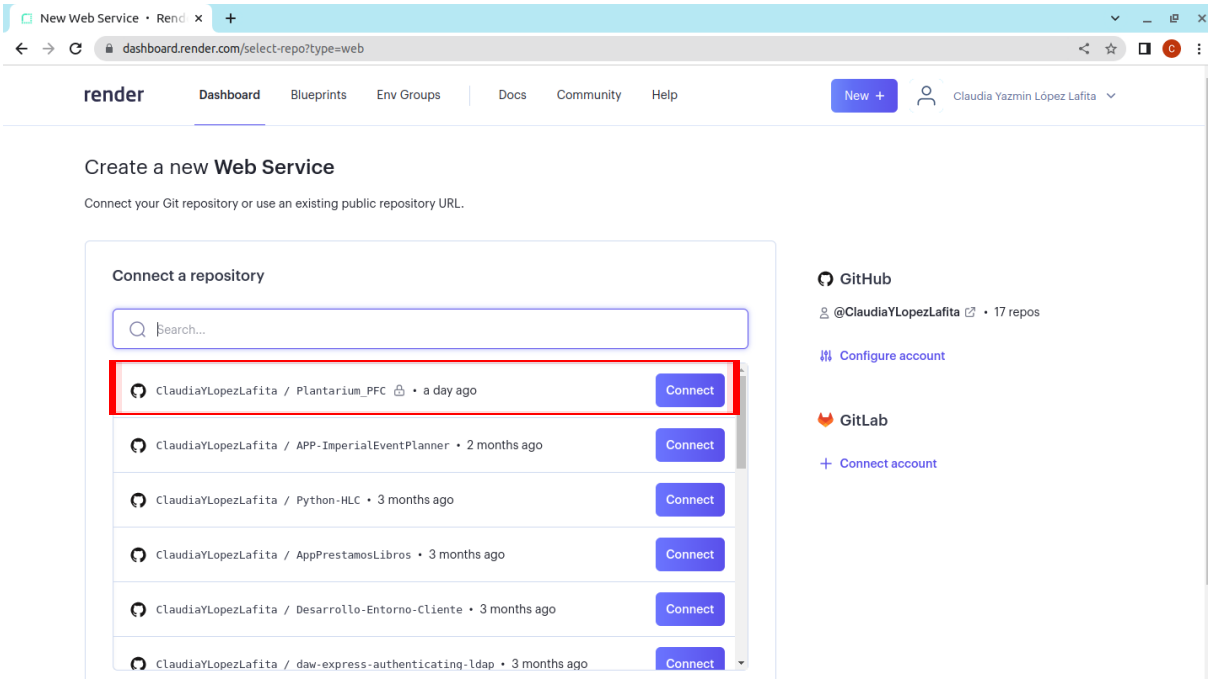


Dentro de nuestro dashboard debemos de indicar como se ve en la imagen un nuevo servicio web.



