

Memoria de Prácticas

Práctica 1

Sistemas Informáticos II

Autores: Ignacio González Porras y Daniel Wentworth Fernández

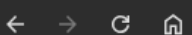
EJERCICIO 1. Despliegue Local y Prueba de Endpoints

1.1 Archivo de entorno (.env)

Su creación es necesaria para tener certeza que la aplicación se despliega en el puerto según lo indicado en el enunciado.

```
P1-base > $ env
1  # IMPORTANT: this file should not be in a repository
2  # To remove a file named env from a Git repository
3  # but keep it in the source (local system), follow these steps:
4  # Remove the file from Git tracking but keep it locally
5  ## git rm --cached env
6  # Add 'env' to .gitignore (so it's not tracked again)
7  ## echo "env" >> .gitignore
8  # Commit the changes
9  ## git commit -m "Removed env from Git tracking and added to .gitignore"
10 # Push the changes to the remote repository
11 ## git push
12 # use sqlite 3
13 ##DATABASE_SERVER_URL=sqlite:///db.sqlite3
14 # use postgres
15 DATABASE_SERVER_URL='postgres://alumnodb:alumnodb@localhost:15432/voto'
16 # The client does not need to store data in any database
17 # so let us define a sqlite in order to avoid warning messages
18 DEBUG=True
19 SECRET_KEY = 'django-insecure-alczftn)j1#$v%xmk@5j(n*px43c8kxgi_ua4%khc+t7g_)s9d'
20
```

1.2 Endpoints disponibles

 127.0.0.1:8000/votoApp/censo/

Introduzca la Informacion Censo de la Persona que Vota (votoSite)

Número de DNI:

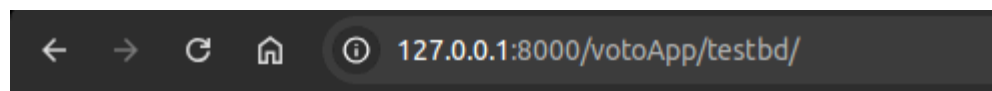
Nombre:

Fecha de Nacimiento:

Código de Autorización:

Aquí tenemos el endpoint cuya funcionalidad es verificar los datos del censo para una cierta persona.

Y aquí tenemos el formulario para registrar votos y consultar el censo.



Test Base de Datos: Registro de Voto (votoSite)

Introduzca los datos del nuevo voto a registrar:

ID Proceso Electoral:

ID Circunscripcion:

ID Mesa Electoral:

Nombre Candidato Votado:

Número de DNI:

Nombre:

Fecha de Nacimiento:

Código de Autorización:

Test Base de Datos: Borrado de Voto

Introduzca el ID del voto a Borrar:

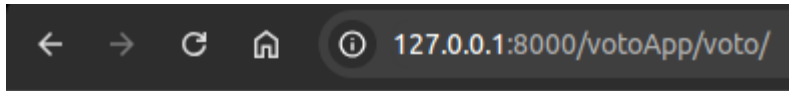
ID del Voto:

Test Base de Datos: Listado de Votos

Introduzca el ID del proceso electoral:

ID del Proceso Electoral:

Aquí vemos el endpoint para crear y gestionar votos.



Registro de Voto (votoSite)

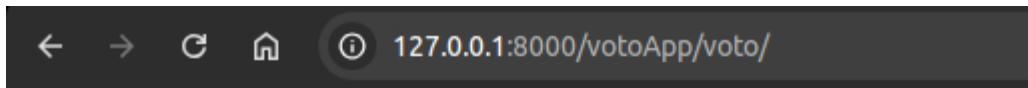
Introduzca los datos del nuevo voto a registrar:

ID Proceso Electoral:

ID Circunscripcion:

ID Mesa Electoral:

Nombre Candidato Votado:



Voto Registrado con Éxito (votoSite)

Id: 1

Codigo Respuesta: 000

Marca Tiempo : Feb. 21, 2025, 6:17 p.m.

