

Test certificados:

```
test > .\test_certificados.py >
1 def test_create_certificado(client, test_user, test_admin_user):
2     headers_user = {"Authorization": f"Bearer {test_user['access_token']}"}
3     headers_admin = {"Authorization": f"Bearer {test_admin_user['access_token']}"}
4
5     # Crear paciente (admin)
6     paciente_response = client.post("/api/v1/pacientes/", json={
7         "usuario_id": test_user["id"],
8         "nombres": "Paciente Certificado",
9         "cedula": "3333333333"
10    }, headers=headers_admin)
11    assert paciente_response.status_code == 201, f"Error creando paciente: {paciente_response.text}"
12    paciente = paciente_response.json()
13
14    # Crear colaborador (admin)
15    colaborador_response = client.post("/api/v1/colaboradores/", json={
16        "usuario_id": test_admin_user["id"],
17        "cedula": "4444444444",
18        "especialidad": "Medicina General"
19    }, headers=headers_admin)
20    assert colaborador_response.status_code == 201, f"Error creando colaborador: {colaborador_response.text}"
21    colaborador = colaborador_response.json()
22
23    # Crear cita (usuario normal)
24    cita_response = client.post("/api/v1/citas/", json={
25        "paciente_id": paciente["id"],
26        "colaborador_id": colaborador["id"],
27        "fecha_hora": "2025-07-21T11:00:00Z",
28        "motivo": "Consulta general"
29    }, headers=headers_user)
30    assert cita_response.status_code == 201, f"Error creando cita: {cita_response.text}"
31    cita = cita_response.json()
32
33    # Crear consulta (admin)
34    consulta_response = client.post("/api/v1/consultas/", json={
35        "cita_id": cita["id"],
36        "texto": "Consulta de rutina"
37    }, headers=headers_admin)
38    assert consulta_response.status_code == 201, f"Error creando consulta: {consulta_response.text}"
39    consulta = consulta_response.json()
40
41    # Verificar que los datos se guardaron correctamente en la base de datos
42    pacientes = client.get("/api/v1/pacientes/").json()
43    colaboradores = client.get("/api/v1/colaboradores/").json()
44    citas = client.get("/api/v1/citas/").json()
45    consultas = client.get("/api/v1/consultas/").json()
46
47    assert len(pacientes) == 1, "Se esperaba 1 paciente"
48    assert len(colaboradores) == 1, "Se esperaba 1 colaborador"
49    assert len(citas) == 1, "Se esperaba 1 cita"
50    assert len(consultas) == 1, "Se esperaba 1 consulta"
51
52    # Verificar que la cita está asociada al paciente y al colaborador
53    cita = citas[0]
54    assert cita["paciente_id"] == paciente["id"], "La cita no está asociada al paciente"
55    assert cita["colaborador_id"] == colaborador["id"], "La cita no está asociada al colaborador"
56
57    # Verificar que la consulta está asociada a la cita
58    consulta = consultas[0]
59    assert consulta["cita_id"] == cita["id"], "La consulta no está asociada a la cita"
60
61    # Limpieza de datos
62    delete_paciente(paciente)
63    delete_colaborador(colaborador)
64    delete_cita(cita)
65    delete_consulta(consulta)
66
67    # Verificar que los datos se eliminaron correctamente
68    pacientes = client.get("/api/v1/pacientes/").json()
69    colaboradores = client.get("/api/v1/colaboradores/").json()
70    citas = client.get("/api/v1/citas/").json()
71    consultas = client.get("/api/v1/consultas/").json()
72
73    assert len(pacientes) == 0, "Se esperaba 0 pacientes"
74    assert len(colaboradores) == 0, "Se esperaba 0 colaboradores"
75    assert len(citas) == 0, "Se esperaba 0 citas"
76    assert len(consultas) == 0, "Se esperaba 0 consultas"
77
78    # Limpieza de tokens
79    delete_token(test_user["access_token"])
80    delete_token(test_admin_user["access_token"])
81
82    # Limpieza de datos de prueba
83    delete_test_data()
84
85    # Limpieza de datos de prueba
86    delete_test_data()
87
88    # Limpieza de datos de prueba
89    delete_test_data()
90
91    # Limpieza de datos de prueba
92    delete_test_data()
93
94    # Limpieza de datos de prueba
95    delete_test_data()
96
97    # Limpieza de datos de prueba
98    delete_test_data()
99
100    # Limpieza de datos de prueba
101    delete_test_data()
102
103    # Limpieza de datos de prueba
104    delete_test_data()
105
106    # Limpieza de datos de prueba
107    delete_test_data()
108
109    # Limpieza de datos de prueba
110    delete_test_data()
111
112    # Limpieza de datos de prueba
113    delete_test_data()
114
115    # Limpieza de datos de prueba
116    delete_test_data()
117
118    # Limpieza de datos de prueba
119    delete_test_data()
120
121    # Limpieza de datos de prueba
122    delete_test_data()
123
124    # Limpieza de datos de prueba
125    delete_test_data()
126
127    # Limpieza de datos de prueba
128    delete_test_data()
129
130    # Limpieza de datos de prueba
131    delete_test_data()
132
133    # Limpieza de datos de prueba
134    delete_test_data()
135
136    # Limpieza de datos de prueba
137    delete_test_data()
138
139    # Limpieza de datos de prueba
140    delete_test_data()
141
142    # Limpieza de datos de prueba
143    delete_test_data()
144
145    # Limpieza de datos de prueba
146    delete_test_data()
147
148    # Limpieza de datos de prueba
149    delete_test_data()
150
151    # Limpieza de datos de prueba
152    delete_test_data()
153
154    # Limpieza de datos de prueba
155    delete_test_data()
156
157    # Limpieza de datos de prueba
158    delete_test_data()
159
160    # Limpieza de datos de prueba
161    delete_test_data()
162
163    # Limpieza de datos de prueba
164    delete_test_data()
165
166    # Limpieza de datos de prueba
167    delete_test_data()
168
169    # Limpieza de datos de prueba
170    delete_test_data()
171
172    # Limpieza de datos de prueba
