

# APP GESTICOLMENAR

JOSÉ VICENTE FALCÓ MILLA  
DESARROLLO DE APLICACIONES WEB  
2º CURSO - MARZO/JUNIO





# Índice

|  |    |
|--|----|
| Agradecimientos.....                         | 5  |
| Resumen.....                                 | 5  |
| Introducción.....                            | 6  |
| Estudio de viabilidad.....                   | 7  |
| Análisis DAFO.....                           | 7  |
| Estudio de mercado.....                      | 9  |
| Viabilidad del proyecto.....                 | 9  |
| Monetizar la aplicación.....                 | 10 |
| Análisis de requisitos.....                  | 11 |
| Descripción de requisitos.....               | 11 |
| Caso de uso general.....                     | 12 |
| Diagrama ER.....                             | 13 |
| Paso a tablas.....                           | 14 |
| Users.....                                   | 14 |
| Places.....                                  | 15 |
| Apiaries.....                                | 16 |
| Queens.....                                  | 17 |
| Beehives.....                                | 18 |
| Products.....                                | 19 |
| Diseases.....                                | 20 |
| Mockups.....                                 | 21 |
| Login Desktop.....                           | 21 |
| Registro Desktop.....                        | 21 |
| Vista principal/Colmenares Desktop.....      | 22 |
| Site Map.....                                | 23 |
| Codificación.....                            | 24 |
| Tecnologías elegidas y su justificación..... | 24 |
| Entorno servidor.....                        | 24 |
| Entorno cliente.....                         | 25 |



## Agradecimientos

A la razón de todo esto, a mi fiel amigo, Noble.

## Resumen

Aplicación Web destinada al manejo de la información relacionada con la apicultura. Poder almacenar, gestionar y visualizar los datos que permitan la correcta gestión, control y administración de los distintos colmenares pertenecientes a un apicultor. Aplicación desarrollada con el framework Laravel (PHP) junto con otras tecnologías tales como SQL, Bootstrap, HTML, CSS .

Aplicació Web destinada a tractar la informació relacionada amb l'apicultura. Poder emmagatzemar, gestionar i visualitzar les dades que permetin la correcta gestió, control i administració dels diferents ruscs pertanyents a un apicultor. Aplicació desenvolupada amb el framework Laravel (PHP) juntament amb altres tecnologies com SQL, Bootstrap, HTML, CSS.

Web application for managing information related to beekeeping. Being able to store, manage and visualize the data that allows the correct management, control and administration of the different apiaries belonging to a beekeeper. Application developed with the (PHP) Laravel framework together with other technologies such as SQL, Bootstrap, HTML, CSS.

## Introducción

Comenzaré la introducción con una pregunta retórica... ¿cuán importante crees que es la vida de las abejas en el planeta tierra?

Sí en efecto, son vitales, sin más. Así mismo, la apicultura, además produce una serie de productos beneficiosos para el ser humano, tales como la miel, edulcorante milenario, el polen y la apitoxina entre otros.

Bueno, pues vamos a explicar el porqué se realiza una aplicación web en la que cualquier apicultor pueda tener de forma ordenada y de gestión sencilla, los datos necesarios más importantes que pueda necesitar tener a mano, consultar y/o almacenar.

Con la App Gesticolmenar, el apicultor podrá tener la información más relevante de sus colmenares, de sus colmenas, datos sobre sus abejas reina, enfermedades activas en la colmena, los datos de las ubicaciones y recordatorios de actividades y tratamientos a llevar a cabo.

Toda esta información, organizada, visual y accesible, hace que, podamos llevar en nuestro bolsillo todo, que podamos modificarla in-situ, que podamos obtener gráficas que nos den un punto de vista más visual de ciertos datos, que no tengamos que rebuscar datos de referencias catastrales para otras gestiones o visitas al apiario de autoridades, por ejemplo.

El ahorro en de tiempo y la ausencia de pérdida de datos entre libretas y papeles varios que son el día a día habitual de la práctica totalidad de los apicultores, hace que trabajar con Gesticolmenar, por fin, haga que puedan seguir con sus actividades favoritas sin tener que gestionar archivadores, libretas y anotaciones por doquier.

# Estudio de viabilidad

## *Análisis DAFO*

### **Fortalezas:**

- Gesticolmenar ofrece una solución para los apicultores, ya que les permite gestionar y llevar un seguimiento de sus colmenares y colmenas de una manera más eficiente y fácil.
- Ofrece la posibilidad de visualizar gráficas que permiten analizar la producción de miel, polen o apitoxina.
- La plataforma es accesible desde cualquier lugar con conexión a internet, lo que permite a los usuarios gestionar sus colmenares y colmenas de forma remota.
- Una aplicación visualmente sencilla, poco cargada y a su vez con toda la funcionalidad necesaria hacen que una vez se empieza a usar, sea tan rápido introducir datos y manejarla, que se mantenga su uso.

### **Debilidades:**

- La aplicación web de gestión de colmenares ya tiene cierta competencia en el mercado, por lo que hay que esforzarse en ofrecer una propuesta única y diferenciada para destacar, como las notificaciones de tareas pendientes.
- El mercado de la apicultura es relativamente pequeño y segmentado, lo que puede limitar el crecimiento potencial de la aplicación.
- Los apicultores en su gran mayoría suelen ser gente de cierta edad o jóvenes que han heredado tanto las costumbres como las ideas y procedimientos, estando la tecnología de la información bastante alejada de su día a día, por lo que, existe cierta reticencia a pasar del papel a las app.
- La falta de experiencia en el desarrollo de aplicaciones web puede afectar a la calidad y funcionalidad de la aplicación.



