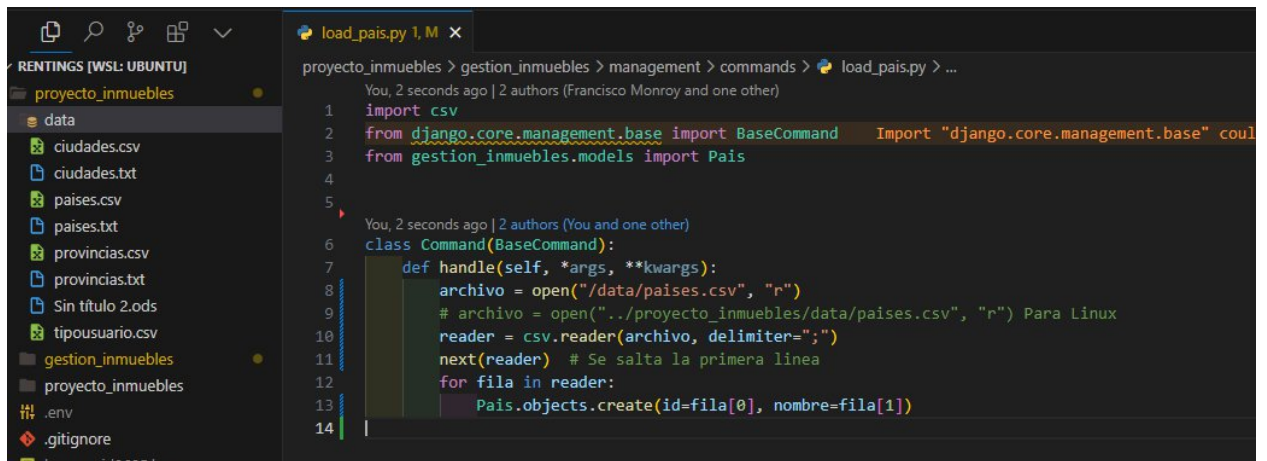


Hito 3

Cargar paises

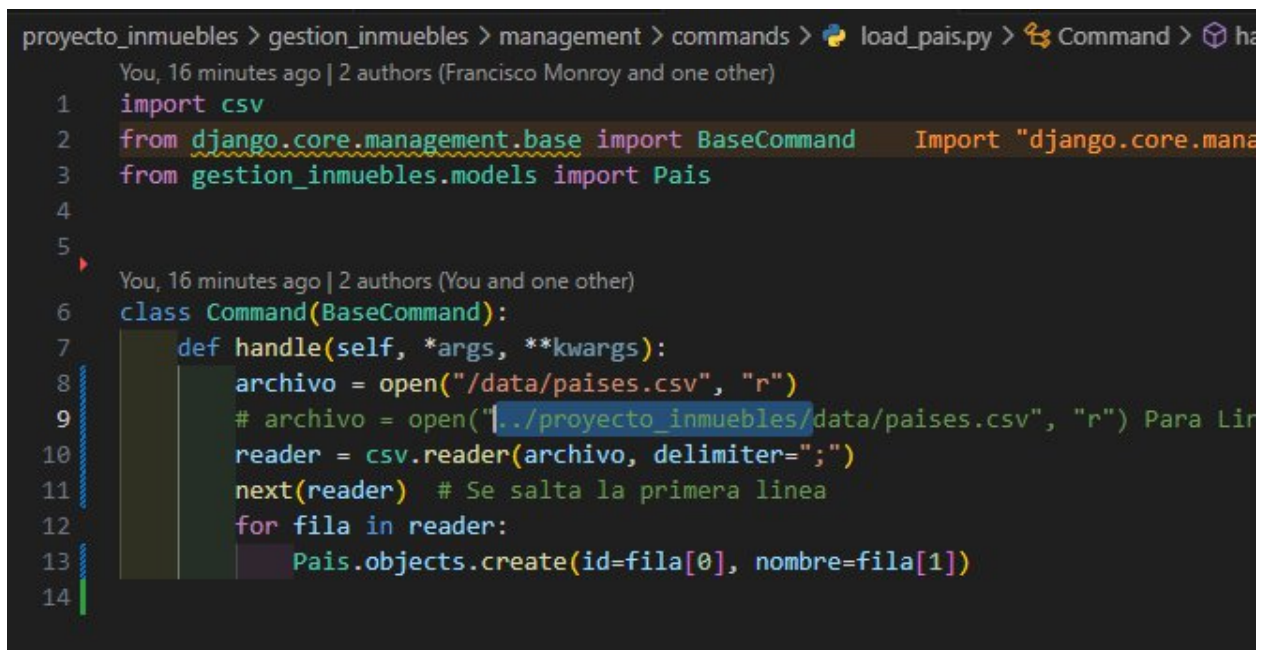


The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'proyecto_inmuebles' with a 'data' folder containing several CSV and TXT files. The code editor shows the 'load_pais.py' file with the following code:

```
1 import csv
2 from django.core.management.base import BaseCommand
3 from gestion_inmuebles.models import Pais
4
5
6 class Command(BaseCommand):
7     def handle(self, *args, **kwargs):
8         archivo = open("/data/paises.csv", "r")
9         # archivo = open("../proyecto_inmuebles/data/paises.csv", "r") Para Linux
10        reader = csv.reader(archivo, delimiter=";")
11        next(reader) # Se salta la primera linea
12        for fila in reader:
13            Pais.objects.create(id=fila[0], nombre=fila[1])
14
```

```
