

ACTIVIDAD 2: CREACIÓN DE MICROSERVICIOS EN PYTHON

GUTIERREZ MARTINEZ VALERIA IVONNE

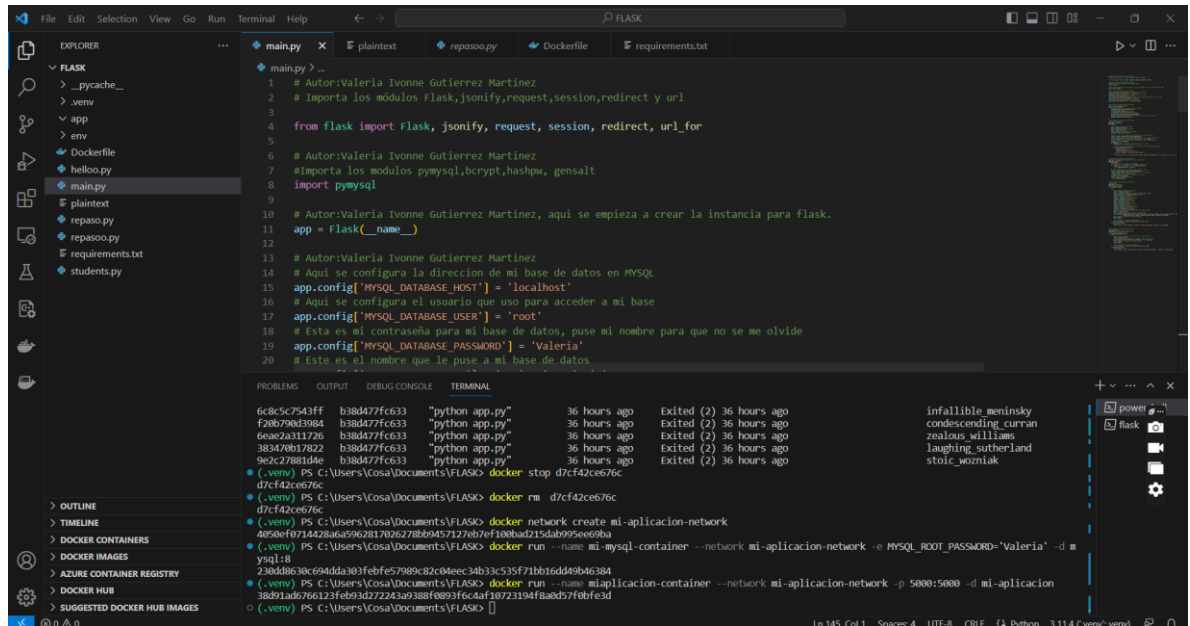


DOCENTE: GABRIEL BARRON RODRIGUEZ
21 de agosto de 2023

Elaborar una aplicación utilizando el lenguaje de programación Python de manera individual utilizando los microservicios.

Crear una imagen y contenedor de una aplicación Python en Docker.