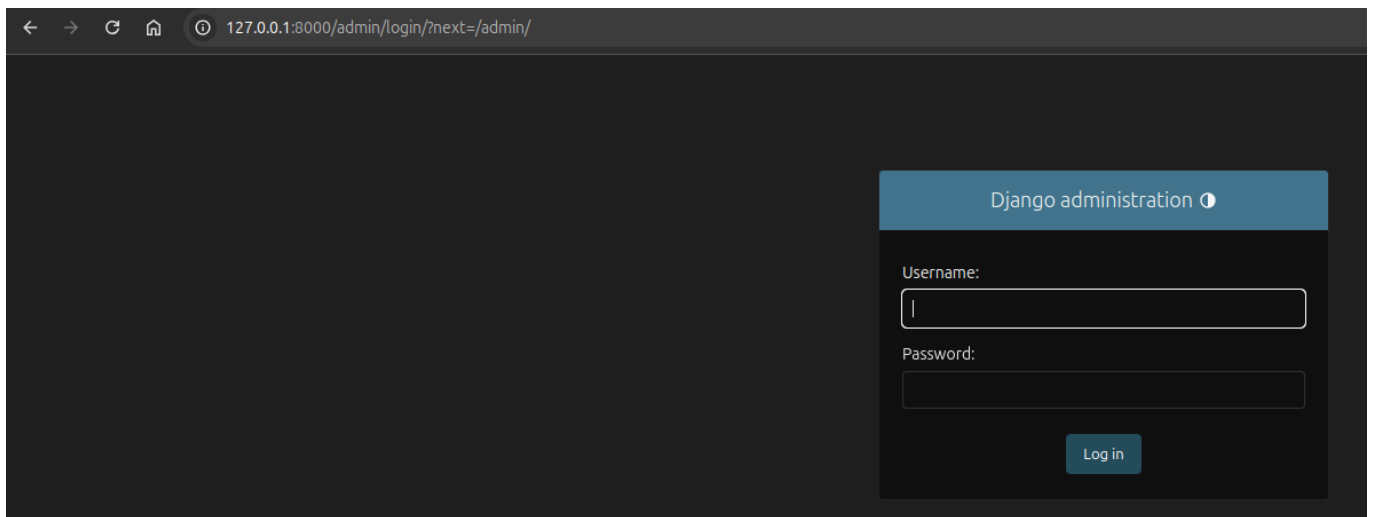
Id Circunscripcion : 1

Id Mesa Electoral : 1

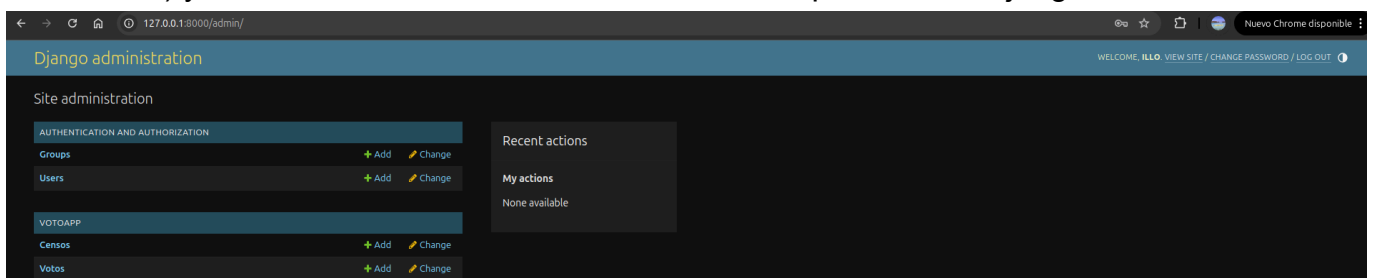
Id Proceso Electoral : 1

Nombre Candidato Votado: 1

Y vemos la respuesta del servidor tras introducir una serie de datos en cada uno de los campos, de manera exitosa.



