PLAN DE GESTIÓN DE LA CONFIGURACIÓN

Control de versiones

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 29/04/2017 | 1.0 | Creación documento | Rommel Chipana Caballero  Wilson Julca Mejía |

**INDICE**

Contenido

[**1.** **Introducción** 4](#_Toc481837455)

[**1.1.** **Propósito** 4](#_Toc481837456)

[**1.2.** **Aplicabilidad** 4](#_Toc481837457)

[**1.3.** **Definiciones, Acrónimos y Abreviaciones** 4](#_Toc481837458)

[**2.** **Gestión de Configuración de Software** 4](#_Toc481837459)

[**2.1.** **Organización** 4](#_Toc481837460)

[**2.2.** **Roles y responsabilidades** 6](#_Toc481837461)

[**2.3.** **Políticas, directrices y procedimientos** 7](#_Toc481837462)

[**2.4.** **Herramientas, entorno e infraestructura** 8](#_Toc481837463)

[**2.5.** **Calendario:** 10](#_Toc481837464)

[**3.** **Actividades de la gestión de la configuración** 10](#_Toc481837465)

[**3.1.** **Identificación** 10](#_Toc481837466)

[**3.1.1.** **Elementos de la configuración** 10](#_Toc481837467)

[***3.1.2.*** **Nomenclatura de la identificación** 12](#_Toc481837468)

[***3.1.3.*** **Lista de Ítems con la nomenclatura.** 13](#_Toc481837469)

**Plan de Gestión de la Configuración (PGC)**

# **Introducción**

La gestión de la configuración coordina el desarrollo de software para minimizar la confusión las modificaciones y así maximizar la productividad y minimizar las equivocaciones.

Kodevian Studio SAC, actualmente utiliza GitLab para almacenar todos sus proyectos en repositorios, utiliza repositorios por área/tecnología (front, back, android, ios) para el mismo proyecto, esto ocasiona el problema de tener muchos repositorios para el mismo proyecto, pero tiene la ventaja de cada grupo de desarrollo de acuerdo a su especialidad tenga que preocuparse solo por subir o traer cambios de su área. También utiliza servidores para las diferentes etapas de desarrollo y entrega: local, test, stage y producción, esto permite tener un mejor forma de trabajo y entrega a los clientes.

