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 (previamente añadida al jardín)
RC-8	Consulta de síntomas de un planta determinada	Un usuario registrado puede consultar los síntomas de una planta (previamente añadida al jardín)
RC-9	Consulta de enfermedad	Un usuario registrado puede consultar las enfermedades determinada por un síntoma de una planta (previamente añadida al jardín)
RC-10	Consulta de tratamiento	Un usuario puede consultar el tratamiento determinado por una enfermedad de una planta (previamente añadida al jardín)
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	Registro de un usuario premium	Un usuario puede suscribirse con privilegios registrándose mediante un formulario y pagando una cuota mensual de 5.95€
RC-14	Baja de un usuario	Un usuario puede eliminar su cuenta, eliminando sus datos en cualquier momento
RC-15	Modificar suscripción	Un usuario registrado podrá cambiar el tipo de suscripción en cualquier momento, modificándose su jardín convenientemente
RC-16	Añadir una planta	Un usuario administrador puede añadir una planta
RC-17	Eliminar una planta	Un usuario administrador puede eliminar una planta
RC-18	Modificar una planta	Un usuario administrador puede modificar una planta
RC-19	Visualizar proveedor	Un usuario registrado podrá visualizar la información de proveedores en cada planta.

RC-20	Añadir proveedor	Un usuario administrador podrá añadir proveedores a cada planta
RC-21	Modificar proveedor	Un usuario administrador podrá modificar la información de un proveedor
RC-22	Eliminar proveedor	Un usuario administrador podrá eliminar de una planta su proveedor (-es)
RC-23	Visualizar estadísticas	Un usuario administrador podrá visualizar gráficas de estadísticas.
RC-24	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 página Jardineriaplantasyflores.com se centran más en publicar artículos sobre cuidados de jardinería y paisajismo, pero también disponen de una amplia biblioteca de fichas de información sobre muchísimos tipos de plantas de todo el mundo. Así como JardineriaOn.com, lleno de consejos y tutoriales para el mantenimiento de plantas y jardines, además del cuidado de diferentes tipos de flores en particular, información sobre paisajismo, semillas, huertos y maquinaria para jardinería.

Otra página a tener en cuenta es MassóGarde.com, encargada de producir productos para el cuidado y tratamiento de enfermedades y plantas. La búsqueda de más aplicaciones web donde encontrar información sobre enfermedades y tratamiento no es tan fructífera como los cuidados de plantas, se centran principalmente en la sección de blog, donde los dueños exponen sus experiencias, como es el caso de verdecora.es.

Si queremos encontrar información sobre tratamiento nos debemos mover por el mundo de las aplicaciones móviles, donde encontramos una gran variedad de aplicaciones dedicadas al cuidado de plantas o como herramienta de ayuda para el riego.

La primera aplicación a la que podemos considerar "competencia" 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. También podemos destacar Enfermedades de las Plantas, esta aplicación contiene información copy-paste de Wikipedia sobre las principales plagas o enfermedades que pueden afectar a los cultivos clasificada por orden alfabético, lo cual complica el tratamiento de la plaga si no sabemos de antemano cuál es.

Una aplicación no tan especializada en enfermedades y plagas, pero que nos hace competencia más directamente es Infojardin, es una aplicación bastante completa sobre todo para consultas de árboles y plantas de jardín. La 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 una disciplina más compleja de lo que muchas personas 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. El principal inconveniente que podemos encontrarnos es que estas personas suelen tener conocimientos de cuidados; pero en caso de ser una enfermedad, puede darse el caso de no tenerlos. Esto nos lleva a la segunda alternativa: búsqueda en internet.

En internet podemos encontrar un sinfín de opciones donde podemos encontrar 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 que buscamos; como podemos ver en el

apartado de alternativas de 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.

Motivos secundarios, pero igual de importantes son: aprender, el uso de tecnologías que no había usado antes, para obtener amplios conocimientos sobre estas y conocer posibles aplicaciones de estas para futuros proyectos. Realizar un aporte a la comunidad, siempre me han gustado las plantas y se lo frustrante que resulta la búsqueda de información y quiero aportar mi pequeño granito de arena; además de también tener beneficio en gran medida de esto.

STACK TECNOLÓGICO

En este apartado se enumeran todos los servicios tecnológicos, sistemas, herramientas y componentes utilizados para crear y ejecutar nuestra aplicación web. Incluiremos los lenguajes 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.

