

APÉNDICE C. PRUEBAS UNITARIAS

Como se planteó en la metodología a este proyecto se le realizaron pruebas al software para confirmar que el funcionamiento sea el indicado. A continuación, se muestran las pruebas realizadas al Back-end para corroborar que el comportamiento de la simulación del software fuese el mismo al comportamiento mostrado por el modelo de Evolución.

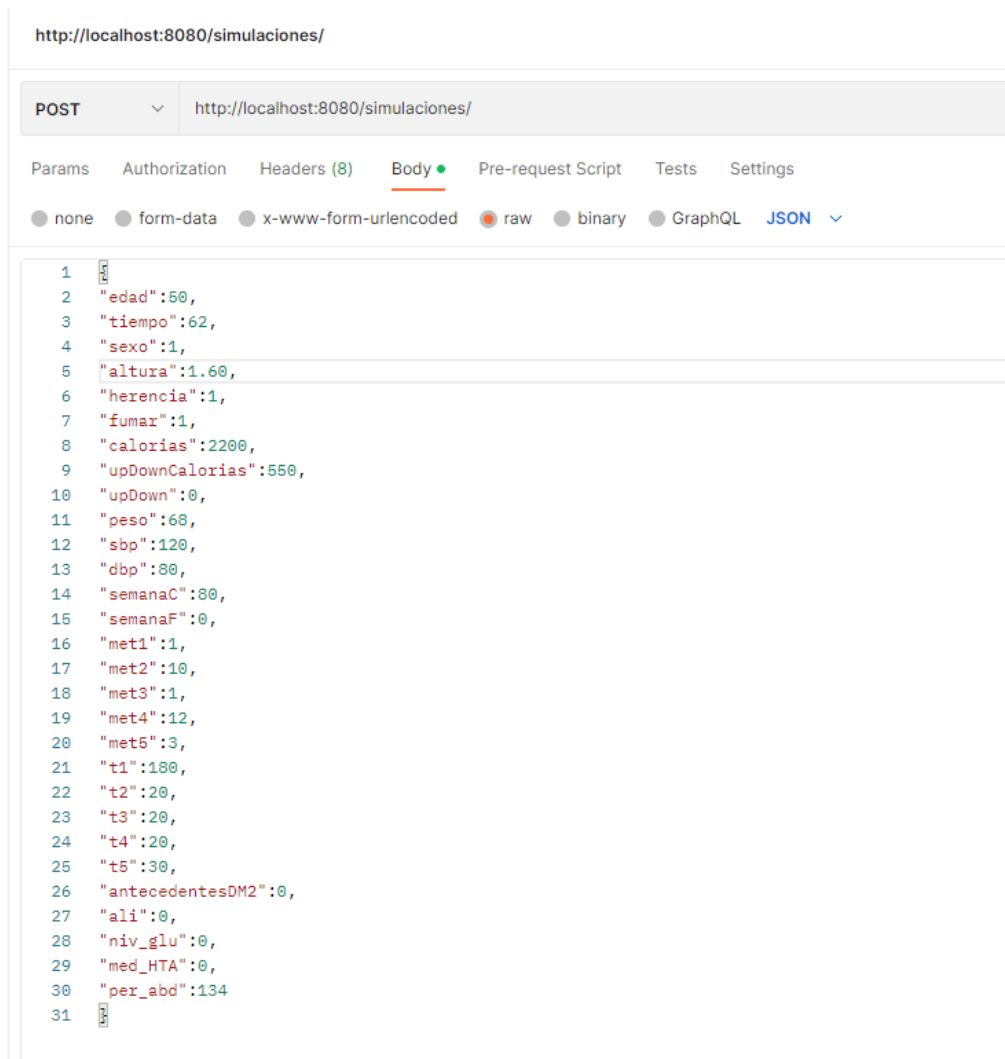
Las pruebas realizadas se llevaron a cabo por medio del aplicativo Postman, el cual permitió observar el funcionamiento de las peticiones que el Front-end iba a solicitar al Back-end.

1.1 Pruebas Back-end

1.1.1 Resultados del modelo

La primera prueba que se realizó fue comprobar que los resultados del software fueran iguales a los resultados del modelo en evolución, para verificar esto se planteó el siguiente escenario:

Figura 1 JSON simulación



A continuación, se muestra los resultados del peso real mostrando todos los pasos de simulación:

Figura 2 Resultados peso real en Evolucion

Iterac.	X:T	P RE
1	50	68
2	50.0027397260274	68.049374025974
3	50.0054794520548	68.0985506078259
4	50.0082191780822	68.1475300250546
5	50.0109589041096	68.1963125567631
6	50.013698630137	68.2448984816595
7	50.0164383561644	68.2932880780575
8	50.0191780821918	68.3414816238765
9	50.0219178082192	68.3894793966427
10	50.0246575342466	68.4372816734894
11	50.027397260274	68.4848887311575
12	50.0301369863014	68.5323008459963
13	50.0328767123288	68.5795182939639
14	50.0356164383562	68.6265413506275
15	50.0383561643836	68.6733702911646
16	50.041095890411	68.7200053903628
17	50.0438356164384	68.766446922621
18	50.0465753424658	68.8126951619494
19	50.0493150684932	68.8587503819706
20	50.0520547945205	68.9046128559197
21	50.0547945205479	68.950282856645
22	50.0575342465753	68.9957606566086
23	50.0602739726027	69.0410465278869
24	50.0630136986301	69.0861407421712
25	50.0657534246575	69.1310435707681
26	50.0684931506849	69.1757552846002
27	50.0712328767123	69.2202761542066
28	50.0739726027397	69.2646064497434
29	50.0767123287671	69.3087464409843
30	50.0794520547945	69.3526963973211
31	50.0821917808219	69.3964565877642
32	50.0849315068493	69.4400272809433
33	50.0876712328767	69.4834087451078
34	50.0904109589041	69.5266012481272
35	50.0931506849315	69.5696050574921
36	50.0958904109589	69.6124204403142

