



Instituto Politécnico Nacional.
Unidad Profesional Interdisciplinaria en
Ingeniería y Tecnologías Avanzadas.



Ingeniería Telemática.

Aplicaciones Distribuidas

Práctica

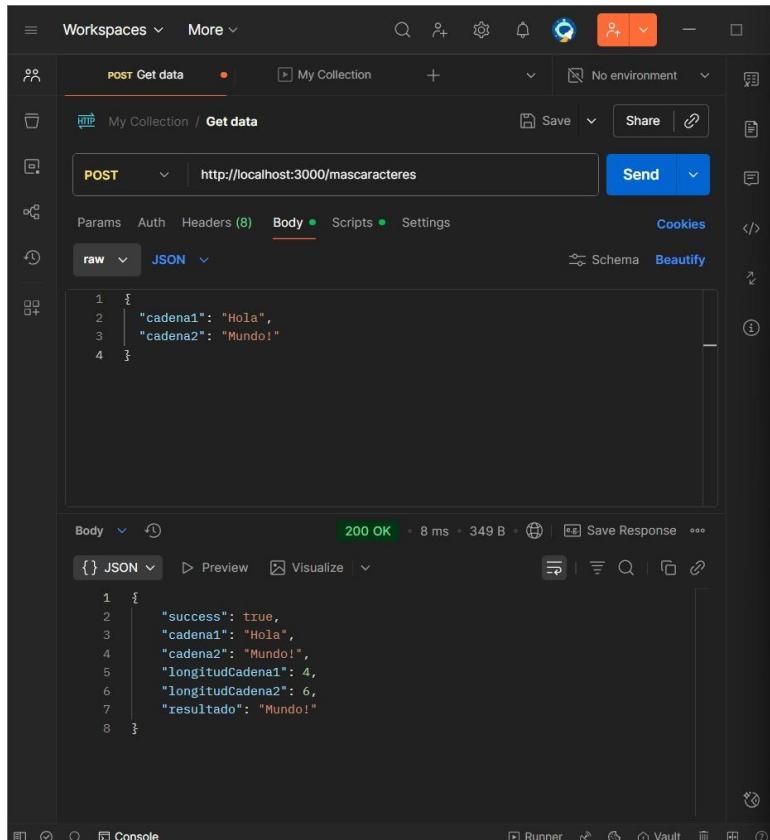
Bautista Uribe Juan Manuel.
Martínez Aparicio Eduardo

Profesor: Sierra Romero Noe

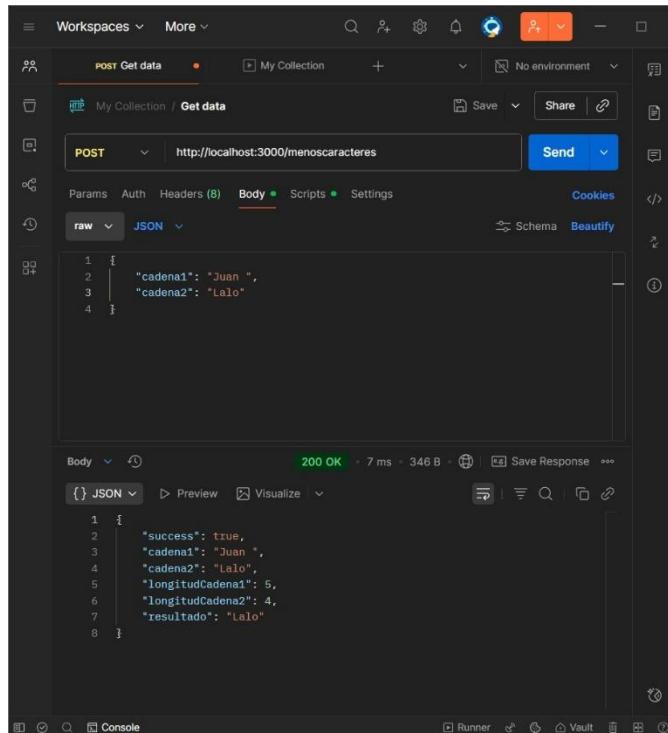
4TM3.

