SNIS

Manual del sistema

Organización Panamericana de la Salud (OPS)

TABLA DE CONTENIDOS

MANUAL CÓDIGO FUENTE	1
Requerimientos	
nstalación de la aplicación	
Acceso al sistema	
ormularios	
Grupos	
Usuarios	7
Cargo	8
Dependencia	
Funcionario	9
Instalación	9
Metadato	
Nivel	10
Servicio	10
Tipo de instalación	
Población	11
Corregimiento	11

Distrito	11
Poblado	12
Provincia	
Región de Salud	12
GESTIÓN DE PERMISOS	13
Usuarios	13
Grupos	15
Niveles de permisos de grupos	
Regional	
Instalación	16
CREACIÓN DE UN WEBSERVICE	17

MANUAL CÓDIGO FUENTE

El Sistema Nacional de Información en Salud es una aplicación web creada con el Framework de desarrollo Django, utilizando el lenguaje Python, hace uso intensivo del patrón de diseño Modelo-Vista-Controlador (MVC).

REQUERIMIENTOS

Django requiere Python 2.5 o superior. No se necesitan otras bibliotecas de Python. No es necesario tener un web server instalado, ya que Django trae su propio servidor para este propósito. Las bases de datos que podemos utilizar son: PostgreSQL, MySQL, MariaDB, Oracle o SQLite.

En nuestro caso debemos tener instalados en nuestro servidor las siguientes dependencias:

- *Python 3.5.6*
- Django 2.2.5
- MariaDB

INSTALACIÓN DE LA APLICACIÓN

El primer paso es clonar la aplicación del repositorio:

git clone https://github.com/campusvirtualsp/snis_minsa.git

Una vez clonado, debemos situarnos en el root de la carpeta del sistema para poder crear un ambiente virtual de Python:

python3 -m venv snisenv

Para activar el ambiente virtual ejecutamos:

source snisenv/bin/activate

Una vez dentro del ambiente debemos realizar las siguientes instalaciones desde la consola:

pip3 install django

pip3 install mysqlclient

pip3 install djangorestframework

pip3 install djangorestframework-simplejwt

Antes que nada, debemos configurar la conexión con la base de datos. Editamos el archivo *snis_minsa/settings.py* y agregamos las variables de entorno que corresponden a las credenciales de conexión de la base de datos:

'NAME': os.environ.get('DB_NAME'),

'USER': os.environ.get('DB_USER_NAME'),

'PASSWORD': os.environ.get('DB_PASSWORD'),

Una vez establecida la conexión con la Base de datos, debemos cargar todas las migraciones que se han hecho y que se encuentran en el repositorio, ejecutamos el siguiente comando:

python3 manage.py migrate

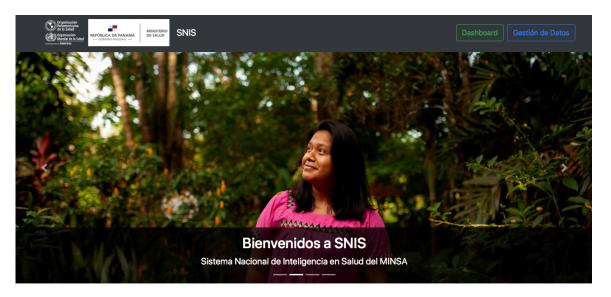
Luego debemos crear el usuario administrador desde la consola:

python manage.py createsuperuser

Finalmente ejecutamos nuestro servidor para probar el sistema:

python manage.py runserver

Para verificar, debemos ir al navegador de preferencia y copiamos http://localhost:8000 en la barra de direcciones. Si todo ha salido bien debe mostrarse nuestra página de bienvenida.



ACCESO AL SISTEMA

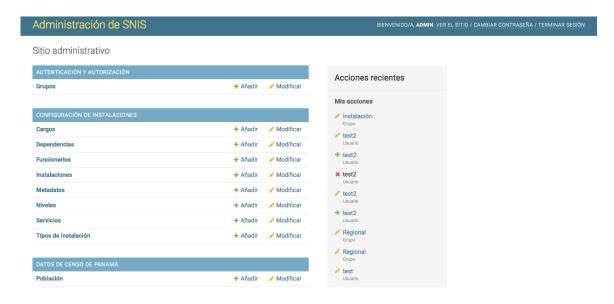
Para acceder a la administración de los datos debemos ir sobre *Gestión* de *Datos*:



Esto nos lleva a la página de Inicio de sesión:



Al iniciar sesión se nos debe presentar la página de gestión de datos principal desde donde tendremos acceso a las diferentes secciones del programa:



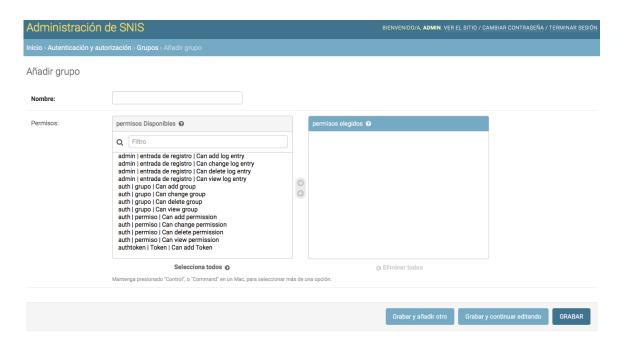
En esta página tenemos las siguientes secciones:

- Gestión de Usuarios y Grupos: En esta sección nos encargamos de los diferentes tipos de usuarios con sus respectivos permisos asociados a los grupos. Está compuesto por las siguientes subsecciones:
 - o Usuarios: usuarios del sistema
 - o Grupos: grupos asociados a los permisos de usuario
- Configuración de Instalaciones: Nos muestra todas las posibles opciones asociadas a una instalación. Está compuesto por:
 - Cargos: Son los posibles cargos (puestos) dentro de la infraestructura del MINSA
 - Dependencias: Es la dependencia o patronato que rige la instalación
 - o Funcionarios: Son los trabajadores de las instalaciones
 - Instalaciones: Son las instalaciones que serán modificadas desde el sistema

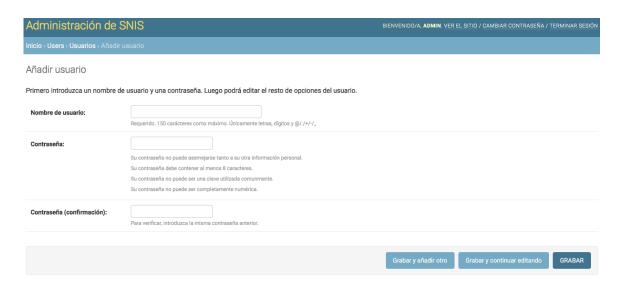
- Metadatos: Son opciones que pueden ser creadas en tiempo real, sin necesidad de ser programadas desde el código fuente
- Niveles: Los niveles de servicio que puede ofrecer una instalación
- Servicios: Todos los posibles servicios que son ofrecidos por una instalación
- Tipos de instalación: Tipo de instalación (hospitales, centros médicos, laboratorios, etc.)
- Datos del censo 2010: Tablas exportadas desde la página de estadística y censo de la contraloría.
 - o Población: la población obtenida por cada poblado del país
- División Política: División política de Panamá
 - Corregimientos
 - o Distritos
 - Poblados
 - o Provincias
 - o Regiones de salud

