

**Tecnologías de la Información y la Comunicación**

**Área: Sistemas Informáticos**

**DESARROLLO DE APLICACIONES WEB**

M.G.T.I. BLANCA ESTELA CHAVIRRA SALINAS

**REPORTE DE SOLUCIÓN DE CASO DE ESTUDIO**

**UNIDAD II**



ITI91N

**Alumno(s):**

Isaí A. Hinojos

Diana L. Leura

Roxana L. Limón

Ivan O. Márquez

Ariel F. Ríos

Chihuahua, Chih. A 15 de octubre de 2020

**Índice**

[PROPÓSITO Y JUSTIFICACIÓN 3](#_Toc85288570)

[DESCRIPCIÓN 3](#_Toc85288571)

[PATRONES DE DISEÑO A UTILIZAR 4](#_Toc85288572)

[Observer 4](#_Toc85288573)

[Singleton 4](#_Toc85288574)

[FRAMEWORK PARA UTILIZAR 5](#_Toc85288575)

[Flask 5](#_Toc85288576)

[ESTRUCTURA DE LA BASE DE DATOS 6](#_Toc85288577)

[SQLite 6](#_Toc85288578)

# **PROPÓSITO Y JUSTIFICACIÓN**

Seedhub es un sistema de provisionamiento y monitoreo para plantas interiores que busca el desarrollo y fortalecimiento de hortalizas y plantas ornamentales a través de un exhaustivo control en los niveles de hidratación, iluminación, propiedades del sustrato y datos de relevancia crítica que se reportarán constantemente en tiempo real al propietario.

Seedhub permite incluso a las personas con el estilo de vida más ocupado desarrollar su jardín bajo el techo de su hogar sin distinción de los espacios interiores, (escuelas, talleres, habitaciones, consultorios, etc…). Puesto que facilita el cuidado y crecimiento de plantas dentro de casa mientras que también permite modificar la configuración de operación para acomodar las distintas etapas del desarrollo de la planta, o acelerar el mismo. Todo esto en un ambiente exclusivo para la planta, de tal suerte que no tiene restricciones respecto a ubicación geográfica o ecosistema en el entorno.

# **DESCRIPCIÓN**

Seedhub consta de un microcontrolador Arduino, emisores, una aplicación web y múltiples sensores que de forma periódica evalúan registran y reportan las estadísticas relacionadas con el estatus de la cápsula. Por medio de un servidor local se proveen los cursos de acción a tomar en caso de que sea menester reacondicionar el ambiente, así como también recolecta la información bruta de los sensores que será posteriormente clasificada y sintetizada en una base de datos. Y que permitirá la presentación clara de estos registros en un sitio web que puede ser visitado a través de dispositivos con conexión a internet para que el usuario pueda monitorear y optimizar el recinto dependiendo de las necesidades y requerimientos de cada huésped.

# **PATRONES DE DISEÑO A UTILIZAR**

## **Observer**

Observer es un patrón de diseño de conducta en el cual los objetos obtienen información acerca de cualquier evento que ocurra a el objeto que se encuentran observando. Este define un método de suscripción, el cual va a actualizar cada objeto que esté suscrito así mismo, asegurándose que cada suscriptor obtenga el mensaje.

Este patrón de diseño nos permitirá crear una interfaz la cual podrá asignar distintos módulos Arduino en caso de que se cuente con ellos, y así ser capaz de actualizar las configuraciones del recinto que dicho módulo esté controlando, así como reportar de vuelta el estatus actual del entorno y de esa forma poder enviarlo a el servidor web, donde posteriormente será almacenado.

## **Singleton**

Singleton es un patrón de diseño de creación el cual nos permite crear una instancia única para un objeto o clase, y utiliza una variable global para almacenar esa instancia. Se pueden utilizar distintas maneras para enforzar la existencia única de dicha instancia porque solo se creará cuando se necesite. Esto previene la existencia de múltiples instancias activas para el mismo objeto, lo cual puede cuásar errores o corrupción de memoria.

El singleton entonces nos ayudará a mejor manejo de la base de datos cuando el recurso compartido sea utilizado. Previniendo la existencia de múltiples instancias de base de datos nos permite asegurarnos que las operaciones son hechas por el mismo objeto, lo cual negará la posibilidad de modificar o eliminar datos mientras otra parte del programa utiliza los recursos de la base de datos.

