

Evaluación N°4: Programación Backend

I. Presentación

Unidad de Aprendizaje 3:

API restful con integración a base de datos.

Aprendizaje esperado

3.1 Codifica una API restful con el fin de conectarse con otras aplicaciones, utilizando autenticación JWT.

Actividades

1. Desarrollar una aplicación API RestFul. Utilizando el framework DjangoRest

En esta actividad práctica desarrollarás un trabajo respondiendo a casos respectivos de situaciones del rubro informático, información que será proporcionada por parte del docente, el cual deben completar con las diferentes partes que se especificarán y detallarán más adelante en esta guía de trabajo.

● Criterios de evaluación

3.1.1 Configura Django Rest framework, según la documentación oficial.

3.1.2 Codifica instrucciones de autenticación, de acuerdo a los requerimientos.

3.1.3 Codifica instrucciones que generen resultados en formato JSON, de acuerdo con los requerimientos.

3.1.4 Codifica una API funcional, que sea restful.

Evaluación

- Evaluación Sumativa 4 con escala de apreciación.
- Ponderación 30%
- Trabajo Individual

II. Instrucciones

1. Revisar el instrumento de evaluación sumativa, que se encuentra al final de este documento. (Escala adjunta)
2. Elabora la aplicación web respondiendo a los requerimientos entregados en la evaluación.
- 2.3 Entregar el proyecto por GitHub, compartiendo la URL en un bloc de notas adjuntando a la plataforma de INACAP.

“Recuerda que este trabajo será bajo la supervisión del docente a cargo de la asignatura y lo más importante, no puedes faltar y/o posponer tu entrega”

III. Actividades

Como futuro **Analista Programador**, debes desarrollar una **API RESTful** para una empresa de mantención industrial de la Región del Biobío, destinada a registrar empresas clientes, equipos, técnicos, planes de mantención y órdenes de trabajo.

La API será utilizada por aplicaciones web y móviles para gestionar y consultar información mediante servicios seguros en **formato JSON**.

Requerimientos

1. Configuración del Framework

- Configurar correctamente **Django REST Framework** en el proyecto.
- Habilitar la **API navegable**.
- Definir **JSON** como formato de respuesta estándar.
- Configurar permisos base de acceso para la API.
- Habilitar un mecanismo de autenticación de usuarios.

2. ORM

Se deben implementar, como mínimo, las siguientes entidades y atributos:

1. Empresa

- name
- address
- rut
- created_at

2. Equipo

- company
- name
- serial_number
- critical
- installed_at

3. Técnico

- user
- full_name
- specialty
- phone

4. Plan de Mantenimiento

- equipment
- name
- frequency_days
- active

5. Orden de Trabajo

- plan
- equipment
- technician
- status
- scheduled_date
- completed_at
- notes

3. Serialización y formato de respuesta

- Implementar mecanismos de **serialización** para cada entidad.
- Todas las respuestas de la API deben entregarse en **formato JSON**.
- Incluir al menos un **endpoint de prueba** que permita validar el correcto funcionamiento general de la API.

4. Desarrollo de la API RESTful

Para cada una de las cinco entidades, la API debe permitir:

- Crear registros (**POST**)
- Consultar registros (**GET**)
- Modificar registros (**PUT/PATCH**)
- Eliminar registros (**DELETE**)

Los endpoints deben:

- Estar organizados por recurso (por ejemplo: /api/companies/, /api/equipments/, etc.).
- Usar correctamente los métodos HTTP según la operación.
- Entregar respuestas en JSON coherentes con la acción realizada.

5. Autenticación y control de acceso (3.1.2)

- Implementar un sistema de **autenticación de usuarios**.
- Definir al menos la siguiente regla de acceso:
 - Usuarios **no autenticados**: solo pueden **consultar** información (lectura).
 - Usuarios **autenticados**: pueden **crear, modificar y eliminar** registros.

6. Documentación del código y del repositorio

El proyecto debe incluir:

- **Comentarios o docstrings** en partes relevantes del código.
- Un archivo **README** en el repositorio que incluya, como mínimo:
 - Descripción breve del proyecto y su contexto.
 - Requisitos para ejecutar el proyecto (dependencias principales).
 - Pasos para ejecutar la API (migraciones, creación de usuario, ejecución del servidor).
 - Ejemplos de uno o dos endpoints de la API (URL y descripción de lo que hacen).

Forma de entrega

La entrega debe realizarse mediante un **repositorio Git** que cumpla con:

- Al menos **5 commits** que muestren un **trabajo progresivo**, por ejemplo:
 - Commit de creación de proyecto base.
 - Commit de modelos.
 - Commit de serializers y vistas.
 - Commit de configuración de autenticación/permisos.
 - Commit de ajustes finales y documentación.

Recursos de apoyo para las actividades	Instrumento(s) de evaluación
No Aplica	Escala de Apreciación 3: Escala_Apreciación_EVA3