

Universidad Gerardo Barrios  
Facultad de Ciencia y Tecnología



Integrantes:

Fabiola Alejandra Benítez Osorto – SMSS111223

Andrea Patricia Ramos Hernández – SMSS101123 Ivette

Azucena Mendiola Requeno – SMSS063723

Docente:

William Alexis Montes Girón

Asignatura:

Programación Computacional II

Fecha de entrega:

Domingo 24 de noviembre de 2024

1. Resolución del plan de trabajo detallado en el tercer avance: Desarrollar las funcionalidades pendientes del proyecto que estén alineadas con las descritas en el segundo avance para poder implementarlas al proyecto dejando pendientes algunas funcionalidades de baja prioridad y ajustes menores.

Las funcionalidades que se aplicaron en este tercer avance fueron la de agregar un botón de eliminar los datos o registros que se realizaron de igual manera añadir una función que permitiría modificar los datos que se pusieron en este caso en nuestro registro de notas.

2. Objetivos Faltantes y Plan de Desarrollo: Describir detalladamente las funcionalidades y componentes planificados para la entrega final. Explicar brevemente cómo se planea desarrollar cada parte faltante (tecnologías, métodos o bibliotecas a utilizar) y cómo ha cambiado su estado con respecto a la entrega anterior.

Al final del proyecto planeamos entregar lo que es una base de datos que presente todos los registros guardados ya que hasta el momento solo cuenta con las funcionalidades descritas en nuestros anteriores avances y con las modificaciones del presente avance que sería el del botón modificar y el botón eliminar.

3. Alcance logrado del desarrollo: Describir brevemente sobre cómo este avance contribuye al cumplimiento del propósito general del proyecto.