### **Oportunidades:**

- La preocupación por el medio ambiente y la apicultura en particular ha ido en aumento en los últimos años, lo que podría generar una mayor demanda de soluciones digitales para la gestión de colmenares.
- La posibilidad de expandir el negocio hacia otros sectores relacionados con la apicultura, como la venta de productos derivados de la miel, como cremas, jabones y otros productos de cuidado personal.
- Generar ventas cruzadas de productos para el cuidado y mantenimiento de colmenares, desde las colmenas hasta los productos sanitarios y la alimentación

### **Amenazas:**

- La competencia en el mercado puede limitar la adopción de tu aplicación por parte de los usuarios.
- La estacionalidad de la actividad apícola puede generar un impacto en los ingresos del negocio, ya que, puede tener períodos de inactividad.
- La posibilidad de que los apicultores no quieran almacenar cierta información en una aplicación, es una posible causa de reticencia al uso.
- La extinción masiva de las abejas pueden afectar a largo plazo al uso de la aplicación.



## ***Estudio de mercado***

El mercado de la apicultura está compuesto principalmente por pequeños productores y empresas dedicadas a la producción de miel y otros productos apícolas. Existen algunas soluciones digitales para la gestión de colmenares, pero son pocas y generalmente de lengua extranjera, enrevesadas y poco prácticas para el día a día. Hay una oportunidad para ofrecer una solución más accesible y fácil de usar para los pequeños productores y apicultores aficionados.

Gesticolmenar puede ser atractiva para los apicultores, ya que les permite gestionar y llevar un seguimiento de sus colmenares y colmenas de una manera más eficiente y fácil, y ofrecer una mayor transparencia sobre la producción. Para llegar a este mercado, será importante tener una estrategia de marketing adecuada, que permita dar a conocer la aplicación y llegar a los posibles clientes.

## ***Viabilidad del proyecto***

### **Recursos Hardware:**

Para alojar la aplicación web, se necesitará un servidor web. El tipo de servidor dependerá del número de usuarios que se esperan y de la cantidad de datos que se manejen. En un principio, podremos utilizar servicios de alojamiento compartido, pero a medida que crezca la base de usuarios, será necesario invertir en un servidor dedicado o en servicios de nube para garantizar la escalabilidad y el rendimiento. En cualquier caso, ninguna de estas soluciones suponen de una inversión grande.

### **Recursos Software:**

Laravel es un framework de PHP que ayuda a acelerar el desarrollo de la aplicación al proporcionar herramientas y características útiles para el desarrollo web, como autenticación, enrutamiento, manejo de sesiones, etc.

Bootstrap es una biblioteca de CSS y JavaScript que se utiliza para el diseño de la interfaz de usuario. Proporciona una variedad de componentes de interfaz de usuario preconstruidos y personalizables que ayudan a crear una aplicación web atractiva y fácil de usar.

Base de datos relacional SQL, para almacenar y recuperar datos de la aplicación, se necesita una base de datos, en este caso, relacional.

Servidor web, para alojar la aplicación, se necesita un servidor web. Se podría utilizar Apache o Nginx, por ejemplo, junto con PHP y MySQL.

Es importante tener en cuenta que estos son solo algunos de los recursos de software necesarios para desarrollar la aplicación web. El proyecto puede requerir herramientas y tecnologías adicionales en función de sus necesidades y objetivos específicos a corto, medio o largo plazo.

### **Recursos humanos:**

Para el desarrollo de la aplicación web de gestión de colmenares, se ha necesitado de un único desarrollador, pero, esto ha dejado patente bastantes carencias en el diseño gráfico de la misma. Así mismo, llegado el momento, se deberá poder atender también a los usuarios con dudas o problemas técnicos que puedan surgir. Así mismo, la comercialización y exposición de la aplicación al mercado, deberá ser gestionada por un buen marketing.

Si es cierto que inicialmente podría encargarme de todas las tareas a sabiendas de las deficiencias, es cierto que conforme la aplicación empiece a establecerse en el mercado, se deberán incluir nuevos roles para el correcto funcionamiento, escalabilidad del producto y cubrir las carencias.

### **Viabilidad temporal:**

En cuanto a la viabilidad temporal del proyecto, es importante tener en cuenta la estacionalidad de la actividad apícola y la necesidad de ofrecer una solución atractiva y funcional para los usuarios. Es posible que la demanda de la aplicación varíe a lo largo del año, y que sea necesario adaptar la oferta y la estrategia de marketing para atraer a nuevos usuarios y fidelizar a los ya existentes. Es importante estar atento a las tendencias del mercado y a las necesidades de los usuarios para mantener la relevancia de la aplicación a largo plazo.

## ***Monetizar la aplicación***

Existen varias maneras de monetizar una aplicación, pero en el caso de Gesticolmenar, lo más importante es encontrar una estrategia de monetización que no requiera una gran inversión inicial o un alto costo de mantenimiento. Algunas opciones podrían ser:

**Publicidad:** La inserción de publicidad en la aplicación podría generar ingresos a través de anuncios en línea, pero es importante tener en cuenta que esto podría afectar la experiencia del usuario.