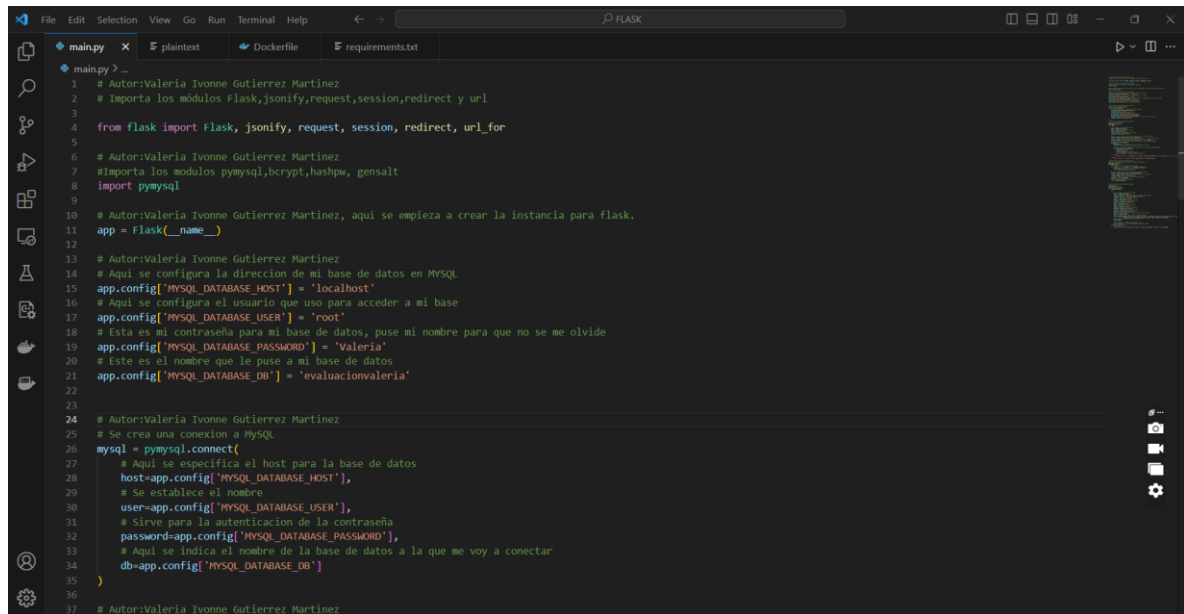
Creación de una red nueva, creación de un contenedor para MySQL y creación de un contenedor para la aplicación.



```
1 # Autor:Valeria Ivonne Gutierrez Martinez
2 # Importa los módulos Flask,jsonify,request,session,redirect y url
3
4 from flask import Flask, jsonify, request, session, redirect, url_for
5
6 # Autor:Valeria Ivonne Gutierrez Martinez
7 #Importa los módulos pymysql,bcrypt,hashpw, gensalt
8 import pymysql
9
10 # Autor:Valeria Ivonne Gutierrez Martinez, aqui se empieza a crear la instancia para flask.
11 app = Flask(__name__)
12
13 # Autor:Valeria Ivonne Gutierrez Martinez
14 # Aqui se configura la direccion de mi base de datos en MySQL
15 app.config['MYSQL_DATABASE_HOST'] = 'localhost'
16 # aqui se configura el usuario que uso para acceder a mi base
17 app.config['MYSQL_DATABASE_USER'] = 'root'
18 # Esta es mi contraseña para mi base de datos, puse mi nombre para que no se me olvide
19 app.config['MYSQL_DATABASE_PASSWORD'] = 'Valeria'
20 # Este es el nombre que le puse a mi base de datos
```

```
6c8c5c7543ff b38d477fc633 "python app.py" 36 hours ago Exited (2) 36 hours ago infallible_meninsky
f20b790d3984 b38d477fc633 "python app.py" 36 hours ago Exited (2) 36 hours ago condescending_curran
6eae2a311726 b38d477fc633 "python app.py" 36 hours ago Exited (2) 36 hours ago zealous_williams
383470b17822 b38d477fc633 "python app.py" 36 hours ago Exited (2) 36 hours ago laughing_sutherland
9e2c27881d4e b38d477fc633 "python app.py" 36 hours ago Exited (2) 36 hours ago stoic_wozniak
(.venv) PS C:\Users\Cosa\Documents\FLASK> docker stop d7cf42ce676c
d7cf42ce676c
(.venv) PS C:\Users\Cosa\Documents\FLASK> docker rm d7cf42ce676c
d7cf42ce676c
(.venv) PS C:\Users\Cosa\Documents\FLASK> docker network create mi-aplicacion-network
4050ef0714428a6a5962817026278bb9457127eb7ef100bad215dab995ee9ba
(.venv) PS C:\Users\Cosa\Documents\FLASK> docker run --name mi-mysql-container --network mi-aplicacion-network -e MYSQL_ROOT_PASSWORD='Valeria' -d mysql
230d48638c694dda303febfe57989c82c04eec34b33c535f71bb16dd49b46384
(.venv) PS C:\Users\Cosa\Documents\FLASK> docker run --name miaplicacion-container --network mi-aplicacion-network -p 5000:5000 -d mi-aplicacion
38d91ad6766123f6b93d272243a9388f0893f6c4af10723194f8a0d57f0bfed3
(.venv) PS C:\Users\Cosa\Documents\FLASK>
```

