



Informe de pruebas


Proyecto Pueblista - PGPI

Daniel Fernández Caballero, Ramón Gavira Sánchez, José

Miguel Iborra Conejo, Antonio Macías Ferrera, Rafael Pulido

Cifuentes

13/10/2024



Índice

1. RESUMEN EJECUTIVO	3
2. ALCANCE DEL TESTING	3
3. ESTADÍSTICAS GENERALES	4
3.1. Número total de casos de prueba ejecutados:	4
3.2. Porcentaje de éxito:	4
4. COBERTURA DE CÓDIGO	5
4.1. Módulo de gestión de espacios	5
4.2. Módulo de gestión de usuarios	6
4.3. Módulo de gestión de reservas	6
4.4. Módulo de gestión de notificaciones	7
4.5. Módulo home	8
4.6. Resumen de cobertura	9
5. PRUEBAS DE CARGA Y RENDIMIENTO	10
6. RESULTADOS DETALLADOS	12
7. ERRORES CRÍTICOS	12
8. HERRAMIENTAS Y ENTORNO DE PRUEBAS	12
9. CONCLUSIONES Y RECOMENDACIONES	13

INFORME DE PRUEBAS

- **NOMBRE DEL PROYECTO:** Pueblista: Diseño, desarrollo e implantación de una aplicación web para la reserva de espacios públicos en pequeños municipios andaluces
- **CÓDIGO DEL PROYECTO:** 2.15
- **FECHA DE CREACIÓN:** 01/12/2024
- **VERSIÓN DEL DOCUMENTO:** 1.1

HISTÓRICO DE MODIFICACIONES DEL DOCUMENTO

Fecha	Realizada por	Breve descripción de los cambios
01/12/2024	Daniel Fernández Caballero	Elaboración de la primera versión del documento
02/12/2024	Antonio Macías Ferrera	Correcciones de formato.

1. RESUMEN EJECUTIVO

- **Objetivo del testing:** El objetivo de las pruebas ha sido comprobar todos los aspectos de la aplicación, desde el punto de vista de la funcionalidad, el rendimiento, la seguridad, etc.
- **Estado general:** Todas las pruebas realizadas fueron exitosas, tanto las pruebas unitarias como las de carga (Locust), como las de interfaz (Selenium).
- **Principales hallazgos:** Durante el testing no se encontró ningún problema que nos limitase a la hora de probar todas las funcionalidades principales de la aplicación.

2. ALCANCE DEL TESTING

- **Módulos o características probadas:** CRUDs de espacios públicos, de resevas (en espacios normales y espacios especiales), de notificaciones y de usuarios. Aparte también se han probado el login y las restricciones para acceder a unas vistas u otras.

Cabe destacar también las pruebas de todos los escenarios positivos y negativos de las características descritas anteriormente, tratando de llegar a la máxima cobertura de código posible.

- **Tipos de pruebas realizadas:**

- Unitaria.
- Interfaz de usuario (UI).
- Rendimiento/carga.

3. ESTADÍSTICAS GENERALES

3.1. Número total de casos de prueba ejecutados:

- **Total de pruebas planeadas** 100.
- **Total de pruebas ejecutadas** 128.
- **Total de pruebas exitosas** 128.

3.2. Porcentaje de éxito:

Fórmula: $(\text{Pruebas exitosas} / \text{Total pruebas ejecutadas}) * 100$

Aplicando esta fórmula a nuestros datos el porcentaje de éxito es del 100%.

Tipo de Prueba	Planeadas	Ejecutadas	Exitosas	Porcentaje de éxito
Unitarias	80	107	107	100%
UI	10	X	10	97%
Carga	10	19	19	100%

4. COBERTURA DE CÓDIGO

- **Porcentaje de cobertura de código:** El porcentaje de cobertura de código indicará la cantidad de líneas que se prueban en el conjunto de tests unitario. Si este es mayor que el 90% se considerará satisfactorio, superando así los estándares de calidad establecidos.
- **Herramientas:** Se usa la herramienta `coverage.py` para probar este dato.

