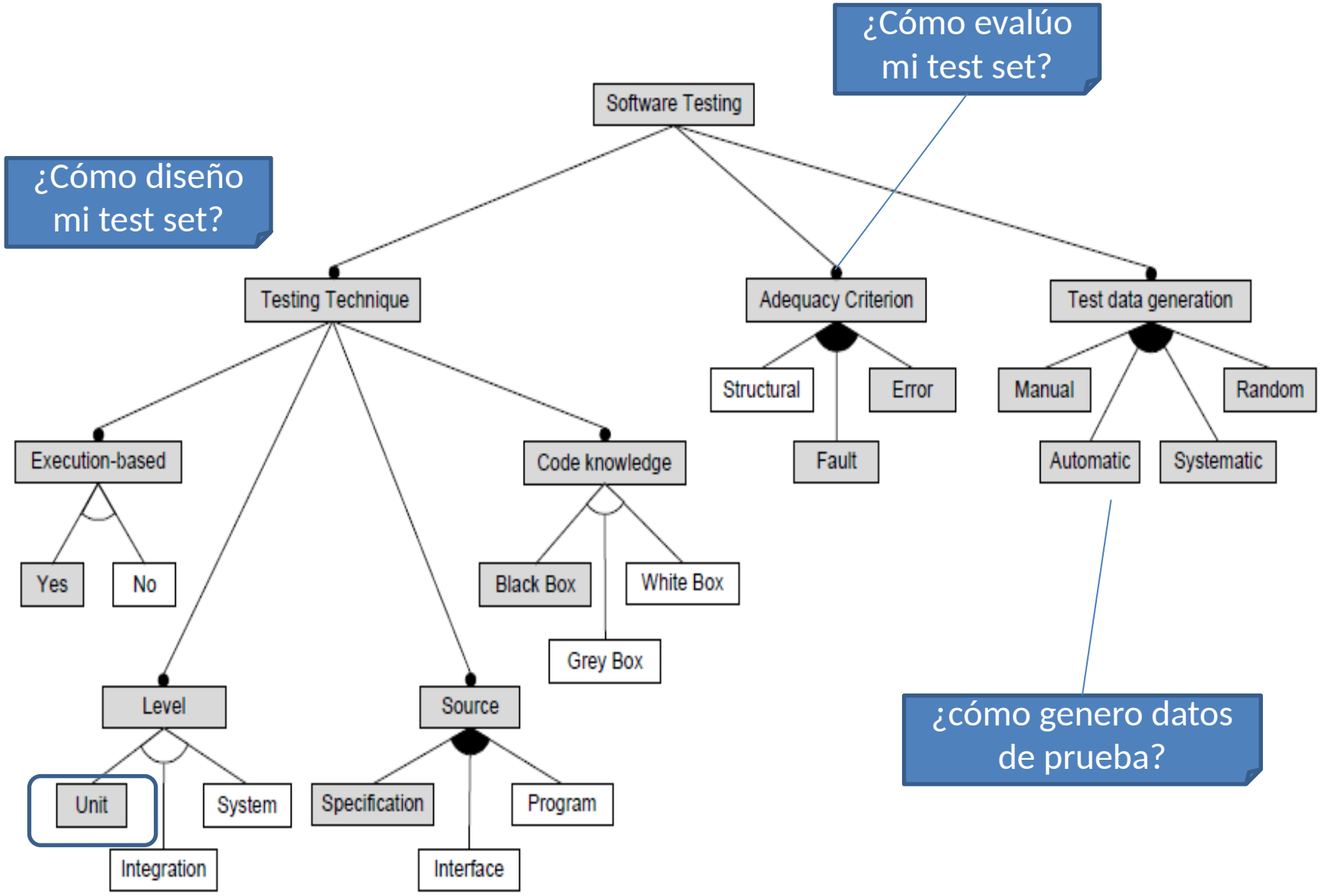
escuela técnica superior
de ingeniería informática

