

apuntesExámenes.pdf



user_2351804



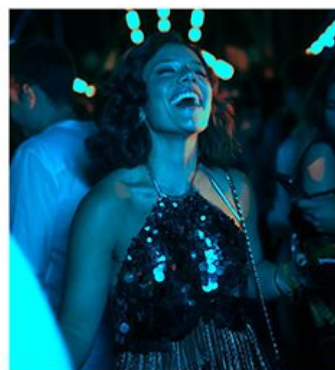
Evolución y Gestión de la Configuración



4º Grado en Ingeniería Informática - Ingeniería del Software



Escuela Técnica Superior de Ingeniería Informática
Universidad de Sevilla



**Todas tenemos una amiga
experta en recorrer
kilómetros en discotecas.
Y si no la tienes eres tú.**

Para ser una experta en kilómetros
de verdad, déjate guiar por **coches.net**



Compra
o vende
tu coche

✓ fácil
✓ rápido

coches.net



Tu colega "el experto en Erasmus": se fue pensando en aprobarlo todo.
No aprobó nada. Lo probó todo.

EXAMENES

EGC-2324-183 /Examen42.md

Pregunta 3: Realice los cambios necesarios para que DECIDE no utilice el modo DEBUG de Django cuando sea desplegado con docker.

En el docker-compose.yml añadimos en esta seccion:

```
web:
  restart: always
  container_name: decide_web
  image: decide_web:latest
  build: .
  command: ash -c "python manage.py migrate && gunicorn -w 4 decide.wsgi --timeout=500 -b 0.0.0.0:5000"
  expose:
    - "5000"
  volumes:
    - static:/app/static
  depends_on:
    - db
  networks:
    - decide
  environment: # Añadimos esto
    - DJANGO_DEBUG=False # Añadimos esto
    - DJANGO_ALLOWED_HOSTS=localhost,example.com # Añadimos esto
```

Pregunta 8: Configure DECIDE para generar releases automáticas mediante el uso de workflows.

Añadimos al workflow un nuevo job:

```
releases:
  runs-on: ubuntu-latest
  steps:
    - uses: actions/checkout@v3
    - name: Set up GitHub CLI
      uses: actions/setup-node@v3
      with:
        node-version: '16'
    - name: Create a release
      env:
        GITHUB_TOKEN: ${ secrets.GITHUB_TOKEN }
      run: |
        gh release create "$(date +%Y-%m-%d_%H-%M-%S)" --title "Release $(date +%Y-%m-%d)" --notes "Automated release from CI/CD"
```

Examen EGC-2324-1630 /Examen32.md

Pregunta 12: Realice un cambio en una línea del README.md. Genere un parche con este cambio.

Pasos a seguir:

1. Editar el archivo `README.md`:

- Abre el archivo `README.md` en tu editor de texto o IDE.
- Realiza un cambio en una línea del archivo (por ejemplo, corrige un error o agrega una línea adicional).

Ejemplo de cambio: Cambia esta línea:

Este es un proyecto de ejemplo.

A:

WUOLAH



Este es un proyecto de ejemplo actualizado.

2. Guardar los cambios:

- Guarda el archivo después de realizar el cambio.

3. Revisar el cambio con Git:

- Abre una terminal en el directorio del proyecto.
- Ejecuta el siguiente comando para comprobar los cambios realizados:

```
git diff README.md
```

Esto mostrará las diferencias entre el archivo original y el modificado.

4. Generar el parche:

- Usa el comando `git diff` para crear un archivo de parche con las diferencias.

```
git diff > cambio_readme.patch
```

Esto generará un archivo llamado `cambio_readme.patch` con las modificaciones.