## **Propósito**

## **Aplicabilidad**

## **Definiciones, Acrónimos y Abreviaciones**

# **Gestión de Configuración de Software**

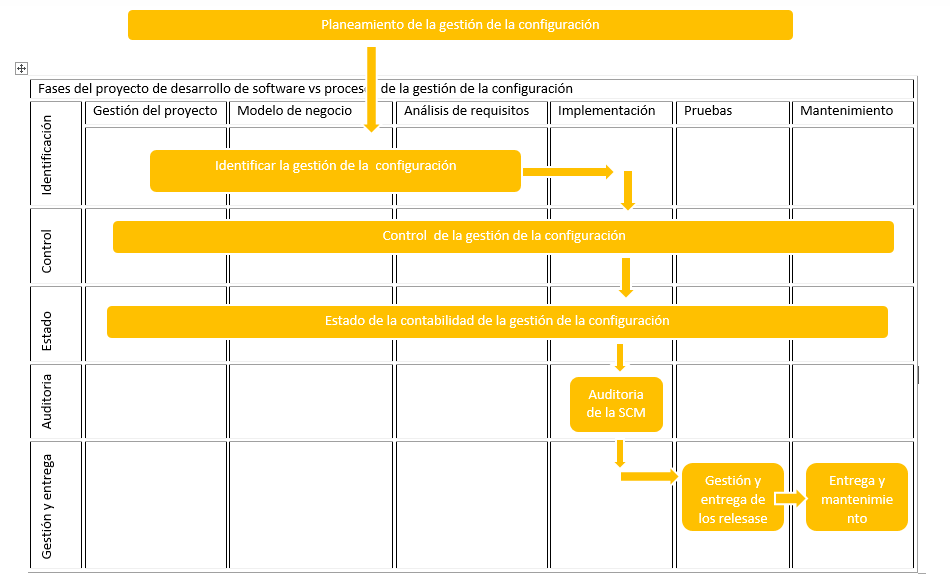
## **Organización**

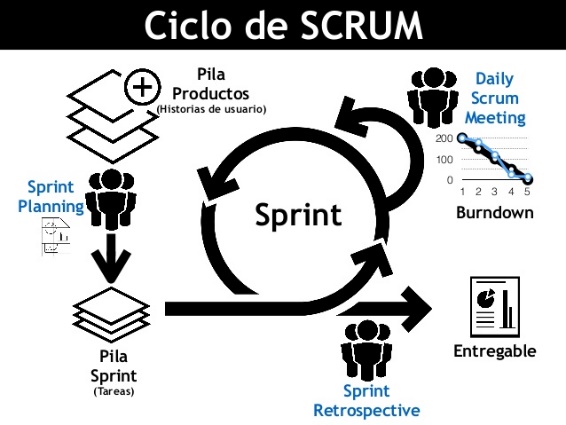
Para el desarrollo del software el equipo de desarrollo sigue la metodología ágil de scrum, trabajando en conjunto, apoyándose unos a otros y así poder aumentar la productividad.

El proceso parte de la lista de objetivos o requisitos que será priorizados del producto, que actúa como plan de proyecto.

En esto el cliente prioriza los objetivos balanceando el valor que le aportan su coste y queda repartidos en iteraciones o entregas, estas entregas ya pueden ser de manera parcial o regular.

El equipo realiza una reunión y se designa un tiempo estimado la cual debe de cumplir.





## **Roles y responsabilidades**

En la siguiente tabla se especificarán los roles y sus respectivas responsabilidades de cada uno. La estructura de la tabla será la siguiente:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Rol** | **Responsabilidad** |
| Esau Ocrospoma Callupe | Comité de control de cambios | Procesar las Solicitudes de cambios e iniciar el proceso de evaluación, aprobar o denegar. También se encargará de la revisión de la implementación. |
| Franco Jimenez Ledesma |
| Romel Chipana Caballero | Bibliotecario | Definir y dar mantenimiento a las bibliotecas que son usadas durante la gestión de configuración. |
| Carlos Ormeño Vargas | Gestor de Configuración | Implementar, mantener y mejorar la gestión de configuración. |
| Lennis Wong Portillo | Auditor | Auditar la Gestión de la configuración |
| Franco Jiménez Ledesma | Jefe de proyectos de Kodevian Studio | Garantizar que se cumplan los objetivos de las planificaciones |
| Wilson Julca Mejía | Desarrollador iOS de Kodevian Studio | Desarrollar sus tareas de acuerdo a la planificación realizada y una vez terminado subir al App Store |
| Luis Campos Rubina | Desarrollador BackEnd de Kodevian Studio | Desarrollar sus tareas de acuerdo a la planificación realizada y estar con constante comunicación con los desarrolladores android, ios y front |
| Percy Tataje Guzmán | Desarrollador Android de Kodevian Studio | Desarrollar sus tareas de acuerdo a la planificación realizada y cuando el producto este finalizado subir al play store |
| Esau Ocrospoma Callupe | Analista de requerimientos y de calidad de Kodevian Studio | Especificar las características operacionales del software (función, datos y rendimientos), indicar la interfaz del software con otros elementos del sistema y establecer las restricciones que debe cumplir el software. |

## **Políticas, directrices y procedimientos**

Los documentos actuales de la empresa que se tomaran en cuenta son:

1. El plan de Gestión de la Configuración está alineado con las siguientes políticas de la organización.  
2. Política de la Seguridad de la Información – PSI  
3. Política de seguridad en los procesos de desarrollo y soporte – PSGSI01 Manual de políticas de seguridad Sistema de Gestión de Seguridad de la información.  
3. Política de gestión de acceso de usuarios – PSGSI02

4. Manual de políticas de seguridad Sistema de Gestión de Seguridad de la información  
5. Política de uso de contraseñas – PSGSI03

6. Manual de políticas de seguridad Sistema de Gestión de Seguridad de la información  
7. Política de Intercambio de Información – PSGSI04

8. Manual de políticas de seguridad Sistema de Gestión de Seguridad de la información

Para llevar a cabo la gestión de la configuración se van a tomar en cuenta los siguientes:

1. Cada vez que se realice un cambio, realizar su commit y push en su rama correspondiente, ubicarse en la rama master hacer pull, merge y push, luego retornar a su rama para seguir realizando cambios

2. Los cambios solo son realizados por usuarios autorizados

## **Herramientas, entorno e infraestructura**

**GitHub**

Durante el proceso de gestión de la configuración se utilizará la herramienta Github para el control de versiones del producto. Cada miembro del equipo debe crear su rama con Nombres y Apellidos en el repositorio que se creó en Github para el proyecto, en donde harán sus modificaciones y así tener al resto del equipo con la última versión.

**Android Studio**

Entorno de desarrollo que está disponible para windows, linux y MacOs destinado para el desarrollo de aplicaciones android.

Proporciona emulador y pruebas de test a las apps desarrolladas en esta.

**XCode**

Entorno de desarrollo para MacOS que contiene un conjunto de herramientas destinadas al desarrollo de aplicaciones iOS.

**PyCharm**

Entorno de desarrollo integrado multiplataforma utilizado para desarrollar en el lenguaje de programación Python. Proporciona análisis de código, depuración gráfica, integración con VCS / DVCS y soporte para el desarrollo web con Django, entre otras bondades

**ENTORNOS:**

**Desarrollo**

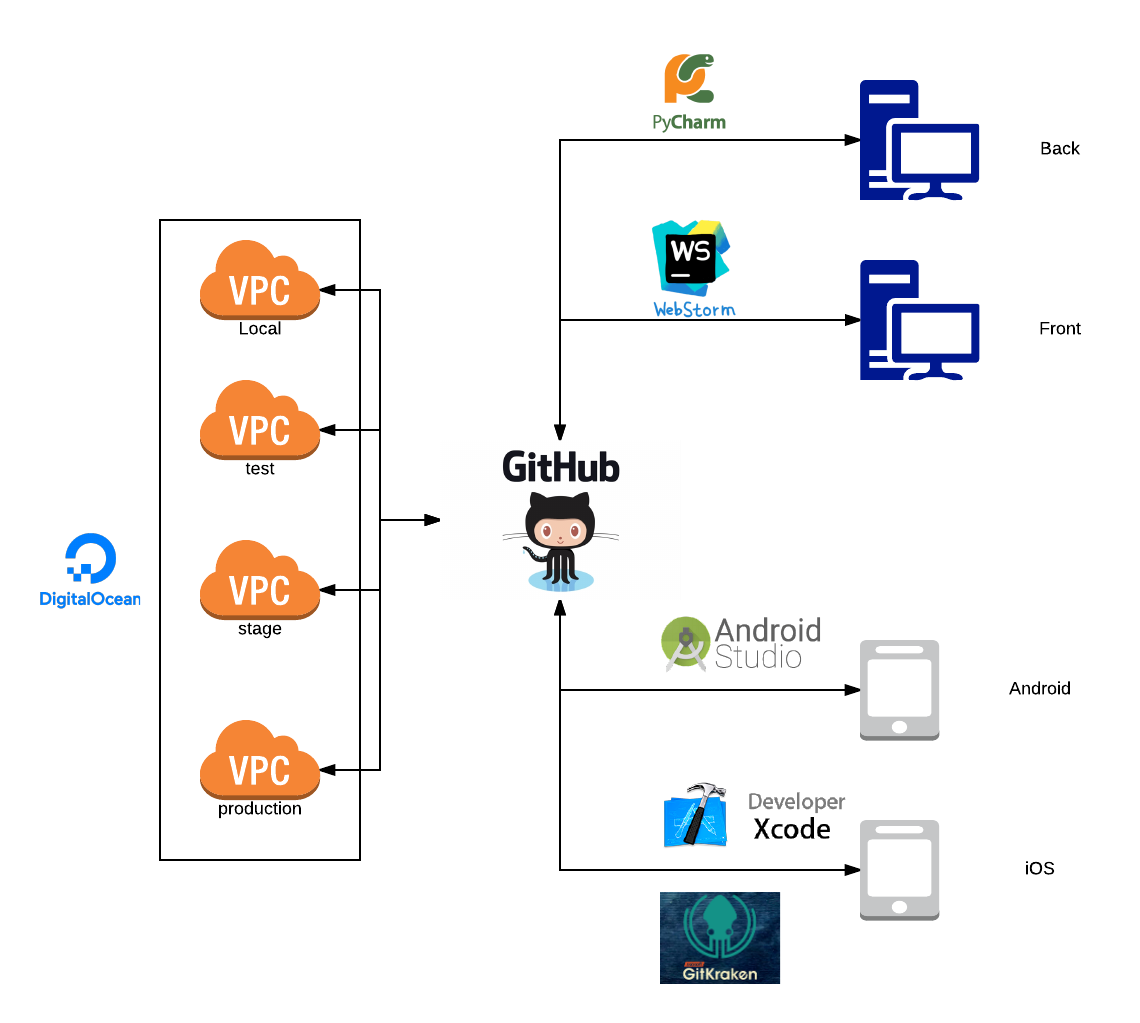
Es el entorno de trabajo para los desarrolladores. Trabajando de forma aislada con el resto de las capas, los desarrolladores pueden probar cambios radicales en el código sin afectar de forma adversa al resto del equipo de desarrollo, modificar la aplicación para añadir nuevas características y/o corregir errores.

