Un dibujo de un perro

Descripción generada automáticamente con confianza mediaLogotipo

Descripción generada automáticamente

Icono

Descripción generada automáticamente

Evaluación 3

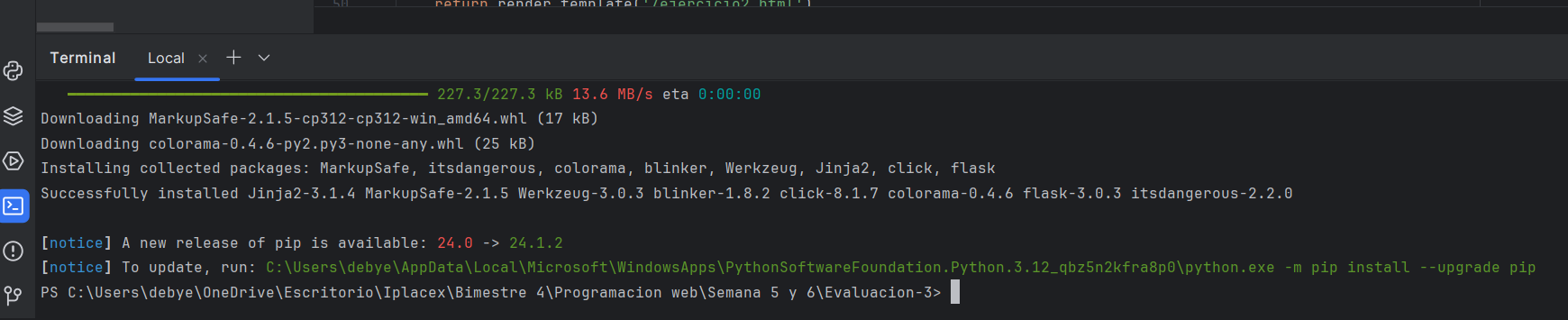
**2024-2B-PROGRAMACIÓN WEB/201-CED-QM**

**Debye Igor Verdejo**

**Instalando FLASK**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente Abrimos la terminal en pycharm y ejecutamos el comando **pip install flask** esperamos que se complete el proceso



Para comprobar la instalación y versión de flask utilizamos el comando **python -m flask –version**

**Texto

Descripción generada automáticamente**

Crear carpeta del proyecto

Seleccionamos en crear nuevo proyecto, le asignamos nombre y elegimos la versión del intérprete de Python y nos aseguramos que tengamos habilitada la opción **Virtualenv** para el ambiente de trabajo   
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Creamos las carpetas del proyecto, haciendo click secundario en la carpeta principal new-directory.

Creamos 2 carptas static y templates.

**Captura de pantalla con la imagen de una pantalla

Descripción generada automáticamente**

Repetimos el proceso y creamos 2 carpetas (css - img) dentro de la carpeta static.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

En el archivo run.py (main) importamos flask y las funciones que usaremos (request y render\_template) **from flask import Flask, request, render\_template**

Iniciamos Flask y configuramos la ruta para la página index. (‘/’)

Texto

Descripción generada automáticamente

En la carpeta templates creamos un nuevo archivo HTML y lo nombramos index

Captura de pantalla de computadora

Descripción generada automáticamente

Configuramos el archivo index.html según el requerimiento con 2 botones que redirijan a las páginas del ejercicio 1 y 2.

Aquí le agregue un título y subtitulo además de crear un contenedor con la clase “container” para todos los elementos dentro de el, a los botones también les agregue una clase.

Para que se refleje correctamente el estilo css incorporado, en el head del documento hay que incorporar la etiqueta **<link rel="stylesheet" href="/static/css/estilos.css">**  que hace referencia a la ubicación de nuestra hoja de estilos.

Texto

Descripción generada automáticamente

Hoja de estilos css, aquí agregamos las atributos para darle diseño a nuestro contenedor y al cuerpo de la página, les cambio el color de fondo para que se note diferencia, entre otros ajustes.

Los archivos CSS se guardan en la carpeta /static/css

Texto

Descripción generada automáticamente

Ejecutamos el archivo run.py haciendo click secundario y presionando run, o en la parte superior derecha en el icono verde de play.

Texto

Descripción generada automáticamente

Nos abre nuestro servidor local, clickeamos la ip y abrimos el proyecto en la pagina de incio

Texto

Descripción generada automáticamente

Nos abre en el navegador nuestro proyecto con los 2 botones

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

**Ejercicio 1**

En templeates creamos la pagina HTML ejercicio1 y la configuramos según el requerimiento, un formulario con campos para ingresar notas y para ingresar la asistencia y que al enviar calcule el promedio y entregue el resultado si esta aprobado.

Importamos la hoja de estilos css en el header ( es importante hay veces que se olvida y no se visualiza correctamente el diseño)

Creamos un contenedor para todos los elementos, en este caso agregue una imagen en la cabecera, las imágenes se guardan en la carpeta static/img,.

Insertamos un titulo y bajo un formulario con la etiqueta **<form method="post" class="form">** es importante especificar aquí el **method="post"**  para la recopilación correcta de los datos( si no esta, no va a funcionar).