> python manage.py load_pais
> dbeaver
```

Cargar provincias

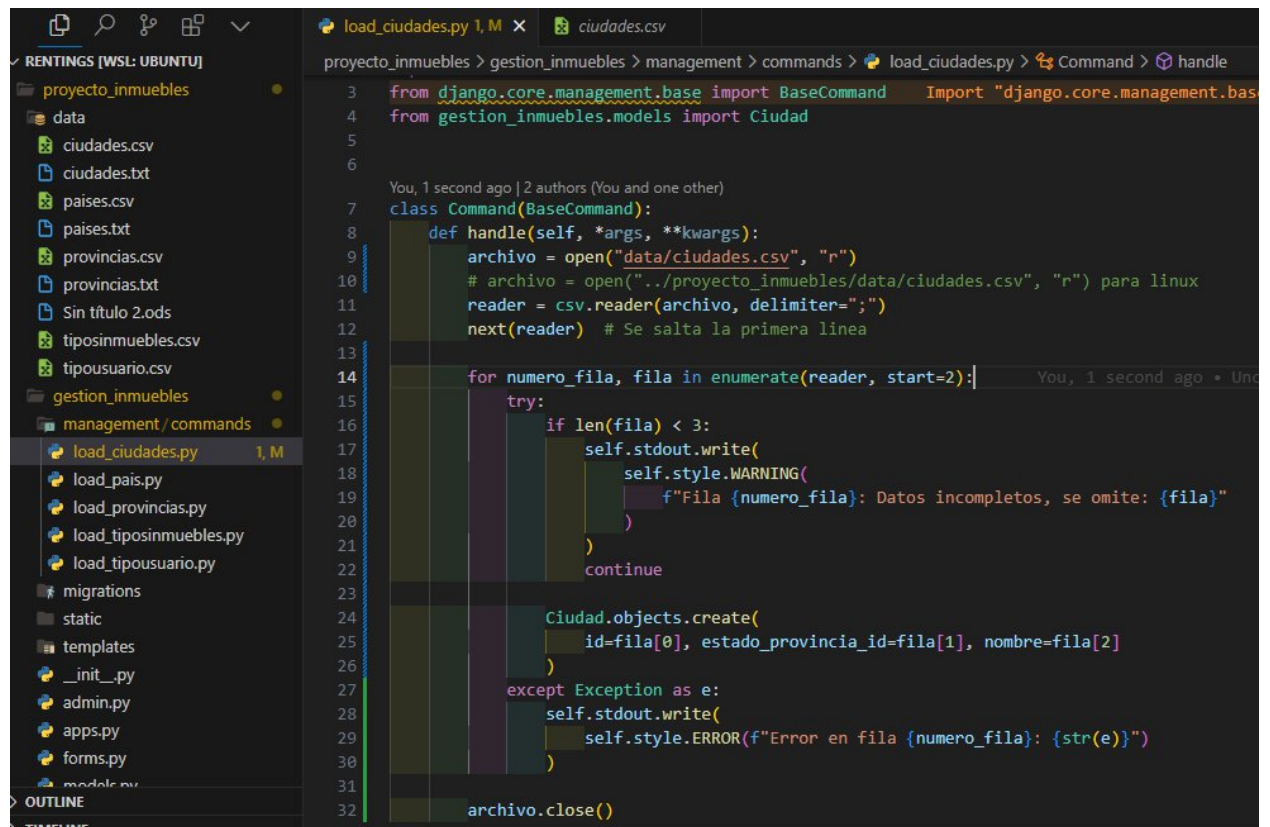


The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'proyecto_inmuebles' with a 'data' folder containing several CSV and TXT files. The code editor shows the 'load_provincias.py' file with the following code:

```
1 import csv
2 from django.core.management.base import BaseCommand
3 from gestion_inmuebles.models import Pais
4
5
6 class Command(BaseCommand):
7     def handle(self, *args, **kwargs):
8         archivo = open("/data/paises.csv", "r")
9         # archivo = open("../proyecto_inmuebles/data/paises.csv", "r") Para Lin
10        reader = csv.reader(archivo, delimiter=";")
11        next(reader) # Se salta la primera linea
12        for fila in reader:
13            Pais.objects.create(id=fila[0], nombre=fila[1])
14
```