Pruebas de software

Departamento de Lenguajes y Sistemas Informáticos

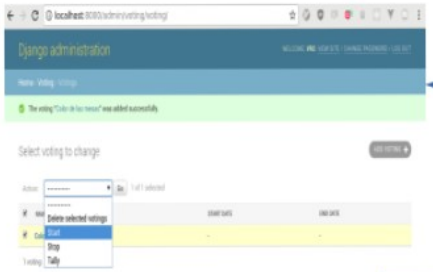
UNIVERSIDAD D SEVILLA

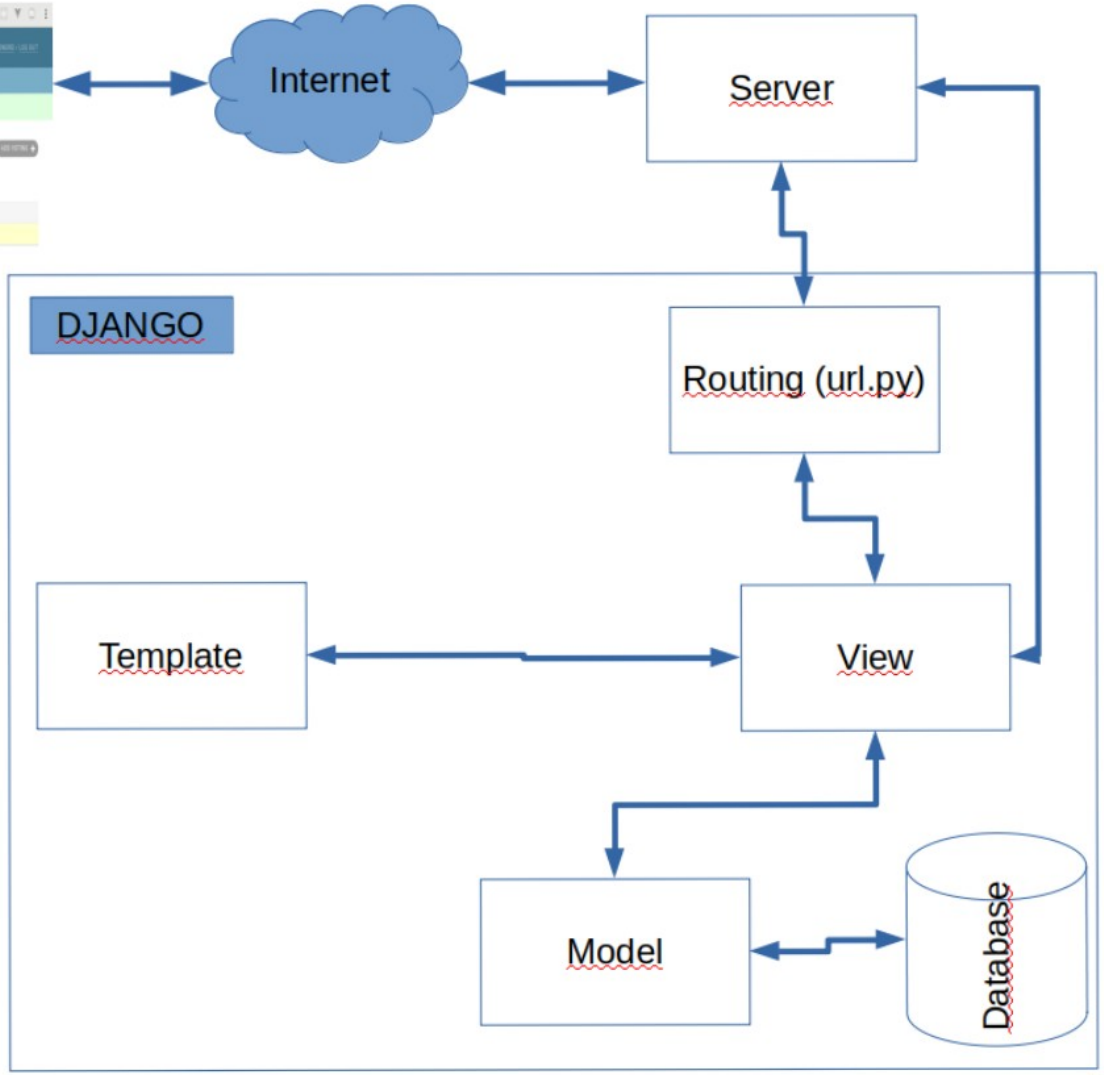
Testing:max 2^26 testing posibles!!



Las pruebas unitarias están diseñadas para ejercitar una parte pequeña y específica de funcionalidad

La estructura de Django





Proceso general de prueba

Implementación de pruebas en django

Un Ejemplo:

```
from django.test import TestCase

class SimpleTest(TestCase):
    def test_basic_addition(self):
    """
    Tests that 1 + 1 always equals 2.
    """
   self.assertEqual(1 + 1, 2)
```

Framework de testing unitario

(**unnittest** → Inspirado en *JUnit*)

Conceptos

- **test fixture** - Preparación necesaria para realizar las pruebas
- **test case** - Caso concreto e individual que se quiere probar
- **test suite** - Conjunto de casos de prueba.
- **test runner** - Componente que ejecuta los tests.

Lugar de implementación y ejecución

- La aplicación crea un fichero **tests.py** por defecto.
- Si necesitamos **más complejidad** →
  Crear nuevos scripts de formato **test*.py**

Una vez escritos, se ejecutan desde la terminal:

```
#Corre todos los tests disponibles
$./manage.py test

#Corre los tests dentro de “voting”
$./manage.py test voting
```

Test de modelos

En Django, los tests referentes a la base de datos no usan la BBDD de **producción**.

