



47GIIN PROYECTO DE INGENIERÍA DEL SOFTWARE

Finalización de la iteración #01

Moisés Sevilla Corrales
PROF. Doctor Horacio Daniel Kuna

05 de diciembre de 2024

Índice

Finalización de la iteración #01.....	3
Consigna:	3
Enlaces de la actividad	9

Finalización de la iteración #01

Consigna:

Documento con los detalles de avances logrados al final de la **iteración #01 del proyecto**.

Adicionalmente, se tendrá que incorporar qué ítems de trabajo se van a implementar en la **siguiente iteración**.

El formato va a ser adaptable según la metodología de gestión seleccionada.

En este caso se debe incluir el **video** de referencia de los avances según lo previsto por la asignatura (se recomienda que sea un enlace para visualizar el mismo que no requiere estar público).

En este documento se deberán incluir los enlaces a:

- **Herramienta de gestión de incidencias.**
- **Herramienta de gestión de versionado de código.**

- Creamos el Desarrollo inicial.
 - Vamos a usar la base de datos de PostgreSQL, con la interfaz de pgAdmin 4.
 - En el Backend usaremos Django REST framework para la comunicación con la base de datos y la creación de endpoints.
 - En el Frontend usaremos Node.js con Vue.js.
 - Para el desarrollo usaremos Windows 11 con Visual Studio Code.
- Empezamos creando la base de datos con un script en pgAdmin.
 - Creamos la base de datos consultorio si no existe.
 - Definimos la tabla paciente.

```
1  -- Crear la base de datos para PostgreSQL
2  CREATE DATABASE IF NOT EXISTS consultorio;
3
4  -- Crear la tabla Paciente
5  CREATE TABLE Paciente (
6      id_paciente SERIAL PRIMARY KEY,
7      dni VARCHAR(9) UNIQUE, -- Nuevo campo DNI
8      nombre VARCHAR(100) NOT NULL,
9      apellido VARCHAR(100) NOT NULL,
10     email VARCHAR(100) UNIQUE NOT NULL,
11     telefono VARCHAR(15),
12     contraseña VARCHAR(100) NOT NULL
13 );
```

- Creamos un proyecto en Django, lo llamamos “consultorio”.
 - Lo configuramos, creamos los endpoints e views.py para el paciente.

```
138 # CRUD Pacientes
139 class PacienteListCreateView(APIView):
140     """
141     GET: Lista todos los pacientes.
142     POST: Crea un nuevo paciente si no existe por id_paciente.
143     """
144     def get(self, request):
145         pacientes = Paciente.objects.all().order_by('id_paciente')
146         serializer = PacienteSerializer(pacientes, many=True)
147         return Response(serializer.data, status=status.HTTP_200_OK)
148
```

```
162 def post(self, request):
163     id_paciente = request.data.get('id_paciente')
164     dni = request.data.get('dni')
165
166     # Verifica si el paciente ya existe por ID
167     if id_paciente and Paciente.objects.filter(id_paciente=id_paciente).exists():
168         return Response({"error": "El paciente ya existe."}, status=status.HTTP_400_BAD_REQUEST)
169
170     # Verifica si el DNI ya existe
171     if dni and Paciente.objects.filter(dni=dni).exists():
172         return Response({"error": "El DNI ya está registrado."}, status=status.HTTP_400_BAD_REQUEST)
173
174     serializer = PacienteSerializer(data=request.data)
175     if serializer.is_valid():
176         serializer.save()
177         return Response(serializer.data, status=status.HTTP_201_CREATED)
178     return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
179
180 class PacienteRetrieveUpdateDeleteView(APIView):
181     """
182     GET: Recupera un paciente por id_paciente.
183     PUT: Actualiza un paciente existente.
184     DELETE: Elimina un paciente existente.
185     """
186     def get(self, request, id_paciente):
187         try:
188             paciente = Paciente.objects.get(id_paciente=id_paciente)
189         except Paciente.DoesNotExist:
190             return Response({"error": "Paciente no encontrado."}, status=status.HTTP_404_NOT_FOUND)
191
192         serializer = PacienteSerializer(paciente)
193         return Response(serializer.data, status=status.HTTP_200_OK)
194
195     def put(self, request, id_paciente):
196         try:
197             paciente = Paciente.objects.get(id_paciente=id_paciente)
198         except Paciente.DoesNotExist:
199             return Response({"error": "Paciente no encontrado."}, status=status.HTTP_404_NOT_FOUND)
200
201         serializer = PacienteSerializer(paciente, data=request.data)
202         if serializer.is_valid():
203             serializer.save()
204             return Response(serializer.data, status=status.HTTP_200_OK)
205         return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
206
207     def delete(self, request, id_paciente):
208         try:
209             paciente = Paciente.objects.get(id_paciente=id_paciente)
210         except Paciente.DoesNotExist:
211             return Response({"error": "Paciente no encontrado."}, status=status.HTTP_404_NOT_FOUND)
212
213         paciente.delete()
214         return Response({"mensaje": "Paciente eliminado exitosamente."}, status=status.HTTP_200_OK)
```