Bueno este avance contribuyo de gran manera porque nuestro objetivo principal es facilitar el registro de las notas que este caso sería para los docentes de igual manera para que tengan un mayor orden al momento del registro.

```
File Edit Selection View Go Run ... < -> Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN\Downloads> registro_notas_actualizado (1).py > ...
1 import sys
2 from PyQt5.QtWidgets import QApplication, QMainWindow, QComboBox, QLineEdit, QPushButton, QVBoxLayout, QWidget, QTableWidgetItem, QTableWidgetItem
3
4
5 class VentanaPrincipal(QMainWindow):
6     def __init__(self):
7         super().__init__()
8
9         # Configuración de la ventana
10        self.setWindowTitle("Información del Docente")
11        self.setGeometry(100, 100, 400, 300)
12
13        # Crear el layout principal
14        layout = QVBoxLayout()
15
16        # Etiquetas y campos para ingresar datos del docente
17        self.docente_label = QLabel("Nombre del Docente:")
18        self.docente_input = QLineEdit()
19
20        self.materia_label = QLabel("Materia:")
21        self.materia_combo = QComboBox()
22        self.materia_combo.addItem("Matemáticas", "Inglés", "Lenguaje", "Sociales", "Ciencias"])
23
24        self.curso_label = QLabel("Curso:")
25        self.curso_combo = QComboBox()
26        self.curso_combo.addItem("7º", "8º", "9º", "1º Bachillerato", "2º Bachillerato")
```

```
File Edit Selection View Go Run ... < -> Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN\Downloads> registro_notas_actualizado (1).py > VentanaPrincipal > __init__
5 class VentanaPrincipal(QMainWindow):
6     def __init__(self):
7
8         # Botón para ingresar
9         self.ingresar_button = QPushButton("Ingresar")
10        self.ingresar_button.clicked.connect(self.abrir_registro_notas)
11
12        # Agregar widgets al layout
13        layout.addWidget(self.docente_label)
14        layout.addWidget(self.docente_input)
15        layout.addWidget(self.materia_label)
16        layout.addWidget(self.materia_combo)
17        layout.addWidget(self.curso_label)
18        layout.addWidget(self.curso_combo)
19        layout.addWidget(self.ingresar_button)
20
21        # Crear un widget central y asignarle el layout
22        container = QWidget()
23        container.setLayout(layout)
24        self.setCentralWidget(container)
25
26        # Aplicar estilo CSS
27        self.setStyleSheet("""
28            QWidget {
29                font-family: Arial, sans-serif;
30                background-color: #f4f4f4;
31            }
32            QLabel {
33                font-size: 14px;
34                margin-bottom: 5px;
35            }
36        """)
```

```
File Edit Selection View Go Run ... < -> Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN>Downloads>registro_notas_actualizado (1).py>VentanaPrincipal>__init__
5 class VentanaPrincipal(QMainWindow):
6     def __init__(self):
7
8         # Aplicar estilo CSS
9         self.setStyleSheet("""
10             QWidget {
11                 font-family: Arial, sans-serif;
12                 background-color: #f4f4f4;
13             }
14             QLabel {
15                 font-size: 14px;
16                 margin-bottom: 5px;
17             }
18             QLineEdit, QComboBox, QPushButton {
19                 padding: 8px;
20                 font-size: 14px;
21                 border: 1px solid #ccc;
22                 border-radius: 5px;
23             }
24             QLineEdit:focus, QComboBox:focus {
25                 border-color: #007bff;
26             }
27             QPushButton {
28                 background-color: #007bff;
29                 color: white;
30                 border: none;
31                 font-weight: bold;
32             }
33             QPushButton:hover {
34                 background-color: #0056b3;
35             }
36         """)
37
38     def abrir_registro_notas(self):
39         # Validar que todos los campos estén llenos
40         if self.docente_input.text() and self.materia_combo.currentText() and self.curso_combo.currentText():
41             # Cerrar esta ventana y abrir la de RegistroNotas
42             self.registro_notas = RegistroNotas()
43             self.registro_notas.show()
44             self.close()
45         else:
46             # Mensaje de error
47             QMessageBox.warning(self, "Error", "Todos los campos son obligatorios.")
48
49     def cerrar_registro_notas(self):
50         self.close()
51
52     def __del__(self):
53         self.close()
54
55 if __name__ == '__main__':
56     app = QApplication(sys.argv)
57     ventana = VentanaPrincipal()
58     ventana.show()
59     sys.exit(app.exec_())
```

```
File Edit Selection View Go Run ... < -> Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN>Downloads>registro_notas_actualizado (1).py>VentanaPrincipal>__init__
5 class VentanaPrincipal(QMainWindow):
6     def __init__(self):
7
8         # Aplicar estilo CSS
9         self.setStyleSheet("""
10             QWidget {