Figura 3 Resultado peso real Back-end

```

HTA-DM - HtaDmApplication [Spring Boot App] [pid: 656]
Edad: 50.0 P_re: 68.0
Edad: 50.0027397260274 P_re: 68.04937402597403
Edad: 50.0054794520548 P_re: 68.09855060782594
Edad: 50.0082191780822 P_re: 68.14753002505456
Edad: 50.0109589041096 P_re: 68.19631255676306
Edad: 50.013698630137 P_re: 68.24489848165952
Edad: 50.0164383561644 P_re: 68.29328807805747
Edad: 50.0191780821918 P_re: 68.34148162387648
Edad: 50.0219178082192 P_re: 68.38947939664267
Edad: 50.0246575342466 P_re: 68.43728167348935
Edad: 50.027397260274 P_re: 68.4848887311575
Edad: 50.0301369863014 P_re: 68.53230084599632
Edad: 50.0328767123288 P_re: 68.57951829396386
Edad: 50.0356164383562 P_re: 68.6265413506275
Edad: 50.0383561643836 P_re: 68.67337029116456
Edad: 50.041095890411 P_re: 68.72000539036279
Edad: 50.0438356164384 P_re: 68.76644692262096
Edad: 50.0465753424658 P_re: 68.81269516194942
Edad: 50.0493150684932 P_re: 68.85875038197064
Edad: 50.0520547945206 P_re: 68.90461285591972
Edad: 50.054794520548 P_re: 68.95028285664502
Edad: 50.0575342465754 P_re: 68.99576065660862
Edad: 50.0602739726028 P_re: 69.04104652788693
Edad: 50.0630136986302 P_re: 69.08614074217121
Edad: 50.0657534246576 P_re: 69.1310435707681
Edad: 50.068493150685 P_re: 69.1757552846002
Edad: 50.0712328767124 P_re: 69.2202761542066
Edad: 50.0739726027398 P_re: 69.26460644974341
Edad: 50.0767123287672 P_re: 69.30874644098431
Edad: 50.0794520547946 P_re: 69.3526963973211
Edad: 50.082191780822 P_re: 69.39645658776422
Edad: 50.0849315068494 P_re: 69.44002728094333
Edad: 50.0876712328768 P_re: 69.48340874510778
Edad: 50.0904109589042 P_re: 69.52660124812724
Edad: 50.0931506849316 P_re: 69.56960505749213
Edad: 50.095890410959 P_re: 69.61242044031427

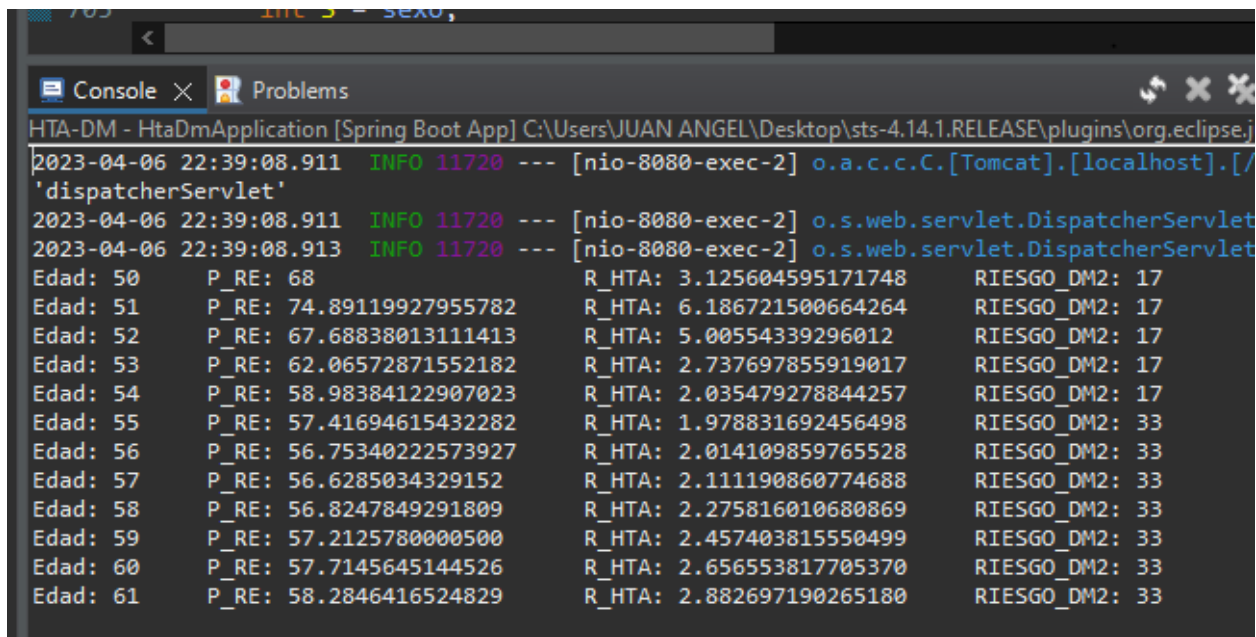
```

Como se puede observar en la ilustración 2 y 3, los valores del peso en el modelo y en evolución son iguales, así que se procedió hacer el mismo escenario, pero esta vez mostrando el comportamiento del peso, el riesgo de sufrir hipertensión y el riesgo de sufrir diabetes, desde los 50 hasta los 61 años y este fue el resultado:

Figura 4 Resultados peso y riesgos Evolucion

Iterac.	X:T	P RE	R HTA	RIESGO DM2
1	50	68	3.12560459517174	17
2	51	74.8911992795578	6.18672150066426	17
3	52	67.6883801311141	5.0055433929601	17
4	53	62.0657287155218	2.73769785591901	17
5	54	58.9838412290702	2.03547927884425	17
6	55	57.4169461543228	1.97883169245649	33
7	56	56.7534022257392	2.01410985976552	33
8	57	56.628503432914	2.11119086077468	33
9	58	56.8247849291807	2.27581601068086	33
10	59	57.2125780000499	2.45740381555049	33
11	60	57.7145645144534	2.65655381770536	33
12	61	58.2846416524826	2.88269719026517	33