- Definimos el modelo de la base de datos en el backend en models.py

```
191 # Modelo para los pacientes
192 class Paciente(models.Model):
193     id_paciente = models.AutoField(primary_key=True) # Identificador único del paciente
194     dni = models.CharField(unique=True, max_length=9) # DNI del paciente
195     nombre = models.CharField(max_length=100) # Nombre del paciente
196     apellido = models.CharField(max_length=100) # Apellido del paciente
197     email = models.CharField(unique=True, max_length=100) # Correo único del paciente
198     telefono = models.CharField(max_length=15, blank=True, null=True) # Teléfono del paciente (opcional)
199     contrasena = models.CharField(max_length=100) # Contraseña del paciente
200
201     class Meta:
202         managed = False
203         db_table = 'paciente'
```

- Luego en la parte del frontend en VUE.js trans configurarlo y cumplir todos los puntos de la iteración nos queda esta interfaz.

Clínica Rehabilita tu alma

Pacientes | Médicos | Agendar

Gestión de Pacientes

DNI:

Nombre:

Apellido:

Email:

Teléfono:

Contraseña:
☐ Mostrar Contraseña

Crear Paciente

Nombre

Ingrese criterio de búsqueda

ID	DNI	Nombre	Apellido	Email	Teléfono	Acciones
6	54547281R	Sofía	Rodríguez	sofia.rodriguez@example.com	777888999	<div>Editar</div> <div>Eliminar</div>
7	83916204N	Miguel	García	miguel.garcia@example.com	333444555	<div>Editar</div> <div>Eliminar</div>
8	51938554K	Laura	Moreno	laura.moreno@example.com	888999000	<div>Editar</div> <div>Eliminar</div>
9	96289843D	Pedro	Ruiz	pedro.ruiz@example.com	666777888	<div>Editar</div> <div>Eliminar</div>
10	00511981J	Elena	Castro	elena.castro@example.com	999000111	<div>Editar</div> <div>Eliminar</div>
11	98765432A	Ana	López	ana.lopez22@example.com	987654321	<div>Editar</div> <div>Eliminar</div>
12	74196328L	Ana	Mena	ana.mena@paciente.com	456852123	<div>Editar</div> <div>Eliminar</div>

- Controlamos todos los campos del formulario para validarlos correctamente antes de crear el paciente, mostramos un CRUD completo del paciente en vista del modo de desarrollador.

- El DNI es único para cada paciente, no se puede repetir, nos mostrara una alerta de que ya está registrado.

The screenshot shows the 'Gestión de Pacientes' interface. At the top, there's a navigation bar with 'Pacientes' and 'Médicos' tabs. A notification box at the top right states: 'localhost:8080 dice Error al guardar el paciente: error: El DNI ya está registrado.' with an 'Aceptar' button. The main form contains fields for DNI, Nombre, Apellido, Email, Teléfono, and Contraseña. The DNI field is filled with '04703910V'. Below the form is a 'Crear Paciente' button and a search bar. A table below the form lists existing patients:


ID	DNI	Nombre	Apellido	Email	Teléfono	Acciones
1	04703910V	Juan	Pérez	juan.perez@paciente.com	444555666	Editar Eliminar

- El email es único para cada paciente, no se puede repetir, nos mostrara una alerta de que ya está registrado.