**Pruebas**

Este entorno debe ser lo más idéntico posible al entorno de producción. El propósito principal del entorno de pruebas es simular al entorno de producción con el fin de testear las actualizaciones (en un entorno similar al de producción) para asegurar que las mismas no corrompen la aplicación existente en los servidores en producción. De esta forma se minimizan las caídas del sistema en producción. Además, este entorno puede funcionar tanto como demo como para entrenamiento y capacitación de los usuarios.

**Producción**

El entorno de producción puede incluir un servidor único o un cluster de servidores. Es el entorno donde trabajan los usuarios finales y se trabaja con los datos de negocio.

**Infraestructura**

## **Calendario:**

# **Actividades de la gestión de la configuración**

## **Identificación**

### **Elementos de la configuración**

La nomenclatura de los ítems de la configuración serán según el siguiente formato:

Para documentos propios de la empresa serán las iniciales de cada las palabras que componen el documento (ACRÓNIMO). Ejemplo: PGC: Plan de Gestión de la Configuración.

Para documentos que pertenecen al desarrollo deberán iniciar con las iniciales del nombre del proyecto y las iniciales del documento, separadas por un guion bajo. [Iniciales del proyecto]\_ [Iniciales del documento]. Ejemplo: KS\_PP.

* + Para documentos que cuentan co **Tipo:** Los ítems según su naturaleza se pueden clasificar en documentos de evolución (E), fuente (F) o soporte (S).
  + **Fuente:** Los ítems provienen de la empresa (E), del proyecto (P), del cliente (C), o proveedor (V).
  + **Extensión:** Corresponde a la tipo de archivo de cada item, que pueden ser: Word (docx), PDF(.pdf), Excel (xslx), MS Project (mpp), entre otros.

**Proyecto:** Son las siglas del proyecto al cual pertenece el ítem. n numeración se considerará nombrarlos con las iniciales del mismo o con parte del documento seguido del número. Ejemplo: CUS01