Aquí vemos el lugar de inicio de sesión de los admins, que tienen acceso a toda la logística de manejo de votos y censos. Se hace login con el superusuario `alumnodb` (contraseña “`alumnodb`”) y se tiene acceso a la administración de la aplicación de django.



EJERCICIO 2. Ejecución de tests

A continuación se adjunta una captura en el que pasa los tests de la práctica:

```
(venv) ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python manage.py test
Found 18 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
..Error: Registrando voto: null value in column "censo_id" of relation "voto" violates not-null constraint
DETAIL: Failing row contains (1, C001, M001, P001, Candidato A, 2025-02-21 17:27:44.825923+00, 000, null).

.....
-----
Ran 18 tests in 0.106s

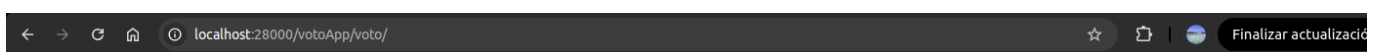
OK
Destroying test database for alias 'default'...
```

Todos los tests que se proporcionan se ejecutan de manera correcta y como deberían. Se detecta de manera puntual un error, en el cual dice que registrando voto tiene un valor nulo que causa un problema, el cual se atribuye a la manera en la cual se inicializan los valores afectados.

EJERCICIO 3. Despliegue en máquinas virtuales

```
GNU nano 7.2 /home/si2/repo/pibase/P1-base/env
# IMPORTANT: this file should not be in a repository
# To remove a file named env from a Git repository
# but keep it in the source (local system), follow these steps:
# Remove the file from Git tracking but keep it locally
## git rm --cached env
# Add 'env' to .gitignore (so it's not tracked again)
## echo "env" >> .gitignore
# Commit the changes
## git commit -m "Removed env from Git tracking and added to .gitignore"
# Push the changes to the remote repository
## git push
# use sqlite 3
##DATABASE_SERVER_URL=sqlite:///db.sqlite3
# use postgres
DATABASE_SERVER_URL='postgres://alumnodb:alumnodb@192.168.86.35:15432/voto'
# DATABASE_SERVER_URL="postgresql://neondb_owner:npg_pXukNK7E0w4Q@ep-curly-hall-a9z21sjf-pooler.gwc.azure.neon.tech/neondb?sslmode=require" # URL de n
# The client does not need to store data in any database
# so let us define a sqlite in order to avoid warning messages
DEBUG=True
SECRET_KEY = 'django-insecure-alczftn)j1#$v%xmK@5j(n*px43c8kxgi_ua4%khc+t7g_)s9d'
```

Aquí en la primera imagen vemos el archivo de entorno empleado para esta sección, no difiere demasiado del usado inicialmente.



Voto Registrado con Éxito (votoSite)

Id: 2

Codigo Respuesta: 000

Marca Tiempo : Feb. 22, 2025, 6:38 p.m.

Id Circunscripcion : 2

Id Mesa Electoral : 2

Id Proceso Electoral : 2

Nombre Candidato Votado: 2

id	idCircunscripcion	idMesaElectoral	idProcesoElectoral	nombreCandidatoVotado	marcaTiempo	codigoRespuesta	censo_id
1	1	1	1	1	2025-02-22 18:37:28.21996+01	000	60060060C
2	2	2	2	2	2025-02-22 18:38:20.054373+01	000	60060060C

(2 rows)

(END)

En esta segunda fotografía, vemos claramente tanto la salida dentro de la conexión por terminal con el voto registrado a través de la URL del navegador que ha sido recibido, como la propia URL dentro del navegador mencionado previamente con la salida de éxito tras haber introducido los datos para registrar el voto.

EJERCICIO 4. Análisis de rendimiento de acceso a datos

Ahora veremos la comparación de los accesos en cada uno de los casos.