**Modelo de suscripción:** Podría establecerse un modelo de suscripción para acceder a funciones premium en la aplicación, como el acceso a estadísticas detalladas o herramientas avanzadas de análisis. Es importante asegurarse de que las funciones premium sean lo suficientemente atractivas como para que los usuarios estén dispuestos a pagar por ellas.

**Venta de datos:** En este modelo de monetización, la aplicación podría recopilar datos sobre la actividad de los usuarios y venderlos a terceros, como empresas de investigación de mercado. Sin embargo, es importante asegurarse de que se cumplan todas las regulaciones de privacidad de datos.

**Venta de productos apícolas:** La aplicación podría establecer acuerdos de colaboración con tiendas de productos apícolas y recibir una comisión por cada venta realizada a través de la aplicación.

Cualquiera de estas opciones podría ser viable para monetizar la aplicación, pero es importante analizar cuidadosamente cuál es la más adecuada, pudiendo ser una combinación de varias o de todas ellas.

## **Análisis de requisitos**

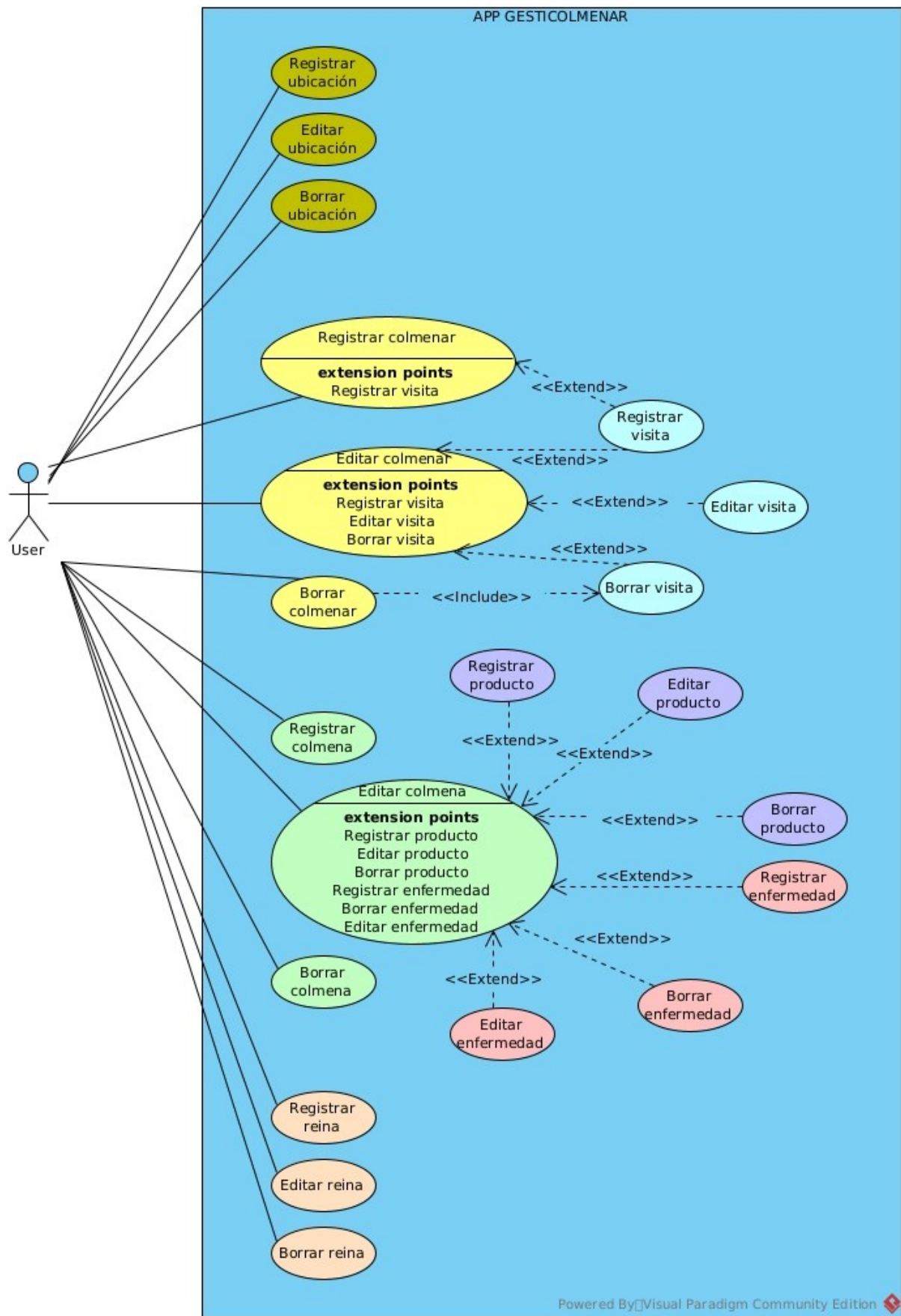
### ***Descripción de requisitos***

Es importante tener en cuenta los requisitos y funcionalidades que deben ser implementadas en la aplicación para satisfacer las necesidades de un apicultor. Por ello, se han tenido en cuenta los aspectos más relevantes del trabajo diario en dicha profesión y han sido analizados e incluidos en el desarrollo.