FORMULARIOS

Grupos



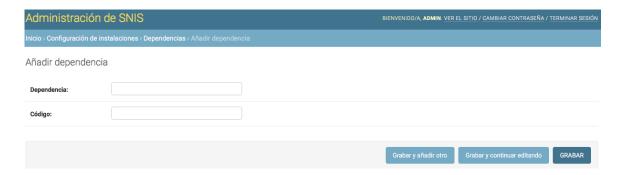
Usuarios



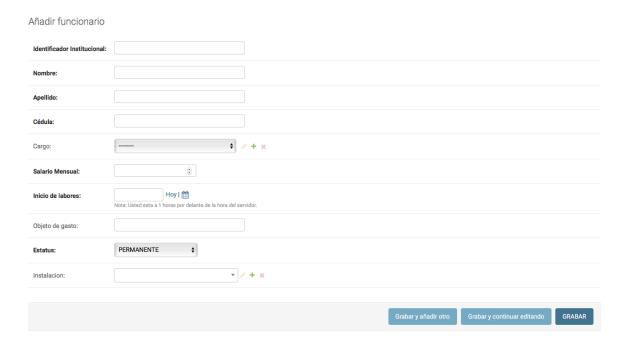
Cargo



Dependencia

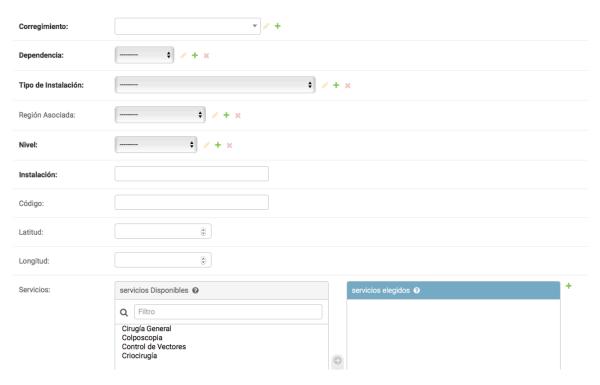


Funcionario

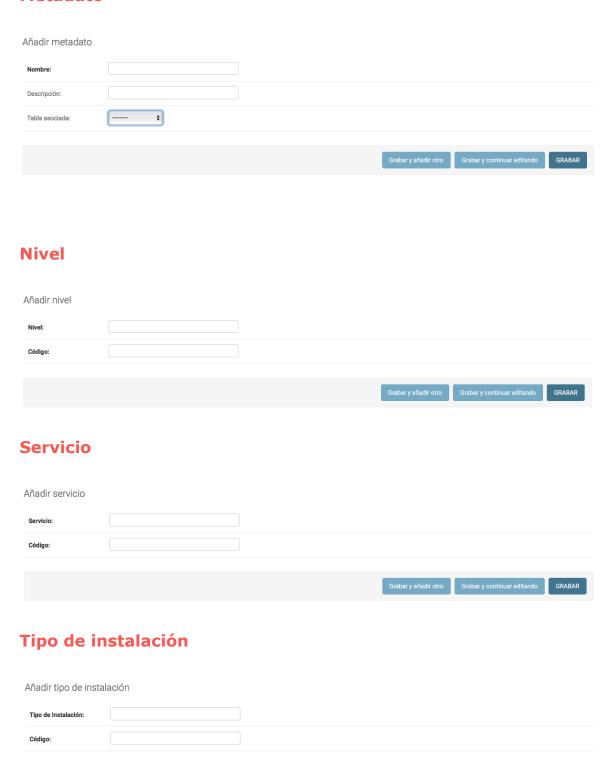


Instalación

Añadir instalación



Metadato

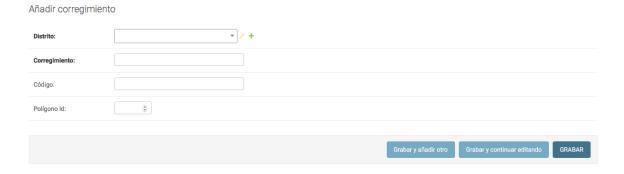


Grabar y añadir otro Grabar y continuar editando GRABAR

Población



Corregimiento



Distrito

Añadir distrito

Región:

Provincia:

Pistrito:

Código:

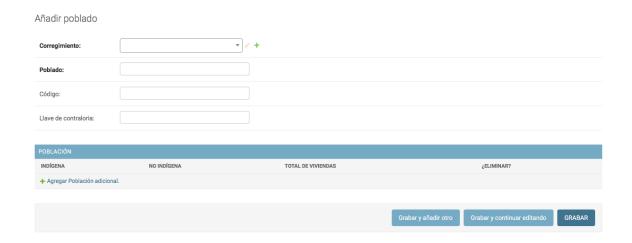
Polígono ld:

Grabar y afiadir otro

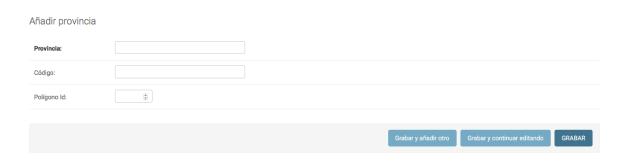
Grabar y continuar editando

GRABAR

Poblado



Provincia



Región de Salud

GESTIÓN DE PERMISOS

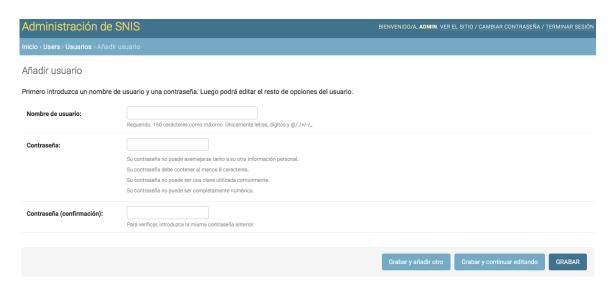
La gestión de permisos consiste en dos partes la gestión de usuarios y la gestión de grupos.