Primero vemos en la máquina virtual 1:

```
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.232160 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.179921 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.211932 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.182815 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.191886 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.217700 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.175741 segundos
```

Luego tenemos los accesos con neon:

```
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 38.649869 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 40.906444 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 38.114235 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 37.186595 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 36.210956 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 34.451960 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 36.344348 segundos
```

Y por último usando django:

```
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.510602 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.407120 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.656592 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.556803 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.426286 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.411587 segundos
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/5o_año/SI2/SI2/P1-base$ python read_1000_entries_from_db.py
Tiempo invertido en buscar las 1000 entradas una a una: 0.556842 segundos
```

Análisis comparativo:

VM1 (Acceso directo):

- Tiempo más rápido
- Desviación estándar pequeña en términos absolutos
- Variabilidad relativa moderada

Django ORM:

- Tiempo intermedio
- Desviación estándar moderada
- Mayor variabilidad relativa de los tres casos

Neon.tech:

- Tiempo más lento
- Mayor desviación estándar en términos absolutos
- Menor variabilidad relativa

VM1: 0.199 ± 0.021 segundos (CV = 10.5%)

Django ORM: 0.504 ± 0.093 segundos (CV = 18.4%)

Neon.tech: 37.409 ± 2.048 segundos (CV = 5.47%)

CUESTIÓN 1.

El acceso directo a VM1 es el más rápido y bastante consistente, Django ORM añade overhead pero mantiene tiempos razonables, aunque con mayor variabilidad y Neon.tech muestra tiempos significativamente más altos debido a la latencia de red, pero sorprendentemente con la menor variabilidad relativa

Esto refuerza la conclusión anterior sobre la necesidad de más repeticiones. Inicialmente se podría pensar que es Neon el cual precisa de este tratamiento pero al analizar la desviación relativa, se trata de especialmente en el caso de Django ORM que muestra la mayor variabilidad relativa de todas, en el cual conviene realizar una mayor cantidad de repeticiones.

EJERCICIO 5. Tests con VM2

Aquí, se trata de una ejecución y comentario exactamente idéntica que en el ejercicio 2. El único cambio reseñable es donde se encuentra la aplicación, que ahora es en la segunda máquina virtual. Esto hará que se tuvo que modificar el archivo de entorno para que tuviera esta pinta:

```
DATABASE_SERVER_URL='postgres://alumnodb:alumnodb@192.168.86.35:15432/voto'
```

```
si2@si2-ubuntu-vm-3:~/repo/p1base/P1-base$ cd /home/si2/repo/p1base/P1-base/
si2@si2-ubuntu-vm-3:~/repo/p1base/P1-base$ python manage.py test
Found 18 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
..Error: Registrando voto: null value in column "censo_id" of relation "voto" violates not-null constraint
DETAIL: Failing row contains (1, C001, M001, P001, Candidato A, 2025-02-24 07:45:17.930261+00, 000, null).

.....
-----
Ran 18 tests in 0.119s

OK
Destroying test database for alias 'default'...
si2@si2-ubuntu-vm-3:~/repo/p1base/P1-base$
```


EJERCICIO 6. Ejecución de tests con WSServer

```
si2@si2-ubuntu-vm-1:~/repo/p1base/P1-ws-server/votoAppWSServer$ python manage.py test
Found 6 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
.....
-----
Ran 6 tests in 0.042s

OK
Destroying test database for alias 'default'...
```

Vemos el comando usado desde votoAppWSServer y que se ejecuta los tests sin ningún problema, pasando todos los que hay.