LENGUAJE DE PROGRAMACIÓN

En el desarrollo de nuestra aplicación web, hay diferentes elementos que debemos de integrar para contemplar un producto final interesante. Para hacer esto, usaremos el lenguaje de programación JavaScript; el cuál es un lenguaje interpretado, dialecto del estándar ECMAScript.

El porqué usar JavaScript consiste en la necesidad de integrar elementos de interactividad, animación y movimiento de datos, entre otros elementos de nuestra página web. Entre las muchas características de JavaScript y las que nos han hecho escoger este lenguaje son:

Permite que nuestra aplicación funcione en cualquier tipo de navegador, ya sea Chrome, Firefox o Safari. Dado su **popularidad** entre el sector de los programadores en la actualidad, hay un elevado número de personas interesadas en compartir su conocimiento, además de publicar sus propios proyectos en repositorios remotos; esto nos permite tener el acceso a la información que nos pueda ser útil. Una de las ventajas más importantes de JavaScript es que trabaja en la **parte del cliente**, por lo que nos ahorra en ancho de banda y por otro lado, acelera la ejecución del programa de código y del sitio web.

LADO BACK-END

Para que nuestra aplicación web funcione, se requiere mucha información y datos que se almacenan en «la parte trasera» del sistema informático. De igual forma se encarga de optimizar otros elementos y recursos como la seguridad y privacidad en un sitio web o aplicación.

Para el desarrollo del lado servidor de nuestro proyecto usaremos el Framework Express.js, el cual se usa en el entorno de ejecución de Nodejs. Primero que nada se debe dejar claro que Node.js no es ni una plataforma para desarrollar aplicaciones ni tampoco es un lenguaje de programación; es un entorno para ejecutar JavaScript fuera del navegador.

Este entorno de ejecución que se basa en eventos asíncronos y en la utilización del motor V8 de Google lo convierte en unos de los entornos con más crecimiento en la actualidad para la creación de aplicaciones web o de escritorio. Además, gracias a la capacidad de poder procesar varias conexiones de forma simultánea le proporciona una gran **escalabilidad**. Se considera un entorno **rápido** y **eficaz** gracias a la ejecución de procesos sin bloqueo y la utilización de menos recursos; y una gran **comunidad**, lo que permite que se mantenga y mejore.

Para agilizar el proceso de desarrollo de nuestro proyecto, usaremos Express.js como Framework, este nos permite agilizar los procesos de desarrollo ya que nos evita tener que escribir código de forma repetitiva, y nos asegura unas buenas prácticas así como la consistencia del código. Se utiliza para una amplia gama de acciones en el ecosistema Node.js, por lo tanto presenta las ventajas del propio entorno de ejecución así como la siguientes características:

Presenta un sistema de **enrutamiento** potente y robusto incorporado por defecto que nos asiste en el desarrollo. Comprende una serie de **middleware** para crear un proceso de desarrollo sin fisuras; gracias a ellos podremos introducir scripts para interceptar el flujo de la aplicación

LADO FRONT-END

El frontend sirve para realizar la interfaz de un sitio web, desde su estructura hasta los estilos, como pueden ser la definición de los colores, tipografías, secciones, etc. Su uso es determinante para que los visitantes tengan una buena experiencia dentro de nuestra aplicación y deseen usarla continuamente. Como hemos mencionado, son los elementos y componentes visibles para los usuarios, y utilizan lenguajes de diseño como CSS, HTML y JavaScript.

En nuestro proyecto usaremos **EJS** (Embedded JavaScript Templating), el cual es uno de los motores de plantillas más populares en JavaScript. Sabiendo que un motor de plantillas se define como un software diseñado para combinar una plantilla con un modelo de datos, lo usaremos para interpolar datos en código HTML. Entre las características que nos han hecho escoger esta tecnología están:

Una de las razones para elegirlo es que el código EJS parece **HTML puro**; ello será mucho más fácil de vincularlo con Bootstrap; el cual es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Los errores de EJS son **fáciles de depurar** dado que son simples excepciones de JavaScript, con números de línea de plantilla incluidos. Y no menos importante es la gran **comunidad** de usuarios activos y la biblioteca está en desarrollo activo.