PASO 3 - Selección de proyecto a desplegar

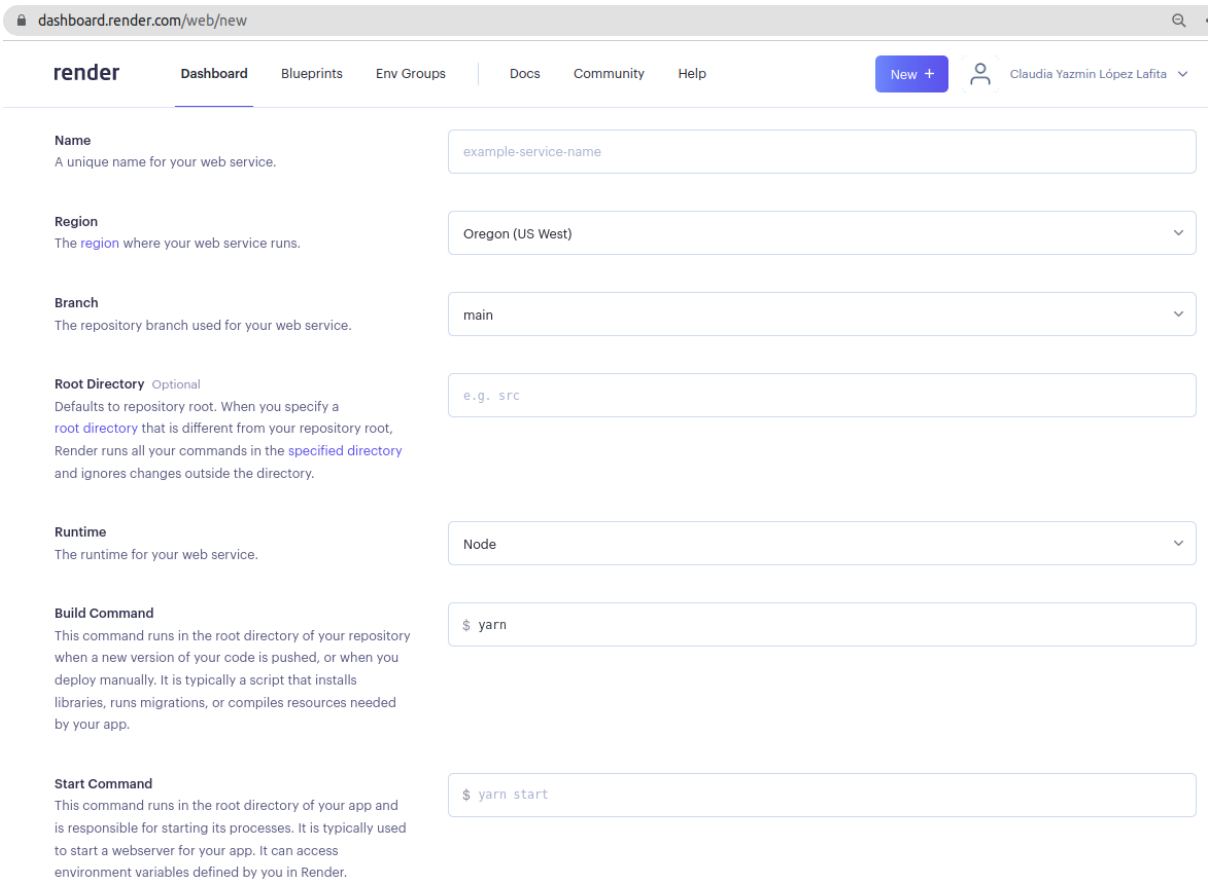
Al tener vinculado el dashboard a nuestro perfil de GitHub, podemos visualizar todos los repositorios que tenemos y por lo tanto simplemente debemos de darle al botón de Connect que se encuentra al lado.



PASO 4 - Datos obligatorios para el despliegue

Para poder realizar el despliegue correctamente es necesario que se rellene el siguiente formulario, para nuestro caso, con los siguientes datos:

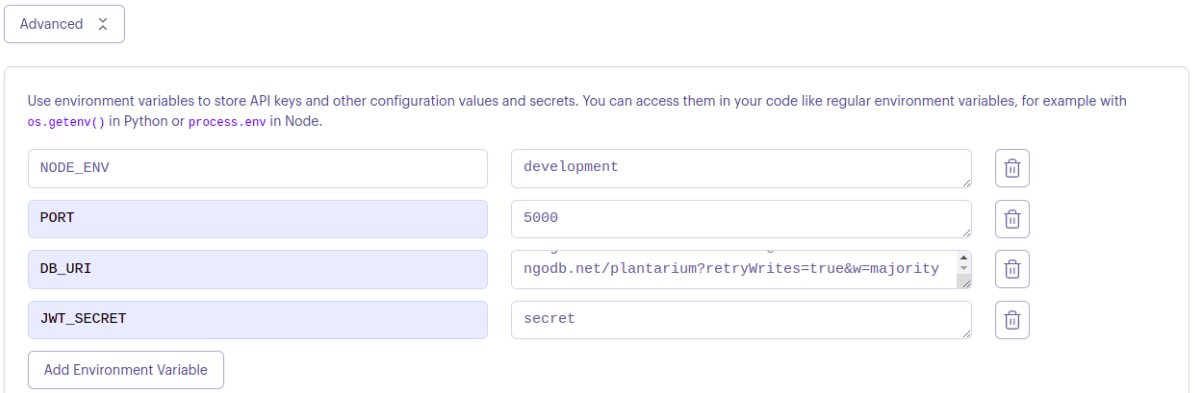
- **Name**: El nombre que quieras darle a tu aplicación.
- **Region**: Es mejor elegir la región más próxima a tu ubicación, en nuestro caso es: ***Frankfurt (EU Central)***
- **Branch**: La rama que queremos desplegar, en nuestro caso es **main**.
- **Root Directory**: Nuestra carpeta raíz que sería **src**.
- **Environment**: Seleccionamos Node.
- **Build Command**: Escribimos ***npm install*** (este script instala las dependencias).
- **Start Command**: Escribimos ***npm run start:dev*** (levanta la aplicación una vez que se haya hecho la build).