173    delete_test_data()
174
175    # Limpieza de datos de prueba
176    delete_test_data()
177
178    # Limpieza de datos de prueba
179    delete_test_data()
180
181    # Limpieza de datos de prueba
182    delete_test_data()
183
184    # Limpieza de datos de prueba
185    delete_test_data()
186
187    # Limpieza de datos de prueba
188    delete_test_data()
189
190    # Limpieza de datos de prueba
191    delete_test_data()
192
193    # Limpieza de datos de prueba
194    delete_test_data()
195
196    # Limpieza de datos de prueba
197    delete_test_data()
198
199    # Limpieza de datos de prueba
200    delete_test_data()
201
202    # Limpieza de datos de prueba
203    delete_test_data()
204
205    # Limpieza de datos de prueba
206    delete_test_data()
207
208    # Limpieza de datos de prueba
209    delete_test_data()
210
211    # Limpieza de datos de prueba
212    delete_test_data()
213
214    # Limpieza de datos de prueba
215    delete_test_data()
216
217    # Limpieza de datos de prueba
218    delete_test_data()
219
220    # Limpieza de datos de prueba
221    delete_test_data()
222
223    # Limpieza de datos de prueba
224    delete_test_data()
225
226    # Limpieza de datos de prueba
227    delete_test_data()
228
229    # Limpieza de datos de prueba
230    delete_test_data()
231
232    # Limpieza de datos de prueba
233    delete_test_data()
234
235    # Limpieza de datos de prueba
236    delete_test_data()
237
238    # Limpieza de datos de prueba
239    delete_test_data()
240
241    # Limpieza de datos de prueba
242    delete_test_data()
243
244    # Limpieza de datos de prueba
245    delete_test_data()
246
247    # Limpieza de datos de prueba
248    delete_test_data()
249
250    # Limpieza de datos de prueba
251    delete_test_data()
252
253    # Limpieza de datos de prueba
254    delete_test_data()
255
256    # Limpieza de datos de prueba
257    delete_test_data()
258
259    # Limpieza de datos de prueba
260    delete_test_data()
261
262    # Limpieza de datos de prueba
263    delete_test_data()
264
265    # Limpieza de datos de prueba
266    delete_test_data()
267
268    # Limpieza de datos de prueba
269    delete_test_data()
270
271    # Limpieza de datos de prueba
272    delete_test_data()
273
274    # Limpieza de datos de prueba
275    delete_test_data()
276
277    # Limpieza de datos de prueba
278    delete_test_data()
279
280    # Limpieza de datos de prueba
281    delete_test_data()
282
283    # Limpieza de datos de prueba
284    delete_test_data()
285
286    # Limpieza de datos de prueba
287    delete_test_data()
288
289    # Limpieza de datos de prueba
290    delete_test_data()
291
292    # Limpieza de datos de prueba
293    delete_test_data()
294
295    # Limpieza de datos de prueba
296    delete_test_data()
297
298    # Limpieza de datos de prueba
299    delete_test_data()
300
301    # Limpieza de datos de prueba
302    delete_test_data()
303
304    # Limpieza de datos de prueba
305    delete_test_data()
306
307    # Limpieza de datos de prueba
308    delete_test_data()
309
310    # Limpieza de datos de prueba
311    delete_test_data()
312
313    # Limpieza de datos de prueba
314    delete_test_data()
315
316    # Limpieza de datos de prueba
317    delete_test_data()
318
319    # Limpieza de datos de prueba
320    delete_test_data()
321
322    # Limpieza de datos de prueba
323    delete_test_data()
324
325    # Limpieza de datos de prueba
326    delete_test_data()
327
328    # Limpieza de datos de prueba
329    delete_test_data()
330
331    # Limpieza de datos de prueba
332    delete_test_data()
333
334    # Limpieza de datos de prueba
335    delete_test_data()
336
337    # Limpieza de datos de prueba
338    delete_test_data()
339
340    # Limpieza de datos de prueba
341    delete_test_data()
342
343    # Limpieza de datos de prueba
344    delete_test_data()
345
346    # Limpieza de datos de prueba
347    delete_test_data()
348
349    # Limpieza de datos de prueba
350    delete_test_data()
351
352    # Limpieza de datos de prueba
353    delete_test_data()
354
355    # Limpieza de datos de prueba
356    delete_test_data()
357
358    # Limpieza de datos de prueba
359    delete_test_data()
360
361    # Limpieza de datos de prueba
362    delete_test_data()
363
364    # Limpieza de datos de prueba
365    delete_test_data()
366
367    # Limpieza de datos de prueba
368    delete_test_data()
369
370    # Limpieza de datos de prueba
371    delete_test_data()
372
373    # Limpieza de datos de prueba
374    delete_test_data()
375
376    # Limpieza de datos de prueba
377    delete_test_data()
378
379    # Limpieza de datos de prueba
380    delete_test_data()
381
382    # Limpieza de datos de prueba
383    delete_test_data()
384
385    # Limpieza de datos de prueba
386    delete_test_data()
387
388    # Limpieza de datos de prueba
389    delete_test_data()
390
391    # Limpieza de datos de prueba
392    delete_test_data()
393
394    # Limpieza de datos de prueba
395    delete_test_data()
396
397    # Limpieza de datos de prueba
398    delete_test_data()
399
400    # Limpieza de datos de prueba
401    delete_test_data()
402
403    # Limpieza de datos de prueba
404    delete_test_data()
405
406    # Limpieza de datos de prueba
407    delete_test_data()
408
409    # Limpieza de datos de prueba
410    delete_test_data()
411
412    # Limpieza de datos de prueba
413    delete_test_data()
414
415    # Limpieza de datos de prueba
416    delete_test_data()
417
418    # Limpieza de datos de prueba
419    delete_test_data()
420
421    # Limpieza de datos de prueba
422    delete_test_data()
423
424    # Limpieza de datos de prueba
425    delete_test_data()
426
427    # Limpieza de datos de prueba
428    delete_test_data()
429
430    # Limpieza de
```