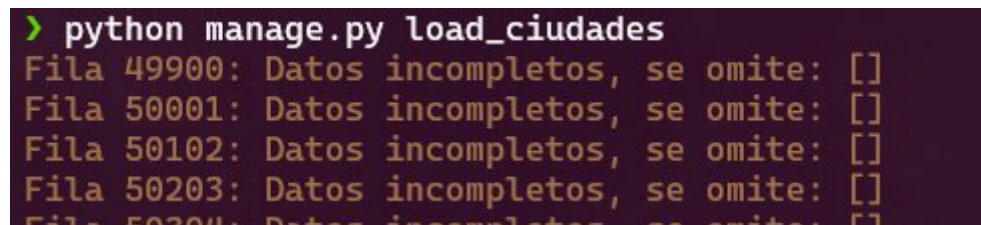
```
> python manage.py load_provincias
```

Cargar ciudades



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'proyecto_inmuebles' with a 'data' folder containing 'ciudades.csv', 'ciudades.txt', 'países.csv', 'países.txt', 'provincias.csv', 'provincias.txt', 'Sin título 2.ods', 'tiposinmuebles.csv', and 'tipousuario.csv'. The 'gestion_inmuebles' folder contains 'management' and 'commands' subfolders. The 'load_ciudades.py' file is selected in the 'commands' folder. The code editor shows the implementation of the 'load_ciudades.py' file, which is a Django management command. The code imports 'BaseCommand' from 'django.core.management.base' and 'Ciudad' from 'gestion_inmuebles.models'. It defines a 'handle' method that opens the 'data/ciudades.csv' file, reads it line by line, and creates 'Ciudad' objects for each line. It includes error handling for incomplete data and exceptions.

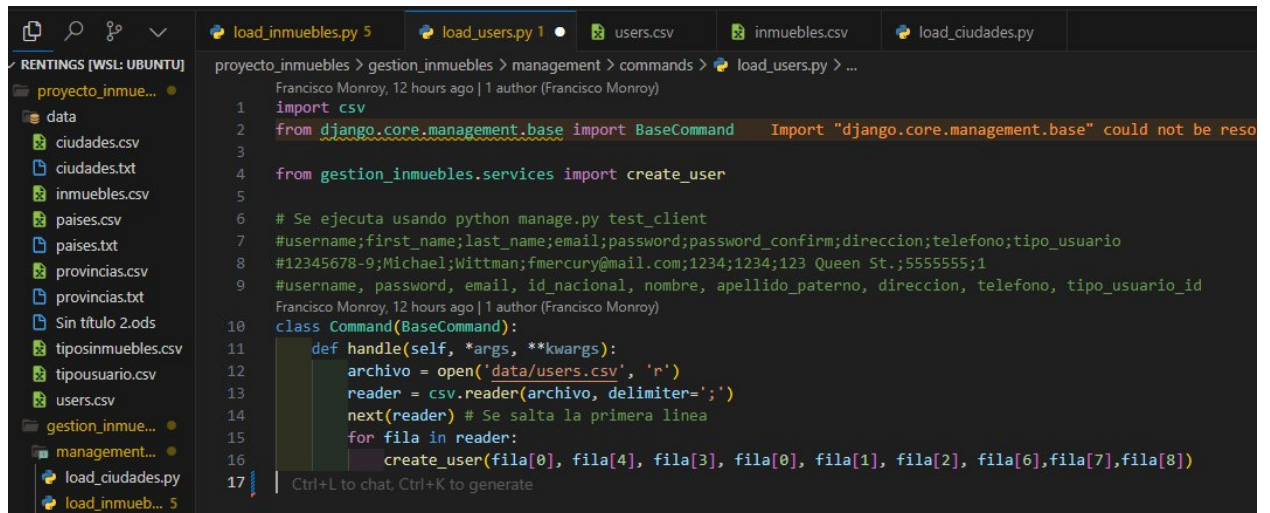
```
3 from django.core.management.base import BaseCommand
4 from gestion_inmuebles.models import Ciudad
5
6
7 class Command(BaseCommand):
8     def handle(self, *args, **kwargs):
9         archivo = open("data/ciudades.csv", "r")
10        # archivo = open("../proyecto_inmuebles/data/ciudades.csv", "r") para linux
11        reader = csv.reader(archivo, delimiter=";")
12        next(reader) # Se salta la primera linea
13
14        for numero_fila, fila in enumerate(reader, start=2):
15            try:
16                if len(fila) < 3:
17                    self.stdout.write(
18                        self.style.WARNING(
19                            f"Fila {numero_fila}: Datos incompletos, se omite: {fila}"
20                        )
21                    )
22                    continue
23
24                    Ciudad.objects.create(
25                        id=fila[0], estado_provincia_id=fila[1], nombre=fila[2]
26                    )
27            except Exception as e:
28                self.stdout.write(
29                    self.style.ERROR(f"Error en fila {numero_fila}: {str(e)}")
30                )
31
32        archivo.close()
```