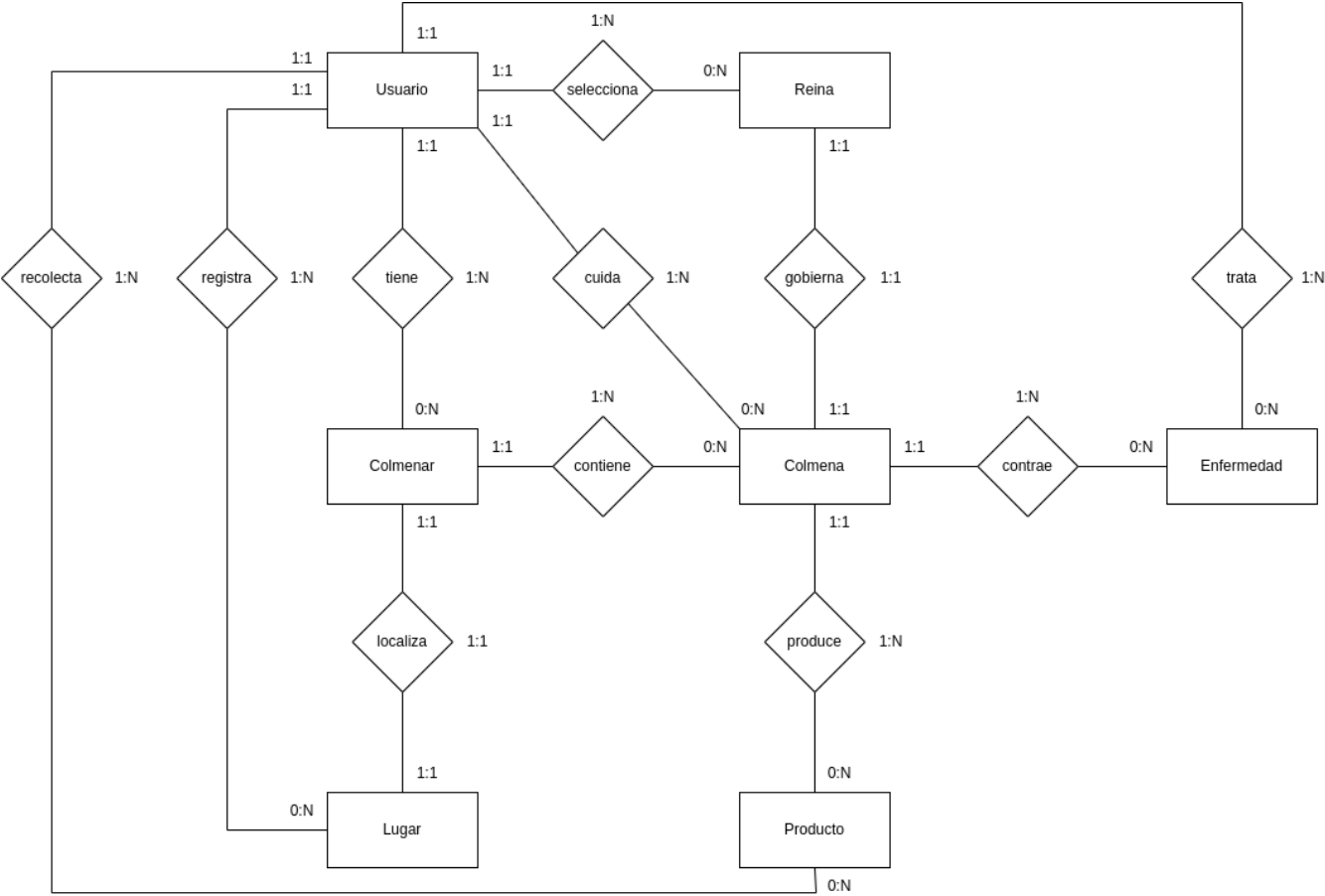
Requisitos:

1. Registro del usuario: la aplicación permite que los usuarios se registren y creen una cuenta.
2. Alta de ubicaciones: se deberán registrar las ubicaciones donde irán cada uno de los colmenares de forma previa al alta de un colmenar, indicando información relevante de dicha ubicación.
3. Alta de colmenares: los usuarios deben tener la capacidad de dar de alta sus propios colmenares, especificando la ubicación de la misma.
4. Alta de abeja reina: el usuario tiene que dar de alta una abeja reina en el sistema de forma previa al alta de una colmena, indicando, datos relevantes de la misma.
5. Alta de colmenas: la aplicación debe permitir al usuario poder dar de alta las colmenas en colmenares previamente dados de alta, especificando además, la reina que gobernará dicha colmena.
6. Indicar enfermedades: el apicultor podrá indicar si alguna colmena presenta alguna o algunas enfermedades o problemas sanitarios, especificando el tipo de las existentes, día en el que se observó el problema y próximo tratamiento en caso de requerirse. Dicho registro de enfermedad, si se indica una fecha de próximo tratamiento y esta es posterior al día actual o el día actual, mostrará un aviso en la sección de tareas. Para quitar el aviso, editaremos la fecha de próximo tratamiento, la eliminaremos o simplemente la dejaremos pasar.
7. Tareas pendientes y próximas visitas: en la gestión de un colmenar, se podrá editar la información del mismo para añadir el día de una visita y el día de la próxima visita, indicar tareas a realizar y campo de texto para anotaciones. La fecha de visita, si es posterior o el día actual, generará un aviso de tareas pendientes. Para quitar el aviso, editaremos la fecha de próxima visita, la eliminaremos o simplemente la dejaremos pasar.
8. Registro de recolección: en cada colmena se podrá editar la información con respecto a la cantidad de kg o gramos recolectados de cada producto a lo largo de la temporada.
9. Gráficas: la aplicación debe mostrar gráficas con los datos de kg totales de miel, polen y gramos de apitoxina, para que los usuarios puedan ver fácilmente el progreso de sus colmenares o la evolución de un colmenar a lo largo de los años si existen datos suficientes para ello.