BASE DE DATOS

La base de datos es esencial para almacenar datos que mostraremos en nuestro servicio, así como aquellos que permiten el funcionamiento correcto del mismo. A partir de ahí, la aplicación puede adaptarse y optimizarse.

Para nuestra base de datos usaremos MongoDB, un sistema de base de datos NoSQL. Los principales motivos para utilizar MongoDB en nuestro proyecto son:

Ofrece una gran **flexibilidad**, debido a que al contrario que las bases de datos relaciones, no siguen ningún esquema que controle cómo se almacena la información, qué tipo de datos se almacenan, etc. Por lo que los campos pueden variar entre documentos y la estructura de datos puede cambiar con el tiempo. Presenta una gran **escalabilidad**, tanto vertical como horizontal; entendiendo como escalabilidad la necesidad de que nuestra base de datos se vaya adaptando a medida que se vá guardando más datos. Esto permite que el rendimiento sea mucho mejor para la aplicación. Gracias a la buena **documentación** oficial que tiene, durante el desarrollo del proyecto se podrá usar como gran recurso para ir mejorando las operaciones que se deben realizar, cómo están documentadas, cómo se usan, etc.

Además de las razones anteriores, que son vistas desde el punto de vista técnico, vamos a destacar otras razones que nos han hecho escoger MongoDB:

Se trata de un software de **código abierto**, que evoluciona continuamente gracias a su gran comunidad. Como mencionamos anteriormente desarrollamos el proyecto con JavaScript y Node.js; los cuales son **complementos** para esta base de datos. Es una base de datos **fácil** de **aprender** así como una herramienta **gratuita** que podemos incorporar en el proyecto sin necesidad de pagar el uso de licencia.

MODELO ARQUITECTÓNICO DEL SOFTWARE

En las primeras aplicaciones se han usado arquitectura monolítica y, aunque se han desarrollado alternativas más sofisticadas, siguen teniendo ventajas importantes; como por ejemplo, al tener un desarrollo centralizado nos facilita tener el código fuente muy localizado; ya que se encuentra en un único lugar. Así mismo, desde el punto de vista de la lógica empresarial proporciona simplicidad en la aplicación, ya que si se necesitan otros datos para nuevas características, ya se encuentran en un mismo paquete. El despliegue y la ejecución son simples; no se requieren herramientas sofisticadas de automatización para desarrollar y desplegar la aplicación.

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:

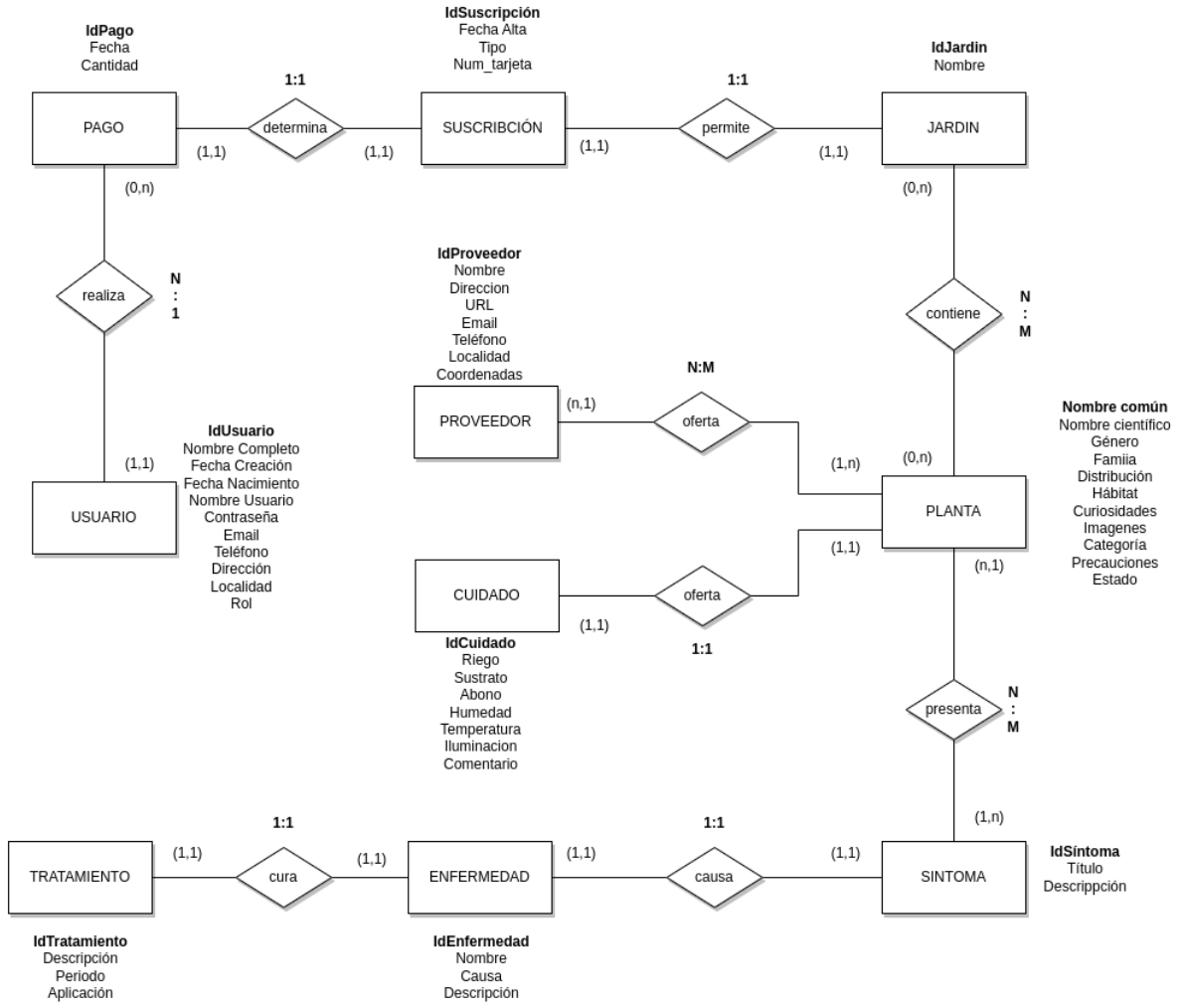

Imagen 1. Diagrama Entidad-Relación. Representación gráfica de la base de datos de Plantarium generada mediante el programa Diagrams. Autoría: Claudia Y. López Lafita.