Test citas:

```
test > @ test_citas.py >
1 def test_create_cita(client, test_user, test_admin_user):
2     headers_user = {"Authorization": f"Bearer {test_user['access_token']}"}
3     headers_admin = {"Authorization": f"Bearer {test_admin_user['access_token']}"}
4
5     # Crear paciente (usuario normal)
6     paciente_response = client.post("/api/v1/pacientes/", json={
7         "usuario_id": test_user["id"],
8         "nombres": "Paciente Cita",
9         "cedula": "1111111111"
10    }, headers=headers_user)
11    assert paciente_response.status_code == 201, f"Error creando paciente: {paciente_response.text}"
12    paciente = paciente_response.json()
13
14    # Crear colaborador (admin)
15    colaborador_response = client.post("/api/v1/colaboradores/", json={
16        "usuario_id": test_admin_user["id"],
17        "cedula": "2222222222",
18        "especialidad": "Pediatria"
19    }, headers=headers_admin)
20    assert colaborador_response.status_code == 201, f"Error creando colaborador: {colaborador_response.text}"
21    colaborador = colaborador_response.json()
22
23    # Crear cita (usuario normal)
24    cita_response = client.post("/api/v1/citas/", json={
25        "paciente_id": paciente["id"],
26        "colaborador_id": colaborador["id"],
27        "fecha_hora": "2025-07-20T10:00:00",
28        "motivo": "Control rutinario"
29    }, headers=headers_user)
30    assert cita_response.status_code == 201, f"Error creando cita: {cita_response.text}"
31
32    cita = cita_response.json()
33    assert cita["estado"] == "SOLICITADA"
34
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS G:\Users\Geovanny\Documents\Github\MedicSystem> python -m pytest test/test_citas.py -v

>>

===== test session starts =====

platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0 -- C:\Python313\python.exe

cachedir: .pytest_cache

rootdir: G:\Users\Geovanny\Documents\Github\MedicSystem

plugins: anyio-4.9.0, cov-6.2.1

collected 1 item

test/test_citas.py::test_create_cita PASSED [100%

Test_colaboradores:

```
test > test_colaboradores.py >
1 def test_create_colaborador(client, test_admin_user):
2     headers_admin = {"Authorization": f"Bearer {test_admin_user['access_token']}"}
3     response = client.post(
4         "/api/v1/colaboradores/",
5         json={
6             "usuario_id": test_admin_user["id"],
7             "cedula": "0987654321",
8             "especialidad": "Cardiologia",
9             "contacto": "0988888888"
10        },
11        headers=headers_admin
12    )
13    assert response.status_code == 201
14    assert "Cardiologia" in response.json()["especialidad"]
15

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS G:\Users\Geovanny\Documents\GitHub\MedicSystem> python -m pytest test/test_colaboradores.py -v
platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0 -- C:\Python313\python.exe
cachedir: .pytest_cache
rootdir: G:\Users\Geovanny\Documents\GitHub\MedicSystem
plugins: anyio-4.9.0, cov-6.2.1
collected 1 item

test/test_colaboradores.py::test_create_colaborador PASSED [100%]
```

Test_consultas:

```
test > test_consultas.py >
1 def test_create_consulta(client, test_user, test_admin_user):
2     headers_user = {"Authorization": f"Bearer {test_user['access_token']}"}
3     headers_admin = {"Authorization": f"Bearer {test_admin_user['access_token']}"}
4
5     # Crear paciente (usuario normal)
6     paciente_response = client.post("/api/v1/pacientes/", json={
7         "usuario_id": test_user["id"],
8         "nombres": "Paciente Consulta",
9         "cedula": "5555555555"
10    }, headers=headers_user)
11    assert paciente_response.status_code == 201, f"Error creando paciente: {paciente_response.text}"
12    paciente = paciente_response.json()
13
14    # Crear colaborador (admin)
15    colaborador_response = client.post("/api/v1/colaboradores/", json={
16        "usuario_id": test_admin_user["id"],
17        "cedula": "6666666666",
18        "especialidad": "Medicina Interna"
19    }, headers=headers_admin)
20    assert colaborador_response.status_code == 201, f"Error creando colaborador: {colaborador_response.text}"
21    colaborador = colaborador_response.json()
22
23    # Crear cita (usuario normal)
24    cita_response = client.post("/api/v1/citas/", json={
25        "paciente_id": paciente["id"],
26        "colaborador_id": colaborador["id"],
27        "fecha_hora": "2025-07-20T10:00:00",
28        "motivo": "Revisión general"
29    }, headers=headers_user)
30    assert cita_response.status_code == 201, f"Error creando cita: {cita_response.text}"
31    cita = cita_response.json()
32
33    # Crear consulta (admin)
34    consulta_response = client.post("/api/v1/consultas/", json={
35        "cita_id": cita["id"],
36        "colaborador_id": colaborador["id"],
37        "diagnostico": "Resfriado común"
38    }, headers=headers_admin)
39
40    PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
41
42    PS G:\Users\Geovanny\Documents\GitHub\MedicSystem> python -m pytest test/test_consultas.py -v
43    platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0 -- C:\Python313\python.exe
44    cachedir: .pytest_cache
45    rootdir: G:\Users\Geovanny\Documents\GitHub\MedicSystem
46    plugins: anyio-4.9.0, cov-6.2.1
47    collected 1 item
48
49    test/test_consultas.py::test_create_consulta PASSED [100%]
```

Test_pacientes:

```
test > test_pacientes.py > ...
1 def test_create_paciente(client, test_user):
2     headers_user = {"Authorization": f"Bearer {test_user['access_token']}"}
3
4     # Ahora permitimos crear paciente si el usuario es admin o es el mismo usuario (es_paciente)
5     response = client.post(
6         "/api/v1/pacientes/",
7         json={
8             "usuario_id": test_user["id"],
9             "nombres": "Paciente Test",
10            "cedula": "1234567890",
11            "contacto": "0999999999"
12        },
13        headers=headers_user
14    )
15    assert response.status_code == 201, f"Error creando paciente: {response.text}"
16
17
18
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS G:\Users\Geovanny\Documents\Github\MedicSystem> python -m pytest test/test_pacientes.py -v
platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0 -- C:\Python313\python.exe
cachedir: .pytest_cache
rootdir: G:\Users\Geovanny\Documents\Github\MedicSystem
plugins: anyio-4.9.0, cov-6.2.1
collected 1 item

test/test_pacientes.py::test_create_paciente PASSED [100%]

Test_reportes:

```
test > test_reportes.py > test_generate_reporte_medico
1 from datetime import datetime, timedelta, timezone
2
3 def test_generate_reporte_medico(client, test_admin_user):
4     headers_admin = {"Authorization": f"Bearer {test_admin_user['access_token']}"}
5     future_date = (datetime.now(timezone.utc) + timedelta(days=1)).strftime("%Y-%m-%d")
6
7     response = client.post(
8         "/api/v1/reportes/medicos",
9         json={"filtros": {"fecha_inicio": future_date, "especialidad": "Cardiología"}},
10        headers=headers_admin
11    )
12    assert response.status_code == 201, f"Error generando reporte médico: {response.text}"
13    reporte = response.json()
14    assert reporte["tipo_reporte"] == "MEDICO"
15    assert "Fecha_creacion" in reporte
16
17
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS G:\Users\Geovanny\Documents\Github\MedicSystem> python -m pytest test/test_reportes.py -v
platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0 -- C:\Python313\python.exe
cachedir: .pytest_cache
rootdir: G:\Users\Geovanny\Documents\Github\MedicSystem
plugins: anyio-4.9.0, cov-6.2.1
collected 1 item

test/test_reportes.py::test_generate_reporte_medico PASSED [100%]

Test_usuarios:

```
test > test_usuarios.py > test_login_user
1 def test_create_user(client):
2     response = client.post(
3         "/api/v1/usuarios/registro",
4         json={
5             "nombre_usuario": "newuser",
6             "correo_electronico": "user@example.com",
7             "contrasena": "securepassword"
8         }
9     )
10    assert response.status_code == 201
11    assert "id" in response.json()
12
13 def test_login_user(client, test_user):
14     response = client.post(
15         "/api/v1/usuarios/login",
16         data={
17             "username": "testuser",
18             "password": "testpass"
19         },
20         headers={"Content-Type": "application/x-www-form-urlencoded"}
21     )
22     assert response.status_code == 200
23     assert "access_token" in response.json()
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS G:\Users\Geovanny\Documents\Github\MedicSystem> python -m pytest test/test_usuarios.py -v

platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0 -- C:\Python313\python.exe
cachedir: .pytest_cache
rootdir: G:\Users\Geovanny\Documents\Github\MedicSystem
plugins: anyio-4.9.0, cov-6.2.1
collected 2 items

test/test_usuarios.py::test_create_user PASSED [50%]
test/test_usuarios.py::test_login_user PASSED [100%]

