Manual tecnico Aplicación Golden Garden

**Diego Armando Muñoz Carvajal**

**Sofia Vanegas Cordoba**

**John D. Rodriguez David**

**Introducción**

Golden garden es una aplicación que busca solucionar la problemática de la difusión de material botánico con fines comerciales de pequeños cultivadores y aficionados a la jardinería, la aplicación cuenta con un estilo de muro tipo post donde los usuarios pueden compartir especies de plantas ya en crecimiento o en semilla, material complementario para el crecimiento de las mismas y productos derivados de las plantas como lo son frutos o flores.

**Tecnologias implementadas**

La página web está construida sobre el lenguaje de programación Python conjunto con el framework para páginas web Django, en este el entorno de ejecución carga las siguientes librerías para su correcto funcionamiento:

-Django: es la base para el desarrollo de aplicaciones web en este proyecto.

-Crispy Forms: Es una librería complementaria para la web que ofrece estilos basados en bootstrap para formularios.

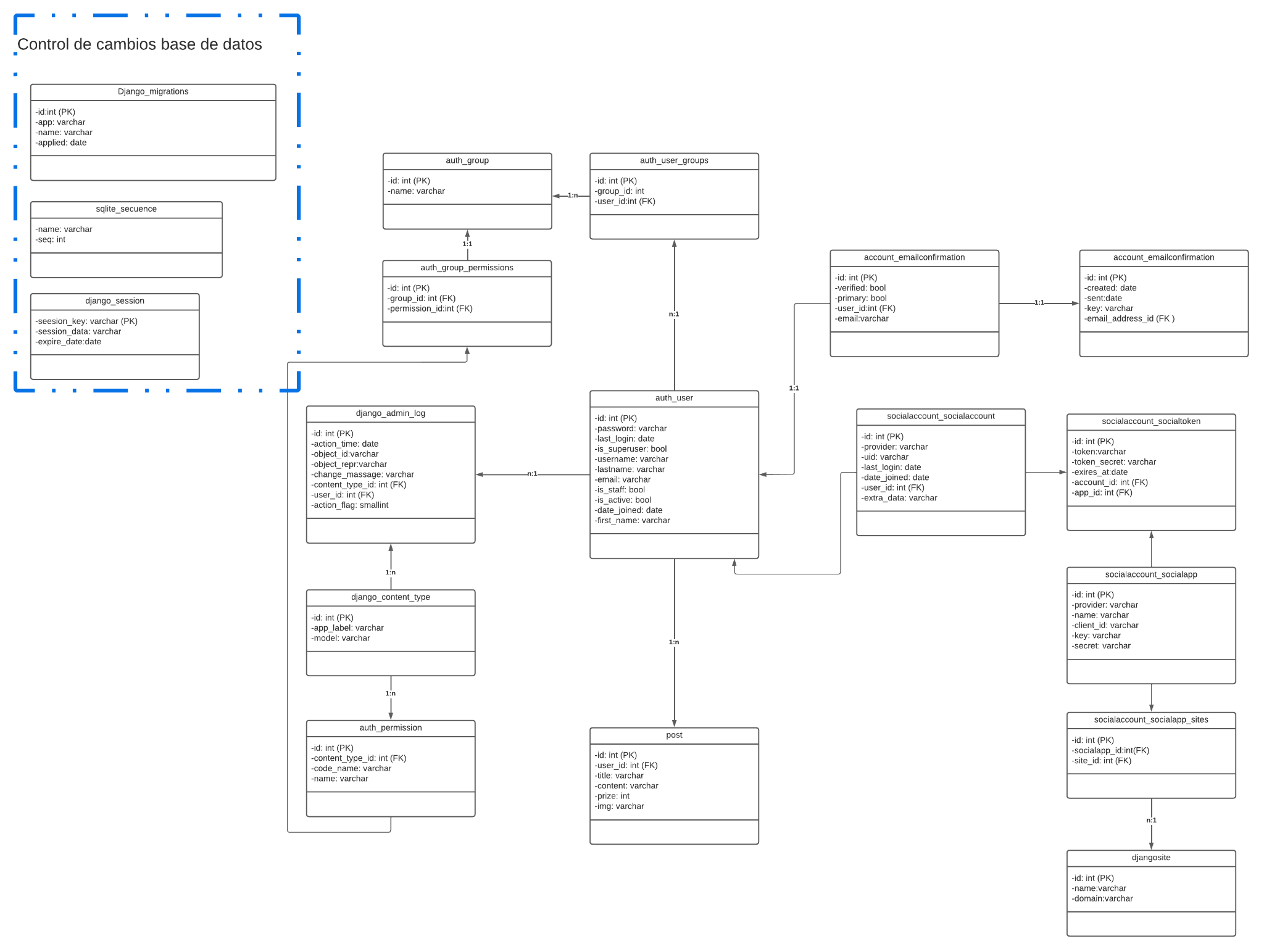
-Django All-Auth: Es un librería complementaria de Django que permite una interfaz de autenticación y registro de usuarios, se encarga también de controlar las vistas para usuarios autentificados, administradores y usuarios no autentificados por medio de permisos.