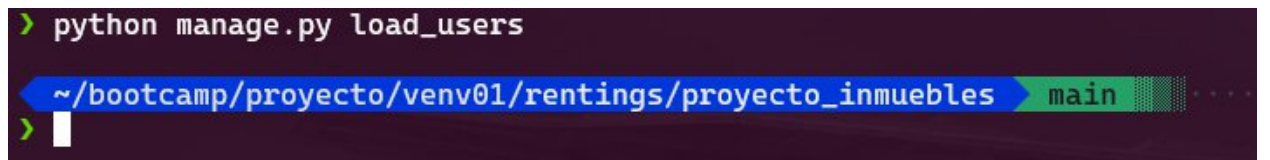
The screenshot shows a terminal window with the command 'python manage.py load_ciudades' executed. The output shows several lines of warnings indicating that data is incomplete for certain rows and is being omitted.

```
> python manage.py load_ciudades
Fila 49900: Datos incompletos, se omite: []
Fila 50001: Datos incompletos, se omite: []
Fila 50102: Datos incompletos, se omite: []
Fila 50203: Datos incompletos, se omite: []
Fila 50304: Datos incompletos, se omite: []
```

Carga de Usuarios

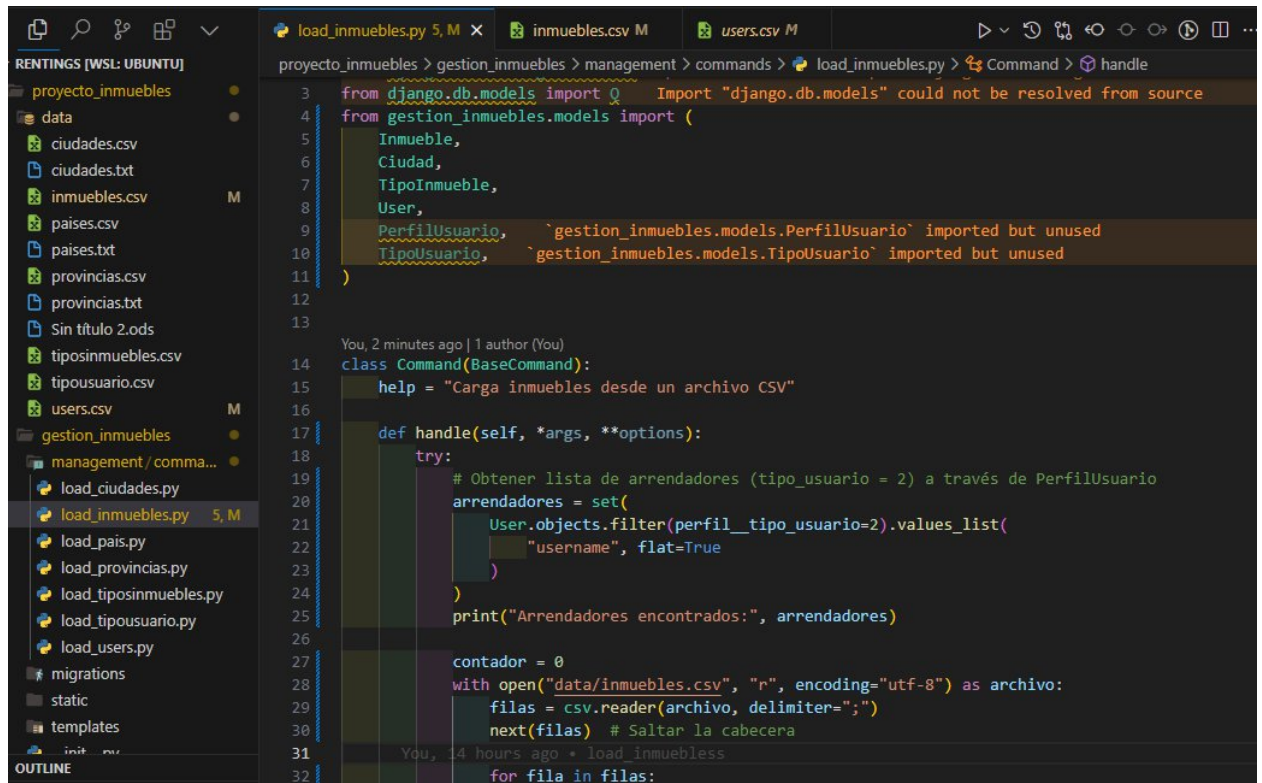


```
projecto_inmuebles > gestion_inmuebles > management > commands > load_users.py > ...
Francisco Monroy, 12 hours ago | 1 author (Francisco Monroy)
1 import csv
2 from django.core.management.base import BaseCommand
3
4 from gestion_inmuebles.services import create_user
5
6 # Se ejecuta usando python manage.py test_client
7 #username;first_name;last_name;email;password;password_confirm;direccion;telefono;tipo_usuario
8 #12345678-9;Michael;Wittman;fmercury@mail.com;1234;1234;123 Queen St.;5555555;1
9 #username, password, email, id_nacional, nombre, apellido_paterno, direccion, telefono, tipo_usuario_id
10 class Command(BaseCommand):
11     def handle(self, *args, **kwargs):
12         archivo = open('data/users.csv', 'r')
13         reader = csv.reader(archivo, delimiter=';')
14         next(reader) # Se salta la primera linea
15         for fila in reader:
16             create_user(fila[0], fila[4], fila[3], fila[0], fila[1], fila[2], fila[6], fila[7], fila[8])
17
```



```
> python manage.py load_users
~/bootcamp/proyecto/venv01/rentings/proyecto_inmuebles main
>
```

