Semana 1 – Plan de Clases

# Semana 1 – Clase 1 (4 horas)

## Tema: Presentación + Diagnóstico inicial + Motivación del Proyecto

## 1. Introducción (30 min)

- Presentación del docente y del programa de la materia.  
- Explicar los objetivos de aprendizaje del cuatrimestre: programación cliente-servidor, APIs, autenticación, tiempo real.  
- Mostrar el pitch del proyecto final (ejemplo: Plataforma de Turnos Inteligente) para motivar a los estudiantes.  
- Conectar el contenido con aplicaciones de la vida real.

Actividad motivadora: preguntar a los estudiantes sobre su experiencia sacando turnos (peluquería, médico, etc.), identificar problemas y comentar si conocen aplicaciones que resuelvan esa situación.

## 2. Encuesta/Examen Nivelatorio (1h)

- Aplicar un formulario en Google Forms (o en papel si no hay conexión).  
- El examen incluye preguntas sobre programación básica, POO, SQL, HTML/CSS/JS y Python.  
- Recalcar que NO es para nota, sino para ajustar el curso al nivel del grupo.

## 3. Discusión de resultados (30 min)

- Mostrar estadísticas anónimas de los resultados (por ejemplo: cuántos saben SQL, cuántos trabajaron con APIs, etc.).  
- Generar una breve discusión con el grupo sobre qué temas creen que necesitarán más repaso.  
- Establecer un acuerdo de trabajo: la materia será intensiva en práctica y cada uno deberá reforzar sus puntos débiles.

## 4. Mini introducción a Cliente-Servidor (1h)

Explicación con ejemplos cotidianos:  
- Cliente = celular del usuario.  
- Servidor = sistema de turnos.  
- Flujo: Pedido (request) → Respuesta (response).  
- Representar con un diagrama simple en la pizarra.

### Práctica corta:

Usar el navegador o Postman para consultar la API pública:  
GET https://jsonplaceholder.typicode.com/users  
  
Mostrar la respuesta en formato JSON y explicar brevemente qué significa cada campo.  
Reflexionar: ¿qué pasó entre el cliente (navegador) y el servidor?

## 5. Cierre y reflexión (30 min)

- Preguntar al grupo: ¿qué fue lo más nuevo que aprendieron hoy?  
- Escuchar breves intervenciones de algunos estudiantes.  
- Anticipar la próxima clase: se practicará cómo Python puede actuar como cliente que solicita datos a un servidor.  
- Dejar como tarea opcional investigar qué es una API y traer un ejemplo de alguna API pública.

# Semana 1 – Clase 2 (4 horas)

## Tema: Introducción práctica a Cliente-Servidor con Python

## 1. Repaso breve (20 min)

- Recordar qué es Cliente-Servidor.  
- Pedir a 2–3 alumnos que lo expliquen con sus propias palabras para afianzar el concepto.

## 2. Primeros pasos con Python (40 min)

Revisión rápida de conceptos básicos: variables, listas, funciones.  
  
def mayor(lista):  
 return max(lista)  
  
print(mayor([2,5,1,8]))

## 3. Python como cliente HTTP (1h 15 min)

Introducir la librería requests para realizar solicitudes HTTP.  
  
import requests  
r = requests.get('https://jsonplaceholder.typicode.com/users')  
print(r.json())  
  
Explicación: en este caso, Python actúa como cliente que consulta un servidor y obtiene una respuesta en formato JSON.

### Práctica guiada:

Los estudiantes deben escribir un script que consulte la API de usuarios y muestre:  
- Nombre  
- Email  
- Ciudad

## 4. Actividad en grupos (1h 15 min)

- Formar grupos de 3–4 alumnos.  
- Cada grupo elige un endpoint distinto de la API pública (ej: posts, comments, todos).  
- Escribir un script en Python para mostrar la información de forma ordenada.  
- Al final, cada grupo presenta brevemente (2 minutos) lo que obtuvo.

## 5. Cierre (30 min)

Recapitulación:  
- Reforzar el concepto de cliente-servidor aplicado en la práctica.  
- Diferenciar entre pedir datos con Postman y hacerlo programáticamente con Python.  
  
Tarea para la próxima clase:  
- Investigar una API pública de interés (clima, deportes, música, etc.) y traer un ejemplo para compartir con la clase.