Práctica de Aplicaciones Distribuidas

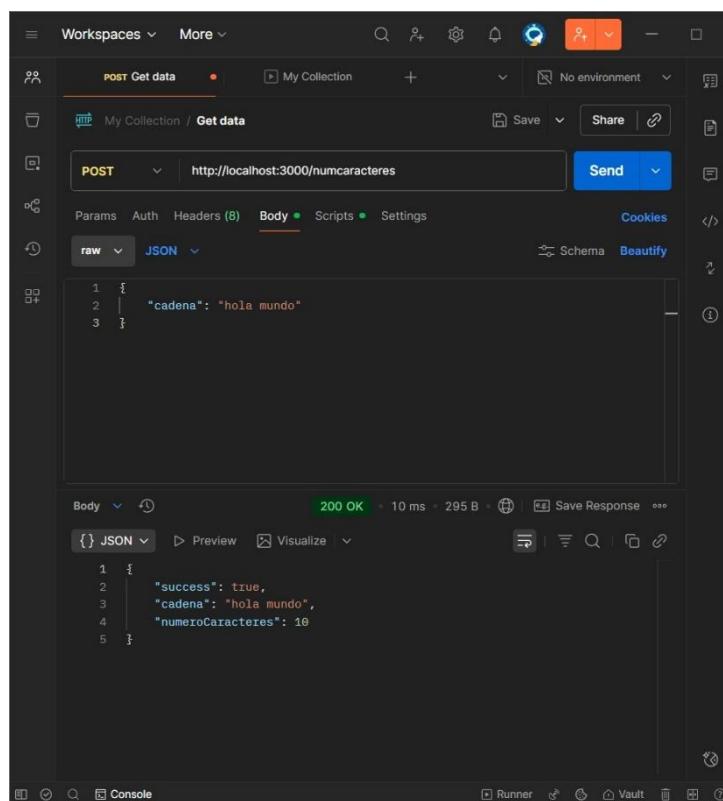
1. Crear una serie de servicios web en la plataforma de RepLit, usando un servidor NodeJS que realicen lo siguiente:
 - a. Requerimientos
 - i. Todos los servicios reciben sus parámetros en una estructura JSON
 - ii. Todos los servicios responden en una estructura JSON
 - iii. Todos los servicios deben validar los parámetros y el resultado de la operación, e incluir un atributo en el JSON de respuesta que indique el resultado o un campo de error indicando el problema
 - b. Tareas a implementar
 - i. **mascaracteres:** recibe dos cadenas y regresa la que tenga más caracteres. Si son iguales, regresa la del primer parámetro



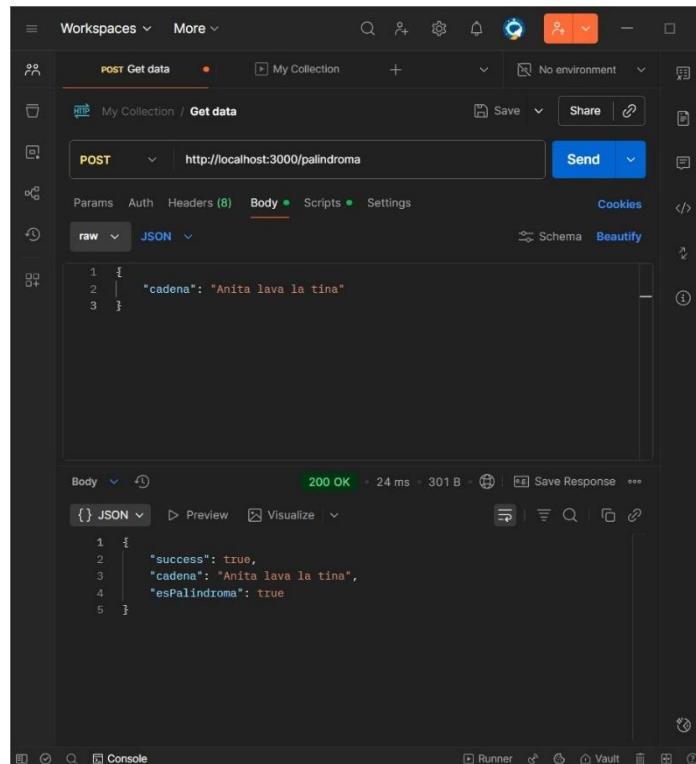
- ii. **menoscaracteres**: recibe dos cadenas y regresa la que tenga menos caracteres. Si son iguales, regresa la del primer parámetro