11                 font-family: Arial, sans-serif;
12                 background-color: #f4f4f4;
13             }
14             QLabel {
15                 font-size: 14px;
16                 margin-bottom: 5px;
17             }
18             QLineEdit, QComboBox, QPushButton {
19                 padding: 8px;
20                 font-size: 14px;
21                 border: 1px solid #ccc;
22                 border-radius: 5px;
23             }
24             QLineEdit:focus, QComboBox:focus {
25                 border-color: #007bff;
26             }
27             QPushButton {
28                 background-color: #007bff;
29                 color: white;
30                 border: none;
31                 font-weight: bold;
32             }
33             QPushButton:hover {
34                 background-color: #0056b3;
35             }
36             QComboBox {
37                 background-color: white;
38                 color: #333;
39             }
40         """)
41
42     def abrir_registro_notas(self):
43         # Validar que todos los campos estén llenos
44         if self.docente_input.text() and self.materia_combo.currentText() and self.curso_combo.currentText():
45             # Cerrar esta ventana y abrir la de RegistroNotas
46             self.registro_notas = RegistroNotas()
47             self.registro_notas.show()
48             self.close()
49         else:
50             # Mensaje de error
51             QMessageBox.warning(self, "Error", "Todos los campos son obligatorios.")
52
53     def cerrar_registro_notas(self):
54         self.close()
55
56     def __del__(self):
57         self.close()
58
59 if __name__ == '__main__':
60     app = QApplication(sys.argv)
61     ventana = VentanaPrincipal()
62     ventana.show()
63     sys.exit(app.exec_())
```

```
registro_notas_actualizado (1).py
C: > Users > ADMIN > Downloads > registro_notas_actualizado (1).py > RegistroNotas > _init_
5  :class VentanaPrincipal(QMainWindow):
80  def abrir_registro_notas(self):
88      # Mostrar un mensaje de error si hay campos vacíos
89      QMessageBox.warning(self, "Error", "Por favor complete todos los campos.")
90
91
92  :class RegistroNotas(QMainWindow):
93  def __init__(self):
94      super().__init__()
95
96      # Configuración de la ventana
97      self.setWindowTitle("Registro de Notas")
98      self.setGeometry(100, 100, 600, 400)
99
100     # Crear el layout principal
101     layout = QVBoxLayout()
102
103     # Selección de estudiante
104     self.estudiante_label = QLabel("Estudiante:")
105     self.estudiante_combo = QLineEdit()
106
107     # Selección de periodo
108     self.periodo_label = QLabel("Periodo:")
109     self.periodo_combo = QComboBox()
110     self.periodo_combo.addItem("Trimestre 1", "Trimestre 2", "Trimestre 3")
111
```

```
registro_notas_actualizado (1).py
C: > Users > ADMIN > Downloads > registro_notas_actualizado (1).py > RegistroNotas > _init_
92  class RegistroNotas(QMainWindow):
93  def __init__(self):
111
112     # Selección de tipo de evaluación
113     self.tipo_eval_label = QLabel("Tipo de Evaluación:")
114     self.tipo_eval_combo = QComboBox()
115     self.tipo_eval_combo.addItem("Examen", "Tareas", "Proyecto")
116
117     # Entrada de la nota
118     self.nota_label = QLabel("Nota:")
119     self.nota_input = QLineEdit()
120
121     # Botones para guardar y eliminar
122     self.save_button = QPushButton("Guardar Nota")
123     self.save_button.clicked.connect(self.guardar_nota)
124
125     self.delete_button = QPushButton("Eliminar Nota")
126     self.delete_button.clicked.connect(self.eliminar_nota)
127
128     # Tabla para mostrar notas
129     self.table = QTableWidgetItem(0, 4) # 4 columnas para estudiante, periodo, tipo de evaluación, nota
130     self.table.setHorizontalHeaderLabels(["Estudiante", "Periodo", "Evaluación", "Nota"])
131
```

```
File Edit Selection View Go Run ... Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN>Downloads>registro_notas_actualizado (1).py>RegistroNotas>__init__
92 class RegistroNotas(QMainWindow):
93     def __init__(self):
132         # Organizar widgets en el layout
133         layout.addWidget(self.estudiante_label)
134         layout.addWidget(self.estudiante_combo)
135         layout.addWidget(self.periodo_label)
136         layout.addWidget(self.periodo_combo)
137         layout.addWidget(self.tipo_eval_label)
138         layout.addWidget(self.tipo_eval_combo)
139         layout.addWidget(self.nota_label)
140         layout.addWidget(self.nota_input)
141         layout.addWidget(self.save_button)
142         layout.addWidget(self.delete_button)
143         layout.addWidget(self.table)
144
145         # Crear un widget central y asignarle el layout
146         container = QWidget()
147         container.setLayout(layout)
148         self.setCentralWidget(container)
149
```

```
File Edit Selection View Go Run ... Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN>Downloads>registro_notas_actualizado (1).py>RegistroNotas>__init__