Figura 5 Resultado peso y riesgos Back-end



```

HTA-DM - HtaDmApplication [Spring Boot App] C:\Users\JUAN ANGEL\Desktop\sts-4.14.1.RELEASE\plugins\org.eclipse.j
2023-04-06 22:39:08.911 INFO 11720 --- [nio-8080-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/
'dispatcherServlet'
2023-04-06 22:39:08.911 INFO 11720 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet
2023-04-06 22:39:08.913 INFO 11720 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet

Edad: 50      P_RE: 68      R_HTA: 3.125604595171748      RIESGO_DM2: 17
Edad: 51      P_RE: 74.89119927955782      R_HTA: 6.186721500664264      RIESGO_DM2: 17
Edad: 52      P_RE: 67.68838013111413      R_HTA: 5.00554339296012      RIESGO_DM2: 17
Edad: 53      P_RE: 62.06572871552182      R_HTA: 2.737697855919017      RIESGO_DM2: 17
Edad: 54      P_RE: 58.98384122907023      R_HTA: 2.035479278844257      RIESGO_DM2: 17
Edad: 55      P_RE: 57.41694615432282      R_HTA: 1.978831692456498      RIESGO_DM2: 33
Edad: 56      P_RE: 56.75340222573927      R_HTA: 2.014109859765528      RIESGO_DM2: 33
Edad: 57      P_RE: 56.6285034329152      R_HTA: 2.111190860774688      RIESGO_DM2: 33
Edad: 58      P_RE: 56.8247849291809      R_HTA: 2.275816010680869      RIESGO_DM2: 33
Edad: 59      P_RE: 57.2125780000500      R_HTA: 2.457403815550499      RIESGO_DM2: 33
Edad: 60      P_RE: 57.7145645144526      R_HTA: 2.656553817705370      RIESGO_DM2: 33
Edad: 61      P_RE: 58.2846416524829      R_HTA: 2.882697190265180      RIESGO_DM2: 33

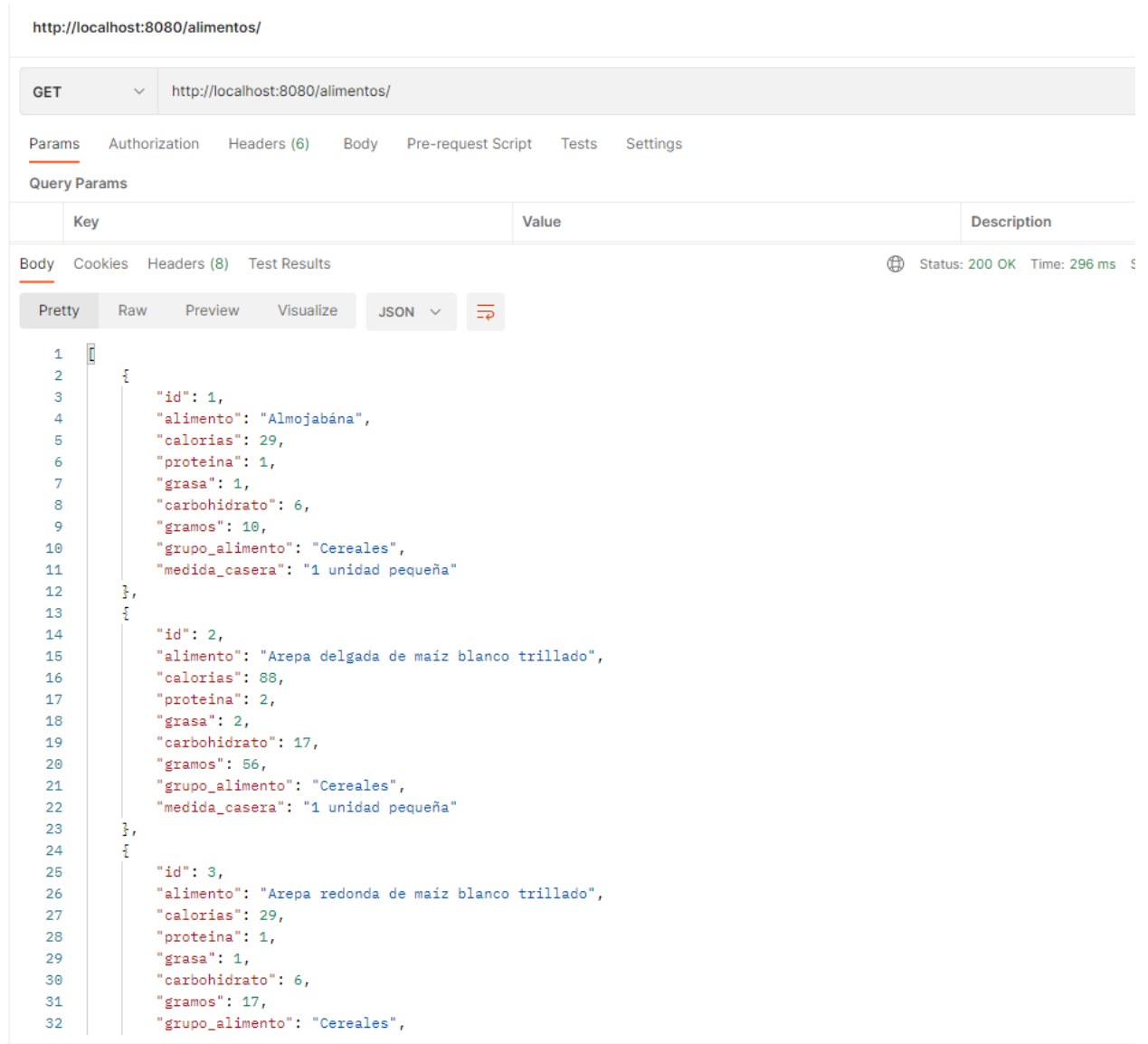
```

Como se puede ver en las ilustraciones 4 y 5, los valores en el modelo y los valores del software tienen comportamientos similares, sin embargo, como el software toma más decimales al momento de hacer la simulación este se desfasa en los últimos decimales.