Desarrollar el microservicio de Login.



```
1 # Autor:Valeria Ivonne Gutierrez Martinez
2 # Importa los módulos Flask,jsonify,request,session,redirect y url
3
4 from flask import Flask, jsonify, request, session, redirect, url_for
5
6 # Autor:Valeria Ivonne Gutierrez Martinez
7 #Importa los módulos pymysql,bcrypt,hashpw, gensalt
8 import pymysql
9
10 # Autor:Valeria Ivonne Gutierrez Martinez, aqui se empieza a crear la instancia para flask.
11 app = Flask(__name__)
12
13 # Autor:Valeria Ivonne Gutierrez Martinez
14 # Aqui se configura la direccion de mi base de datos en MySQL
15 app.config['MYSQL_DATABASE_HOST'] = 'localhost'
16 # aqui se configura el usuario que uso para acceder a mi base
17 app.config['MYSQL_DATABASE_USER'] = 'root'
18 # Esta es mi contraseña para mi base de datos, puse mi nombre para que no se me olvide
19 app.config['MYSQL_DATABASE_PASSWORD'] = 'Valeria'
20 # Este es el nombre que le puse a mi base de datos
21 app.config['MYSQL_DATABASE_DB'] = 'evaluacionvaleria'
22
23
24 # Autor:Valeria Ivonne Gutierrez Martinez
25 # Se crea una conexion a MySQL
26 mysql = pymysql.connect(
27     # Aqui se especifica el host para la base de datos
28     host=app.config['MYSQL_DATABASE_HOST'],
29     # Se establece el nombre
30     user=app.config['MYSQL_DATABASE_USER'],
31     # Sirve para la autentificacion de la contraseña
32     password=app.config['MYSQL_DATABASE_PASSWORD'],
33     # Aqui se indica el nombre de la base de datos a la que me voy a conectar
34     db=app.config['MYSQL_DATABASE_DB']
35 )
36
37 # Autor:Valeria Ivonne Gutierrez Martinez
```

```
File Edit Selection View Go Run Terminal Help FLASK
main.py x plaintext Dockerfile requirements.txt
main.py > ...
36
37 # Autor:Valeria Ivonne Gutierrez Martinez
38 #Inicio del metodo POST
39 @app.post('/register')
40 def login():
41
42     # Para obtener los datos desde JSON
43     datos = request.get_json()
44     # Para obtener el email de los datos
45     email = datos.get('email')
46     # Para obtener la contraseña de los datos
47     password = datos.get('password')
48
49     # Aqui se creo un cursor para tener interaccion con la base de datos
50     cursor = mysql.cursor(mysql.cursors.Dictcursor)
51     # Aqui se hace una consulta SQL para tener el usuario con el email que se dio
52     cursor.execute('SELECT * FROM usuarios WHERE email = %s', (email,))
53     # Sirve para dar el primer usuario que se parezca al email que se ingreso
54     usuarios = cursor.fetchone()
55     if usuarios:
56         stored_password = usuarios['password'] #Obtiene la contra
57
58         # Sirve si la contraseña almacenada coincide con la contraseña proporcionada
59         if stored_password == password:
60             # Inicia una sesion
61             session['logged_in'] = True
62             # Da un mensaje de inicio de sesion
63             return {'message': 'Inicio de sesión exitoso'}
64         else:
65             return {'error': 'La contraseña no coincide, intentalo nuevamente'} # Da un mensaje si no es correcto
66     else:
67         return {'error': 'El correo no está registrado en la base de datos'}
68
69 # Autor:Valeria Ivonne Gutierrez Martinez
70 #Inicio del metodo GET, para mostrar la tabla de los datos
71 @app.get('/showtable')
72 def ver_tabla():
73
74     # Para verificar si el usuario inicio sesion
75     if 'logged_in' not in session or not session['logged_in']:
76         #Redirige al registro si aun no se inicia la sesion
77         return redirect(url_for('register'))
78
79     #Se crea un cursor para la interaccion con la base de datos
80     cursor = mysql.cursor(mysql.cursors.Dictcursor)
81     #Aqui se hace una consulta SQL para obtener todos los usuarios
82     cursor.execute('SELECT * FROM usuarios')
83     # Obtener todos los usuarios en formato de diccionario
84     usuarios = cursor.fetchall()
85     # Para obtener los datos desde JSON
86     return jsonify(usuarios)
87
88 # Autor:Valeria Ivonne Gutierrez Martinez
89 # Metodo POST
90 @app.post('/add_user')
91 # Para agregar usuarios
92 def agregar_usuario():
93
94     try:
95         # Para obtener los datos desde JSON
96         datos = request.get_json()
97         # Aqui se obtiene el valor del numero de control de los datos
98         numero_de_control = datos.get('numero_de_control')