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **N.º de IC** | **Nombre** | **Tipo** | **Extensión** | **Fuente** | **Proyecto** |
| 01 | Plan de Gestión de la configuración del software | Evolución | DOCX | Empresa | -- |
| 02 | Plan de Gestión de Alcance | Evolución | DOCX | Empresa | -- |
| 03 | Plan de Gestión de Cambios | Evolución | DOCX | Empresa | -- |
| 04 | Política de la Seguridad de la Información | Evolución | DOCX | Empresa | -- |
| 05 | Política de seguridad en los procesos de desarrollo y soporte | Evolución | DOCX | Empresa | -- |
| 06 | Política de gestión de acceso de usuarios | Evolución | DOCX | Empresa | -- |
| 07 | Política de uso de contraseñas | Evolución | DOCX | Empresa | -- |
| 08 | Política de Intercambio de Información | Evolución | DOCX | Empresa | -- |
| 09 | Project Charter del Proyecto | Evolución | DOCX | Proyecto | AINNI |
| 10 | Acta de reunión de Coordinación del Proyecto | Evolución | DOCX | Proyecto | AINNI |
| 11 | Documento de Análisis del Sistema | Evolución | DOCX | Proyecto | AINNI |
| 12 | Documento de Diseño del proyecto | Evolución | PDF | Proyecto | AINNI |
| 13 | Planificación de Proyecto | Evolución | XLSX | Proyecto | AINNI |
| 14 | Manual de Usuario | Evolución | DOCX | Proyecto | AINNI |
| 15 | Plan de Pruebas | Evolución | XLSX | Proyecto | AINNI |
| 16 | Manual de Instalación | Evolución | DOCX | Proyecto | AINNI |
| 17 | Código Fuente Python | Fuente | PY? | Proyecto | AINNI |
| 18 | Codigo Fuente Android | Fuente | APK | Proyecto | AINNI |
| 19 | Código Fuente IOS | Fuente | IPA? | Proyecto | AINNI |
| 20 | S.O Android 5.0 - 7.1.2 | Soporte | ISO | Usuario | -- |
| 21 | S.O IOS 7 - 10 | Soporte | .iso | Usuario | -- |

### **Nomenclatura de la identificación**

Para identificar correctamente a cada uno de los productos o ítems de la configuración que se van generando a lo largo del desarrollo del proyecto, se ha diseñado la siguiente nomenclatura para los nombres de los elementos o ítems:

* Los Ítems que no son específicos a un proyecto único, tales como políticas, descripciones de procesos y guías (Documentos generales), son identificados únicamente por su acrónimo.

Ejemplo: PGC (PLAN DE GESTION DE LA CONFIGURACION).

* Los ítems que son específicos de un PY pero no están asociados con un componente del PY utilizan un identificador de 2 partes:

[ACRÓNIMO\_NOMBRE\_PROYECTO]\_[ACRÓNIMO\_TIPO\_ARTEFACTO][ACRÓNIMO\_NOMBRE\_DOCUMENTO]

Ejemplo, para identificar el modelo de diseño, que es del tipo de artefacto Diseño, del proyecto SGE, tenemos: **SGE\_DIMD**

* SGE: Sistema de gestión de encuestas (nombre de proyecto).
* DI: Tipo de artefacto (Diseño).
* MD: Modelo de diseño (nombre del ítem).
* Los documentos de caos de uso del sistema tendrán el siguiente identificador de 3 partes:

ACRÓNIMO\_NOMBRE\_PROYECTO]\_[ACRÓNIMO\_TIPO\_ARTEFACTO][ACRÓNIMO\_NOMBRE\_DOCUMENTO]\_[NUMERO]

EJEMPLO : -SGE\_ANCUS\_01

En la tabla 4, se indica el acrónimo de los tipos de artefactos utilizados para la nomenclatura de los ítems:



Tabla 4: Acrónimos de los tipos de artefactos.

**Nota:** En caso de que se agregue al repositorio un nuevo archivo, y la sigla del nombre de dicho archivo ya le pertenece a otro elemento de la configuración, este tendrá las cuatro primeras letras de la palabra de la última letra de la sigla de dicho archivo, siendo la primera letra en mayúscula y las siguientes tres en minúscula.

Por ejemplo: Si ya contamos con la siguiente nomenclatura: SGE\_ANMT que hace referencia al ítem: MATRIZ TRAZAVILIDAD UC vs REQUISITOS y deseamos agregar un nueva nomenclatura para el ítem: MATRIZ TRAZAVILIDAD UC vs CLASES, agregaremos la siguiente nomenclatura: SGE\_ANMTuccl

### **Lista de Ítems con la nomenclatura.**

En la tabla 5, se muestra la nomenclatura de los entregables considerados en la gestión de la configuración, donde:

* Nomenclatura: Son las siglas que se le asignará a cada entregable según la estructura indica en el anexo 3.2 (Nomenclatura de la clasificación).
* Entregable: Es el nombre del documento o ítems
* Proyecto: Son las siglas del proyecto al cual pertenece el entregable.