EJERCICIO 7. Endpoints con REST API.

```
(venv) ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_año/S12/S12/P1-ws-server/votoAppWSServer$ curl -X DELETE http://127.0.0.1:28000/votoAppWSServer/restapiserver/voto/5
{"message":"Voto eliminado"}
```

Esta primera conexión, la realizamos a la URL

<http://127.0.0.1:28000/votoAppWSServer/restapiserver/voto/5>

Usamos el método DELETE para eliminar un voto de la base de datos, lo cual se puede ver en la misma tras la ejecución de este acceso:

```
voto=# select * from voto;
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | codigoRespuesta | censo_id
-----+-----+-----+-----+-----+-----+-----+-----
 5 | CIRC127          | MESA130         | 2025               | Francisco Gibson      | 2025-02-22 20:51:19.615684+01 | 000             | 525525530
(1 row)

voto=# select * from voto;
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | codigoRespuesta | censo_id
-----+-----+-----+-----+-----+-----+-----+-----
(0 rows)
```

Se ve el antes y el después de la ejecución del DELETE sobre el voto con id 5, que se ve en la URL proporcionada.

```
(venv) ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_año/S12/S12/P1-ws-server/votoAppWSServer$ curl -X POST http://127.0.0.1:28000/votoAppWSServer/restapiserver/censo/ -H "Content-Type: application/json" -d '{
  "numeroDNI": "93993993X",
  "nombre": "Francisco Gibson Coll",
  "fechaNacimiento": "06/09/83",
  "anioCenso": "2025",
  "codigoAutorizacion": "249"
}'
(venv) ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_año/S12/S12/P1-ws-server/votoAppWSServer$ curl -X POST http://127.0.0.1:28000/votoAppWSServer/restapiserver/voto/ -H "Content-Type: application/json" -d '{
  "idCircunscripcion": "CIRC127",
  "idMesaElectoral": "MESA130",
  "idProcesoElectoral": "2025",
  "nombreCandidatoVotado": "Francisco Gibson",
  "censo_id": "525525530"
}'
{"idCircunscripcion": "CIRC127", "idMesaElectoral": "MESA130", "idProcesoElectoral": "2025", "nombreCandidatoVotado": "Francisco Gibson", "censo_id": "525525530", "marcaTiempo": "2025-02-22T20:51:19.615684+01:00"}
```

En este otro acceso, nos conectaremos a la URL

<http://127.0.0.1:28000/votoAppWSServer/restapiserver/censo/>

Usamos el método POST para en este caso añadir un voto, con los datos que vemos proporcionados:

idCircunscripcion: "CIRC127"

idMesaElectoral: "MESA130"

idProcesoElectoral: "2025"

nombreCandidatoVotado: "Francisco Gibson"

censo_id: "525525530"

El resultado lo podemos observar ya en la BBDD:

```
voto=# select * from voto;
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | codigoRespuesta | censo_id
-----+-----+-----+-----+-----+-----+-----+-----
 5 | CIRC127           | MESA130         | 2025               | Francisco Gibson      | 2025-02-22 20:51:19.615684+01 | 000             | 525525530
(1 row)
```

A continuación, dejamos una captura con pruebas hechas con datos inválidos:

```
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_ano/SI2/SI2/P1-base$ curl -X POST http://127.0.0.1:28000/votoAppWSServer/restapiserver/censo/ -H "Content-Type: application/json" -d '{
  "numeroDNI": "999999999M",
  "nombre": "Francisco Martinez",
  "fechaNacimiento": "06/09/83",
  "anioCenso": "2025",
  "codigoAutorizacion": "111"
}'
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_ano/SI2/SI2/P1-base$ curl -X POST http://127.0.0.1:28000/votoAppWSServer/restapiserver/voto/ -H "Content-Type: application/json" -d '{
  "idCircunscripcion": "CIRC999",
  "idMesaElectoral": "MESA999",
  "idProcesoElectoral": "2025",
  "nombreCandidatoVotado": "Francisco Martinez",
  "censo_id": "123456789"
}'
ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_ano/SI2/SI2/P1-base$ curl -X DELETE http://127.0.0.1:28000/votoAppWSServer/restapiserver/voto/48
{"message": "Voto no encontrado"}ignacioglezz@ignacioglezz:~/Escritorio/UNIVERSIDAD/So_ano/SI2/SI2/P1-base$
```