99         # Aqui se obtiene el valor de usuario
100         usuario = datos.get('usuario')
101         # Aqui se obtiene el valor de password
102         password = datos.get('password')
103         # Aqui se obtiene el valor de email
104         email = datos.get('email')
105         # Aqui se obtiene el valor de nombre
106         nombre = datos.get('nombre')
107         # Aqui se obtiene el valor de telefono
108         telefono = datos.get('telefono')
109
110         # Aqui se crea un cursor para la interaccion con la base de datos
111         cursor = mysql.cursor(mysql.cursors.Dictcursor)
112         # Aqui se hace una consulta SQL para obtener todos los usuarios
113         cursor.execute('SELECT * FROM usuarios')
114         # Obtener todos los usuarios en formato de diccionario
115         usuarios = cursor.fetchall()
116         # Para obtener los datos desde JSON
117         return jsonify(usuarios)
118
119     except:
120         return {'error': 'Error al agregar usuario'}
```

```
File Edit Selection View Go Run Terminal Help FLASK
main.py x plaintext Dockerfile requirements.txt
main.py > ...
72 def ver_tabla():
73     #Para verificar si el usuario inicio sesion
74     if 'logged_in' not in session or not session['logged_in']:
75         #Redirige al registro si aun no se inicia la sesion
76         return redirect(url_for('register'))
77
78     #Se crea un cursor para la interaccion con la base de datos
79     cursor = mysql.cursor(mysql.cursors.Dictcursor)
80     #Aqui se hace una consulta SQL para obtener todos los usuarios
81     cursor.execute('SELECT * FROM usuarios')
82     # Obtener todos los usuarios en formato de diccionario
83     usuarios = cursor.fetchall()
84     # Para obtener los datos desde JSON
85     return jsonify(usuarios)
86
87 # Autor:Valeria Ivonne Gutierrez Martinez
88 # Metodo POST
89 @app.post('/add_user')
90 # Para agregar usuarios
91 def agregar_usuario():
92
93     try:
94         # Para obtener los datos desde JSON
95         datos = request.get_json()
96         # Aqui se obtiene el valor del numero de control de los datos
97         numero_de_control = datos.get('numero_de_control')
98         # Aqui se obtiene el valor de usuario
99         usuario = datos.get('usuario')
100         # Aqui se obtiene el valor de password
101         password = datos.get('password')
102         # Aqui se obtiene el valor de email
103         email = datos.get('email')
104         # Aqui se obtiene el valor de nombre
105         nombre = datos.get('nombre')
106         # Aqui se obtiene el valor de telefono
107         telefono = datos.get('telefono')
108
109         # Aqui se crea un cursor para la interaccion con la base de datos
110         cursor = mysql.cursor(mysql.cursors.Dictcursor)
111         # Aqui se hace una consulta SQL para obtener todos los usuarios
112         cursor.execute('SELECT * FROM usuarios')
113         # Obtener todos los usuarios en formato de diccionario
114         usuarios = cursor.fetchall()
115         # Para obtener los datos desde JSON
116         return jsonify(usuarios)
117
118     except:
119         return {'error': 'Error al agregar usuario'}
```

```
File Edit Selection View Go Run Terminal Help
main.py x plaintext Dockerfile requirements.txt
183 # Aqui se obtiene el valor de email
184 email = datos.get('email')
185 # Aqui se obtiene el valor de nombre
186 nombre = datos.get('nombre')
187 # Aqui se obtiene el valor de telefono
188 telefono = datos.get('telefono')
189 # Aqui se obtiene el valor de municipio
190 municipio = datos.get('municipio')
191 #Se crea un cursor para la interaccion con la base de datos
192 cursor = mysql.cursor()
193 #para una consulta SQL para insertar un nuevo usuario
194 cursor.execute('INSERT INTO usuarios (numero_de_control, usuario, password, email, nombre, telefono, municipio) VALUES (Xs, Xs, Xs, Xs, Xs, Xs, Xs)',
195               (numero_de_control, usuario, password, email, nombre, telefono, municipio))
196 #Para guardar los cambios en la base de datos
197 mysql.commit()
198
199 #Da un mensaje si el usuario se agrego
200 return {'message': 'Usuario agregado exitosamente'}, 201
201 except Exception as e:
202     #Da mensaje de error
203     return {'error': 'No se agrego el usuario, intentelo nuevamente', 'details': str(e)}, 404
204
205
206
```

