

# MANUAL TECNICO

## PROYECTO 2

Aplicación elaborada por Saúl Barbero

Guatemala 03 de mayo 2021

# Introducción

La siguiente aplicación fue diseñada en Visual Code, con los lenguajes de Python, HTML5 y JavaScript. Además de CSS y otros lenguajes necesarios para la programación web.

Esta es una página web funcional de un hospital universitario.

# Objetivos

## General

- Aplicar los conocimientos obtenidos a lo largo del curso de IPC1.

## Específicos

- Crear un Back y un Front.
- Montar un servidor web que despliegue la página creada.

# Especificación Técnica

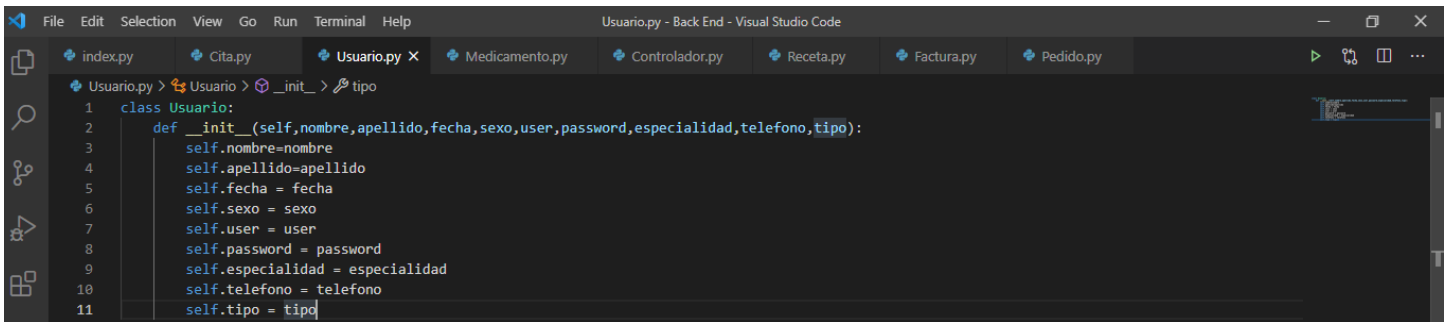
## Requisitos Mínimos de Software

- RAM: 128 MB, 64 MB para Windows XP (32 bits)
- Espacio en disco: 124 MB
- Java JRE de última versión
- Exploradores: Internet Explorer 9 y superior, Firefox
- Conexión a internet

## Requisitos Mínimos de Hardware

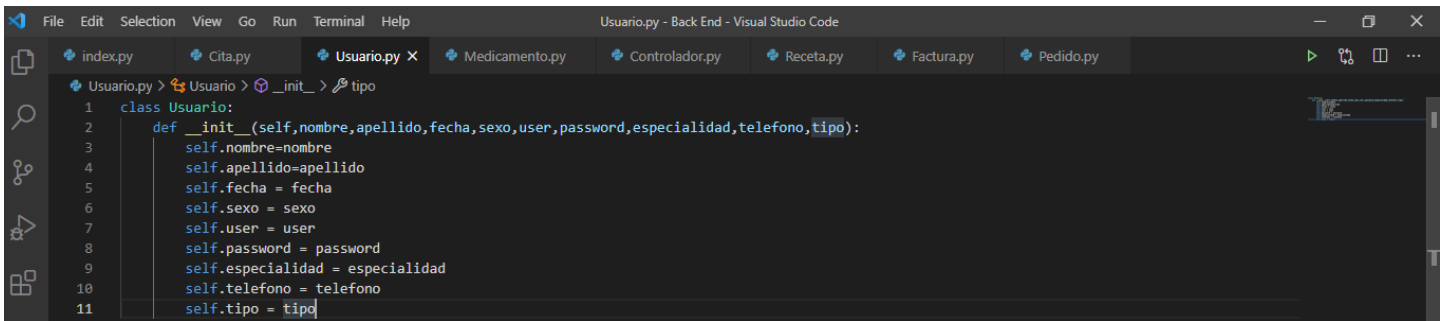
- Teclado
- Mouse
- Pantalla con resolución mínima de 240p y 144p
- CPU