## Caso de uso general



# Diagrama ER



## Paso a tablas

### *Users*

**users** (id, name, surname, dni, explotation\_code, email, email\_verified\_at, password)

PK: id

| Users             |                      |
|-------------------|----------------------|
| id                | Unsigned Big Integer |
| explotation_code  | String               |
| name              | String               |
| surname           | String               |
| dni               | String               |
| email             | String               |
| email_verified_at | String (nullable)    |
| password          | String               |

**id:** Campo usado como PK creado por Laravel.

**explotation\_code:** Todo ganadero de abejas ha de estar dado de alta en el REGA y debe tener el código de explotación, el cual ha de ir impreso de forma indeleble en todas y cada una de las colmenas.

**name:** Nombre del usuario.

**surname:** Los apellidos del usuario.

**dni:** Documento identificativo del usuario.

**email:** Email de contacto del usuario.

## Places

**places** (id, name, catastral\_code, poligon, parcel, postal\_code, has\_water, user\_id)

PK: id

FK: user\_id → users

| Places         |                      |
|----------------|----------------------|
| id             | Unsigned Big Integer |
| name           | String               |
| catastral_code | String (unique)      |
| poligon        | String               |
| parcel         | String               |
| postal_code    | String               |
| has_water      | Boolean              |
| user_id        | Unsigned Big Integer |

**id:** Campo usado como PK creado por Laravel.

**catastral\_code:** Referencia Catastral del lugar de 20 caracteres alfanuméricos.

**poligon:** Referencia del polígono cartográfico de la ubicación.

**parcel:** Referencia de la parcela asociada al polígono.

**postal\_code:** Código postal del lugar donde está el colmenar.

**has\_water:** Boleano que nos indique si el lugar posee acceso a agua.

**user\_id:** Foreign Key



## Apiaries

**apiaries** (id, last\_visit, next\_visit, beehives\_quantity, clear\_apiary, refill\_water, collect\_honey, collect\_pollen, collect\_apitoxine, food, others, user\_id, place\_id)

PK: id

FK: user\_id → users

FK: places\_id → places

UK: place\_id → places

| Apiaries          |                      |
|-------------------|----------------------|
| id                | Unsigned Big Integer |
| last_visit        | Date (nullable)      |
| next_visit        | Date (nullable)      |
| beehives_quantity | Integer              |
| clear_apiary      | Boolean              |
| refill_water      | Boolean              |
| collect_honey     | Boolean              |
| collect_pollen    | Boolean              |
| collect_apitoxine | Boolean              |
| food              | Boolean              |
| others            | Text (nullable)      |
| user_id           | Unsigned Big Integer |
| place_id          | Unsigned Big Integer |

**id:** Campo usado como PK creado por Laravel.

**beehives\_quantity:** Número total de colmenas que contiene el colmenar.

**clear\_apiary:** Necesidad o no de hacer limpiezas en el apiario, desde cortar hierba, limpiar caminos, recoger restos... etc.

**refill\_water:** Indicar si el apiario necesita rellenar el/los depósitos o medios por los que se suministre agua al ganado.

**collect\_honey/collect\_pollen/collect\_apitoxine:** indicar si hay que ir al apiario a recoger algún producto.

**food:** Indicar si hay que llevar comida al ganado, normalmente apipasta o asimilados, pues, en la recolección de los productos de la colmena, tanto miel como polen, son, los alimentos que ellas recolectan y por ende, hay que sustituir su alimento por otros, tales como azúcares/edulcorantes/fructosas...

**others:** Campo de texto para poder realizar anotaciones o añadir recordatorios no contemplados.

**user\_id:** Foreign Key

**place\_id:** Foreign Key

## Queens

**queens** (id, race, color, start\_date, end\_date, is\_inseminated, is\_zanganera, is\_new\_blood, user\_id)

PK: id

FK: user\_id → users

| Queen          |                      |
|----------------|----------------------|
| id             | Unsigned Big Integer |
| race           | String               |
| color          | String               |
| start_date     | String               |
| end_date       | String               |
| is_inseminated | Boolean              |
| is_zanganera   | Boolean              |
| is_new_blood   | Boolean              |
| user_id        | Unsigned Big Integer |

**id:** Campo usado como PK creado por Laravel.

**race:** Las reinas son las que van a trasladar la genética y raza a toda la población de abejas y es un parámetro muy importante para saber si van a ser abejas más o menos agresivas, más o menos productoras de miel o de otros productos de la colmena.

**color:** Existe un código internacional de colores aceptado por todos los apicultores que identifica el año de nacimiento de una reina: Azul (años terminados en 0 ó 5), Blanco (terminados en 1 ó 6), Amarillo (terminados en 2 ó 7), Rojo (terminados en 3 ó 8) y Verde (terminados en 4 ó 9).