(No es necesario declararla en settings.py)

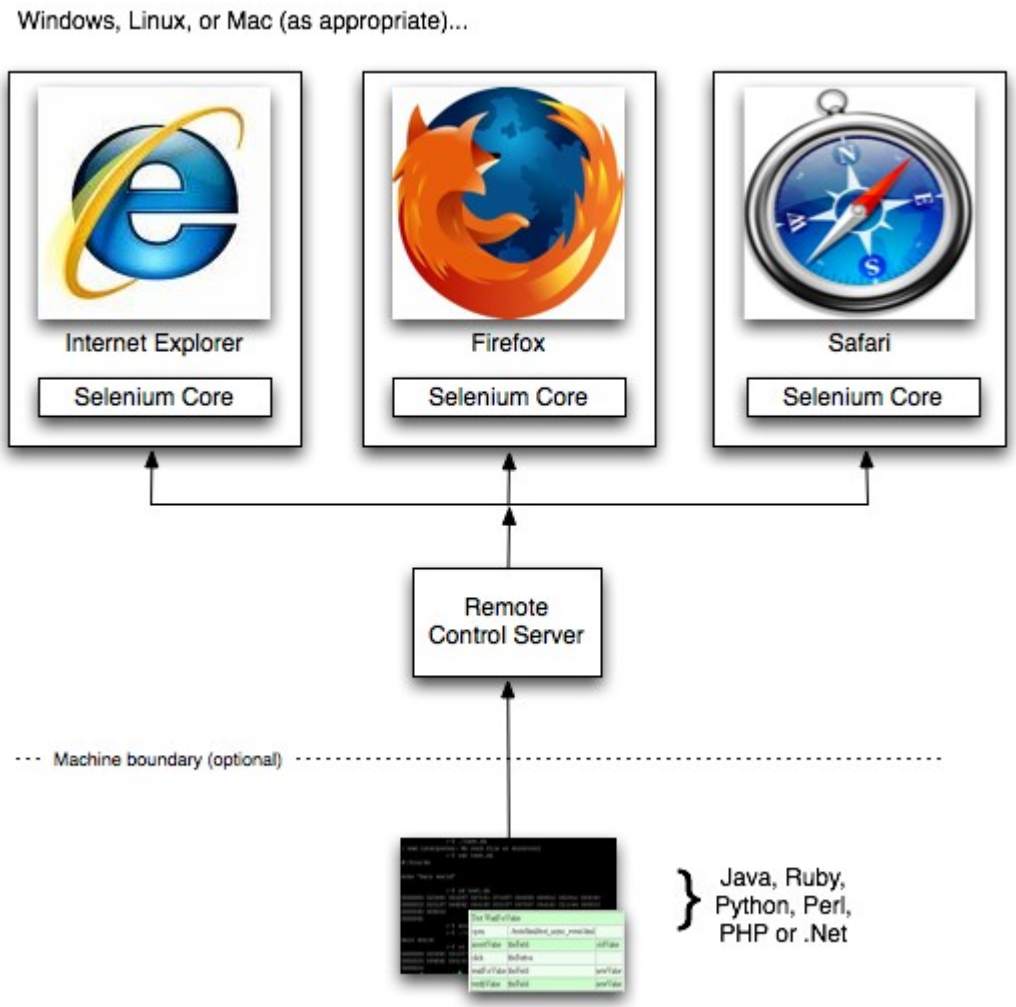
Test de modelos

```
def test_poll(self):
    """
    Test that Poll is correctly created and saved in DB
    """
    ca = Ca.objects.create(id=1312, name="Andalucia")
        census = Census.objects.create(id=1222, title="Jose",
postalCode=11510, ca=ca)
        poll = Poll.objects.create(id=1319, title="Prueba",
        description="Votación de prueba", startDate="2017-01-13",
        endDate="2018-01-10",census=census, participantes=0,
        votos=0)
self.assertEqual(poll.id, 1319)
```