1.1.2 Listar alimentos

Para listar todos los alimentos que se encuentran en la base de datos, solamente se hace una petición GET como se muestra a continuación:

Figura 6 listar alimentos



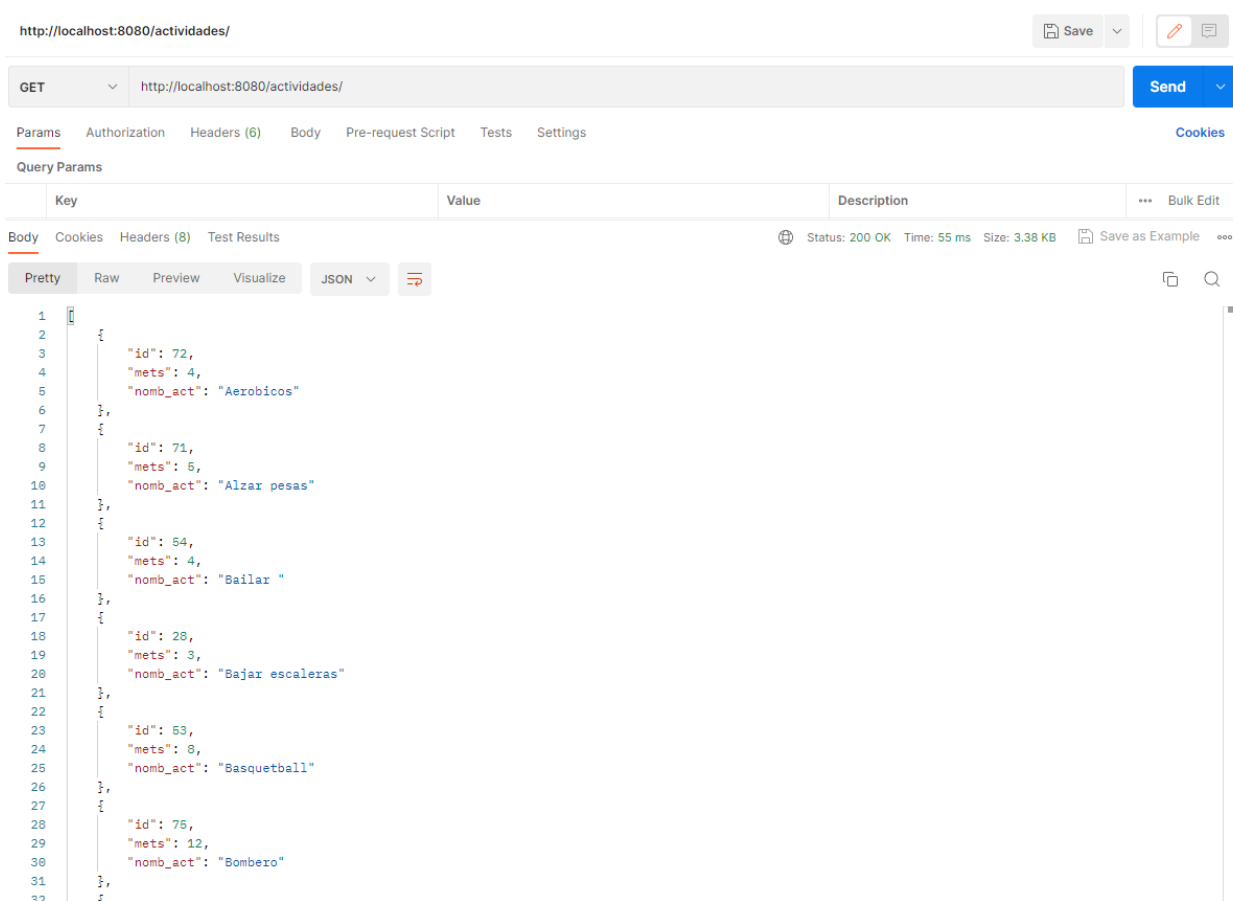
Como se puede ver en postman, la solicitud tuvo un Status 200 lo cual nos indica que la petición no tuvo ningún problema.

1.1.3 Listar actividades

Para listar las actividades físicas que se encuentra en la base de datos, al igual que para listar los alimento, se utiliza un método GET.

A continuación, se muestra el resultado en postman:

Figura 7 listar actividades



1.1.4 Listar paciente

Para el caso de listar pacientes se verifico que al utilizar la petición GET se obtuvieran todos los pacientes y después que solo se filtrara al paciente por cedula como se muestra en la siguiente imagen:

Figura 8 listar todos los pacientes

http://localhost:8080/pacientes/

GET http://localhost:8080/pacientes/

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

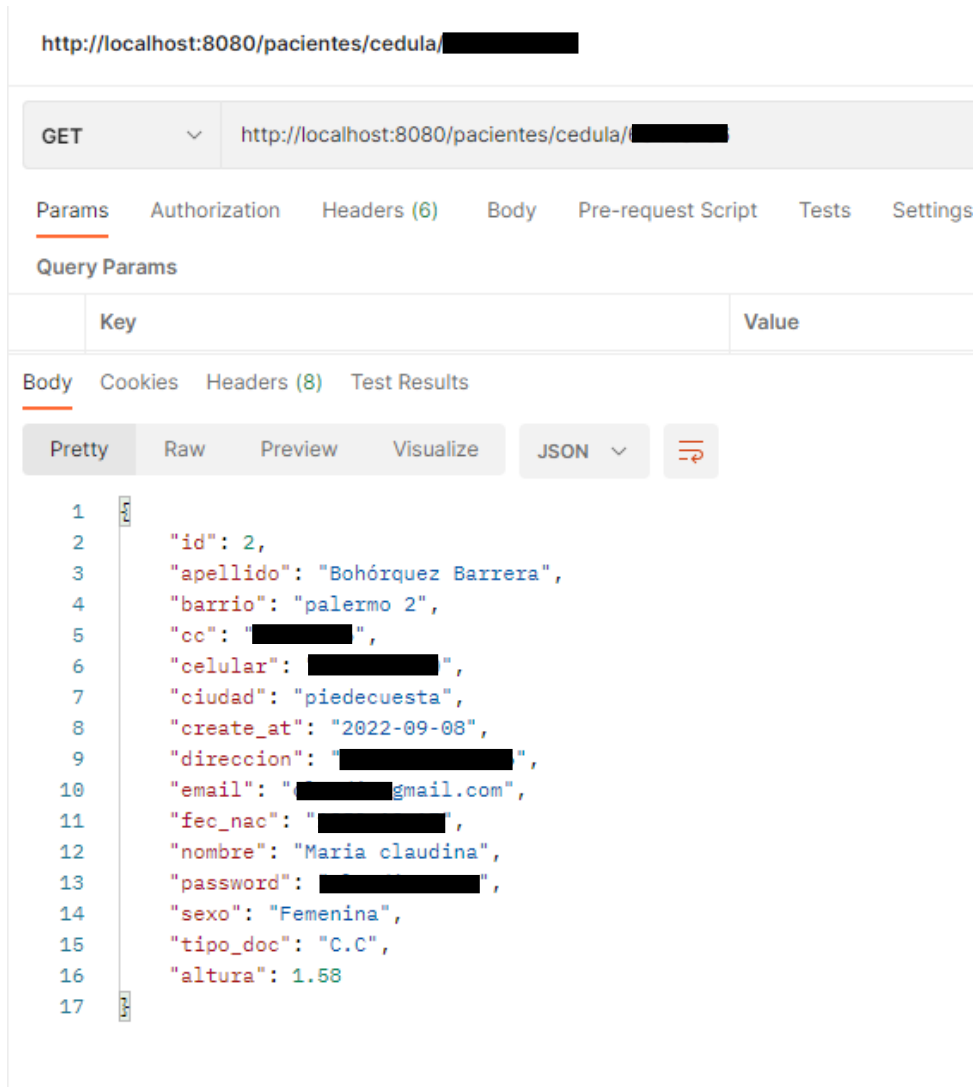
Key	Value	Description
-----	-------	-------------