# DESCRIPCION



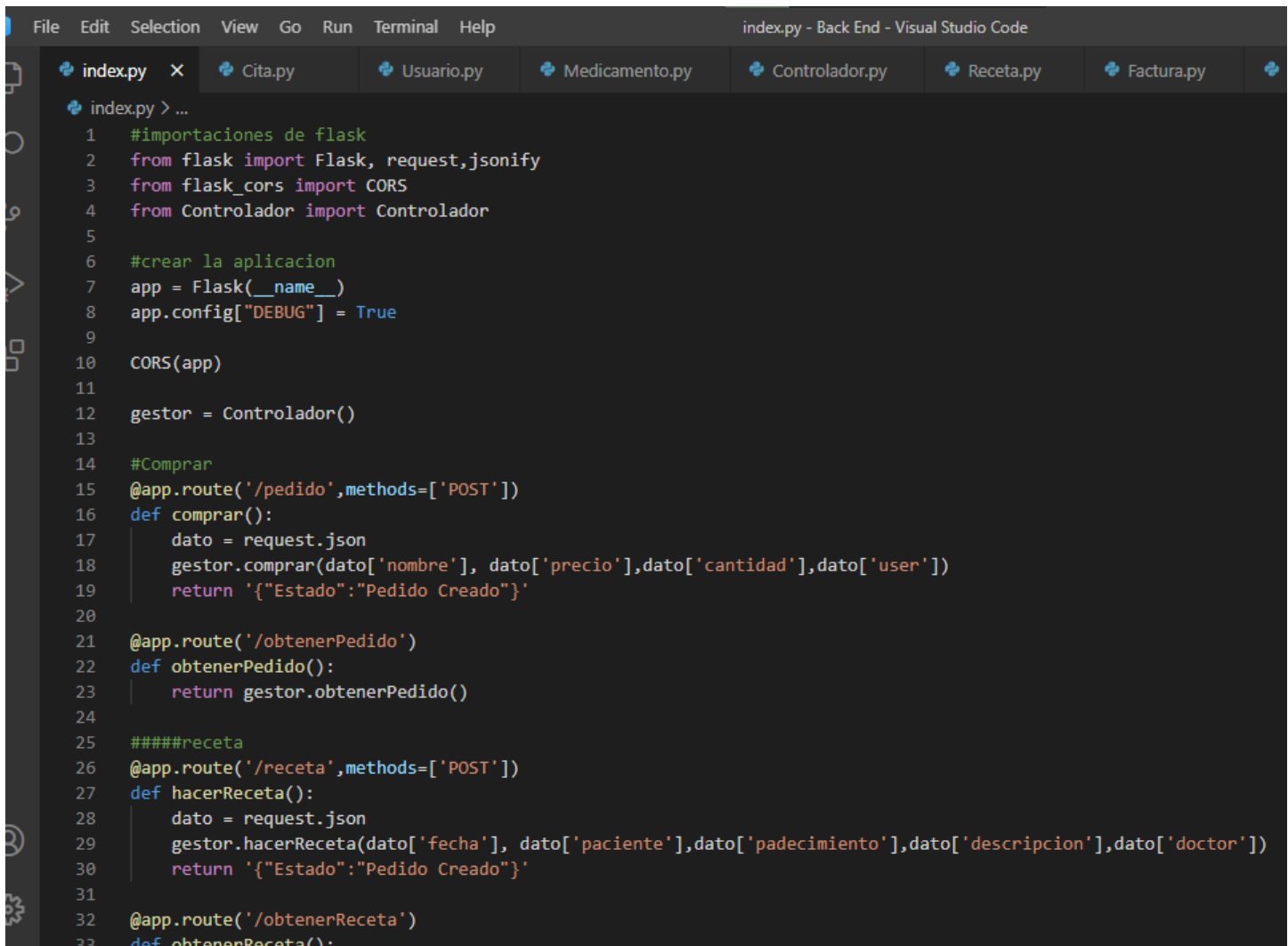
```
1 class Usuario:
2     def __init__(self,nombre,apellido,fecha,sexo,user,password,especialidad,telefono,tipo):
3         self.nombre=nombre
4         self.apellido=apellido
5         self.fecha = fecha
6         self.sexo = sexo
7         self.user = user
8         self.password = password
9         self.especialidad = especialidad
10        self.telefono = telefono
11        self.tipo = tipo
```