Carga de tipos de inmuebles

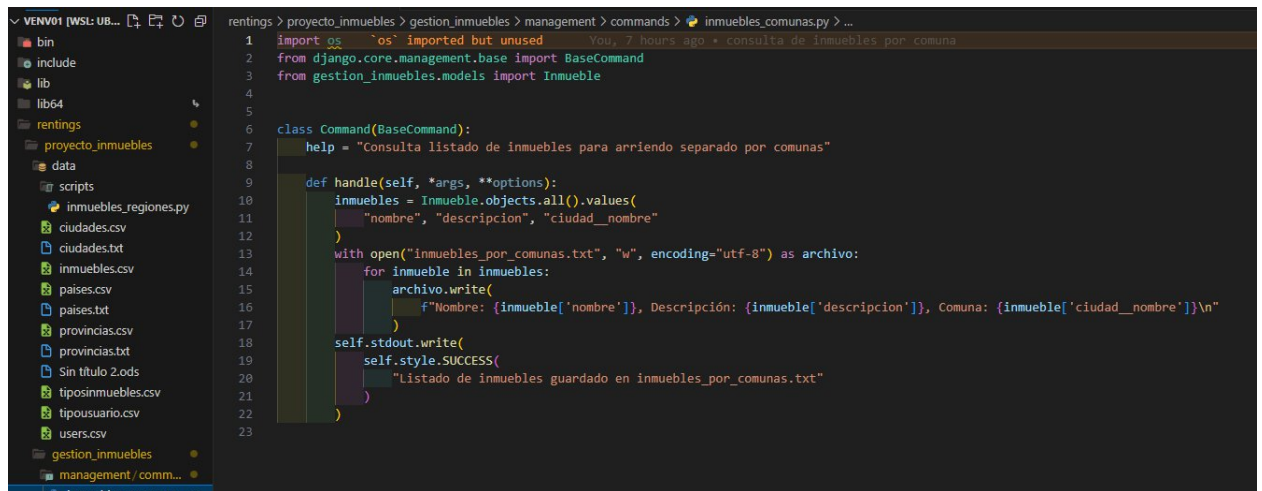


```
projecto_inmuebles > gestion_inmuebles > management > commands > load_inmuebles.py > Command > handle
3 from django.db.models import Q
4 from gestion_inmuebles.models import (
5     Inmueble,
6     Ciudad,
7     TipoInmueble,
8     User,
9     PerfilUsuario, `gestion_inmuebles.models.PerfilUsuario` imported but unused
10    TipoUsuario, `gestion_inmuebles.models.TipoUsuario` imported but unused
11 )
12
13
14 You, 2 minutes ago | 1 author (You)
15 class Command(BaseCommand):
16     help = "Carga inmuebles desde un archivo CSV"
17
18     def handle(self, *args, **options):
19         try:
20             # Obtener lista de arrendadores (tipo_usuario = 2) a través de PerfilUsuario
21             arrendadores = set(
22                 User.objects.filter(perfil_tipo_usuario=2).values_list(
23                     "username", flat=True
24                 )
25             )
26             print("Arrendadores encontrados:", arrendadores)
27
28             contador = 0
29             with open("data/inmuebles.csv", "r", encoding="utf-8") as archivo:
30                 filas = csv.reader(archivo, delimiter=";")
31                 next(filas) # Saltar la cabecera
32                 for fila in filas:
```

```
> python manage.py load_inmuebles
Arrendadores encontrados: {'67890123-4', '56789012-3', '45678901-2', '23456789-0', '90123456-7', '78901234-5', '11234787-8', '89012345-6', '34567890-1', '12345678-9'}
Se cargaron 10 inmuebles exitosamente

~/bootcamp/proyecto/venv01/rentings/proyecto_inmuebles main !3 venv01 Py 11:18:39
```