|  |  |  |
| --- | --- | --- |
| **Nomenclatura** | **Entregable** | **Proyecto** |
| PGC.docx | PLAN DE GESTION DE LA CONFIGURACION | - |
| CRO.mpp | CRONOGRAMA | - |
| PC.docx | POLÍTICA DE CONFIDENCIALIDAD |  |
| PS.docx | POLÍTICA DE SEGURIDAD | - |
| PGSI.docx | POLTÍCA DE GESTIÓN DE LA SEGURIDAD DE LA INFORMACIÓN | - |
| SGE\_IMES.php | EJECUTABLE DEL SISTEMA | SGE |
| SGE\_ANDN.docx | DOCUMENTO DE NEGOCIO | SGE |
| SGE-COCF.nj | CODIGO FUENTE | SGE |
| SGE-COSBD.sql | SCRIPTS DE BASE DE DATOS | SGE |
| SGE-COAXML.xml | ARCHIVOS XML | SGE |
| SGE-ANPP.docx | PLAN DE PROYECTO | SGE |
| SGE-ANCP.mpp | CRONOGRAMA DEL PROYECTO | SGE |
| SGE-ANLRQ.xlsx | LISTA DE REQUISITOS | SGE |
| SGE-ARDA.docx | MODELO DE ARQUITECTURA | SGE |
| SGE-DIMD.docx | MODELO DE DISEÑO | SGE |
| SGE-DIDMD.docx | DOCUMENTO DE MICRODISEÑO POR UC | SGE |
| SGE-ANDA.dcox | DOCUMENTO DE ANALISIS | SGE |
| SGE-ANMT.xlsx | MATRIZ TRAZAVILIDAD UC vs REQUISITOS | SGE |
| SGE-ANMTuccl.xlsx | MATRIZ TRAZAVILIDAD UC vs CLASES | SGE |
| SGE-COMBD,mwb | MODELO DE BASE DE DATOS | SGE |
| SGE-PRDCP.docx | DOCUMENTO DE CASOS DE PRUEBA | SGE |
| SGB-ANDN.docx | DOCUMENTO DE NEGOCIO | SGB |
| SGB-COCF.nj | CODIGO FUENTE | SGB |
| SGB-PRDCP.docx | DOCUMENTO DE CASOS DE PRUEBA | SGB |
| SGV-ANDN.docx | DOCUMENTO DE NEGOCIO | SGV |
| SGV-DIMD.docx | MODELO DE DISEÑO | SGV |
| SGV-COMBD.docx | MODELO DE BASE DE DATOS | SGV |
| STU-ARDA.docx | MODELO DE ARQUITECTURA | STU |
| STU-ANDN.docx | DOCUMENTO DE NEGOCIO | STU |
| STU-ANLRQ.xlsx | LISTA DE REQUISITOS | STU |
| SGE\_ANCUS\_01.docx | ESPECIFICACIÓN DE CUS GESTIONAR TEMAS | SGE |
| SGE\_ANCUS\_02.docx | ESPECIFICACIÓN DE CUS GESTIONAR PREGUNTAS | SGE |
| SGE\_ANCUS\_03.docx | ESPECIFICACIÓN DE CUS GESTIONAR ENCUESTAS | SGE |
| SGE\_ANCUS\_04.docx | ESPECIFICACIÓN DE CUS GESTIONAR ADMINISTRADORES | SGE |
| SGE\_ANCUS\_05.docx | ESPECIFICACIÓN DE CUS VISUALIZAR RESULTADO DE ENCUESTAS | SGE |

Tabla 5: Lista de ítems con nomenclatura.

### 

**3.2 Control**

**3.2.2 DEFINICIÓN DE LA ESTRUCTURA DE LIBRERÍAS**

GrupoRPP

Proyecto 2

A

I

N

N

I

PGC

PGCA

Si se modifica interviene el comité de control de cambios

AINNI

Gestión

Negocio

Análisis

Implementación

AINNI

Documentos

Documentos

Codificación

Diseño

Análisis

Clientes (Reléase)

Trabajo (Desarrollo)

Librería principal (aceptados)

PGC=Plan de gestión de la configuración

PGCA=Plan de gestión de la configuración de cambios

**ESTRUCTURA DE LIBRERÍAS ESPECÍFICA**

**LIBRERÍA PRINCIPAL:**

**WEB:**

Será desarrollado en lenguaje Python, utilizando el framework Django, que nos servirá como marco base de trabajo para el desarrollo del proyecto. Unas de las principales características del framework antes mencionado, es la facilidad para usar librerías, la facilidad de gestionar las versiones de cada librería, y hasta la facilidad de instalación de las mismas.