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, required: true, index:{unique: true}}
/* Nombre de usuario, será el identificador de tipo cadena y obligatorio, que el usuario rellene cuando se registre en la página y no esté ya almacenado en la base de datos. */
password: {type: String, required: true}
/* Contraseña del usuario, siendo campo obligatorio de tipo cadena que se guardará en la base de datos de manera encriptada*/
creationdate: { type: Date, default: Date.now }
```

/* Fecha de alta del usuario, será por defecto la fecha actual y de tipo Date*/

fullName: {type: String, required: true}

/* Nombre “verdadero” del usuario, campo obligatorio de tipo cadena*/

email: {type: String, required: true}

/*Correo electrónico del usuario, atributo de tipo cadena que será obligatorio*/

birthdate: {type: Date, required: true}

/* Fecha de alta del usuario, la cuál será por defecto la fecha actual*/

address: {type: String, required: true}

/* Fecha de nacimiento del usuario, la cuál será obligatorio y de tipo Date*/

locality: {type: String, required: false}

/* Localidad actual del usuario, atributo de tipo cadena que no es obligatorio, pero sí necesario para la ubicación de proveedores*/

phone: {type: number, required: false}

/* Teléfono del usuario, atributo numérico no obligatorio*/

role: {type: String, enum: ['admin', 'subscriber'], default: 'subscriber'}

/* Rol del usuario, que solo podrá tener dos valores dependiendo de cómo interactúa con la aplicación; será por defecto suscriptor. Atributo de tipo cadena */

payments: [{type: Date, ref: ‘Pay’, default: null}]

/* Array de fechas de cada pago que realice el usuario, por defecto estará vacío dado que el pago no es obligatorio*/

PAY:

codPay:{type:String, required: true, index:{unique: true}}

/*Código de pago que será el identificador, de tipo cadena obligatorio*/

date: {type: Date, required: true, index:{unique: true}, default: Date.now}

/* Fecha de cobre del pago, tipo date que tendrá por valor la fecha en la cual un usuario paga la suscripción premium*/

amount: {type: Float, required: true, default: 5.95}

/* Cantidad a cobrar al usuario con suscripción premium, que será de tipo float y por defecto será 5.95€ */

subscription:{type:String, ref: 'Subscription', required: true}

/* Número de suscripción a la que está asociado el pago, hace referencia a la entidad suscripción por lo tanto es de tipo de cadena y obligatorio*/

SUBSCRIPTION:

numSubscription: {type: String, required: true, index:{unique: true}}

/* El identificador de la entidad de tipo cadena será único para cada usuario, se creará aleatoriamente cuando se dé de alta.*/

date: {type: Date, required: true, default: Date.now}

/* Fecha de suscripción que coincide principalmente con la fecha de alta del usuario, pero que puede cambiar si este cambia el tipo. Es de tipo Date y por defecto es la fecha actual*/

type: {type: String, enum: ['premium', 'general'], default: 'general'}

/* Tipo de suscripción que elige el usuario entre dos opciones, es un atributo de tipo cadena que por defecto tendrá el valor de general.*/

numCard:{type: number, required:false}

/* Número de la tarjeta de crédito por donde se le cobrará el usuario, es de tipo numérico y es obligatorio solo en caso de escoger el tipo de suscripción premium*/

garden:{type:String, ref: 'Garden', required: true}

/* Código del jardín que se genera cuando el usuario se suscribe a la aplicación web, es de tipo cadena y requerido*/

GARDEN:

codGarden: {type: String, required:true, index:{unique:true}}

/* Identificador de la entidad que será único, generado aleatoriamente, obligatorio y de tipo cadena*/

name: {type: String, required: false, default: randomString}

/*Nombre del jardín dado por el usuarios que será opcional y de tipo cadena*/

plants: [{type: String, ref: 'Plant', default: null}]

/*Array de los nombre científicos de las plantas que el usuario irá añadiendo*/

PLANT:

codPlant: {type: String, required:true, index:{unique:true}}

/*Código de la planta que será un atributo de tipo cadena que es obligatorio y que será el identificador de la planta*/

sciName:{type: String, required:true, index:{unique:true}}

/* Nombre científico de la planta que será el identificador de la entidad, atributo de tipo cadena que es obligatorio*/

comName: {type: String, required: true}

/* Nombre común de la planta que será obligatorio y de tipo cadena*/

genus: {type: String, required: true}

/* Género de la planta que será obligatorio y de tipo cadena*/

family: {type: String, required: true}

/* Familia de la planta que será obligatorio y de tipo cadena*/

distribution: {type: String, enum: [‘Cosmopolita’, ‘Endémico’], default: ‘Cosmopolita’}

/* Distribución de la planta, solo puede tener dos valores: cosmopolita o endémico y por defecto será cosmopolita. Atributo de tipo cadena*/

habitat: {type: String, required: true}

/* Campo que nos informa del hábitat de la planta, es obligatorio y de tipo cadena*/

curiosities: {type: String, required: false}

/* Campo que nos informa de la ecología de la planta, no es obligatorio y de tipo cadena*/

precautions: {type: String, required: false}

/* Campo optativo donde se indicarán precauciones que se ha de tener con esta planta. Es de tipo cadena*/

category: {type: Array, required: true}

/* Array donde se guardaran las categorías que se le asigne a la planta, es un atributo obligatorio*/

images: {type: Array, required: true}