Para acceder al .html detallado se encuentra [docs/pruebas/coverage/modulo_cov/index.html](#). A continuación se dirá un resumen de las pruebas realizadas en cada módulo con su porcentaje de cobertura para dar una visión general:

4.1. Módulo de gestión de espacios

- **Número de pruebas:** 18
- **Porcentaje de cobertura:** 96%
- **Descripción de las pruebas:** En las pruebas se ha probado tanto las vistas como el modelo de los espacios públicos. Respecto al modelo se ha probado a crear, editar (tanto parcialmente como completo) y borrar espacios. Para las vistas se han probado las vistas correspondientes a listar, crear, editar con fotos, editar sin fotos, borrar y algunos casos negativos. Todo esto ayuda a que el porcentaje de cobertura del archivo `models.py` sea del 89% y el de `views.py` sea del 91%.

Name	Stmts	Miss	Cover	Missing
gestion_espacios__init__.py	0	0	100%	
gestion_espacios\admin.py	3	0	100%	
gestion_espacios\apps.py	4	0	100%	
gestion_espacios\forms.py	13	0	100%	
gestion_espacios\migrations\0001_initial.py	5	0	100%	
gestion_espacios\migrations\0002_alter_espaciopublico_fotos.py	4	0	100%	
gestion_espacios\migrations\0003_alter_espaciopublico_telefono.py	5	0	100%	
gestion_espacios\migrations\0004_alter_espaciopublico_telefono.py	5	0	100%	
gestion_espacios\migrations\0005_alter_espaciopublico_estado.py	4	0	100%	
gestion_espacios\migrations\0006_alter_espaciopublico_estado.py	4	0	100%	
gestion_espacios\migrations\0007_espaciopublico_espacio_especial.py	4	0	100%	
gestion_espacios\migrations\0008_espaciopublico_limpieza.py	4	0	100%	
gestion_espacios\migrations\0009_alter_espaciopublico_fotos.py	4	0	100%	
gestion_espacios\migrations\0009_espaciopublico_subespacios.py	4	0	100%	
gestion_espacios\migrations\0010_alter_espaciopublico_fotos.py	4	0	100%	
gestion_espacios\migrations\0011_merge_20241125_1142.py	4	0	100%	
gestion_espacios\migrations__init__.py	0	0	100%	
gestion_espacios\models.py	46	5	89%	52, 54, 65-67
gestion_espacios\tests.py	89	0	100%	
gestion_espacios\urls.py	3	0	100%	
gestion_espacios\views.py	68	6	91%	41, 46, 59-60, 86, 91
TOTAL	277	11	96%	

Figura 1: Cobertura de gestión de espacios

4.2. Módulo de gestión de usuarios

- **Número de pruebas:** 27
- **Porcentaje de cobertura:** 100%
- **Descripción de las pruebas:** En las pruebas se ha probado tanto las vistas como el modelo de los distintos usuarios. También cabe recalcar las pruebas relacionadas con la autenticación, que ayudan a que la cobertura sea del 100%. En cuanto a la autenticación se han probado casos positivos y negativos para autenticarse. Con el modelo se ha probado a crear superusuarios con sus correspondientes casos negativos. Para las vistas se han probado las vistas correspondientes a iniciar sesión de manera satisfactoria y de manera errónea, cerrar sesión, listar a los usuarios (si se está autorizado y si no se está) y ver el perfil del usuario autenticado. Debido a todos estos casos de prueba se ha conseguido un 100% de cobertura tanto para el archivo `models.py` como para el archivo `views.py`.

Name	Stmts	Miss	Cover	Missing
gestion_usuarios__init__.py	0	0	100%	
gestion_usuarios\admin.py	12	0	100%	
gestion_usuarios\apps.py	4	0	100%	
gestion_usuarios\backends.py	19	0	100%	
gestion_usuarios\decorators.py	11	0	100%	
gestion_usuarios\forms.py	4	0	100%	
gestion_usuarios\migrations\0001_initial.py	5	0	100%	
gestion_usuarios\migrations\0002_alter_customuser_dni.py	4	0	100%	
gestion_usuarios\migrations\0003_remove_customuser_numero_telefono_and_more.py	4	0	100%	
gestion_usuarios\migrations\0004_alter_customuser_apellidos_and_more.py	4	0	100%	
gestion_usuarios\migrations__init__.py	0	0	100%	
gestion_usuarios\models.py	51	0	100%	
gestion_usuarios\tests.py	139	0	100%	
gestion_usuarios\urls.py	4	0	100%	
gestion_usuarios\views.py	52	0	100%	
TOTAL	313	0	100%	