En Django, las librerías se suelen subir a un repositorio centralizado que se gestionan con PIP, el foro y comunidad de desarrolladores de Python, se encarga del mantenimiento, gestión y control de las mismas.

En la estructura del proyecto que nos provee el framework, se encuentra un archivo llamado “requirements.txt”, en el cual se procede a instalar los paquetes que usaremos en nuestro proyecto.

**MÓVIL iOS:**

Será desarrollado en lenguaje Swift3, utilizando la arquitectura MVC que XCode genera.

En iOS, las librerías se suelen subir a un repositorio centralizado que se gestionan con POD, el foro y comunidad de desarrolladores de Swift3, se encarga del mantenimiento, gestión y control de las mismas.

En la estructura del proyecto que nos provee el framework, se encuentra un archivo llamado Podfile en el cual se procede a instalar los paquetes que usaremos en nuestro proyecto.

**RESPONSABLES:**

La responsabilidad recae en el Jefe de desarrollo y los programadores, quienes son que van a interactuar en directo con las mismas.

**ACTIVIDADES:**

Las actividades principales en cuanto a la gestión de los paquetes, son sobre todo el control de las versiones, puesto que las gemas (como cualquier librería), tienden a actualizarse cada cierto tiempo, ya sea por errores encontrados o por nuevas funcionalidades que se desea tener. Los roles y el tipo de acceso a cada rol son presentadas en la Tabla 1.

|  |  |
| --- | --- |
| **ROL** | **TIPO DE ACCESO** |
| Gerente del proyecto | * Supervisar * Leer |
| Jefe de desarrollo | * Leer * Supervisar * Escribir * Ejecutar |
| Desarrolladores | * Leer * Ejecutar * Escribir |
| Arquitecto de Software | * Supervisar |

Tabla 1. Roles y Tipos de Acceso

**LIBRERÍAS WEB CONTROLADAS:**

A continuación listamos algunos paquetes que utilizaremos en el desarrollo del proyecto, que serán de gran ayuda en el despliegue de funcionalidades:

* **Django:** Paquete que tiene todas los módulos del framework web Django
* **Django Rest Framework:** Paquete que usaremos para realizar los servicios web siguiendo la filosofía de los formularios de django.
* **Django Debug Toolbar:** Paquete que sirve para mostrar la cantidad de queries, uso del CPU en cada página.
* **Schema Plus (schema\_plus):** Es una Gema que nos facilitará la gestión de llaves foráneas e índices en nuestra base de datos física. Dando paso a una gestión más fácil en la relación de modelos (base de datos a nivel lógico).
* **Pillow:** Paquete que sirve para el manejo de imágenes.
* **django-cryspi-forms:** Paquete sirve para la creación rápida de formularios.
* **psycopg2:** Paquete que sirve para la conexión con la base de datos PostgreSQL.

**LIBRERÍAS MÓVIL iOS CONTROLADAS:**

A continuación listamos algunos paquetes que utilizaremos en el desarrollo del proyecto, que serán de gran ayuda en el despliegue de funcionalidades:

* **Alamofire:** Paquete que permite el manejo de las solicitudes a los servicios REST
* **Alamofire-image:** Paquete que permite la descarga de imágenes asíncronamente.
* **RealmSwift:** Paquete que permite manejar la base de datos en Realm.
* **Schema Plus (schema\_plus):** Es una Gema que nos facilitará la gestión de llaves foráneas e índices en nuestra base de datos física. Dando paso a una gestión más fácil en la relación de modelos (base de datos a nivel lógico).

**NVActivityIndicatorView:** Paquete que nos permite mostrar un loader mientras se consume los servicios REST o mientras se realiza una acción.