Body Cookies Headers (8) Test Results Status: 200 OK Time: 29 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "apellido": "Castellanos Sandoval",
4   "barrio": "palermo 2",
5   "cc": "██████████",
6   "celular": "██████████",
7   "ciudad": "Bucaramanga",
8   "create_at": "2022-09-08",
9   "direccion": "██████████",
10  "email": "██████████@██████████.com",
11  "fec_nac": "██████████",
12  "nombre": "Edgar",
13  "password": "██████████",
14  "sexo": "Masculino",
15  "tipo_doc": "C.C",
16  "altura": 1.69
17 },
18 {
19   "id": 2,
20   "apellido": "Bohórquez Barrera",
21   "barrio": "palermo 2",
22   "cc": "██████████",
23   "celular": "██████████",
24   "ciudad": "piedecuesta",
25   "create_at": "2022-09-08",
26   "direccion": "██████████",
27   "email": "██████████@██████████.com",
28   "fec_nac": "██████████",
29   "nombre": "Maria claudina",
30   "password": "██████████",
31   "sexo": "Femenina",
32 }
```

Figura 9 listar paciente por cedula



1.1.5 Login paciente

Para el login del paciente se verifico que al estar correcto el usuario y la contraseña devolviera todos los datos relacionados con el paciente y que al encontrarse mal alguno de los dos devolviera un null ya que de esta manera el Front reconoce que la contraseña o el usuario son incorrectos.