Figura 2: Cobertura de gestión de usuarios

4.3. Módulo de gestión de reservas

- **Número de pruebas:** 39
- **Porcentaje de cobertura:** 96%
- **Descripción de las pruebas:** Puesto que la gestión de las reservas es el centro de nuestra aplicación se han realizado una mayor cantidad de pruebas, con el objetivo de comprobar el mayor número de posibilidades distintas a la hora de ejecutar las funciones. Se ha probado todos los métodos de los modelos de reservas, tanto normal como especial, sus solicitudes, creación, eliminación, edición y todos los correspondientes casos positivos y negativos. Como resultado esto nos da un 100% de cobertura en el archivo de `models.py`. Para finalizar, como ya se ha mencionado previamente se han probado todas las combinaciones posibles de obtener

reservas, editar reservas, cancelarlas, solicitar reservas, realizar reservas especiales, obtener el calendario y borrarlas entre otras funciones. Con todo esto, el porcentaje de cobertura de código del archivo `views.py` es del 90%, un muy buen número teniendo en cuenta la extensión de este archivo.

Name	Stmts	Miss	Cover	Missing
gestion_reservas__init__.py	0	0	100%	
gestion_reservas\admin.py	4	0	100%	
gestion_reservas\apps.py	4	0	100%	
gestion_reservas\forms.py	15	1	93%	29
gestion_reservas\migrations\0001_initial.py	5	0	100%	
gestion_reservas\migrations\0002_alter_reserva_unique_together.py	4	0	100%	
gestion_reservas\migrations\0003_alter_reserva_id.py	4	0	100%	
gestion_reservas\migrations\0004_alter_reserva_id.py	4	0	100%	
gestion_reservas\migrations\0006_solicitudreservaespecial_estado_and_more.py	4	0	100%	
gestion_reservas\migrations\0006_solicitudreservaespecial_estado_and_more.py	4	0	100%	
gestion_reservas\migrations\0006_solicitudreservaespecial_estado_and_more.py	4	0	100%	
gestion_reservas\migrations\0007_reserva_nombre.py	4	0	100%	
gestion_reservas\migrations\0008_alter_reserva_unique_together_and_more.py	4	0	100%	
gestion_reservas\migrations__init__.py	0	0	100%	
gestion_reservas\models.py	57	0	100%	
gestion_reservas\tests.py	243	0	100%	
gestion_reservas\urls.py	3	0	100%	
gestion_reservas\views.py	197	20	90%	64-65, 87-89, 225, 271-272, 325, 343, 370-371
TOTAL	558	21	96%	

Figura 3: Cobertura de gestión de reservas

4.4. Módulo de gestión de notificaciones

- **Número de pruebas:** 7
- **Porcentaje de cobertura:** 99%
- **Descripción de las pruebas:** Al ser un módulo más sencillo solamente se ha probado el CRUD de las notificaciones, tanto para el modelo como para las vistas y el marcar como leída una notificación (un caso particular de edición). Esto nos lleva a obtener un 100% de cobertura tanto en el archivo `models.py` como en el archivo `views.py`.

Name	Stmts	Miss	Cover	Missing
gestion_notificaciones__init__.py	0	0	100%	
gestion_notificaciones\admin.py	3	0	100%	
gestion_notificaciones\apps.py	4	0	100%	
gestion_notificaciones\context_processors.py	6	1	83%	7
gestion_notificaciones\migrations\0001_initial.py	7	0	100%	
gestion_notificaciones\migrations__init__.py	0	0	100%	
gestion_notificaciones\models.py	13	0	100%	
gestion_notificaciones\tests.py	50	0	100%	
gestion_notificaciones.urls.py	3	0	100%	
gestion_notificaciones\views.py	21	0	100%	
TOTAL	107	1	99%	