Consulta 1



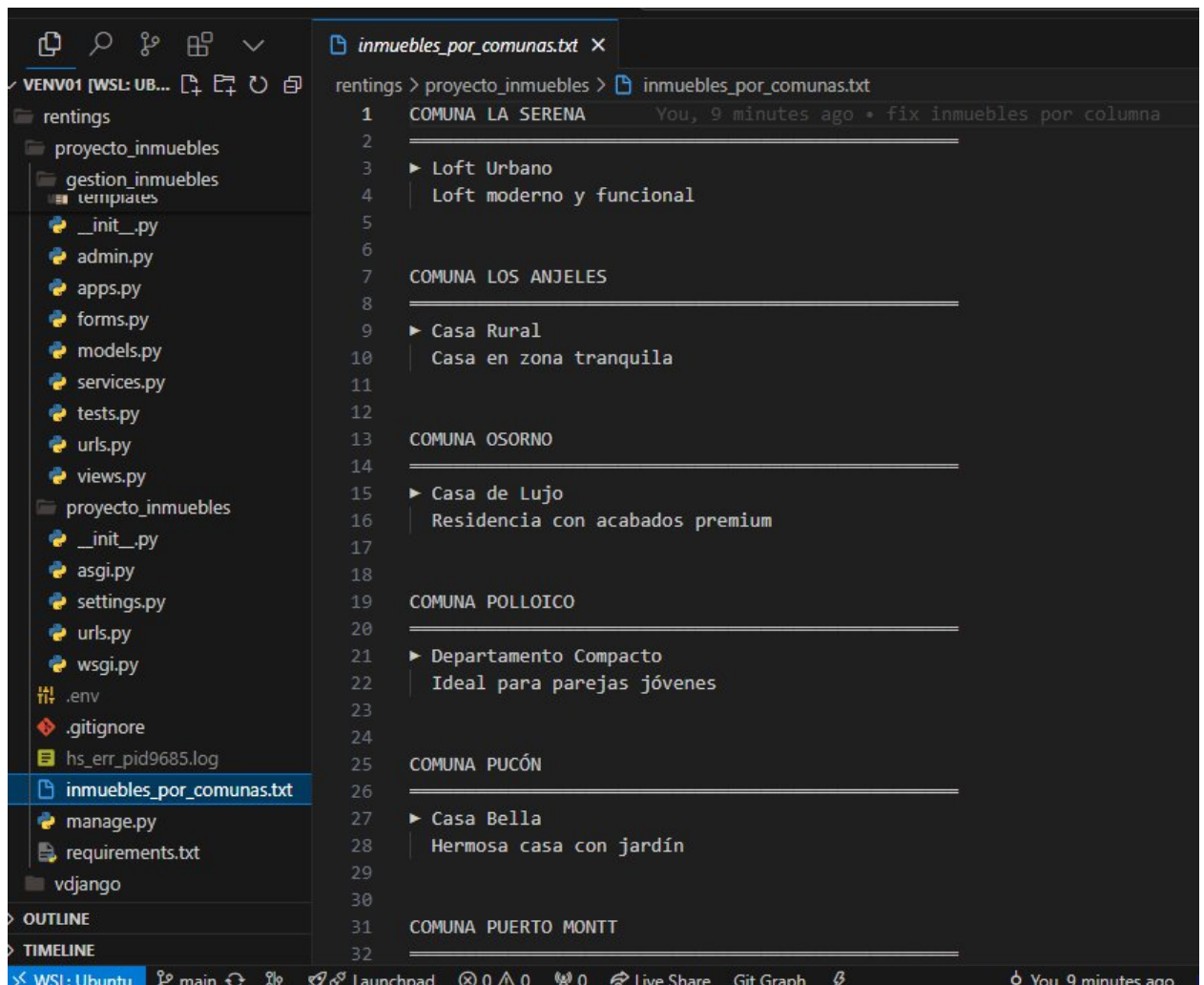
The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like 'bin', 'include', 'lib', 'lib64', 'rentings', and 'proyecto_inmuebles'. The code editor shows the file 'inmuebles_comunas.py' with the following code:

```
1 import os  # os` imported but unused You, 7 hours ago • consulta de inmuebles por comuna
2 from django.core.management.base import BaseCommand
3 from gestion_inmuebles.models import Inmueble
4
5
6 class Command(BaseCommand):
7     help = "Consulta listado de inmuebles para arriendo separado por comunas"
8
9     def handle(self, *args, **options):
10         inmuebles = Inmueble.objects.all().values(
11             "nombre", "descripcion", "ciudad_nombre"
12         )
13         with open("inmuebles_por_comunas.txt", "w", encoding="utf-8") as archivo:
14             for inmueble in inmuebles:
15                 archivo.write(
16                     f'Nombre: {inmueble["nombre"]}, Descripción: {inmueble["descripcion"]}, Comuna: {inmueble["ciudad_nombre"]}\n'
17                 )
18             self.stdout.write(
19                 self.style.SUCCESS(
20                     "Listado de inmuebles guardado en inmuebles_por_comunas.txt"
21                 )
22             )
23
```