USUARIOS

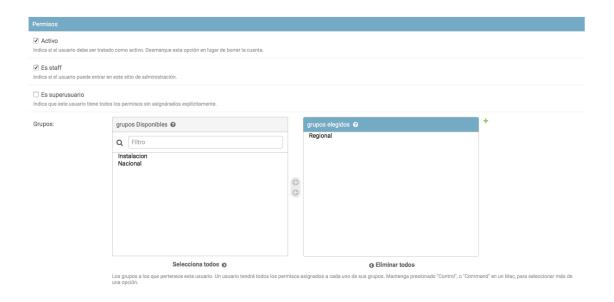
Para crear un usuario simplemente debemos irnos a la sección de usuarios y hacer clic en *Añadir Usuario* como se puede apreciar en la siguiente imagen:



Al hacer esto nos llevará al formulario de creación de usuario:



Una vez creado el usuario podemos editar sus propiedades. En este caso mostramos la selección de permisos de grupo Regional y seleccionamos su región





Dependiendo del tipo de grupo podemos asignar o no una región o una instalación. Debemos recordar que para los usuarios Nacionales no es necesario ya que tendrán acceso a toda la información del país.

GRUPOS

Los permisos están gestionados por grupos de usuarios. En nuestro caso tenemos 3 grupos en los cuales podemos dividir los usuarios

- Nacional
- Regional
- Usuario de Instalación

El grupo *Nacional* se refiere a los usuarios que van a gestionar completamente todos los datos del sistema a nivel nacional. Sin embargo, no tendrá permisos de súper administrador.

El grupo Regional se refiere a aquellos usuarios que gestionan los datos asociados a su región.

El grupo de la instalación se refiere al usuario que solo gestionará la información de su instalación.

NIVELES DE PERMISOS DE GRUPOS

Nacional

• Población: acceso total

• Dependencia: acceso total

• Funcionarios: acceso total

Metadatos: acceso total

Horarios: acceso total

• Instalaciones: acceso total

• Niveles: acceso total

Contacto: acceso total

Cargos: acceso total

• Servicios: acceso total

• Tipos de instalación: acceso total

• Corregimiento: acceso total

• Distrito: acceso total

Poblado: acceso total

Provincia: acceso total

• Región de salud: acceso total

Regional

Debemos recordar que el usuario regional solo tiene acceso a su región asignada

- Población: acceso total a su región
- Funcionarios: acceso total a su región
- Instalaciones: acceso total a su región
- Corregimiento: acceso total a su región
- Distrito: acceso total a su región
- Poblado: acceso total a su región
- Provincia: acceso de solo lectura
- Región de salud: acceso de solo lectura

Instalación

Debemos recordar que el usuario por instalación solo tiene acceso a su instalación asignada

- Población: acceso total la población de la instalación
- Funcionarios: acceso total los funcionarios de la instalación
- Instalaciones: acceso total a su instalación
- Corregimiento: acceso de solo lectura

CREACIÓN DE UN WEBSERVICE

Instalar via pip el servicio web de Django.

```
pip install djangorestframework
```

Agregar 'rest_framework' en la seccion INSTALLED_APPS de setting.py y la autenticación.

En el archivo *rest_api/views.py* se asignan los permisos de accesos a la información a través de un ViewSet. Tambien se hace el link con la clase

serializadora que se encarga de devolver la información. Ejemplo del ViewSet

```
class InstalacionViewSet(viewsets.ModelViewSet):
   permission_classes = (IsAuthenticated,)
   queryset = Instalacion.objects.all().order_by('id')
   serializer_class = InstalacionSerializer
```

Por cada tabla (modelo) de la aplicación se podrá extraer la información necesaria. En el archivo rest_api/serializers.py se creará un Serializer por cada servicio. Por ejemplo, para el modelo Instalacion se creará una clase serializadora InstalacionSerializer con los respectivos campos que serán devueltos. A continuación, se muestra la clase que devuelve la información en formato JSON:

```
class InstalacionSerializer(serializers.Serializer):
   id = serializers.IntegerField()
   instalacion = serializers.CharField()
   cod_inst = serializers.CharField()
   tipo_instalacion = serializers.StringRelatedField()
   nivel = serializers.StringRelatedField()
   latitude = serializers.DecimalField(max_digits=9, decimal_places=6)
   longitude = serializers.DecimalField(max_digits=9, decimal_places=6)
   corregimiento = serializers.StringRelatedField()
   dependencia = serializers.StringRelatedField()
   region_dependencia = serializers.StringRelatedField()
   servicios = serializers.StringRelatedField(many=True)

class Meta:
   model = Instalacion
```

Para el redireccionamiento url se usa el archivo rest_api/urls.py, en el siguiente caso se asigna al url la terminación instalaciones y se contacta el ViewSet.

```
router = routers.DefaultRouter()
router.register(r'instalaciones', views.InstalacionViewSet)
```