



UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Taller 2 - Corte II

Alvaro Javier Leon Lugo

20242020279

Angie Xiomara Fuentes Montañez

20242020265

Programación Avanzada

Universidad Distrital

Respuesta a preguntas

1. ¿Qué ventajas encontraste al aplicar el patrón MVC en tu proyecto?

Gracias a la aplicación del modelo MVC se redujo significativamente el código dentro del main, además de que permite que las funciones estén de manera más organizada en el código, además si lo pensamos desde la aplicabilidad, si quiere extenderse el código no requiere realizar grandes cambios a una cantidad considerable de código, si no esa parte en específica, sin tener que afectar a todas.

2. ¿Qué dificultades surgieron al separar las responsabilidades entre modelo, vista y controlador?

Era difícil pensar que clases iban en qué parte del programa, puesto que al momento de hacer la ejecución del programa existían inconvenientes para la división de modelo/vista/controlador, y en cuanto a la ejecución hubo problemas iniciales en cuanto a la comunicación entre las carpetas, ya que la comunicarlas salía un error el cual decía que no sabía donde estaba esa carpeta, sin embargo pudimos suplirlas al final, gracias a entender la organización inicial de las carpetas.

3. ¿Qué aprendiste sobre la ejecución de sentencias SQL y su relación con la estructura del código?

Relacionado con la estructura para nuestro código permitió que existiera una división por responsabilidades (como la lógica de modelo), además de que ayuda a que el código sea más organizado. SQL está especializado en bases de datos por lo que permite una interacción más sencilla con la base de datos, como las comparaciones o búsquedas. Además los comandos en sí mismos hacían más fácil entender a qué parte o que componente afectan, lo cual contribuye a la forma en cómo se estructuró.

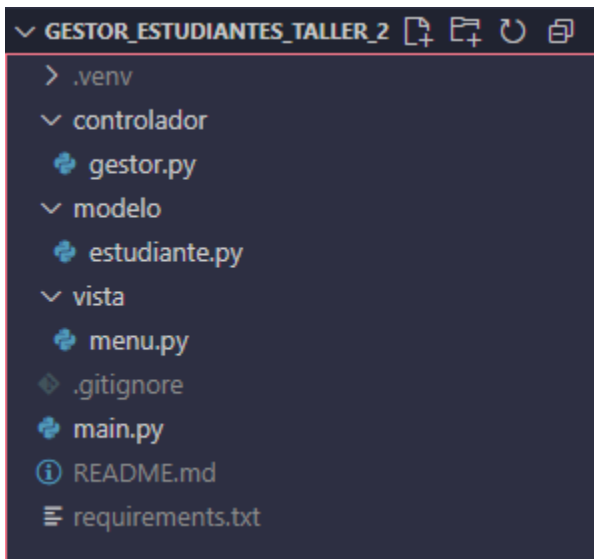
4. ¿Qué mejorarías si tuvieras que ampliar el sistema?

Podrían separarse muchísimo más las partes del código, siguiendo más estrictamente el principio de responsabilidad única, lo cual ayudaría a la hora de agregar nuevas funcionalidades. Además se podría plantear el uso de clases abstractas, lo cual ayudaría a la hora de extender funcionalidades y reducir el código.

Por otro lado se podrían haber creado más funcionalidades de validación de datos, procurando que el código no se vare por un error, dando mensajes claros y concisos sobre el error.

Evidencias de ejecución

Estructura general del programa: Separación por carpetas modelo/vista/controlador, cada una con su respectivo [archivo.py](#) gestor/esutidante/menu




Menu inicial al ejecutar main.py: Al iniciar el programa lo primero que se desplegará será este menú, en el cual se encuentran todas las opciones disponibles del programa, además otra particularidad, siempre al concluir exitosamente una acción (excepto la opción 0), se volverá a ejecutar automáticamente.

```
--- Gestor Estudiantes ---
1. Crear base de datos
2. Agregar estudiante
3. Listar estudiantes
4. Consultar por nota minima
5. Actualizar nota
6. Eliminar estudiante por nombre
7. Eliminar estudiantes con nota < 3.0
8. Eliminar estudiantes con nota > 3.0
9. Buscar estudiante por una cadena de texto
10. Mostrar estudiantes ordenados por nota descendente
0. Salir
Seleccione una opcion:
```

Función #1: Crear base de datos: Si se selecciona la opción 1, saldrá el siguiente mensaje de confirmación, y además creará un archivo.db en el que se estara la base de datos.

```
Seleccione una opcion: 1
[OK] Base de datos estudiantes.db creada y lista para uso.
```

A screenshot of a file explorer showing a file named 'estudiantes.db'.