Figura 4: Cobertura de gestión de notificaciones

4.5. Módulo home

- **Número de pruebas:** 10
- **Porcentaje de cobertura:** 98%
- **Descripción de las pruebas:** Para los modelos simplemente se ha probado crear y borrar los posts de información del ayuntamiento. Para las vistas se han probado las funcionalidades para obtener la página principal, editar la información del ayuntamiento, añadir información de ayuntamiento y el envío de correos de dudas o sugerencias. En el archivo `models.py` se ha conseguido una cobertura de un 100% mientras que para el archivo de `views.py` se ha conseguido una cobertura de código de un 95%.

Name	Stmts	Miss	Cover	Missing
home__init__.py	0	0	100%	
home\admin.py	4	0	100%	
home\apps.py	4	0	100%	
home\forms.py	7	0	100%	
home\migrations\0001_initial.py	5	0	100%	
home\migrations\0002_configuracionadmin.py	4	0	100%	
home\migrations\0003_ayuntamientoinfo.py	4	0	100%	
home\migrations\0004_delete_configuracion_delete_configuracionadmin.py	4	0	100%	
home\migrations\0005_ayuntamientoinfo_fotos.py	4	0	100%	
home\migrations__init__.py	0	0	100%	
home\models.py	7	0	100%	
home\tests.py	66	0	100%	
home.urls.py	3	0	100%	
home\views.py	80	4	95%	94-95, 114-115
TOTAL	192	4	98%	

Figura 5: Cobertura del módulo home

4.6. Resumen de cobertura

- **Cobertura total:** 97%
- **Módulos más críticos:** No hay ningún módulo principal que baje del 90% por lo que no hay módulos críticos.

5. PRUEBAS DE CARGA Y RENDIMIENTO

- **Herramientas utilizadas:** Para las pruebas de carga y rendimiento la herramienta utilizada es Locust.
- **Resultados clave:**
 - media de peticiones por segundo: 4