A continuación, se muestran los resultados de la prueba:

Figura 10 login exitoso

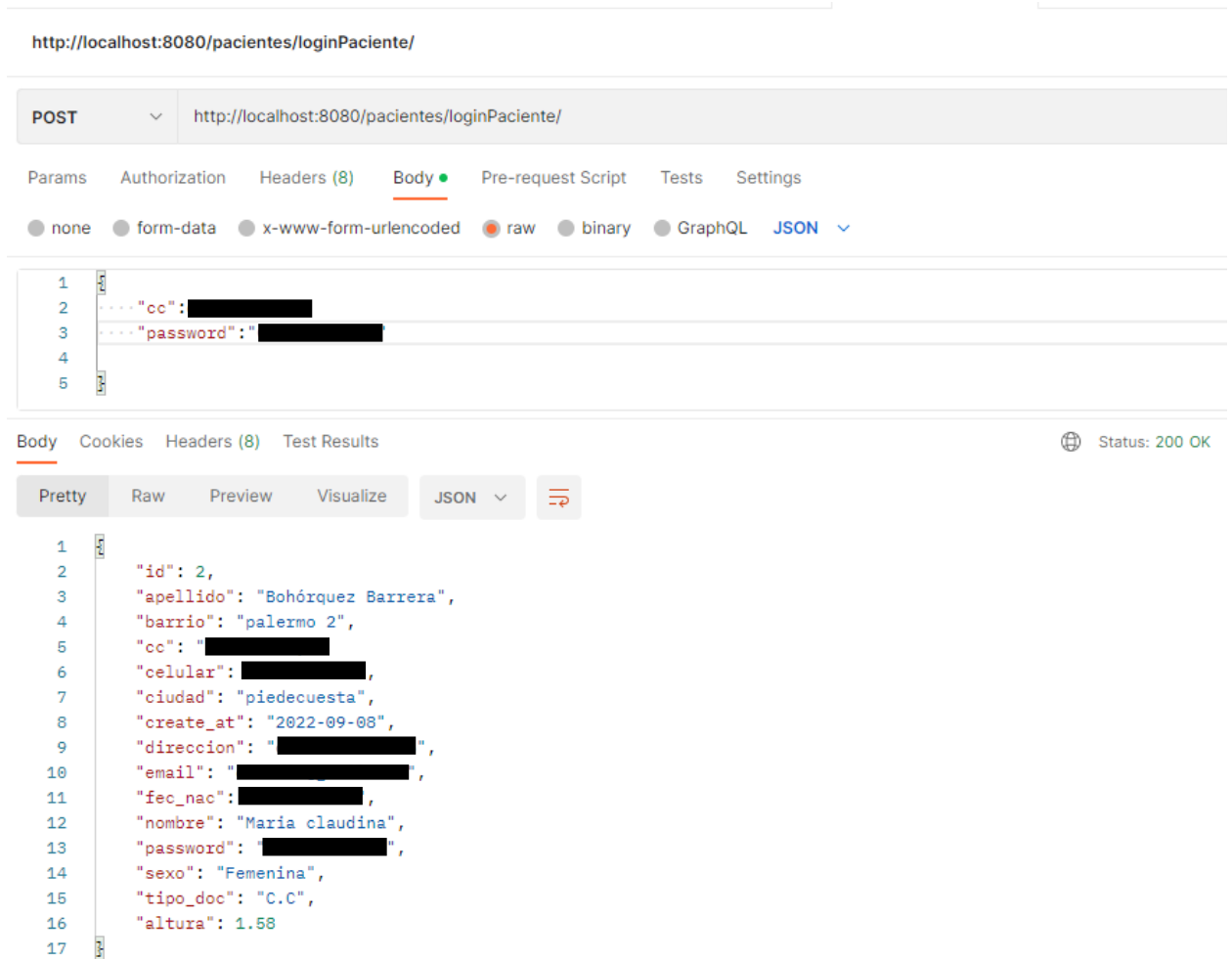


Figura 11 login fallido por cc erróneo

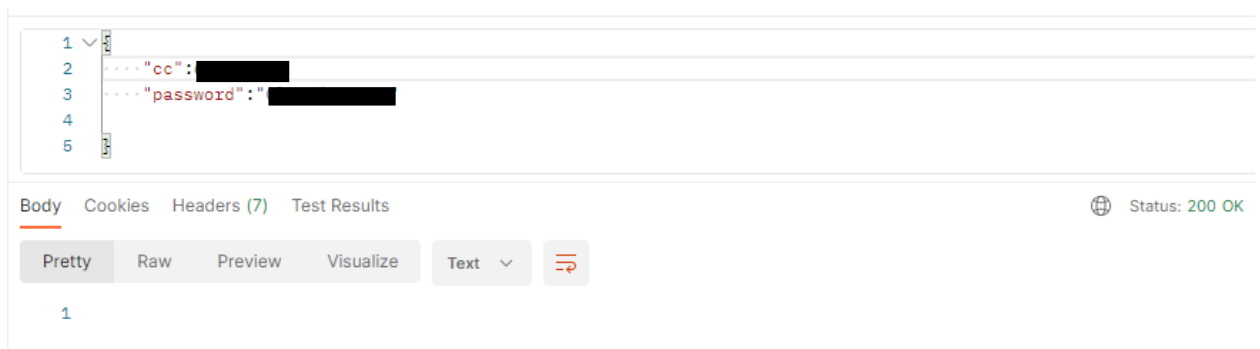
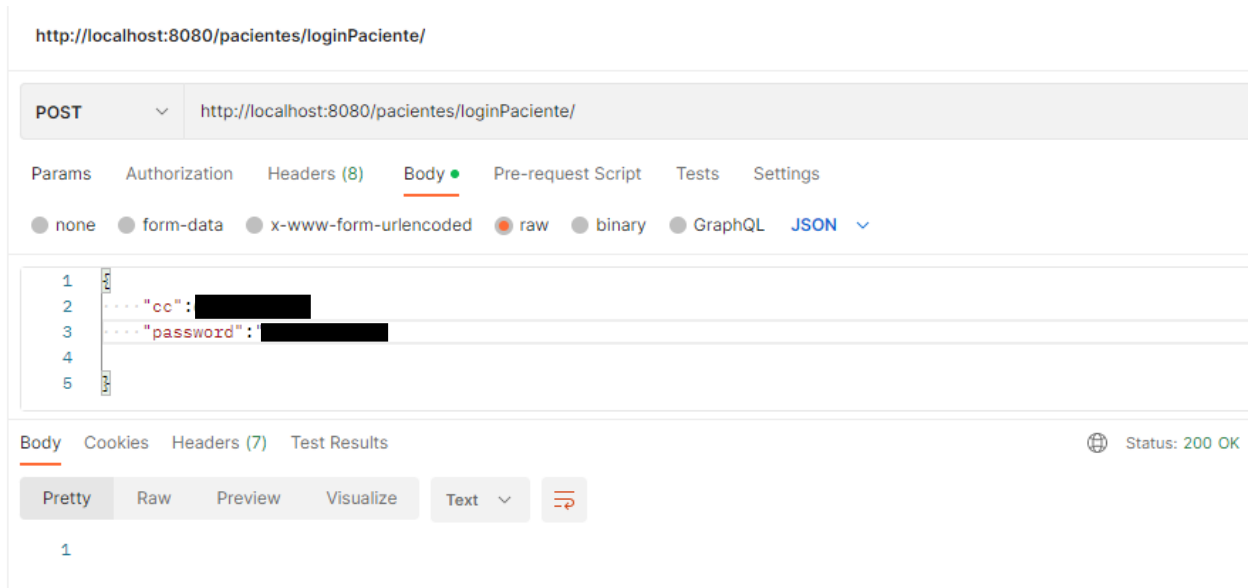


Figura 12 login fallido por password erróneo



1.1.6 Guardar paciente

Para guardar a los pacientes se probó por medio de un método post, que mandara todos los datos del paciente, así como lo haría el Front como se muestra en la siguiente ilustración:

Figura 13 guardar paciente

http://localhost:8080/pacientes/

POST http://localhost:8080/pacientes/

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1  {
2    "apellido": "Prueba",
3    "barrio": "palermo 2",
4    "cc": "123456789",
5    "celular": "██████████",
6    "ciudad": "piedecuesta",
7    "direccion": "██████████",
8    "email": "prueba@gmail.com",
9    "fec_nac": "1950-12-17",
10   "nombre": "prueba",
11   "password": "Prueba12345",
12   "sexo": "Femenina",
13   "tipo_doc": "C.C",
14   "altura": 1.58
15 }