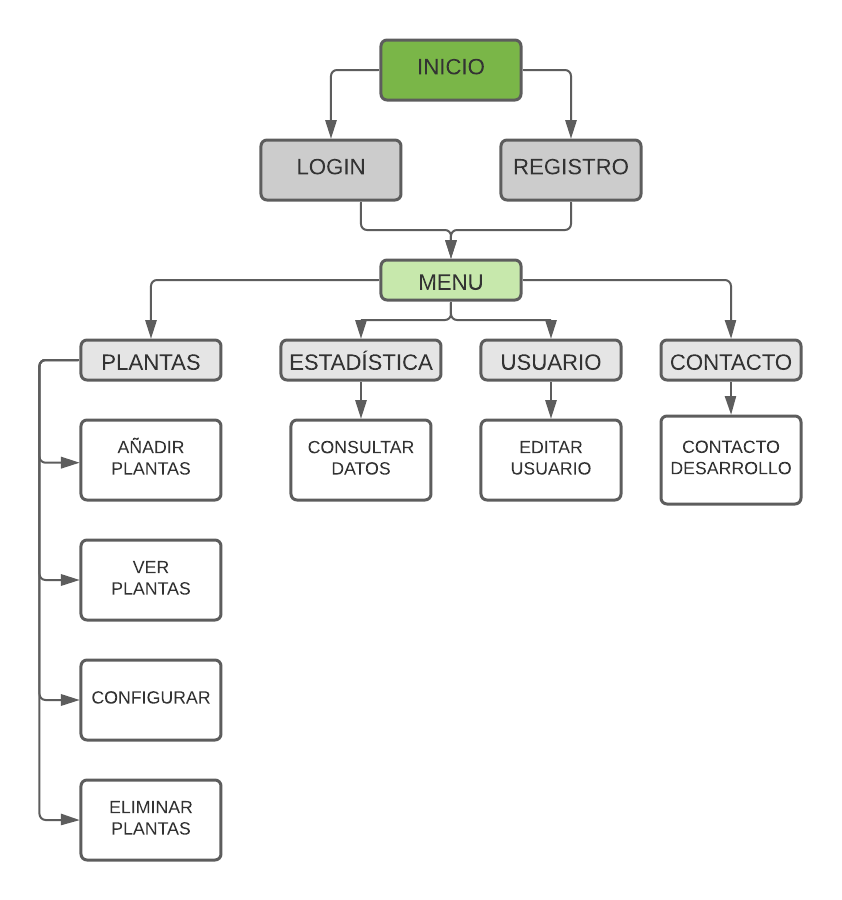
# **FRAMEWORK PARA UTILIZAR**

## **Flask**

Flask es framework de Python ligero para aplicaciones web de tipo Web Server Gateway Interface (WSGI), y fue creado con el propósito de simplicidad y amigabilidad a el usuario, pero con potencial para escalar aplicaciones complejas. Flask le ofrece a el desarrollador múltiples elecciones a el momento de desarrollar una aplicación, provee con herramientas, librerías y mecánicas que permiten la construcción de un sitio web pero no enforzará dependencias o te dirá como debe estar estrictamente estructurado.

Flask entonces nos permitirá desarrollar una aplicación web mínima y ligera la cual puede ser montada localmente o escalada a un servidor que permite la interoperabilidad de las aplicaciones con distintos usuarios, tantos como nuestro servidor web pueda sostener. Al ser Flask un framework basado en Python, nos permite acceder a las librerías que nos dan funcionabilidad para todo tipo de programa sin necesidad de instalar dependencias o paquetes de terceros.

# **DIAGRAMA DE NAVEGACIÓN**



# **ESTRUCTURA DE LA BASE DE DATOS**

## **SQLite**

La misma flexibilidad de Python nos permite utilizar una base de datos SQLite, la cual implementa la funcionalidad de las bases de datos relacionales sin la necesidad de un servicio de base de datos corriendo en nuestro servidor, de esa forma aligerando la carga en el mismo. SQLite nos permite ejecutar consultas como en cualquier otro modelo relacional sin la necesidad de un sistema administrador de base de datos.



# **MOCKUPS**