Coverage:

```
PS G:\Users\Geovanny\Documents\GitHub\MedicSystem> python -m pytest --cov-app --cov-report=term-missing test/
>>>
===== test session starts =====
platform win32 -- Python 3.13.1, pytest-8.4.1, pluggy-1.6.0
rootdir: G:\Users\Geovanny\Documents\GitHub\MedicSystem
plugins: anyio-4.9.0, cov-6.2.1
collected 8 items

test\test_certificados.py . [ 12%]
test\test_citas.py . [ 25%]
test\test_colaboradores.py . [ 37%]
test\test_consultas.py . [ 50%]
test\test_pacientes.py . [ 62%]
test\test_reportes.py . [ 75%]
test\test_usuarios.py .. [100%]
```

coverage: platform win32, python 3.13.1-final-0				
Name	Stmts	Miss	Cover	Missing
app__init__.py	5	0	100%	
app\api__init__.py	2	0	100%	
app\api\v1__init__.py	17	0	100%	
app\api\v1\certificados.py	16	2	88%	18, 29
app\api\v1\citas.py	16	2	88%	19, 32
app\api\v1\colaboradores.py	13	1	92%	18
app\api\v1\consultas.py	21	7	67%	19, 31-42
app\api\v1\pacientes.py	21	7	67%	20, 33-44
app\api\v1\reportes.py	14	1	93%	14
app\api\v1\usuarios.py	24	3	88%	27, 41, 49
app\core__init__.py	4	0	100%	
app\core\config.py	17	0	100%	
app\core\database.py	11	4	64%	11-15
app\core\password.py	6	0	100%	
app\core\security.py	46	9	80%	16, 19, 30-31, 41, 44, 47, 52, 57
app\main.py	19	6	68%	10-15
app\models__init__.py	9	0	100%	
app\models\base.py	5	0	100%	
app\models\certificado.py	13	0	100%	
app\models\cita.py	14	0	100%	
app\models\colaborador.py	17	0	100%	
app\models\consulta.py	19	0	100%	
app\models\paciente.py	14	0	100%	
app\models\reporte.py	12	0	100%	
app\models\usuario.py	14	0	100%	
app\repositories__init__.py	9	0	100%	
app\repositories\base.py	38	15	61%	18, 24, 32-43, 46-49
app\repositories\certificado.py	10	1	90%	11
app\repositories\cita.py	16	3	81%	10, 16, 22
app\repositories\colaborador.py	10	1	90%	11
app\repositories\consulta.py	10	2	80%	8, 11
app\repositories\paciente.py	8	0	100%	
app\repositories\reporte.py	10	2	80%	8, 11
app\repositories\usuario.py	24	2	92%	29, 31
app\schemas__init__.py	9	0	100%	
app\schemas\base.py	3	0	100%	
app\schemas\certificado.py	33	0	100%	
app\schemas\cita.py	26	0	100%	
app\schemas\colaborador.py	21	0	100%	
app\schemas\consulta.py	32	0	100%	
app\schemas\paciente.py	23	0	100%	
app\schemas\reporte.py	24	0	100%	
app\schemas\usuario.py	23	0	100%	
app\services__init__.py	9	0	100%	
app\services\base.py	29	13	55%	15-21, 24, 30-36, 39-45
app\services\certificado.py	28	6	79%	17, 31-37, 40
app\services\cita.py	31	14	55%	17-18, 26-37, 40-47
app\services\colaborador.py	15	2	87%	13, 20
app\services\consulta.py	40	18	55%	16, 27-40, 51-54, 57
app\services\paciente.py	13	1	92%	13
app\services\reporte.py	25	4	84%	20, 23, 35-41

app\services\usuario.py	22	3	86%	16, 25, 30
app\utils__init__.py	3	0	100%	
app\utils\enums.py	17	0	100%	
app\utils\helpers.py	13	3	77%	13, 17, 21

TOTAL	943	132	86%	