92 class RegistroNotas(QMainWindow):
93     def __init__(self):
149
150         # Aplicar estilo CSS
151         self.setStyleSheet("""
152             QWidget {
153                 font-family: Arial, sans-serif;
154                 background-color: #f4f4f4;
155             }
156             QLabel {
157                 font-size: 14px;
158                 margin-bottom: 5px;
159             }
160             QLineEdit, QComboBox, QPushButton {
161                 padding: 8px;
162                 font-size: 14px;
163                 border: 1px solid #ccc;
164                 border-radius: 5px;
165             }
166             QLineEdit:focus, QComboBox:focus {
167                 border-color: #007bff;
168             }
169             QPushButton {
170                 background-color: #28a745;
171                 color: white;
172                 border: none;
173                 font-weight: bold;
174             }
175             QPushButton:hover {
176                 background-color: #218838;
177             }
178         """)
```

```
File Edit Selection View Go Run ... < -> Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN>Downloads>registro_notas_actualizado (1).py>RegistroNotas>__init__
92 class RegistroNotas(QMainWindow):
93     def __init__(self):
175         QPushButton {
176             background-color: #218838;
177         }
178         QComboBox {
179             background-color: white;
180             color: #333;
181         }
182         QTableWidget {
183             background-color: white;
184             border-radius: 5px;
185             border: 1px solid #ccc;
186             margin-top: 10px;
187         }
188         QTableWidgetItem {
189             padding: 5px;
190         }
191         QHeaderView::section {
192             background-color: #007bff;
193             color: white;
194             padding: 5px;
195         }
196     """
197
198     def guardar_nota(self):
```

```
File Edit Selection View Go Run ... < -> Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN>Downloads>registro_notas_actualizado (1).py>RegistroNotas>guardar_nota
92 class RegistroNotas(QMainWindow):
198     def guardar_nota(self):
199         # Obtener los datos ingresados
200         estudiante = self.estudiante_combo.text()
201         periodo = self.periodo_combo.currentText()
202         tipo_eval = self.tipo_eval_combo.currentText()
203         nota = self.nota_input.text()
204
205         # Verificar que la nota sea válida
206         if nota.isdigit() and 0 <= int(nota) <= 10:
207             # Agregar los datos a la tabla
208             fila = self.table.rowCount()
209             self.table.insertRow(fila)
210             self.table.setItem(fila, 0, QTableWidgetItem(estudiante))
211             self.table.setItem(fila, 1, QTableWidgetItem(periodo))
212             self.table.setItem(fila, 2, QTableWidgetItem(tipo_eval))
213             self.table.setItem(fila, 3, QTableWidgetItem(nota))
214
215             # Limpiar los campos de entrada
216             self.estudiante_combo.clear()
217             self.nota_input.clear()
218         else:
219             # Mostrar un mensaje de error
220             QMessageBox.warning(self, "Error", "Ingrese una nota válida entre 0 y 10.")
221
222     def eliminar_nota(self):
```

```
File Edit Selection View Go Run ... Fabiola Benitez
registro_notas_actualizado (1).py
C:\Users\ADMIN\Downloads> registro_notas_actualizado (1).py > RegistroNotas > guardar_nota
92 class RegistroNotas(QMainWindow):
198     def guardar_nota(self):
220         QMessageBox.warning(self, "Error", "Ingrese una nota válida entre 0 y 10.")
221
222     def eliminar_nota(self):
223         # Obtener la fila seleccionada
224         fila_seleccionada = self.table.currentRow()
225         if fila_seleccionada >= 0:
226             self.table.removeRow(fila_seleccionada)
227         else:
228             QMessageBox.warning(self, "Error", "Seleccione una fila para eliminar.")
229
230
231 # Código para ejecutar la aplicación
232 if __name__ == "__main__":
233     app = QApplication([])
234     ventana_principal = VentanaPrincipal()
235     ventana_principal.show()
236     app.exec_()
237
```

File Edit Selection View Go Run ...

Información del ...

Nombre del Docente:

Materia:

Matemáticas

Curso:

7°

Ingresar

self.setWindowTitle("I



Registro de Notas

Estudiante:

patricia

Periodo:

Trimestre 1

Tipo de Evaluación:

Examen

Nota:

9

Guardar Nota

Eliminar Nota

Estudiante	Periodo	Evaluación	Nota
------------	---------	------------	------

Registro de Notas

Estudiante:

Periodo:

Trimestre 1

Tipo de Evaluación:

Examen

Nota:

Guardar Nota

Eliminar Nota

	Estudiante	Periodo	Evaluación	Nota
1	patricia	Trimestre 1	Examen	9

◀

▶

Registro de Notas

Estudiante:

Periodo:

Trimestre 1

Tipo de Evaluación:

Examen


Nota:

Guardar Nota

Eliminar Nota

	Estudiante	Periodo	Evaluación	Nota
	patricia	Trimestre 1	Examen	9

Error



Seleccione una fila para eliminar.

OK