- iii. numcaracteres: recibe una cadena y regresa el número de caracteres que la cadena tiene



- iv. palindroma: recibe una cadena y regresa true si la cadena es una palindroma, y false en caso contrario



POST <http://localhost:3000/palindroma>

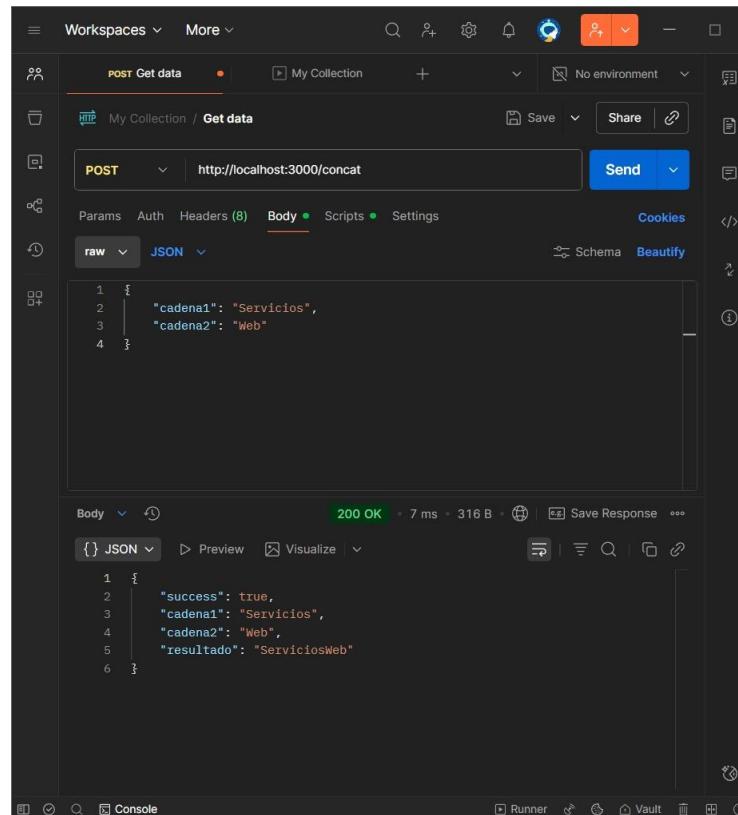
```
1 {
2   "cadena": "Anita lava la tina"
3 }
```

Body [Raw](#) [JSON](#)

200 OK 24 ms 301 B

```
1 {
2   "success": true,
3   "cadena": "Anita lava la tina",
4   "esPalindroma": true
5 }
```

- v. concat: recibe dos cadenas y regresa la concatenación iniciando con el primer parámet



POST [http://localhost:3000\(concat](http://localhost:3000(concat)

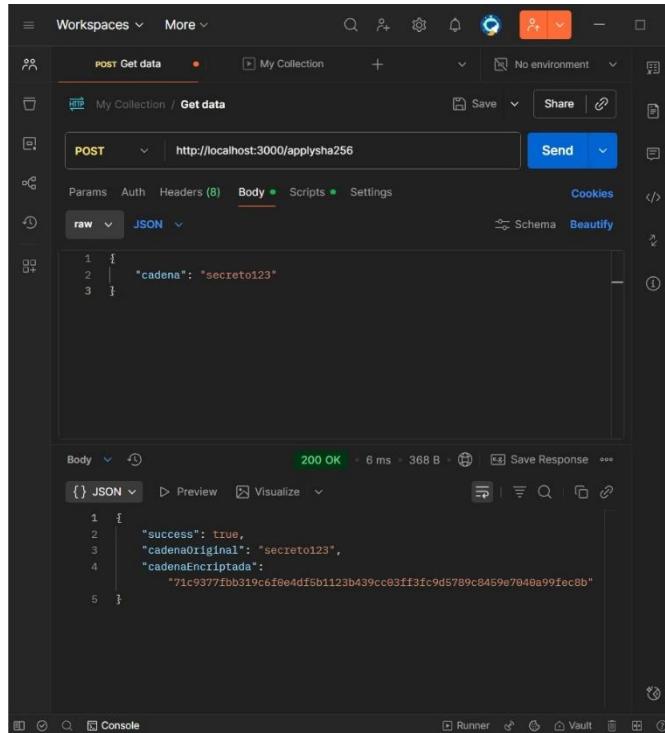
```
1 {
2   "cadena1": "Servicios",
3   "cadena2": "Web"
4 }
```