-Pillow: Permite el manejo adecuado de imágenes.

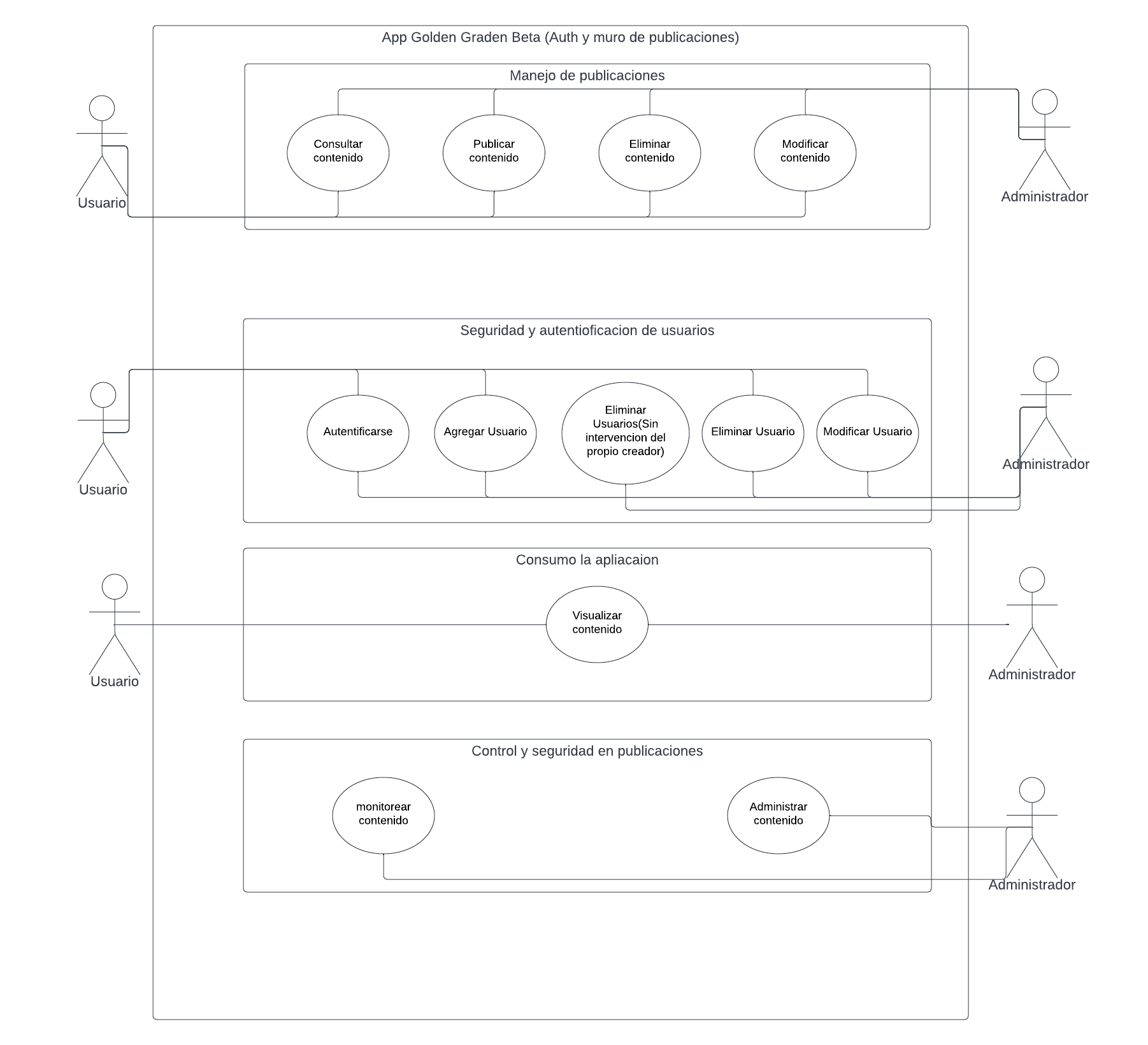
Para ejecutar el proyecto se aconseja hacer uso de un entorno virtual con python 3.10.4. para instalar cualquiera de las librerías usar el comando **pip install <nombre librería>**