Creamos los Label e input asociados entre si y los configuramos para que sean de tipo numero, con un mínimo de 0 y máximo de 70 en las notas y mínimo 0 y máximo 100 en la asistencias, además de hacerlos obligatorios. El ultimo input lo asignamos con tipo **submit**

Texto

Descripción generada automáticamente

Bajo el formulario agregamos unos textos con las variables que vamos a trabajar  
y bajo esto agregamos un botón para volver al inicio.

Abrimos el archivo run.py y configuramos la nueva página y sus funciones

**@app.route('/ejercicio1', methods=['GET', 'POST'])** Seleccionamos la ruta y agregamos los métodos para la obtención de los datos.

Recopilamos los datos de las variables del formulario con la función **request** y además le damos el formato de INT a la variable, **creamos la variable promedio** para calcularlo y además con un if hacemos las comparaciones y **creamos la variable** **estado.**

En este caso agregue más opciones de comparación para que muestre un mensaje personalizado para cada situación.

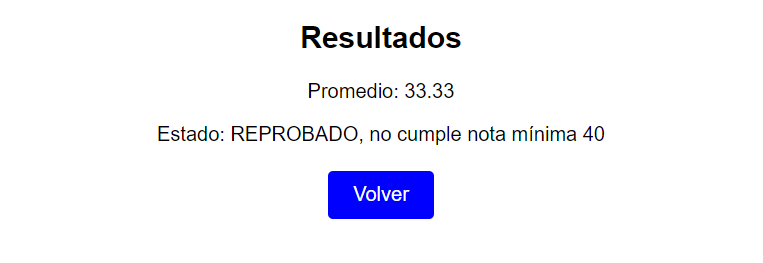
Renderizamos la página con las variables que agregamos en la pagina ejercicio1 y luego renderizamos toda la página para su correcta visualización respetando la estructura.

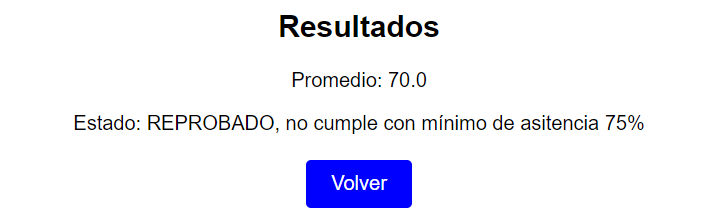
Texto

Descripción generada automáticamente

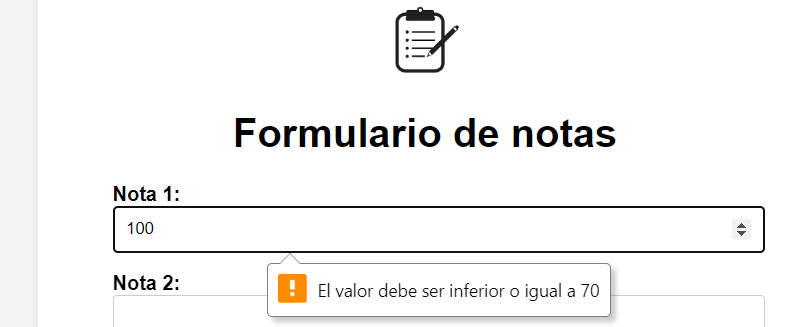
Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteEjecutamos, accedemos a la página 1 y realizamos pruebas









Cargando archivos a GitHub (Carga de este documento al repositorio)

Texto

Descripción generada automáticamente

**Ejercicio 2**

Creamos una nueva pagina HTML en la carpeta templates y le asignamos el nombre ejercicio2

Creamos el formulario para ingresar los nombres y en este caso los tipos de los campos son de texto, además de crear una nueva clase llamada “form-2” para este formulario, sus requerimientos y forma de mostrar los resultados

Texto

Descripción generada automáticamente

En el archivo run.py agregamos la ruta y creamos la función para resolver el requerimiento.

En la ruta agregamos los métodos y recopilamos la información del formulario.

Luego ejecutamos un if para realizar las comparaciones con la función **len** para identificar la cantidad de caracteres de cada nombre y asignar los valores a las variables resultado y cantidad. Realizamos los render correspondientes y ejecutamos el programa.

Nota: Haciendo este Word me di cuenta que también se puede realizar el análisis del largo de las variables, agregándolas todas a una lista y luego trabajar con la lista, no lo he probado pero quizás es menos el código que se escribe)

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

**Autoevaluación**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criterios de revisión del informe con la información recopilada:** | **Completamente logrado**  **(4)** | **Medianamente logrado**  **(3)** | **Parcialmente logrado**  **(2)** | **No logrado**  **(0)** |
| 1. Incorporé los elementos básicos de Python | x |  |  |  |
| 1. Codifiqué correctamente los requerimientos solicitados para el encargo. | x |  |  |  |
| 1. Utilicé correctamente los comandos de GIT para subir el proyecto a mi repositorio. | x |  |  |  |
| 1. Finalicé la actividad, comprendiendo cada uno de los ejercicios desarrollados | x |  |  |  |
| 1. Logré el objetivo planteado para esta actividad | x |  |  |  |
| **Puntaje Total** | **20 puntos** | | | |
| **Puntaje Obtenido** | **20 Puntos obtenidos** | | | |
|  |  | | | |