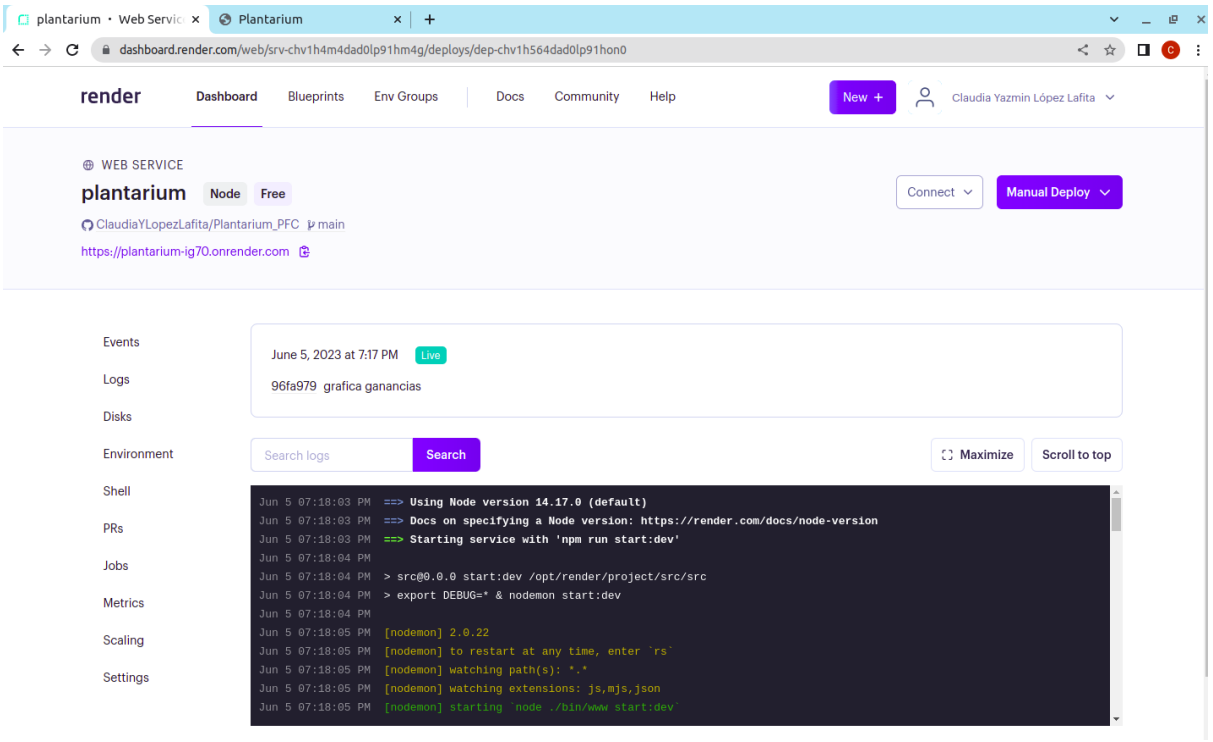
Elegimos el tipo de instancia para nuestro despliegue, que sería gratuita; este formato nos proporciona 500h/mes.

PASO 5 - Datos necesario para el despliegue

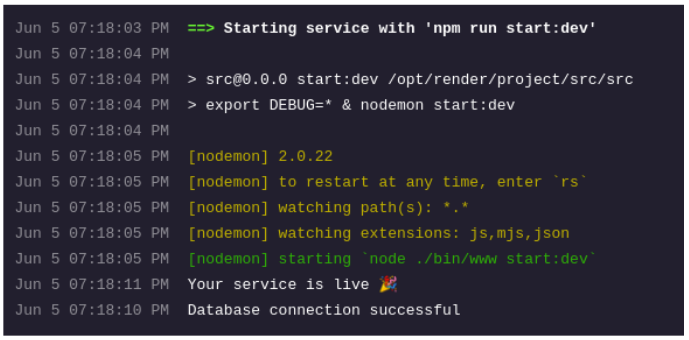
En caso de que en el proyecto se esté usando variables de entorno, como es nuestro caso, es necesario que se definan. Para eso tras la selección de de tipo de instancia clicamos en Advances y se desplegará una zona donde podemos indicar la llave y el valor de estas variables.