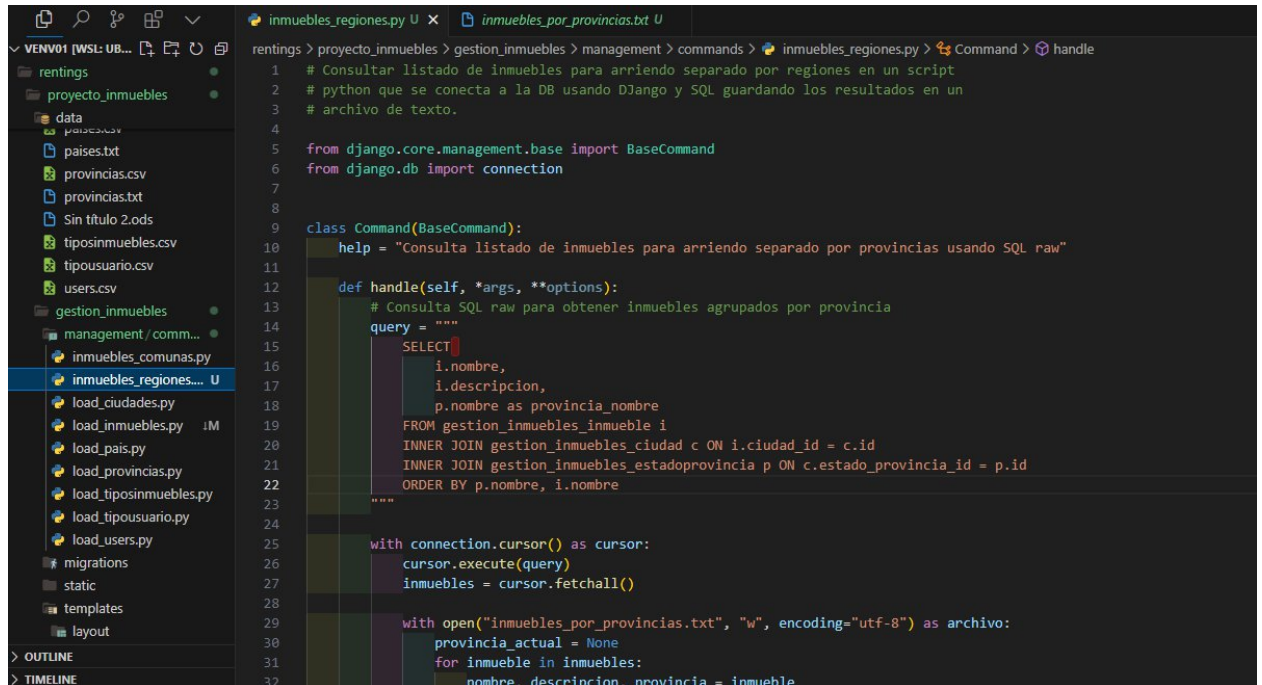
```
> python manage.py inmuebles_comunas
Listado de inmuebles guardado en inmuebles_por_comunas.txt
```

```
~/bootcamp/proyecto/venv01/rentings/proyecto_inmuebles main !2
```

```
>
```