```

Body Cookies Headers (9) Test Results Status: 201 Created

Pretty Raw Preview Visualize JSON

```

1  {
2    "id": 4,
3    "apellido": "Prueba",
4    "barrio": "palermo 2",
5    "cc": "123456789",
6    "celular": "██████████",
7    "ciudad": "piedecuesta",
8    "create_at": "2023-04-09",
9    "direccion": "██████████",
10   "email": "prueba@gmail.com",
11   "fec_nac": "1950-12-17",
12   "nombre": "prueba",
13   "password": "Prueba12345",
14   "sexo": "Femenina",
15   "tipo_doc": "C.C",
16   "altura": 1.58
17 }

```

Como se puede ver en la ilustración 13, el estatus de fue 201 el cual indica que se creó al paciente correctamente.

1.1.7 Listar visitas

Para probar que el back listara las visitas de una manera correcta se procedió a utilizar el método GET para listar todas las visitas y las visitas por la cedula del paciente.

El resultado de estos métodos se muestra en las siguientes ilustraciones:

Figura 14 listar todas las visitas

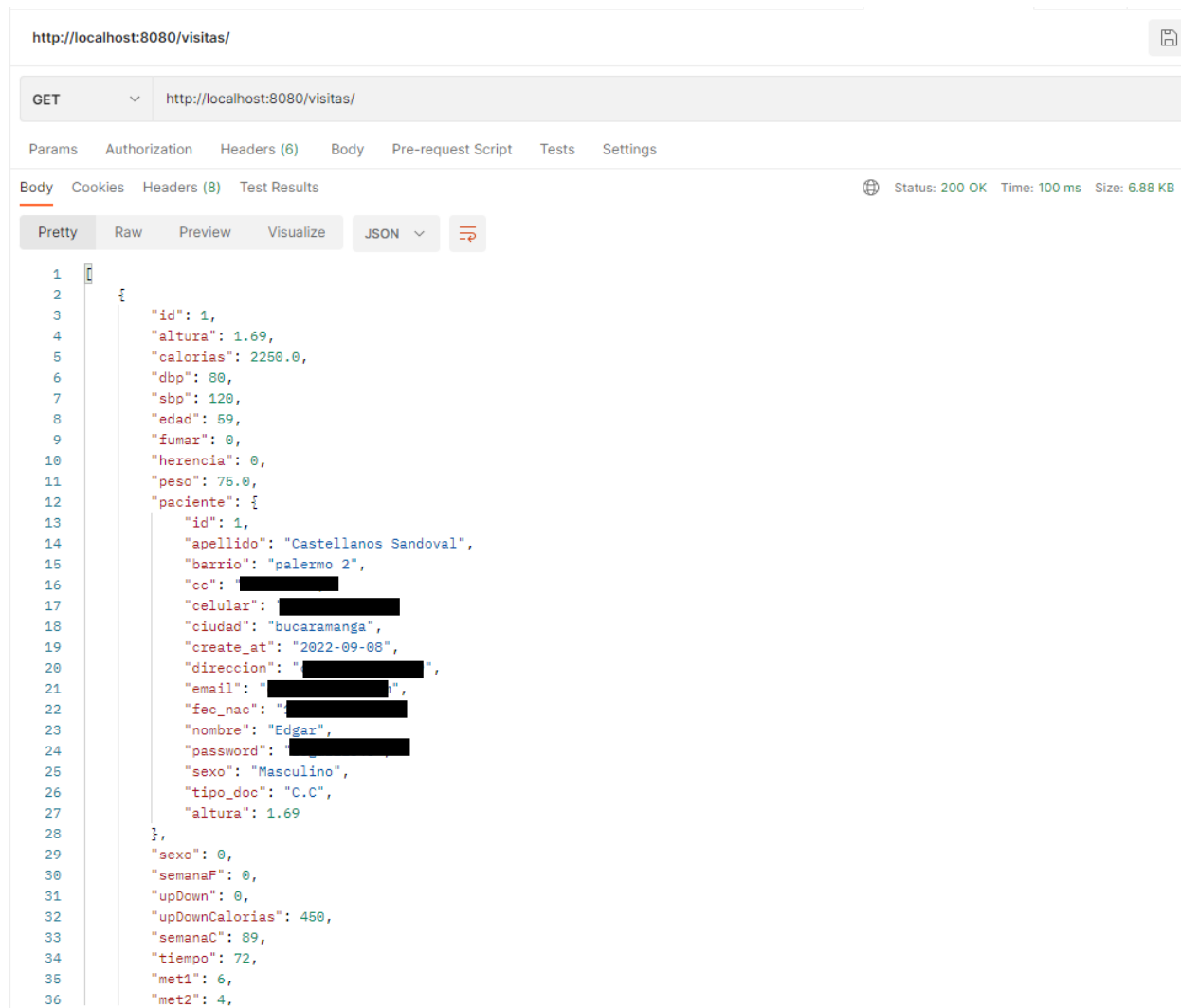
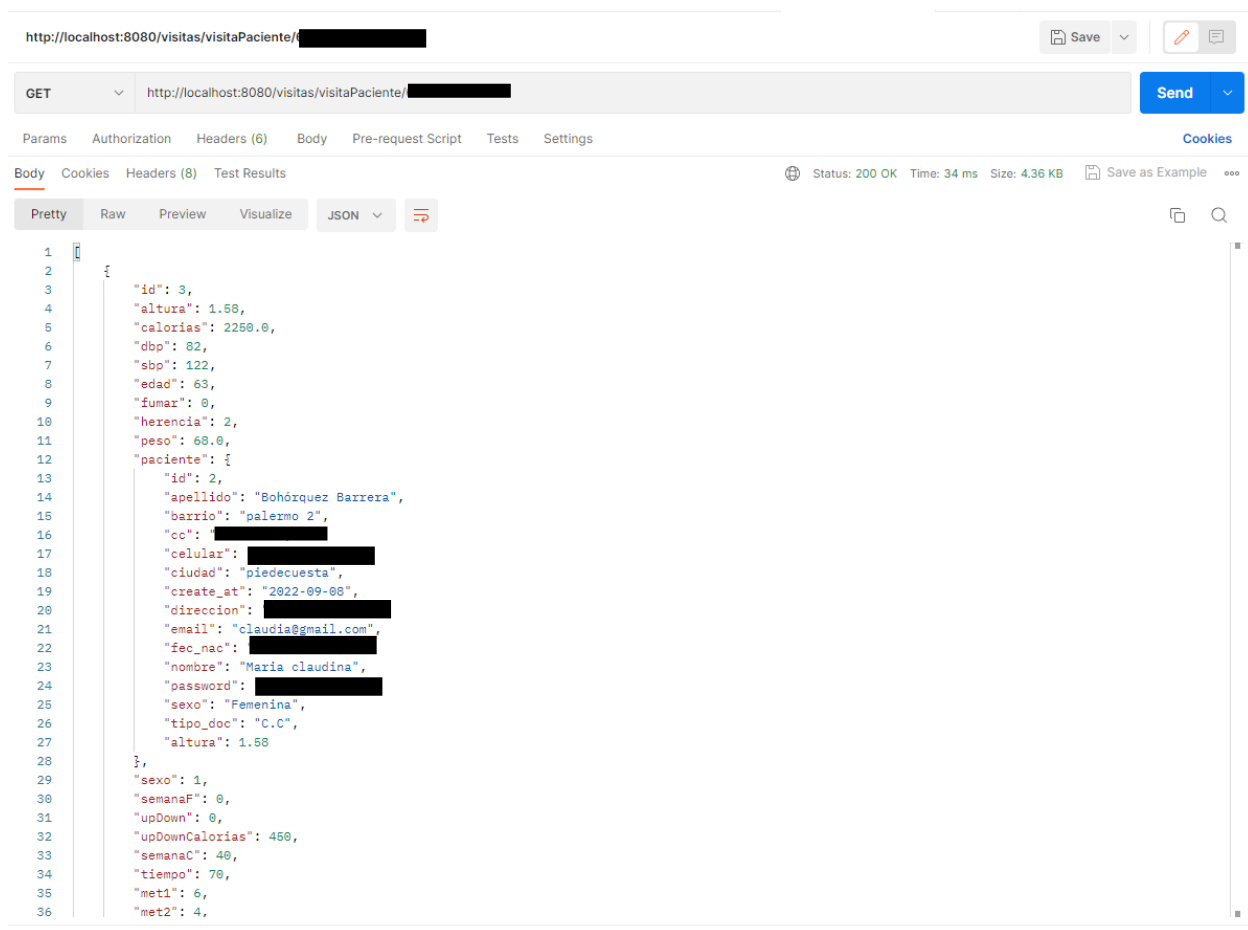