/* Array con las url de la ruta de la carpeta donde se localizan las imágenes que se mostrarán en las fichas de cada una de las plantas, será con una capacidad de 6 imágenes. Atributo obligatorio*/

status:{type: String, enum: ['Visible', 'No visible']}

/*Atributo que nos indica si la planta está visible o no para los usuarios, es obligatorio y por defecto es 'No visible'*/

gardens: [{type: String, ref: 'Garden', default: null}]

/* Array de referencia, donde se encontrará el código de los jardines donde se localiza la planta. */

suppliers: [{type: String, ref: 'Garden', default: null}]

/* Array de referencia, donde se encontrará el código de los proveedores donde se puede comprar la planta*/

ATTENDANCE:

codAttendace: {type: String, required:true, index:{unique:true}}

/* Identificador de la entidad que será único, generado aleatoriamente, obligatorio y de tipo cadena*/

water: {type: String, enum: ['Poco frecuente', 'Frecuente', 'Muy Frecuente'], default: 'Frecuente'}

/* Frecuencia de riego, de los tres posibles valores que puede tomar el valor por defecto será la de 'Frecuente', atributo de tipo cadena obligatorio*/

soil: {type: String, required: true}

/* Tipo de sustrato que necesita la planta, es un atributo de tipo cadena y obligatorio*/

compost: {type: String, required: true}

/* Tipo de abono complementario que necesita la planta para su desarrollo, es un atributo de tipo cadena y obligatorio*/

moisture: {type: String, required: false}

/* Humedad a la cual se debe de mantener la planta, es un atributo de tipo cadena obligatorio*/

temperature: {type: String, required: true}

/* Rango de temperatura a la cual se debe de mantener la planta, es un atributo de tipo cadena obligatorio*/

lightning: {type: String, enum: [‘Alta’, ‘Moderada’, ‘Baja’], default: ‘Moderada’}

/* Grado de iluminación que debe de tener la planta para un desarrollo óptimo, sólo podrá escoger entre tres opciones y por defecto será moderado; atributo de tipo cadena*/

comment: {type: String, required: false}

/* Campo donde se especificarán los cuidados especiales, si es necesario, de las plantas. No es obligatorio y es de tipo cadena.*/

plant: {type:String, ref: ‘Plant’, required: true}

/* Referencia del identificador de la planta a la que corresponde el cuidado*/

SYMPTOM:

codSymptom: {type: String, required: true, index:{unique:true}}

/* Código identificativo único para cada síntoma, es de tipo cadena y obligatorio*/

title: {type: String, required: true}

/* “Título” o nombre del síntoma, es un atributo de tipo cadena y obligatorio*/

description: {type: String, required: true}

/* Campo donde se describe el síntoma como complemento de las imágenes. Es obligatorio y es de tipo cadena.*/

disease: {

codDisease: {type: String, required: true, index:{unique:true}}

/* Código identificativo único para la enfermedad, es de tipo cadena y obligatorio*/

name: {type: String, required: true}

/* “Título” o nombre del síntoma, es un atributo de tipo cadena y obligatorio*/

cause: {type: String, enum: [‘Cuidados erróneos’, ‘Patógeno’]}
/* Campo que indica la causa de la enfermedad, si son por los cuidados o son por patógenos. Es tipo cadena obligatorio*/
description: {type: String, required: true}
/* Campo donde se describe la enfermedad. Es obligatorio y es de tipo cadena.*/
treatment:{
codDisease:{type: String, required: true, index:{unique:true}}
/* Código identificativo único para el tratamiento, es de tipo cadena y obligatorio*/
period: {type: String, required: true}
/* Campo donde se especifica el periodo de tiempo durante el cual se ha de realizar el tratamiento. Es un atributo de tipo cadena obligatorio*/
application: {type: String, required: true}
/* Campo donde se especifica el método de aplicación del tratamiento. Es un atributo de tipo cadena obligatorio*/
comment: {type: String, required: false}
/* Campo donde se describe la enfermedad. Es obligatorio y es de tipo cadena.*/
}
/* Documento embebido en enfermdad, que corresponde el tratamiento de la enfermedad donde se localiza, es obligatorio*/
}
/* Documento embebido en síntoma, que corresponde con la enfermedad que causa el síntoma, es obligatorio*/
plants: [{type:String, ref: ‘Plant’, required: true}]
/* Referencia del identificador de las plantas que pueden tener este síntoma*/

SUPPLIER:

codSupplier: {type: String, required: true, index:{unique:true}}

/* Código identificativo único para cada proveedor, es de tipo cadena y obligatorio*/

name: {type: String, required: true}

/* Nombre empresarial del proveedor, atributo de tipo cadena obligatorio*/

address: {type: String, required: true}

/* Dirección del proveedores, es de tipo cadena y obligatorio*/

email: {type: String, required: true}

/* Correo electrónico del proveedores, es de tipo cadena y obligatorio*/

url: {type: String, required: true}

/* Url de la página web del proveedor, es un campo de tipo cadena no obligatorio, pero sí necesario para la generación de un acceso directo*/

phone: {type: Number, required: false}

/* Número de contacto del proveedor. Es un atributo de tipo numérico no obligatorio*/

locality: {type: String, required: true}

/* Localidad donde se ubica el proveedor. Es un atributo de tipo cadena no obligatorio*/

coordinates:{ type: Point, required: true}

/* Coordenadas del proveedor obligatorias para la representación en un mapa, es un atributo de tipo point que es obligatorio*/

plants: [{type: String, ref: 'Plant', default: null}]

/* Array de los nombre científicos 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.