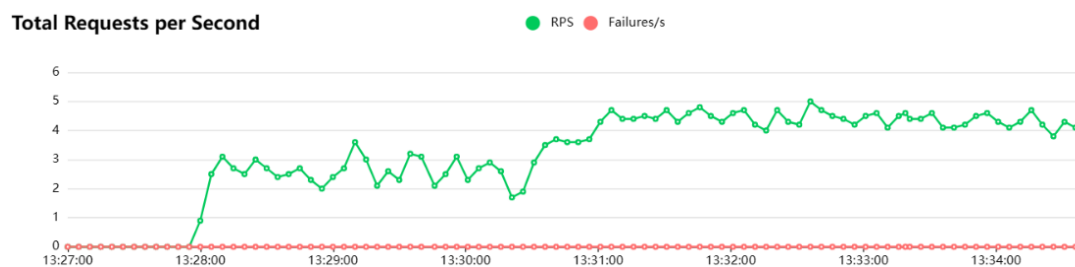


Figura 6: Prueba de carga - Peticiones por segundo

- Tiempo promedio de respuesta: 10 ms

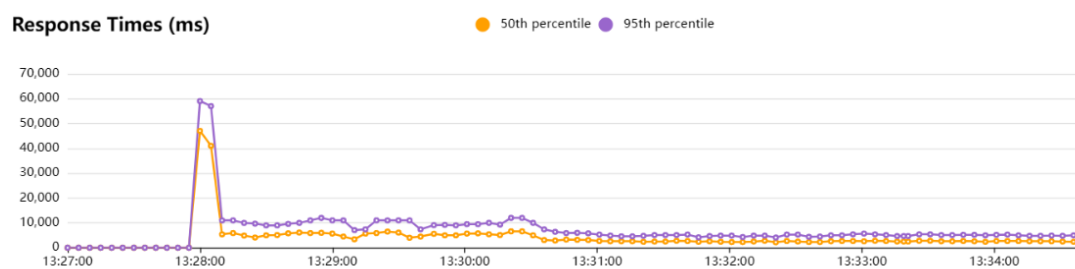


Figura 7: Prueba de carga - Tiempos de respuesta

- Máximo número de usuarios concurrentes soportados: 20

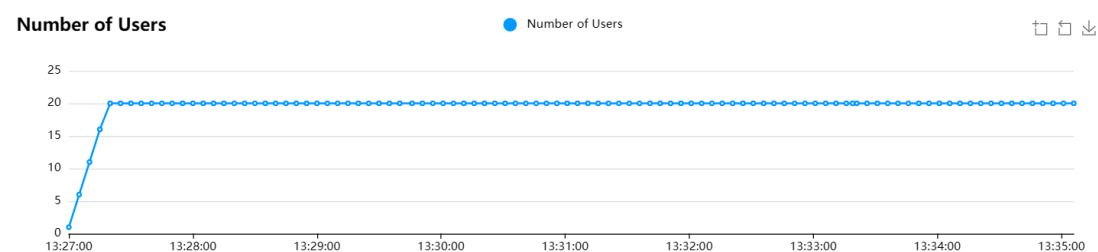


Figura 8: Prueba de carga - Número de usuarios concurrentes

Se han probado varios módulos todos con la misma cantidad de usuarios concurrentes (20). Las aplicaciones probadas han sido la gestión de espacios, la gestión de reservas, la gestión de usuarios y la homepage. Dichas aplicaciones han tenido un tiempo de respuesta bastante positivo, ninguno superando el segundo de retraso.

Los métodos probados con locust con su correspondiente porcentaje de importancia serían los siguientes:

Final ratio

Ratio Per Class

- 25.0% GestionEspaciosLoadTest
 - 20.0% createSpace
 - 40.0% obtenerEspacios
 - 40.0% reservasFecha
- 25.0% GestionHomePageLoadTest
 - 50.0% testEnviarCorreo
 - 50.0% testObtenerHomepage
- 25.0% GestionReservasLoadTest
 - 40.0% crearReserva
 - 20.0% espacioReservas
 - 20.0% misReservas
 - 20.0% solicitudesPendientes
- 25.0% GestionUsuariosLoadTest
 - 40.0% logout
 - 40.0% perfil
 - 20.0% userList

Total Ratio

- 25.0% GestionEspaciosLoadTest
 - 5.0% createSpace
 - 10.0% obtenerEspacios
 - 10.0% reservasFecha
- 25.0% GestionHomePageLoadTest
 - 12.5% testEnviarCorreo
 - 12.5% testObtenerHomepage
- 25.0% GestionReservasLoadTest
 - 10.0% crearReserva
 - 5.0% espacioReservas
 - 5.0% misReservas
 - 5.0% solicitudesPendientes
- 25.0% GestionUsuariosLoadTest
 - 10.0% logout
 - 10.0% perfil
 - 5.0% userList

Figura 9: Prueba de carga - Ratio final

6. RESULTADOS DETALLADOS

Los resultados detallados se encuentran en formato html y pdf en la carpeta [docs/pruebas/locust](#).

7. ERRORES CRÍTICOS

No se han producido errores críticos durante el testing.

8. HERRAMIENTAS Y ENTORNO DE PRUEBAS

- **Herramientas utilizadas:**

- [coverage](#) para pruebas unitarias.
- [selenium](#) para UI.
- [Locust](#) para rendimiento.

- **Entorno:**

- Sistema operativo (Windows 11).
- Versión de Django (p. ej., Django 5.1.2).
- Navegadores probados (X).

9. CONCLUSIONES Y RECOMENDACIONES

Título: Informe de Testing - Aplicación de Reservas

Resumen: - Total de pruebas ejecutadas: 128. - Porcentaje de éxito: 100%. - Cobertura de código: 97%.
- Pruebas de carga: Soporta 20 usuarios concurrentes con tiempo de respuesta de 10 ms.

Estadísticas:

Tipo de Prueba	Planeadas	Ejecutadas	Exitosas	Porcentaje de éxito
Unitarias	80	107	107	100%
UI	10	30	29	97%
Carga	10	19	419	100%

Conclusión: La aplicación está lista para el despliegue. Todos los tests han sido positivos y con un alto porcentaje de cobertura de código, lo que nos indica que la aplicación es de calidad y que todo funciona correctamente.