**start\_date:** Año en el que esa reina entra a gobernar una colmena.

**end\_date:** Las reinas tienen un promedio de correcta ovulación fértil de aproximadamente 5 años, normalmente se sustituyen a lo largo del quinto año de su ciclo de vida, por tanto, aquí indicaremos el año en que teóricamente esa reina debe ser sustituida.

**is\_inseminated:** Una reina, ha de realizar el “vuelo nupcial” en el que, uno o diversos zánganos la inseminarán, por tanto, si no es correctamente inseminada pondrá huevos no fértiles, indicándonos si ha sido o no inseminada.

**is\_zanganera:** Una reina, por muchas y diversas razones puede volverse “zanganera”, normalmente, la causa principal es debido a que su “espermanteca” ha llegado a su fin y, empiezan a poner huevos que en su mayoría darán larvas de zángano y no de obreras, por ende, si se sospecha de esto, hay que indicarlo y observarla, de ser zanganera, habrá que indicarlo y proceder al cambio.

**is\_new\_blood:** En la actualidad es muy común adquirir reinas producidas y en muchos casos ya fertilizadas en laboratorios o granjas específicas. Saber si una reina es de nuestras colmenas o es sangre nueva, es un dato importante para saber si la nueva línea es o no interesante.

**user\_id:** Foreign key

## Beehives

**beehives** (id, user\_code, type, honey\_frames, pollen\_frames, brood\_frames, user\_id, apiary\_id, queen\_id)

PK: id

FK: user\_id → user

FK: apiary\_id → apiary

FK: queen\_id → queen

| Beehives      |                      |
|---------------|----------------------|
| id            | Unsigned Big Integer |
| type          | String               |
| honey_frames  | Integer              |
| pollen_frames | Integer              |
| brood_frames  | Integer              |
| user_id       | Unsigned Big Integer |
| apiary_id     | Unsigned Big Integer |
| queen_id      | Unsigned Big Integer |

**id:** Campo usado como PK creado por Laravel.

**type:** Existen diferentes modelos de colmenas: Langstroth, Dadant o Layens.

**honey\_frames:** De los marcos/cuadros que hay en la colmena, número de ellos que se destinan a producción de Miel.

**pollen\_frames:** De los marcos/cuadros que hay en la colmena, número de ellos que se destinan a dejarlos con polen para la propia alimentación de las abejas.

**brood\_frames:** De los marcos/cuadros que hay en la colmena, número de ellos que se destinan a la cría de nuevas obreras.

**user\_id:** Foreign key

**apiary\_id:** Foreign key

**queen\_id:** Foreign key

## Products

**products** (id, type, grams, year, user\_id, beehive\_id)

PK: id

FK: user\_id → users

FK: beehive\_id → beehives

| Products   |                      |
|------------|----------------------|
| id         | Unsigned Big Integer |
| type       | String               |
| grams      | Integer              |
| year       | String               |
| user_id    | Unsigned Big Integer |
| beehive_id | Unsigned Big Integer |

**id:** Campo usado como PK creado por Laravel.

**type:** los distintos tipos de producto que podríamos sacar de una colmena serían: miel, polen y apitoxina. Existen más productos que se pueden obtener, pero, por norma general no son apiarios al uso, son instalaciones más específicas para la obtención de jalea real y propóleo, del mismo modo, la cera de abeja, no suele comercializarse ya que es empleada para formar los nuevos panales para nuevas campañas.

**grams:** cantidad del producto extraído, se expresa en gramos ya que, tanto la apitoxina se extrae en pequeñas cantidades, es de fácil conversión a kg para la miel o el polen.

**user\_id:** Foreign key

## ***Diseases***

**diseases** (id, name, treatment\_start\_date, treatment\_repeat\_date, user\_id, beehive\_id)

PK: id

FK: user\_id → users

FK: beehive\_id → beehives

| Diseases              |                      |
|-----------------------|----------------------|
| id                    | Unsigned Big Integer |
| name                  | String               |
| reatment_start_date   | Date                 |
| treatment_repeat_date | Date                 |
| user_id               | Unsigned Big Integer |
| beehive_id            | Unsigned Big Integer |

**id**: Campo usado como PK creado por Laravel.

**name**: Existen distintos tipos de enfermedades relacionadas con las abejas, algunas de ellas parasitarias, tales como Varroa Destructor, Loque Americana... etc.