**Solución**

Para elaborar la solución se planteó realizar una pequeña red social donde los usuarios registrados tuvieran acceso a un botón para crear nuevas publicaciones, cada publicación tendrá un título, un valor del artículo, una etiqueta - la cual servirá en un futuro para realizar búsquedas de ciertos tipos de productos- y una imagen. Estos valores están asociados a un usuario el cual tendría la facultad de modificar o eliminar su publicación.

En principio usando la librería allauth se maneja todo el registro de usuarios y permiso, para manejar este se debe consultar la documentación y manejar la app auth por medio del archivo settings, para el manejo de los post se crea una estructura que tiene como clave foránea el id del usuario de Auth all, contiene los atributos ya mencionados, de esta forma la base de datos se forma de la siguiente manera: [Diagrama bd\_GG.pdf](https://drive.google.com/file/d/1B4HzzUU-k4poPy_zVQJMevvSIBWQbemB/view?usp=sharing)

Adicionalmente se debe considerar los permisos del usuario se debe tener un usuario administrador que tenga acceso a el control de la aplicación por lo que se definen los siguientes casos de uso:



El manejo de permisos en cuanto a post se maneja desde el back end, solo los post que son propiedad del usuario pueden ser modificados desde la página web, con la excepción de los administradores que por medio del centro de control de django tiene acceso a eliminar contenido desde la base de datos.

**Clases**

Las clases siguen el esquema de la arquitectura de proyectos de Django, se tiene una clase forms, que organiza los formularios acorde a un modelo y una clase view que contiene la api. ver el código para más detalles.

**views:**

* **about, inicio, tips:** redirige al usuario la pestaña de about, home y tips respectivamente.
* **crear post**: se encarga del manejo del método post para crear nuevas publicaciones en el muro y realiza la validación de la petición.
* **editarPost**: Se encarga de manejar las peticiones de update, válida si el post editado es válido para su anexo.
* **verify**: válida si el usuario está seguro de eliminar un post
* **eliminarPost**: elimina un post.

**Ejecutar la aplicación**

Si se hace uso de un entorno virtual se debe correr en un terminal - puede ser a preferencia del usuario- el archivo activate ubicado dentro de la parte de scripts de el entorno virtual, seguido de esto el entorno estará listo para correr el proyecto de django, solo ubíquese dentro de la raíz del proyecto de Django, por defecto es en donde se encuentra el archivo manage.py - este es el archivo que permite la ejecución y manejo del proyecto- para correr el proyecto use el comando

>Python manage.py runserver

**Realizar cambios en la base de datos**

Para realizar cambios en los modelos se deben hacer por medio del archivo models, los cambios se harán efectivos luego de hacer migraciones y migrar el proyecto por medio de los siguientes comandos dentro de una terminal con la ubicación en la raiz del proyecto:

>Python manage.py makemigration

>Python manage.py migrate