Función #2: Agregar estudiante: Si se selecciona la opción 2, pedirá el nombre, el cual se deberá ingresar por teclado, el correo, el cual se deberá ingresar por teclado y la nota la el cual se deberá ingresar por teclado, una vez realizado ello se confirmara que se agregó.

```
Seleccione una opcion: 2
Nombre: Pablo
Correo: pablo@gmail.com
Nota: 3.4
[OK] Estudiante agregado correctamente.
```

Función #3: Listar estudiantes: Si se selecciona la opción 2, esta nos desplegará la base datos en la pantalla, la cual estará organizada evidenciando toda la información que se le ha agregado, (solo si ya se adiciono estudiantes).

```
Seleccione una opcion: 3

[INFO] Listado de estudiantes:
-----
1 | Pablo      | pablo@gmail.com   | Nota: 3.40
2 | Ana        | ana@gmail.com     | Nota: 2.50
3 | Daniel     | daniel@yahoo.com  | Nota: 2.80
```

Funcion #4: Consultar por nota mínima: Si se selecciona la opción 4, pedirá primero un parámetro el cual deberá ingresar por teclado, en el que suministra la nota mínima para filtrar los datos que se van a mostrar en pantalla.

```
Seleccione una opcion: 4
Ingrese la nota minima: 2.6

[INFO] Estudiantes con nota >= 2.6
-----
Pablo      | 3.40
Daniel     | 2.80
```

Funcion #5: Actualizar nota: Si se selecciona la opción 5, pedirá primero ingresar por teclado el nombre del estudiante exacto al cual se le quiere cambiar la nota, una vez comprobado, se pedirá ingresar por teclado la nota a la que se actualiza. Como vemos al consultar otra vez los datos con la opción 3, exitosamente se cambia.

```
Seleccione una opcion: 5
Nombre del estudiante: Ana
Nueva nota: 3.4
[OK] Nota actualizada correctamente.
```

```
2 | Ana        | ana@gmail.com     | Nota: 3.40
```

Función #6: Eliminar estudiante por nombre: Si se selecciona la opción 6, pedirá ingresar por teclado el nombre del estudiante exacto al cual se le quiere eliminar del registro, una vez encontrarlo lo elimina, sale el mensaje de confirmación, y al consultar otra vez los datos con la opción 3, ya no saldra dentro del registro.

```
5 | Carlos     | carlos@gmail.com  | Nota: 2.50
Nombre del estudiante a eliminar: Carlos
[OK] Estudiante eliminado correctamente.
```

Función #7: Eliminar estudiantes con nota < 3.0: (listado antes, listado después)

Si se selecciona la opción 7, automáticamente eliminará todos los registros que tenemos que tengan una nota inferior a 3. Como vemos en las imágenes tenemos varios registros, se ejecuta la opción 7, y al regresar con la opción 3, ya no salen los que tenían la nota inferior, así que funciona exitosamente.

```
[INFO] Listado de estudiantes:
-----
1 | Pablo      | pablo@gmail.com      | Nota: 3.40
2 | Ana        | ana@gmail.com        | Nota: 3.40
3 | Daniel     | daniel@yahoo.com     | Nota: 2.80
4 | Nicol      | nicol@outlook.com    | Nota: 2.10
5 | Carlos     | carlos@gmail.com     | Nota: 2.50
6 | Luna       | luna@gmail.com       | Nota: 4.60
```

```
Seleccione una opcion: 7
[OK] Registros con nota menor a 3.0 eliminados.
```

```
[INFO] Listado de estudiantes:
-----
1 | Pablo      | pablo@gmail.com      | Nota: 3.40
2 | Ana        | ana@gmail.com        | Nota: 3.40
6 | Luna       | luna@gmail.com       | Nota: 4.60
```

Función #8: Eliminar estudiantes con nota >3.0: Si se selecciona la opción 8, automáticamente eliminará todos los registros que tenemos que tengan una nota mayor a 3. Se ejecuta la opción 8, y al regresar con la opción 3, ya no salen los que tenían la nota mayor a 3, así que funciona exitosamente.

```
Seleccione una opcion: 8
[OK] Registros con nota mayor a 3.0 eliminados.
```

```
[INFO] Listado de estudiantes:
-----
7 | Daniel     | daniel@yahoo.com     | Nota: 2.80
8 | Nicol      | nicol@outlook.com    | Nota: 2.10
9 | Carlos     | carlos@gmail.com     | Nota: 2.50
```

Función #9: Si se selecciona la opción 9, lo primero es que pedirá es la cadena de caracteres ingresada por teclado y la buscará dentro de la base de datos y solo listará los que cumplan ese parámetro, como vemos tenemos muchos registros, ingresamos la cadena, y solo lista los que cumplen con ello.

```
[INFO] Listado de estudiantes:
-----
7 | Daniel     | daniel@yahoo.com     | Nota: 2.80
8 | Nicol      | nicol@outlook.com    | Nota: 2.10
9 | Carlos     | carlos@gmail.com     | Nota: 2.50
10 | Javier     | javier@gmail.com     | Nota: 3.10
11 | Milena     | milena@gmail.com     | Nota: 4.50
```

```
Seleccione una opcion: 9
Ingrese parte del nombre a buscar: mil
```

```
[INFO] Resultados de bussqueda:
-----
Milena      | milena@gmail.com     | Nota: 4.50
```

```
Seleccione una opcion: 9
Ingrese parte del nombre a buscar: pab
```

Función #10: Estudiantes ordenados por nota descendiente: El sistema mostrará una lista con los estudiantes registrados y organizará las notas de mayor a menor, en donde se mostrará a quién pertenece cada nota.

Seleccione una opcion: 10

[INFO] Estudiantes ordenados por nota (DESC):

| | | |
|--------|-------------------|------------|
| Milena | milena@gmail.com | Nota: 4.50 |
| Javier | javier@gmail.com | Nota: 3.10 |
| Daniel | daniel@yahoo.com | Nota: 2.80 |
| Carlos | carlos@gmail.com | Nota: 2.50 |
| Nicol | nicol@outlook.com | Nota: 2.10 |

Función #0: Salir del sistema: Finaliza la ejecución del programa.

Seleccione una opcion: 0

Saliendo del sistema