EJERCICIO 8. Tests ws-client

Igual que en el apartado 6, ejecutamos los tests y tenemos resultados sin problemas.

```
.deleting votes
DELETE 1
This test only works if (1) http://localhost:28000/restapiserver/
server is up and running and (2) the database is populated.
Very likely in the future ths test should be done with mocks
and not real calls to the server.

INSERT 0 1
.deleting votes
DELETE 0
This test only works if (1) http://localhost:28000/restapiserver/
server is up and running and (2) the database is populated.
Very likely in the future ths test should be done with mocks
and not real calls to the server.

INSERT 0 1
INSERT 0 1
INSERT 0 1
.deleting votes
DELETE 3
This test only works if (1) http://localhost:28000/restapiserver/
server is up and running and (2) the database is populated.
Very likely in the future ths test should be done with mocks
and not real calls to the server.

.deleting votes
DELETE 1
This test only works if (1) http://localhost:28000/restapiserver/
server is up and running and (2) the database is populated.
Very likely in the future ths test should be done with mocks
and not real calls to the server.

.
-----
Ran 8 tests in 0.721s

OK
Destroying test database for alias 'default'...
```


EJERCICIO 9. Despliegue web de server y client

Registro de Voto (votoSite)

Introduzca los datos del nuevo voto a registrar:

ID Proceso Electoral:

ID Circunscripcion:

ID Mesa Electoral:

Nombre Candidato Votado:

Terminal output:

```
ignacioglezz@ignacioglezz: ~  
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | cod  
igoRespuesta | censo_id  
-----+-----+-----+-----+-----+-----+-----  
(0 rows)  
(END)
```

Terminal output (server):

```
OK  
Destroying test database for alias 'default'...  
si2@si2-ubuntu-vm-3:~/repo/pibase/P1-ws-client/votoAppWSClient$ python manage.py runserver 0.0.0.0:8000  
Watching for file changes with StatReloader  
Performing system checks...  
  
System check identified no issues (0 silenced).  
February 23, 2025 - 13:10:02  
Django version 4.2.13, using settings 'votoSite.settings'  
Starting development server at http://0.0.0.0:8000/  
Quit the server with CONTROL-C.  
  
[23/Feb/2025 13:13:06] "POST /votoApp/ HTTP/1.1" 302 0  
[23/Feb/2025 13:13:06] "GET /votoApp/voto/ HTTP/1.1" 200 1081
```

Vemos el despliegue completo de tanto el cliente como el servidor.

Voto Registrado con Éxito (votoSite)

Id:

Codigo Respuesta:

Marca Tiempo : 2025-02-23T13:13:41.888596+01:00

Id Circunscripcion : 1

Id Mesa Electoral : 1

Id Proceso Electoral : 1

Nombre Candidato Votado: 1

Terminal output:

```
ignacioglezz@ignacioglezz: ~  
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | codigoRespuesta | censo_id  
-----+-----+-----+-----+-----+-----+-----+-----  
14 | 1 | 1 | 1 | 1 | 2025-02-23 13:13:41.888596+01 | 000 | 60060060C  
(1 row)  
:
```

Terminal output (server):

```
Destroying test database for alias 'default'...  
si2@si2-ubuntu-vm-3:~/repo/pibase/P1-ws-client/votoAppWSClient$ python manage.py runserver 0.0.0.0:8000  
Watching for file changes with StatReloader  
Performing system checks...  
  
System check identified no issues (0 silenced).  
February 23, 2025 - 13:10:02  
Django version 4.2.13, using settings 'votoSite.settings'  
Starting development server at http://0.0.0.0:8000/  
Quit the server with CONTROL-C.  
  
[23/Feb/2025 13:13:06] "POST /votoApp/ HTTP/1.1" 302 0  
[23/Feb/2025 13:13:06] "GET /votoApp/voto/ HTTP/1.1" 200 1081  
[23/Feb/2025 13:13:41] "POST /votoApp/voto/ HTTP/1.1" 200 402
```

Ahora se ha registrado un voto que se ve reflejado tanto en el éxito del formulario, como en la petición POST enviada como el voto detectado ya en la base de datos.