The screenshot shows the 'Gestión de Pacientes' interface. At the top, there's a navigation bar with 'Pacientes' and 'Médicos' tabs. A notification box at the top right states: 'localhost:8080 dice Error al guardar el paciente: email: Ya existe paciente con este email.' with an 'Aceptar' button. The main form contains fields for DNI, Nombre, Apellido, Email, Teléfono, and Contraseña. The Email field is filled with 'juan.perez@paciente.com'. Below the form is a 'Crear Paciente' button and a search bar. A table below the form lists existing patients:

ID	DNI	Nombre	Apellido	Email	Teléfono	Acciones
1	04703910V	Juan	Pérez	juan.perez@paciente.com	444555666	Editar Eliminar

- Vemos el Sprint 1 que teníamos definido y finalizamos la iteración 1, para continuar con la iteración 2.

<input type="checkbox"/> Tablero Sprint 1 8 ene – 10 ene (32 incidencias)			9,5	10	5
<input type="checkbox"/> SGRCM-92	Crear el Desarrollo inicial		FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-93	Crear el modelo de datos para el paciente en la base de datos		FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-94	Implementar el formulario de ingreso de datos de paciente		FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-95	Programar la validación de los campos obligatorios en el formulario		FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-96	Implementar la función de guardar los datos en la base de datos		FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-110	Desarrollar un mensaje de confirmación para el registro si es correcto		FINALIZADA		
<input type="checkbox"/> SGRCM-97	Registro de un nuevo paciente	CREAR GESTIÓN DE PA...	FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-28	Agregar paciente	CREAR GESTIÓN DE PA...	FINALIZADA		
<input type="checkbox"/> SGRCM-102	Consultar pacientes registrados	CREAR GESTIÓN DE PA...	FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-33	Buscar paciente	CREAR GESTIÓN DE PA...	FINALIZADA		
<input type="checkbox"/> SGRCM-103	Actualizar datos de un paciente	CREAR GESTIÓN DE PA...	FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-29	Editar paciente	CREAR GESTIÓN DE PA...	FINALIZADA		
<input type="checkbox"/> SGRCM-104	Eliminar pacientes	CREAR GESTIÓN DE PA...	FINALIZADA		
<input checked="" type="checkbox"/> SGRCM-31	Eliminar paciente	CREAR GESTIÓN DE PA...	FINALIZADA		
<input type="checkbox"/> SGRCM-115	Registro de especialistas	CREAR ESPECIALISTAS	EN CURSO		
<input checked="" type="checkbox"/> SGRCM-72	Agregar especialista	CREAR ESPECIALISTAS	EN CURSO		
<input checked="" type="checkbox"/> SGRCM-73	Editar especialista	CREAR ESPECIALISTAS	EN CURSO		
<input checked="" type="checkbox"/> SGRCM-74	Eliminar especialista	CREAR ESPECIALISTAS	EN CURSO		
<input checked="" type="checkbox"/> SGRCM-75	Buscar especialista	CREAR ESPECIALISTAS	EN CURSO		
<input type="checkbox"/> <input checked="" type="checkbox"/> SGRCM-76	Ver especialista 	CREAR ESPECIALISTAS	EN CURSO		

Enlaces de la actividad

Video de presentación de este entregable (YouTube):

URL: <https://youtu.be/t45tf4bSV-4>

Herramienta de gestión de incidencias (Jira):

URL: [https://moisessevilla.atlassian.net/jira/software/projects/SGRCM/boards/1/backlog?selecte
dIssue=SGRCM-92](https://moisessevilla.atlassian.net/jira/software/projects/SGRCM/boards/1/backlog?selecte
dIssue=SGRCM-92)

Herramienta de gestión de versionado de código (GitHub):

URL: https://github.com/moisessevilla/47GIIN_Proyecto_Consultorio