La parte del BACKend fue diseñada exclusivamente con Python.



```
1 class Usuario:
2     def __init__(self,nombre,apellido,fecha,sexo,user,password,especialidad,telefono,tipo):
3         self.nombre=nombre
4         self.apellido=apellido
5         self.fecha = fecha
6         self.sexo = sexo
7         self.user = user
8         self.password = password
9         self.especialidad = especialidad
10        self.telefono = telefono
11        self.tipo = tipo
```

Se crearon diferentes objetos donde se almacenan los usuarios, las citas, medicamentos, facturas, pedidos y recetas para su posterior acceso.



```
index.py > ...
1  #importaciones de flask
2  from flask import Flask, request, jsonify
3  from flask_cors import CORS
4  from Controlador import Controlador
5
6  #crear la aplicacion
7  app = Flask(__name__)
8  app.config["DEBUG"] = True
9
10 CORS(app)
11
12 gestor = Controlador()
13
14 #Comprar
15 @app.route('/pedido', methods=['POST'])
16 def comprar():
17     dato = request.json
18     gestor.comprar(dato['nombre'], dato['precio'], dato['cantidad'], dato['user'])
19     return '{"Estado": "Pedido Creado"}'
20
21 @app.route('/obtenerPedido')
22 def obtenerPedido():
23     return gestor.obtenerPedido()
24
25 #####receta
26 @app.route('/receta', methods=['POST'])
27 def hacerReceta():
28     dato = request.json
29     gestor.hacerReceta(dato['fecha'], dato['paciente'], dato['padecimiento'], dato['descripcion'], dato['doctor'])
30     return '{"Estado": "Pedido Creado"}'
31
32 @app.route('/obtenerReceta')
33 def obtenerReceta():
```

Se crearon rutas de acceso a los métodos CRUD para cada objeto creado, de esta manera se puede realizar los cambios necesarios desde el front al back.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8" />
5   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6   <meta http-equiv="X-UA-Compatible" content="ie=edge" />
7   <title>UHospital - Administración</title>
8
9   <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Open+Sans:400,600" />
10  <link rel="stylesheet" href="../../Administrador/partesVer/css/all.min.css" />
11  <link rel="stylesheet" href="../../Administrador/partesVer/css/bootstrap.min.css" />
12  <link rel="stylesheet" type="text/css" href="../../Administrador/partesVer/slick/slick.css"/>
13  <link rel="stylesheet" type="text/css" href="../../Administrador/partesVer/slick/slick-theme.css"/>
14  <link rel="stylesheet" href="../../Administrador/partesVer/css/templatemo-style.css" />
15  <link rel="stylesheet" href="../../Administrador/partesVer/style.css">
16
17  <meta name="description" content="">
18  <meta name="viewport" content="width=device-width, initial-scale=1">
19
20  <script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/1.4.1/jspdf.min.js"></script>
21  <script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf-autotable/2.3.5/jspdf.plugin.autotable.min.js"></script>
22
23  <!-- Bootstrap CSS -->
24  <link rel="stylesheet" href="../../Administrador/partesVer/assets/css/bootstrap.min.css">
25
26  <!-- Style CSS -->
```

La parte del front fue diseñada con HTML, se utilizó CSS para darle un estilo más presentable al proyecto

```

administrador > JS acciones.js > crearpdf > then() callback
1  function createHeaders(keys) {
2      var result = [];
3      for (var i = 0; i < keys.length; i += 1) {
4          result.push({
5              id: keys[i],
6              name: keys[i],
7              prompt: keys[i],
8              width: 55,
9              align: "center",
10             padding: 0
11         });
12     }
13     return result;
14 }
15
16 > function convertirdata(paciente){ ...
51     }
52
53
54
55
56
57 function crearpdf(){
58
59     fetch('http://34.121.228.56:5000/obtenerPacientes')
60     .then(response => response.json())
61     .then(data=>{
62         //Declarando los headers

```

Las acciones y la forma de unir el front con el back se realizo por medio de javascript.

Aquí se tienen los métodos para la creación de pdf y la recolección de datos para los objetos del back.