Test de modelos

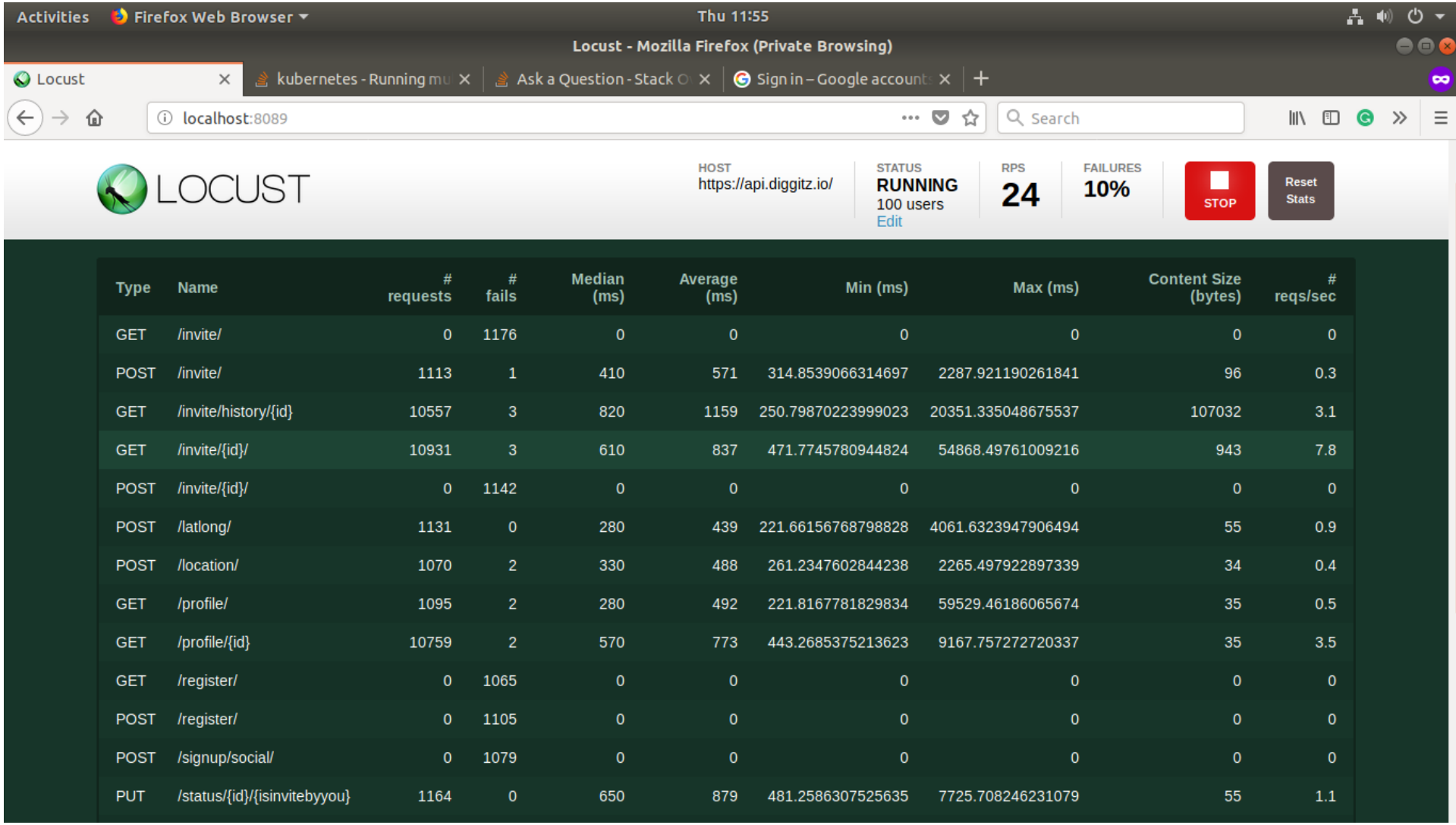
```
def setUp(self):
    super().setUp()
    self.census = Census(voting_id=1, voter_id=1)
    self.census.save()

def tearDown(self):
    super().tearDown()
    self.census = None

def test_store_census(self):
    self.assertEqual(Census.objects.count(), 1)
```

Probando las vistas



Probando la carga del sistema

Source: Experian Marketing Services' Hitwise



Type	Name	# requests	# fails	Median (ms)	Average (ms)	Min (ms)	Max (ms)	Content Size (bytes)	# reqs/sec
GET	/invite/	0	1176	0	0	0	0	0	0
POST	/invite/	1113	1	410	571	314.8539066314697	2287.921190261841	96	0.3
GET	/invite/history/{id}	10557	3	820	1159	250.79870223999023	20351.335048675537	107032	3.1
GET	/invite/{id}/	10931	3	610	837	471.7745780944824	54868.49761009216	943	7.8
POST	/invite/{id}/	0	1142	0	0	0	0	0	0
POST	/latlong/	1131	0	280	439	221.66156768798828	4061.6323947906494	55	0.9
POST	/location/	1070	2	330	488	261.2347602844238	2265.497922897339	34	0.4
GET	/profile/	1095	2	280	492	221.8167781829834	59529.46186065674	35	0.5
GET	/profile/{id}	10759	2	570	773	443.2685375213623	9167.757272720337	35	3.5
GET	/register/	0	1065	0	0	0	0	0	0
POST	/register/	0	1105	0	0	0	0	0	0
POST	/signup/social/	0	1079	0	0	0	0	0	0
PUT	/status/{id}/{isinvitebyyou}	1164	0	650	879	481.2586307525635	7725.708246231079	55	1.1