Componente GIT

5. Suponga que tiene un repositorio vacío en Azure llamado “solución-legal” y es necesario llevar el código que se encuentra de manera local en su máquina al repositorio, describa el paso a paso que usted realizaría para llevar su código local al repositorio por medio de instrucciones GIT, desde cómo se conectaría a Azure hasta como realizaría el despliegue de la herramienta.

Guía: Subir Código Local al Repositorio Vacío en Azure DevOps

Esta guía describe el proceso paso a paso para llevar el código almacenado de manera local en una máquina al repositorio vacío en Azure DevOps llamado “solución-legal”. El procedimiento incluye la conexión inicial, subida del código y la configuración opcional de despliegue.

Paso 1: Configurar la conexión a Azure

1. Inicia sesión en Azure DevOps:
- Ve a https://dev.azure.com/ e inicia sesión con tu cuenta.
2. Crea un proyecto (si no existe):
- Haz clic en “Nuevo Proyecto,” proporciona un nombre (por ejemplo, “Solución Legal”), completa los detalles y haz clic en “Crear.”

Paso 2: Crear el repositorio “solución-legal”

1. Dentro de tu proyecto en Azure DevOps:
- Ve a la sección “Repos” desde el menú.
- Haz clic en “+ Nuevo repositorio.”
- Nombra tu repositorio como “solución-legal", selecciona “Git” como tipo y haz clic en “Crear.”

Paso 3: Clonar el repositorio localmente

1. Obtén la URL del repositorio:
- Dentro de “Repos,” haz clic en “Clonar” y copia la URL HTTPS o SSH del repositorio.
2. Clona el repositorio en tu máquina local:

```
git clone <URL_DEL_REPOSITORIO>
```

Paso 4: Agregar tu código al repositorio

1. Navega al repositorio clonado:

```
cd solucion-legal
```

2. Agrega los archivos de tu proyecto al repositorio.
3. Prepara los cambios para el commit:

```
git add .
```

4. Realiza el primer commit:

```
git commit -m "Primer commit: Agregar código inicial de solución-legal"
```

5. Sube los cambios al repositorio remoto:

```
git push origin main
```

Paso 5: Gestionar cambios futuros

1. Realiza modificaciones, agrega o elimina archivos en tu código local.
2. Prepara los cambios:

```
git add <nombre-del-archivo>
```

3. Haz el commit de los cambios:

```
git commit -m "Descripción de los cambios realizados"
```

4. Sube los cambios al repositorio remoto:

```
git push
```

Paso 6: Crear y fusionar solicitudes de extracción (Pull Requests)

1. Crea una nueva rama para los cambios:

```
git checkout -b <nombre-nueva-rama>
```

2. Realiza cambios, prepáralos y haz commit:

```
git add .
git commit -m "Trabajo en nueva funcionalidad"
git push origin <nombre-nueva-rama>
```

3. Crea una solicitud de extracción en Azure DevOps:
- Ve a “Repos > Pull Requests.”
- Haz clic en “Nueva solicitud de extracción,” selecciona la rama origen y destino, agrega un título y descripción, y haz clic en “Crear.”
4. Revisa y fusiona la solicitud de extracción:
- Revisa los cambios, haz clic en “Aprobar” y luego en “Completar” para fusionar los cambios en la rama principal.

Paso 7: Configurar y ejecutar CI/CD (opcional)

1. Configura un pipeline en Azure DevOps:
- Ve a la sección “Pipelines” en el menú del proyecto.
- Haz clic en “Nuevo Pipeline” y sigue las instrucciones para conectar el repositorio.
2. Define los pasos del pipeline:
- Configura las tareas necesarias para compilar, probar y desplegar tu código.
3. Ejecuta el pipeline:
- Una vez configurado, ejecuta el pipeline para validar el proceso de despliegue.

Conclusión

Al completar estos pasos, habrás subido exitosamente tu código local al repositorio vacío “solución-legal” en Azure DevOps. También habrás aprendido a gestionar cambios futuros y, opcionalmente, configurar pipelines CI/CD para automatizar procesos de compilación y despliegue.

6. ¿Tienes conocimiento sobre desarrollo web? ¿En qué proyecto? ¿Cuál fue tu aporte?

Sí, tengo experiencia en desarrollo web. Durante mi trabajo en la Vicepresidencia de Auditoría, en la Gerencia de auditoria Financiera, Jurídica y Cumplimiento de Bancolombia, desarrollé una herramienta para la evaluación de casos por la línea ética utilizando **Streamlit**. Mi aporte principal fue construir tanto el backend como el frontend de la herramienta. Implementé el backend en **Python** para procesar los datos y manejar la lógica del negocio, mientras que el frontend se diseñó en **Streamlit** para ofrecer una interfaz interactiva y fácil de usar para los auditores.

7. ¿Qué lenguajes de programación web manejas?

Manejo **Python** para el desarrollo web, específicamente en frameworks como **Django** y herramientas como **Streamlit**. También tengo conocimientos en **HTML** para diseño básico y he trabajado en proyectos de web scraping utilizando **Selenium** para automatizar tareas y extraer datos de páginas web.