Body [Raw](#) [JSON](#)

200 OK 7 ms 316 B

```
1 {
2   "success": true,
3   "cadena1": "Servicios",
4   "cadena2": "Web",
5   "resultado": "ServiciosWeb"
6 }
```

- vi. `applysha256`: recibe una cadena, le aplica una encriptación SHA256 y regresa como resultado la cadena original y la encriptada



```
POST /applysha256
```

```
Params: Headers: Body: JSON
```

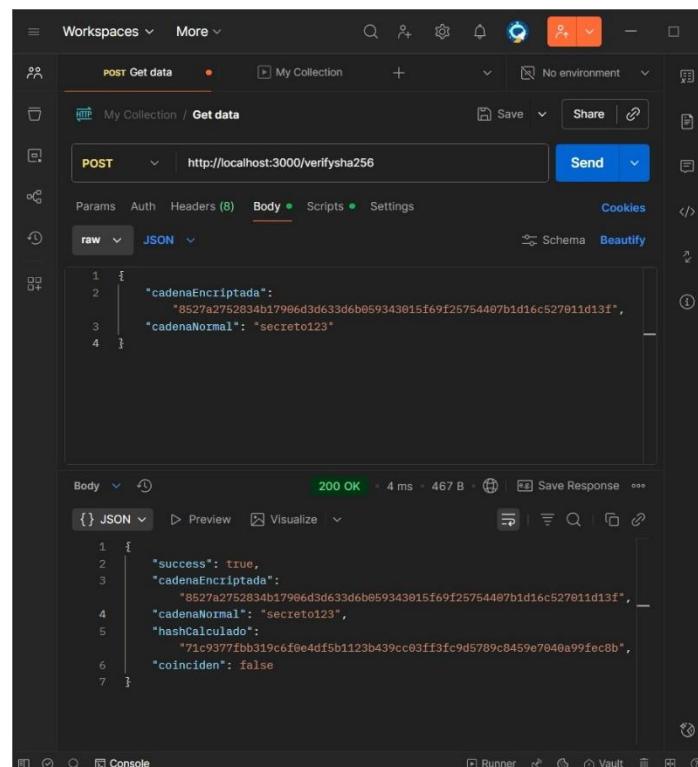
```
{ "cadena": "secreto123" }
```

```
Body: JSON
```

```
200 OK
```

```
{ "success": true, "cadenaOriginal": "secreto123", "cadenaEncriptada": "71c9377fbb319c6f0e4df5b1123b439cc03ff3fc9d5789c8459e7040a99fec8b" }
```

- vii. `verifysha256`: recibe una cadena encriptada, una cadena normal, a la cadena normal le aplica SHA256, la compara con la cadena encriptada y regresa true si coinciden, y false en otro caso



```
POST /verifysha256
```

```
Params: Headers: Body: JSON
```

```
{ "cadenaEncriptada": "8527a2752834b17906d3d633d6b059343015f69f25754407b1d16c527011d13f", "cadenaNormal": "secreto123" }
```

```
Body: JSON
```

```
200 OK
```

```
{ "success": true, "cadenaEncriptada": "8527a2752834b17906d3d633d6b059343015f69f25754407b1d16c527011d13f", "cadenaNormal": "secreto123", "hashCalculado": "71c9377fbb319c6f0e4df5b1123b439cc03ff3fc9d5789c8459e7040a99fec8b", "coinciden": false }
```