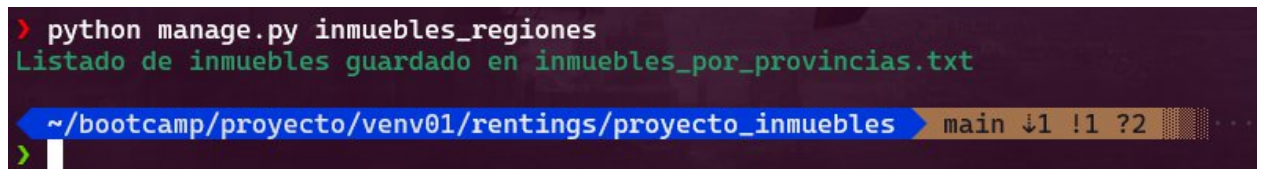


Consulta 2



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like 'rentings', 'proyecto_inmuebles', 'data', 'gestion_inmuebles', and 'management'. The code editor shows a Python script named 'inmuebles_regiones.py' with the following content:

```
1 # Consultar listado de inmuebles para arriendo separado por regiones en un script
2 # python que se conecta a la DB usando Django y SQL guardando los resultados en un
3 # archivo de texto.
4
5 from django.core.management.base import BaseCommand
6 from django.db import connection
7
8
9 class Command(BaseCommand):
10     help = "Consulta listado de inmuebles para arriendo separado por provincias usando SQL raw"
11
12     def handle(self, *args, **options):
13         # Consulta SQL raw para obtener inmuebles agrupados por provincia
14         query = """
15             SELECT
16                 i.nombre,
17                 i.descripcion,
18                 p.nombre as provincia_nombre
19             FROM gestion_inmuebles_inmueble i
20             INNER JOIN gestion_inmuebles_ciudad c ON i.ciudad_id = c.id
21             INNER JOIN gestion_inmuebles_estadoprovincia p ON c.estado_provincia_id = p.id
22             ORDER BY p.nombre, i.nombre
23         """
24
25         with connection.cursor() as cursor:
26             cursor.execute(query)
27             inmuebles = cursor.fetchall()
28
29         with open("inmuebles_por_provincias.txt", "w", encoding="utf-8") as archivo:
30             provincia_actual = None
31             for inmueble in inmuebles:
32                 nombre, descripcion, provincia = inmueble
```



The screenshot shows a terminal window with the following content:

```
> python manage.py inmuebles_regiones
Listado de inmuebles guardado en inmuebles_por_provincias.txt

~/bootcamp/proyecto/venv01/rentings/proyecto_inmuebles main ↓1 !1 ?2
>
```

VENVO1 [WSL: UB...]

rentings

projecto_inmuebles

gestion_inmuebles

__init__.py

admin.py

apps.py

forms.py

models.py

services.py

tests.py

urls.py

views.py

projecto_inmuebles

__init__.py

asgi.py

settings.py

urls.py

wsgi.py

.env

.gitignore

hs_err_pid9685.log

inmuebles_por_comunas.txt

inmuebles_por_provinc... U

manage.py

requirements.txt

vdjango

OUTLINE

TIMELINE

inmuebles_regiones.py U

inmuebles_por_provincias.txt U x

rentings > proyecto_inmuebles > inmuebles_por_provincias.txt

1 PROVINCIA ARAUCANÍA

2

3 ▶ Casa Bella

4 | Hermosa casa con jardín

5

6 ▶ Casa Bella

7 | Hermosa casa con jardín

8

9 ▶ Casa Familiar

10 | Casa amplia y luminosa

11

12 ▶ Casa Familiar

13 | Casa amplia y luminosa

14

15 ▶ Departamento Moderno

16 | Departamento céntrico con terraza

17

18 ▶ Departamento Moderno

19 | Departamento céntrico con terraza

20

21

22 PROVINCIA BÍO-BÍO

23

24 ▶ Casa Rural

25 | Casa en zona tranquila

26

27 ▶ Casa Rural

28 | Casa en zona tranquila

29

30

31 PROVINCIA COQUIMBO

32