**reatment\_start\_date**: fecha en la que se lleva a cabo el tratamiento de la colmena.

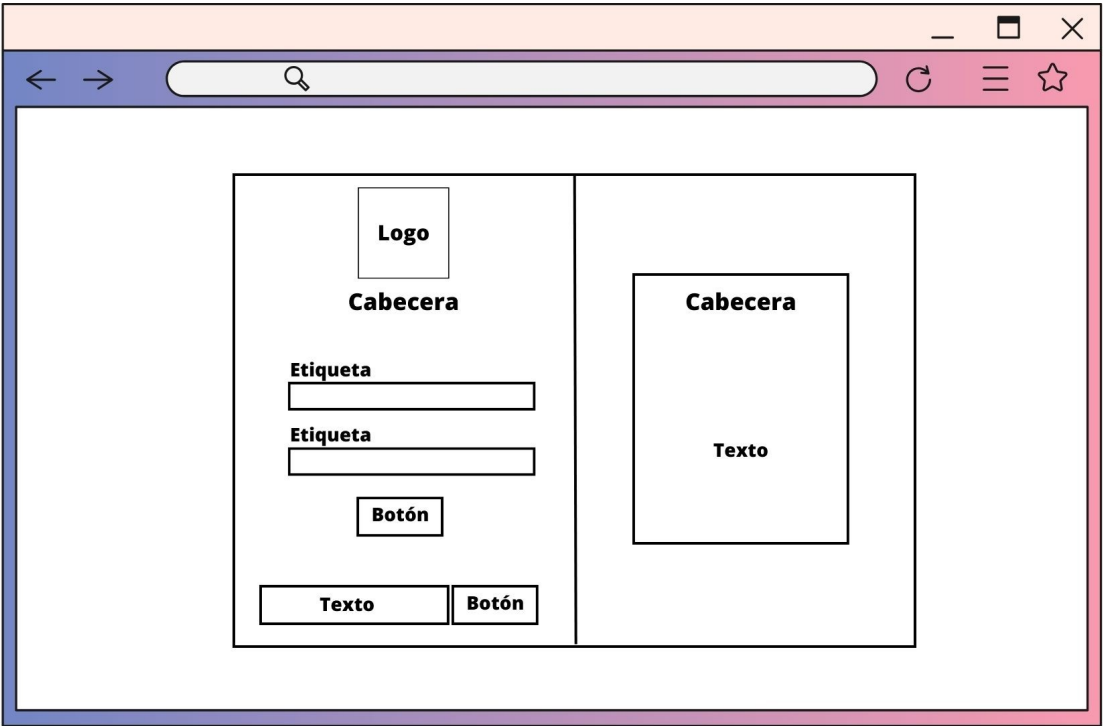
**treatment\_repeat\_date**: fecha en la que se debería volver a repetir el tratamiento, en caso necesario.

**user\_id**: Foreign key

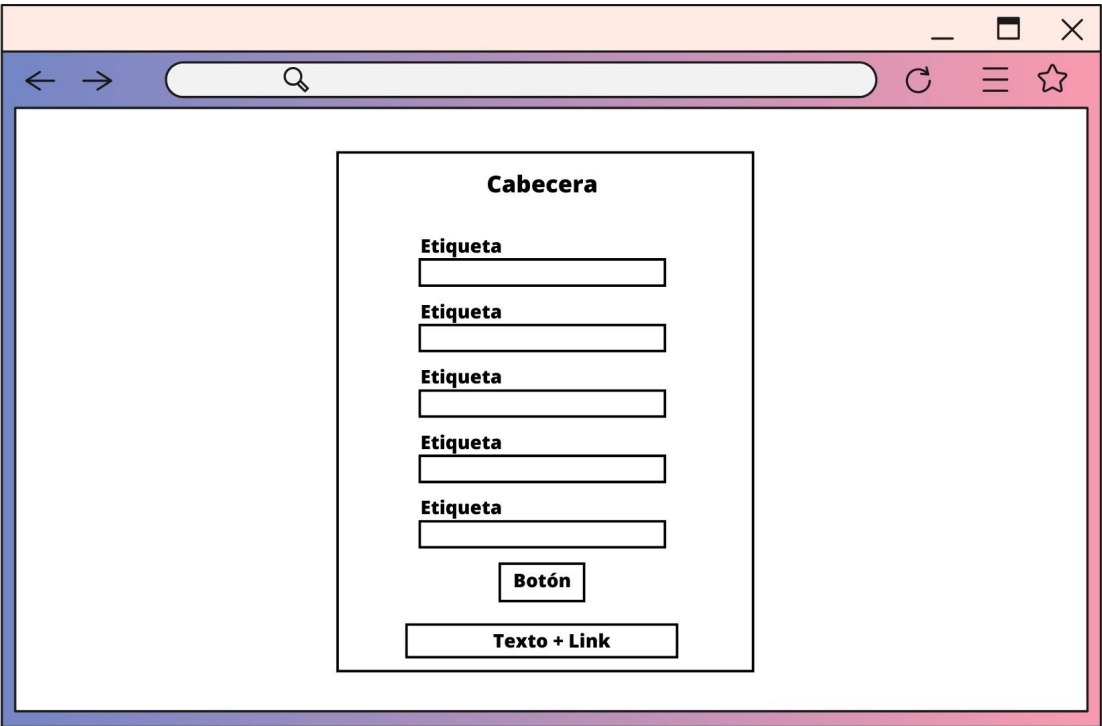
**beehive\_id**: Foreign key

# Mockups

## Login Desktop



## Registro Desktop




## Vista principal/Colmenares Desktop


Logo

Link

Link

Link














Success text (condicional)

Botón

Botón

Cabecera

|  |  |   |   |   |
|--|--|---|---|---|
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Texto ▾