MySQL

MySQL Workbench

Gestion local de MySQL X

File Edit View Query Database Server Tools Scripting Help

Navigation

SCHEMAS

Filter objects

evaluationvaleria

Tables

Views

Stored Procedures

Functions

sys

Administration Schemas

Information

No object selected

DATABASES

Limit to 1000 rows

```
1 use evaluationvaleria;
2 CREATE TABLE usuarios (
3     id INT AUTO_INCREMENT PRIMARY KEY,
4     numero_de_control VARCHAR(20) NOT NULL,
5     usuario VARCHAR(50) NOT NULL,
6     password VARCHAR(100) NOT NULL,
7     email VARCHAR(100) NOT NULL,
8     nombre VARCHAR(100) NOT NULL,
9     telefono VARCHAR(20),
10    municipio VARCHAR(100)
11 );
12
13 select * from usuarios;
14 INSERT INTO usuarios (numero_de_control, usuario, password, email, nombre, telefono, municipio)
15 VALUES
16 (1220100003, 'ANZO0001', 'Valeria', 'anzo.mario@gmail.com', 'ANZO AVALOS MARIA CITLALLI', '4181234567', 'Dolores Hidalgo'),
17 (1219100524, 'ARRE0001', 'Valeria', 'arredondo.daniel@gmail.com', 'ARREDONDO GONZALEZ DANIEL ENRIQUE', '4182345678', 'Dolores Hidalgo'),
18 (1220100317, 'DUAR0001', 'Valeria', 'duarte.daniel@gmail.com', 'DUARTE VELAZQUEZ DANIEL', '4183456789', 'San Felipe'),
19 (1220100629, 'GUTT0001', 'Valeria', 'gutierrez.valeria@gmail.com', 'GUTIERREZ MARTINEZ VALERIA IVONNE', '4184567890', 'Dolores Hidalgo'),
20 (1220100632, 'LUNA0001', 'Valeria', 'luna.angel@gmail.com', 'LUNA CANTERO ANGEL IVAN', '4185678901', 'San Miguel de Allende'),
21 (1220100209, 'MART0001', 'Valeria', 'martinez.guadalupe@gmail.com', 'MARTINEZ RAMIREZ GUADALUPE MONSERRAT', '4186789012', 'San Miguel de Allende'),
22 (1220100053, 'REY0001', 'Valeria', 'reyes.salvador@gmail.com', 'REYES MORALES SALVADOR', '4187890123', 'Dolores Hidalgo'),
23 (1220100597, 'SALA0001', 'Valeria', 'salazar.maria@gmail.com', 'SALAZAR LEON MARIA GUADALUPE', '4188901234', 'San Felipe'),
24 (1220100596, 'TADO0001', 'Valeria', 'tadeo.alejandra@gmail.com', 'TADEO MARTINEZ ALEJANDRO', '4189012345', 'Dolores Hidalgo'),
25 (1220100075, 'TORR0001', 'Valeria', 'torres.jose@gmail.com', 'TORRES GARCIA JOSE ROSELIO', '4180123456', 'Dolores Hidalgo'),
26 (1220100595, 'TRAN0001', 'Valeria', 'tranqueño.oscar@gmail.com', 'TRANQUEÑO HERNANDEZ OSCAR ARMANDO', '4181234567', 'Dolores Hidalgo');
```

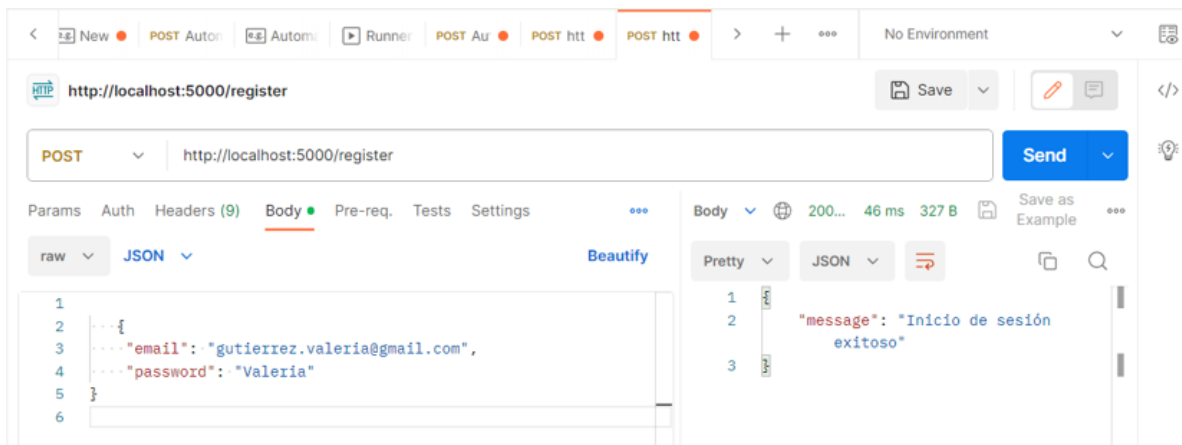
SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