Una vez definida podemos irnos al final de la página y presionar el botón: **Create Web Service.**

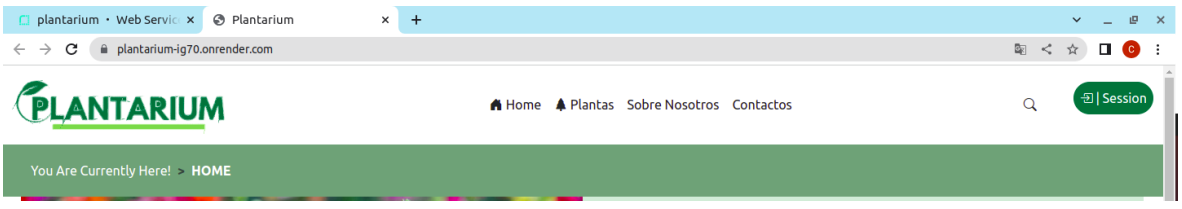


En consola podemos ver cómo se van instalando las dependencias, y cómo se va levantando la aplicación. En caso de ser todo correcto nos saldrá por por consola el mensaje que establecimos en nuestra app.js para que nos indique que el servicio se ha levantado correctamente (“ ”):



PASO 6 - Accedemos a nuestra aplicación

Una vez ya desplegada podemos acceder a la aplicación cliqueando en la URL que nos proporciona Rende.



CONCLUSIÓN

El proyecto "Plantarium" es una aplicación desarrollada con el objetivo de ofrecer información, cuidados y tratamientos para plantas ornamentales. Los usuarios pueden acceder a una enciclopedia digital de plantas, encontrar proveedores y suscribirse para tener un jardín virtual personalizado. Además, se proporciona un servicio de enfermería virtual para diagnosticar y tratar enfermedades de las plantas. En resumen, "Plantarium" brinda a los amantes de las plantas una herramienta completa para el cuidado exitoso de sus especies.

Durante el desarrollo de este proyecto, hemos adquirido habilidades sólidas en la creación de aplicaciones web utilizando tecnologías como Node.js, Express y Mongoose. También hemos trabajado con plantillas EJS para generar vistas dinámicas y atractivas. Nuestra experiencia en el desarrollo de "Plantarium" ha sido invaluable para mejorar nuestras habilidades de programación, especialmente en el manejo eficiente de bases de datos y la implementación de rutas en el framework Express. Además, hemos adquirido conocimientos sobre la integración de servicios de suscripción y hemos aprendido a construir interfaces interactivas que brindan a los usuarios una experiencia fluida y agradable.

En resumen, este proyecto nos ha permitido ampliar nuestro conjunto de habilidades y fortalecer nuestra comprensión de la creación de aplicaciones web eficientes y atractivas.

ANEXO

Anexo I: índice MockUps de la aplicación web.

MK_G_Página_Inicio

MK_G_Inicio_Sesión

MK_G_Contacto

MK_G_Sobre_Nosotros

MK_G_Perfil_Planta_Información

MK_G_Perfil_Planta_Cuidados

MK_G_Perfil_Planta_Síntomas

MK_G_Perfil_Planta_Proveedores

MK_G_Perfil_Planta_Pop_Up1

MK_G_Perfil_Planta_Mapa

MK_G_Plantas

MK_UR_Perfil

MK_UR_Planta_Información

MK_UR_Perfil_Planta_Cuidados

MK_UR_Perfil_Planta_Síntomas

MK_UR_Perfil_Planta_Proveedores

MK_UR_Perfil_Planta_Pop_Up2

MK_UR_Perfil_Planta_Pop_Up3

MK_G_Perfil_Planta_Pop_Up4

MK_UR_Perfil_Detalle

MK_UR_Perfil_Edición

MK_UR_Perfil_Edición_Pop_Up

MK_UR_Perfil_Edición_Actualizado

MK_UR_Perfil_Edición_Pop_Up

MK_UR_Perfil_Subscripcion_Pop_Up

MK_UR_Jardin_General

MK_UR_Jardin_Premium

MK_A_Perfil

MK_A_Accion_Plantas

MK_A_Planta_Pop_Up1

MK_A_Nueva_Planta1

MK_A_Nueva_Planta1_Datos

MK_A_Nueva_Planta2

MK_A_Nueva_Planta2_Datos

MK_A_Nueva_Planta3

MK_A_Nueva_Planta3_Datos

MK_A_Nueva_Planta4

MK_A_Nueva_Planta4_Datos

MK_A_Edicion_Planta

MK_A_Proveedores

MK_A_Proveedores_Pop_Up1

MK_A_Nuevo_Proveedores

MK_A_Nuevo_Proveedores_Datos

MK_A_Edicion_Proveedores

MK_A_Panel_Control_Estadísticas

MK_A_Estadisticas_Plantas

MK_A_Estadisticas_Ganancias

MK_A_Estadisticas_Suscripcione