¡Voto eliminado correctamente!

```
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | codigoRespuesta | censo_id
-----+-----+-----+-----+-----+-----+-----+-----
(0 rows)

voto=#
```

```
ignacioglezz@ignacioglezz: ~
Django version 4.2.13, using settings 'votoSite.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.

[23/Feb/2025 13:13:06] "POST /votoApp/ HTTP/1.1" 302 0
[23/Feb/2025 13:13:06] "GET /votoApp/voto/ HTTP/1.1" 200 1081
[23/Feb/2025 13:13:41] "POST /votoApp/voto/ HTTP/1.1" 200 402
Not Found: /votoApp/testdb/
[23/Feb/2025 13:14:31] "GET /votoApp/testdb/ HTTP/1.1" 404 3193
[23/Feb/2025 13:14:39] "GET /votoApp/testbd/ HTTP/1.1" 200 2495
[23/Feb/2025 13:15:10] "POST /votoApp/testbd/delvoto/ HTTP/1.1" 200 186
```

Ahora proseguimos con una prueba más en la que procedemos a eliminar dicho voto, que a su vez se ve reflejado en todas las ventanas que tenemos, igual que en la instancia anterior.

Votos Registrados (votoSite)

id	idCircunscripcion	idMesaElectoral	Candidato Votado	Marca Tiempo	Codigo Respuesta
1	1	1	1	2025-02-23T13:16:48.055345+01:00	
1	1	1	1	2025-02-23T13:17:15.512329+01:00	
1	5	2	1	2025-02-23T13:17:42.268111+01:00	

```
ignacioglezz@ignacioglezz: ~
voto=# select * from voto;
id | idCircunscripcion | idMesaElectoral | idProcesoElectoral | nombreCandidatoVotado | marcaTiempo | codigoRespuesta | censo_id
-----+-----+-----+-----+-----+-----+-----+-----
15 | 1 | 1 | 1 | 1 | 2025-02-23 13:16:48.055345+01 | 000 | 801801800
16 | 1 | 1 | 1 | 1 | 2025-02-23 13:17:15.512329+01 | 000 | 98598598R
17 | 1 | 5 | 1 | 2 | 2025-02-23 13:17:42.268111+01 | 000 | 59059059U
(3 rows)

voto=#
```

```
ignacioglezz@ignacioglezz: ~
[23/Feb/2025 13:14:31] "GET /votoApp/testdb/ HTTP/1.1" 404 3193
[23/Feb/2025 13:14:39] "GET /votoApp/testbd/ HTTP/1.1" 200 2495
[23/Feb/2025 13:15:10] "POST /votoApp/testbd/delvoto/ HTTP/1.1" 200 186
[23/Feb/2025 13:16:48] "POST /votoApp/testbd/ HTTP/1.1" 200 402
[23/Feb/2025 13:16:49] "GET /votoApp/testbd/ HTTP/1.1" 200 2495
[23/Feb/2025 13:17:15] "POST /votoApp/testbd/ HTTP/1.1" 200 402
[23/Feb/2025 13:17:19] "GET /votoApp/testbd/ HTTP/1.1" 200 2495
[23/Feb/2025 13:17:42] "POST /votoApp/testbd/ HTTP/1.1" 200 402
[23/Feb/2025 13:17:43] "GET /votoApp/testbd/ HTTP/1.1" 200 2495
[23/Feb/2025 13:18:04] "POST /votoApp/testbd/getvotos/ HTTP/1.1" 200 782
```

Por último vemos la lista completa de votos al finalizar.