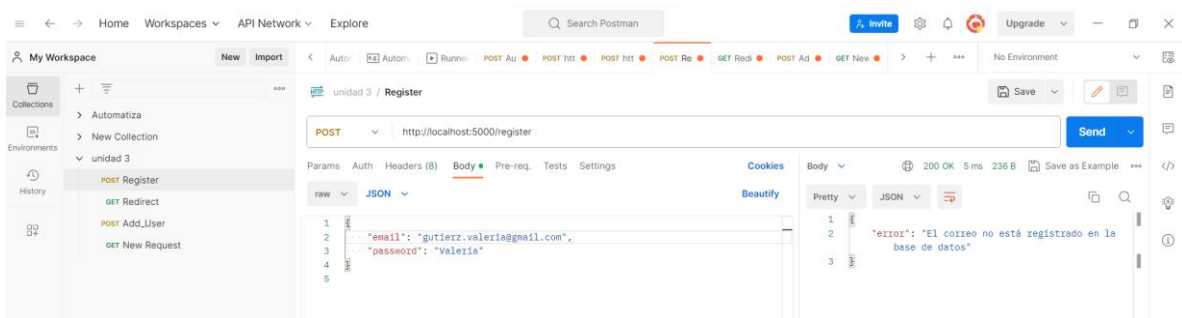
Context Help Snippets

Postman

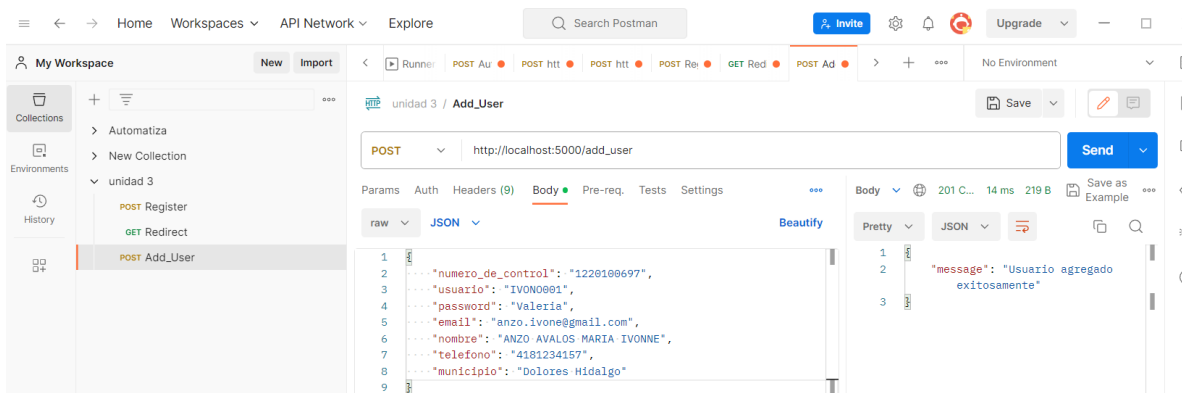
POST registro



Error



POST add-user



Error

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of requests under "unidad 3", including "POST Register", "GET Redirect", and "POST Add_User". The main panel shows the details of the "POST Add_User" request, which is configured with the URL "http://localhost:5000/add_user". The request body is in JSON format, containing a single key-value pair: `{ "numero_de_control": "1234567890" }`. The response status is "404 Not Found", and the response body contains an error message: `{ "details": "400 Bad Request: Failed to decode JSON object: Expecting \':\' delimiter: line 2 column 25 (char 27)", "error": "No se agrego el usuario, intentelo nuevamente" }`.

My Workspace

unitad 3 / Add_User

POST http://localhost:5000/add_user

Params Auth Headers (9) Body Pre-req. Tests Settings

raw JSON Beautify

```
1 {
2   "numero_de_control": "1234567890"
}
```

Body 404 N... 6 ms 351 B Save as Example

Pretty JSON

```
1 {
2   "details": "400 Bad Request:
3     Failed to decode JSON
4     object: Expecting \':\'
       delimiter: line 2 column 25
       (char 27)",
5   "error": "No se agrego el
6     usuario, intentelo
7     nuevamente"
8 }
```