**Examen Técnico para Desarrollador Junior**

**Parte 1: Preguntas Teóricas**  
*1. Django*

¿Qué es Django y por qué se usa?

Django es un framework web de Python, y se usa porque con él se pueden crear aplicaciones web de manera rápida, es escalable, incluye protecciones contra vulnerabilidades y viene con diferentes funcionalidades integradas.

Explica brevemente el patrón MVC (Modelo-Vista-Controlador) y cómo se implementa en Django.

En Django el MVC se le conoce como MVT (Model-View-Template) y funciona de manera similar. Mediante esta arquitectura es posible dividir el desarrollo de una aplicación web en tres componentes principales:

* El modelo representa la estructura que tendrán los datos dentro de la aplicación y cómo será su interacción con la base de datos.
* La vista contiene la lógica de la aplicación y maneja las solicitudes que se generen en ella. Recibe datos del modelo y los procesa.
* En la plantilla se define como se van a presentar los datos. Por lo regular se trata de archivos HTML con una sintaxis que permite insertar los datos dentro de él.

¿Qué es un modelo en Django y cómo se define?

El modelo es una clase que se encarga de representar la estructura de los datos. Se utiliza para definir cómo se almacenan y manipulan los datos en la aplicación. Con Django es posible mapear estas clases con la base de datos. Cada atributo de la clase representa un campo de una tabla de la base de datos.

*2. Python*

¿Qué es un diccionario en Python y cómo se diferencia de una lista?

Un diccionario en Python es una estructura de datos que almacena datos en pares que tienen la estructura de clave-valor. Una lista almacena elementos singulares y se accede a ellos mediante un índice, mientras que en los diccionarios se accede a los elementos mediante su clave.

Explica la diferencia entre append() y extend() en una lista de Python.

Con append es posible agregar un único elemento al final de una lista, mientras que con extend es posible agregar una estructura como otra lista o una tupla al final de una lista.

¿Qué es un decorador en Python? Proporciona un ejemplo simple.

Un decorador es una función que recibe como argumento otra función y puede modificar su comportamiento de manera que otra serie de instrucciones se puedan ejecutar antes y/o después de la función original.

*3. MySQL*

¿Qué es una base de datos relacional?

Es un conjunto de tablas, relaciones y registros que cumplen con el modelo relacional. El modelo relacional está basado en la teoría de conjuntos.

Explica la diferencia entre una clave primaria y una clave foránea.

Una llave primaria identifica de manera única a un registro en una tabla, mientras que una llave foránea es un campo o conjunto de campos que hacen referencia a la clave primaria de otra tabla y establece la relación que existe entre ambas tablas.

¿Cómo se realiza una consulta SELECT básica en MySQL?

SELECT [conjunto de campos] FROM [tabla] WHERE [condición]

* Conjunto de campos: son los campos que se quieren consultar de una tabla. En caso de que se quieran consultar todos los campos, basta con escribir un asterisco (\*).
* Tabla: es la tabla de la base de datos de donde se van a consultar el conjunto de datos.
* Condición: solo los campos que cumplan con la condición serán visualizados.

*4. API*

¿Qué es una API y para qué se utiliza?

Es un conjunto de reglas y protocolos establecidos para que dos aplicaciones puedan comunicarse entre sí. De esta manera las dos aplicaciones podrán intercambiar datos y funcionalidades.

Explica brevemente la diferencia entre una API REST y una API SOAP.

Una API REST puede ser usada preferentemente para aplicaciones web o móviles, y donde se requieran utilizar los métodos HTTP estándar, mientras que una API SOAP puede ser usada preferentemente cuando se requiera de una seguridad más robusta y se trabaje en entornos empresariales más estrictos.

¿Qué son los métodos HTTP comunes utilizados en una API REST?

Son las operaciones básicas de un CRUD y se utilizan para crear peticiones a un servidor.

* GET: solicita visualizar un recurso.
* POST: crea un nuevo recurso
* PUT: manda un cambio en el estado de un recurso
* DELETE: elimina un recurso

**Parte 2: Ejercicios Prácticos**  
*1. Django*  
Crea un modelo Django para una aplicación de biblioteca con las siguientes características: Libros (título, autor, fecha de publicación) y Autores (nombre, fecha de nacimiento).  
 *2. Python*  
Escribe una función en Python que reciba una lista de números y devuelva una nueva lista con los números ordenados de menor a mayor.  
Escribe un script en Python que lea un archivo de texto y cuente la cantidad de palabras que contiene.  
  
*3. MySQL*  
Escribe una consulta MySQL para crear una tabla llamada "Usuarios" con los campos "id", "nombre", "email" y "fecha\_de\_registro".  
Inserta tres registros en la tabla "Usuarios".  
 *4. API*  
Diseña una API simple en Django que permita realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en un recurso llamado "Tareas". Proporciona los endpoints necesarios.

**Entrega:**

* Proporciona el código fuente en un repositorio de Git.
* Incluye un archivo README.md con instrucciones claras sobre cómo configurar y ejecutar el proyecto.