5. (Opcional) Verificar el contenido del parche:

- Para asegurarte de que el parche contiene las modificaciones correctas, puedes abrir el archivo `cambio_readme.patch` en un editor de texto o usar:

```
cat cambio_readme.patch
```

6. (Opcional) Aplicar el parche en otro lugar:

- Si necesitas aplicar este parche en otro repositorio o entorno, usa:

```
git apply cambio_readme.patch
```

7. Entregar el parche:

- Sube el archivo `cambio_readme.patch` o inclúyelo en la respuesta del examen según las instrucciones del evaluador.

Ejemplo de salida del archivo `cambio_readme.patch`:

```
diff --git a/README.md b/README.md
index abc123..def456 100644
--- a/README.md
+++ b/README.md
@@ -1,1 @@
- Este es un proyecto de ejemplo.
+ Este es un proyecto de ejemplo actualizado.
```

Pregunta 13: Cambie a una rama diferente y aplique el parche en esa rama.

Pasos a seguir:

1. Crear o cambiar a una rama diferente:

- Usa el siguiente comando para cambiar a una rama existente:

```
git checkout <nombre_de_la_rama>
```

- Si necesitas crear una nueva rama y cambiar a ella, usa:

```
git checkout -b <nombre_de_la_rama>
```

2. Aplicar el parche en la nueva rama:

- Asegúrate de que el archivo de parche (`cambio_readme.patch`) esté disponible en el directorio actual.
- Aplica el parche con el siguiente comando:

```
git apply cambio_readme.patch
```

3. Verificar los cambios aplicados:

- Comprueba que el parche se haya aplicado correctamente ejecutando:

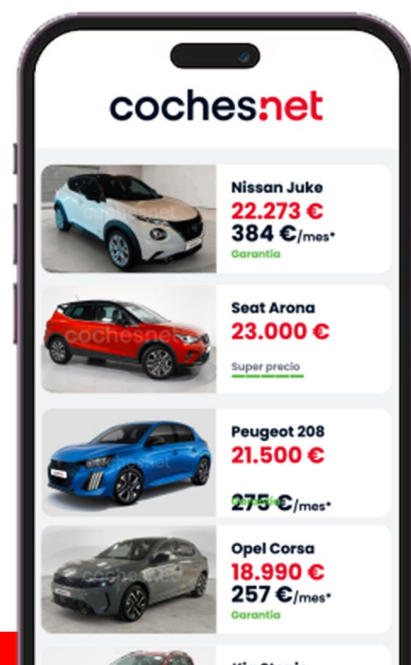
Nacida para sacar matrículas, experta en tener que matricularme tres veces de lo mismo.

Si quieres ser una verdadera experta, confía en coches.net



Vende o compra
tu coche

- ✓ Nuevos
- ✓ Renting
- ✓ Km 0
- ✓ Segunda Mano



Evolución y Gestión de la Co...



Comparte estos flyers en tu clase y consigue más dinero y recompensas



Banco de apuntes de la

WUOLAH

1

Imprime esta hoja

2

Recorta por la mitad

3

Coloca en un lugar visible para que tus compis puedan escanar y acceder a apuntes

4

Llévate dinero por cada descarga de los documentos descargados a través de tu QR



```
git diff
```

Esto mostrará los cambios que se aplicaron en la rama actual.

4. Guardar los cambios en la rama:

- Añade los cambios al área de staging:

```
git add README.md
```

- Confirma los cambios con un mensaje de commit:

```
git commit -m "Aplicar parche al README.md en la rama <nombre_de_la_rama>"
```

5. Entregar el parche aplicado:

- Si es necesario, sube la rama al repositorio remoto:

```
git push origin <nombre_de_la_rama>
```

Pregunta 3: Realice los cambios necesarios en la configuración de docker para que el servicio web lanzado mediante docker compose utilice 4 workers de gunicorn.

```
- command: ash -c "python manage.py migrate && gunicorn -w 5 decide.wsgi --timeout=500 -b 0.0.0.0:5000"
+ command: ash -c "python manage.py migrate && gunicorn -w 4 decide.wsgi --timeout=500 -b 0.0.0.0:5000"
```