Botón

Botón

Texto ▾

Botón

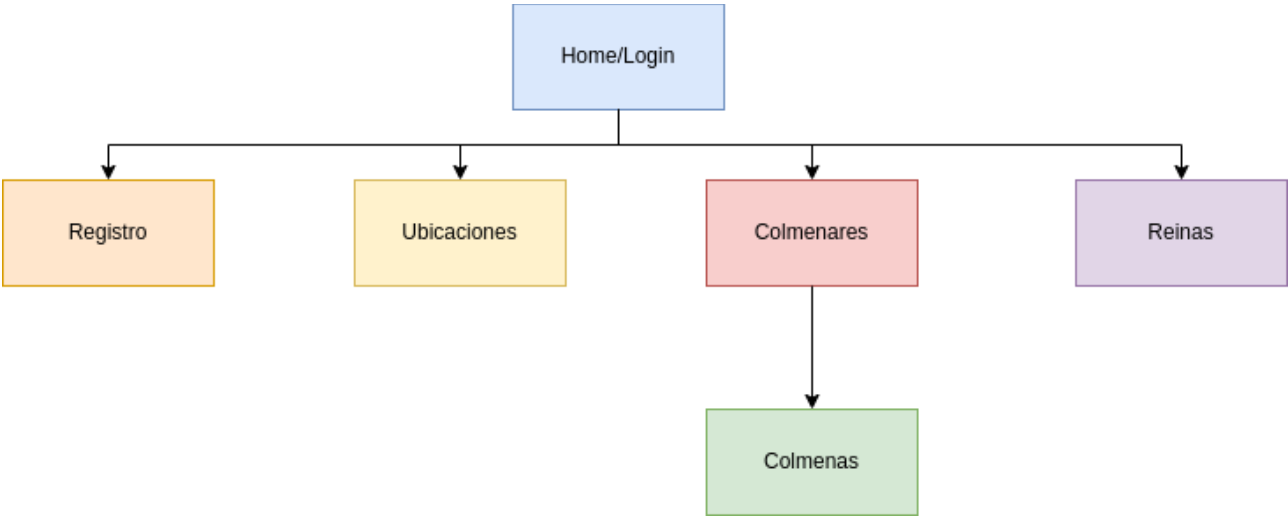
Botón

Texto



Site Map



## **Codificación**

### ***Tecnologías elegidas y su justificación***

En este proyecto he elegido como framework de PHP a Laravel, ya que, es muy popular, relativamente fácil de aprender y usar y me permite desarrollar la lógica de la aplicación web de manera estructurada y escalable. Además, Laravel ofrece una gran cantidad de características y herramientas, como el enrutamiento, la gestión de sesiones y autenticación de usuarios, manejo de la base de datos mediante el ORM de Eloquent...

Así mismo, existen multitud de librerías externas y una gran comunidad, por ello, también se ha usado una librería para mostrar los datos en formato gráfico, en este caso, se ha usado la librería “ConsoleTV/Charts”: <https://packagist.org/packages/consoleTVs/charts>

Así mismo, se ha usado junto a Laravel, Blade, que es un sistema de plantillas/vistas para el frontend que está integrado con el framework de Laravel, permitiendo de esta manera, separar la lógica de la presentación, manteniendo el código limpio y organizado.

Por otro lado y para visualizar el sitio correctamente y con un estilo limpio, se ha utilizado el framework de Bootstrap, que no deja de ser un conjunto de herramientas para desarrollar sitios y aplicaciones web. Ofrece una gran cantidad de estilos y componentes predefinidos para poder crear sitios responsive y que sea atractivo visualmente.

Para dar algunos estilos que no están incluidos con Bootstrap se ha empleado CSS y no SCSS ya que la cantidad de líneas de código han sido muy pocas no aportando nada el uso de SCSS.

Finalmente, para el sistema de gestión de la base de datos se ha usado uno de los más populares y más ampliamente utilizados, MySQL. Se ha empleado dicho sistema ya que Laravel lo incorpora de forma nativa y se puede integrar de forma sencilla la aplicación con la base de datos.

Laravel: Es un framework de PHP muy popular y fácil de usar que te permitirá desarrollar la lógica de la aplicación web de manera estructurada y escalable. Laravel ofrece una gran cantidad de características y herramientas, como el enrutamiento, la gestión de sesiones, la autenticación de usuarios y la base de datos ORM.

En resumen, las tecnologías que elegidas considero que son una muy buena combinación para desarrollar GESTICOLMENAR de manera eficiente y efectiva. Laravel te permite manejar la lógica de la aplicación, Blade te ayuda a separar la lógica de la presentación, Bootstrap te ofrece una amplia gama de componentes para la interfaz de usuario, MySQL te permite almacenar y recuperar datos de manera eficiente y CSS te permitirá personalizar la apariencia de tu aplicación.

### ***Entorno servidor***

En cuanto a la descripción general, diría que la aplicación será alojada en un servidor web como Apache o Nginx y base de datos MySQL.

En cuanto al tema de la seguridad, usando Laravel, se disponen de herramientas de validación de datos integradas para los campos de entrada de los formularios, por lo tanto se evitan así los ataques de inyección SQL.

Se usa encriptación para datos sensibles como podría ser la contraseña.

Además, Laravel nos incluye un sistema de autenticación y autorización para restringir el acceso a ciertas partes de la aplicación sólo a usuarios autenticados, protegiendo también el enrutamiento de la misma forma.

## ***Entorno cliente***

Se ha asegurado que la aplicación sea compatible con diferentes navegadores tales como Chrome, Firefox y Brave, así mismo, también se ha validado que se visualice correctamente en dispositivos móviles mediante las herramientas de desarrollo que incorpora Chrome y su posibilidad de visualizar en distintos dispositivos móviles, asegurando de este modo que GESTICOLMENAR es una app responsive y que funciona como es debido.

Para obtener un correcto funcionamiento responsive, se han basado las vistas en Bootstrap a través de Blade, proporcionando esa adaptación correcta, independientemente del viewport, posicionando los elementos en las posiciones correctas para su legibilidad, usabilidad y con un buen user experience.