Figura 15 listar visita por cedula del paciente

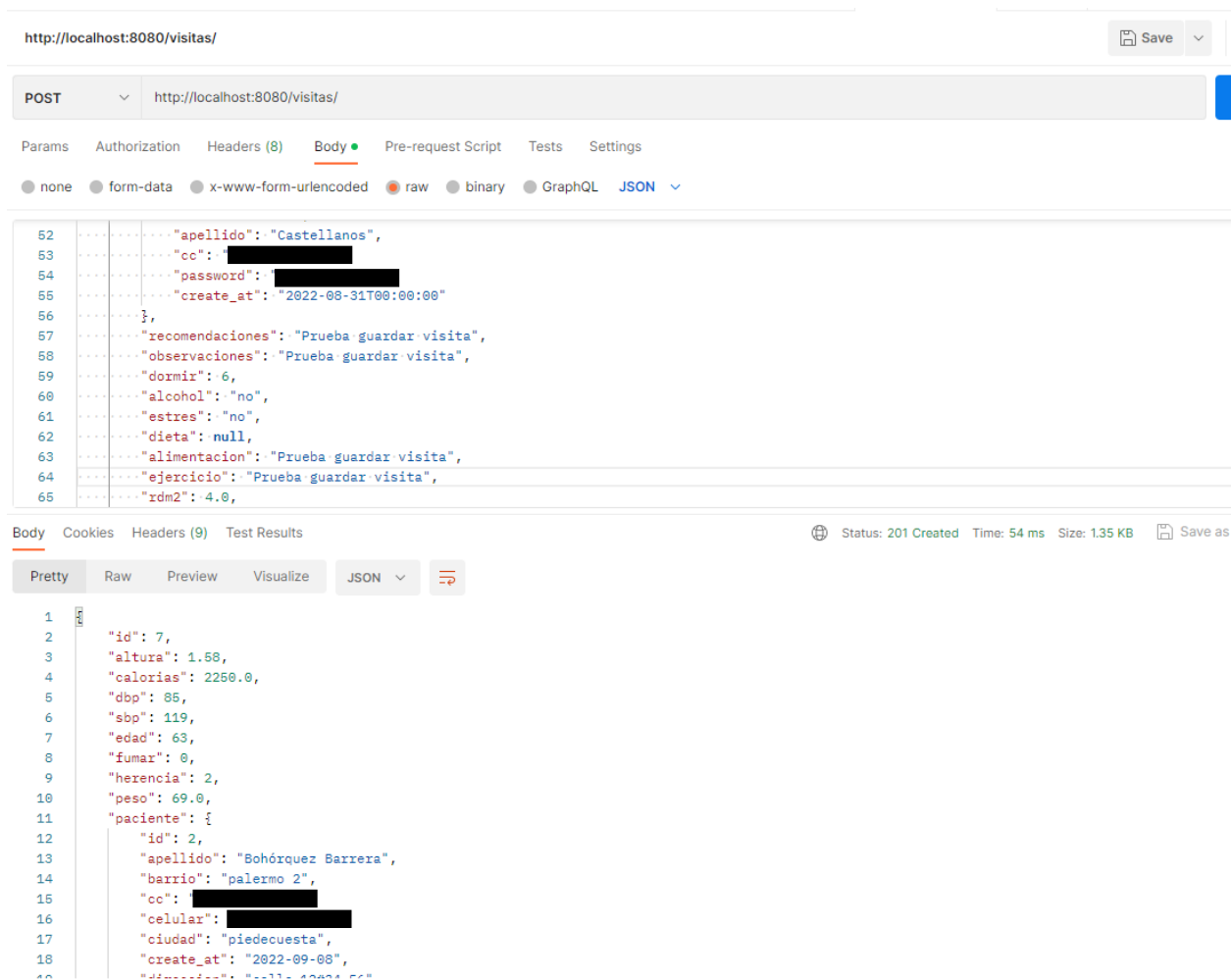


1.1.8 Guardar visita de un paciente

Una de las funciones más importantes del Back es guardar las visitas del paciente para que posteriormente el personal de salud pueda dar seguimiento al estado de salud del paciente, por tal motivo fue indispensable asegurar el correcto funcionamiento de esta función.

A continuación, se muestra los resultados de guardar los datos de la visita en la base de datos:

Figura 16 guardar visita



Como se puede ver en la ilustración 16, el status del método POST utilizado fue 201, lo cual indica que se guardó correctamente la información de la visita.