BryanSegundoParcialOperativos

Estudiante	Código
Bryan Estiben Pérez Parra	12203030

Tutorial de Ejecución

Modulo 1: Preparación de La Aplicación, dentro de la Maquina Virtual

URL Del Repositorio:

https://github.com/Bryan100/BryanSegundoParcialOperativos.git

URL Del Archivo README.md

https://github.com/Bryan100/BryanSegundoParcialOperativos/blob/master/README.md

Paso 0. Inicie sesión en su Maquina Virtual (preferiblemente con la Consola de Putty.exe), ingresando su username y password.

Paso 1. Verificar que la interfaz-puente de la maquina virtual esté arriba. De lo contrario, ejecute el sgte comando:

```
  $ ifup ethx, donde el caractér 'x' varía según el número asignado a la interface puente
```

Paso 2. Ir a la carpeta /home y crear un directorio con el nombre "jenkinUser" (Ver Comandos Abajo):

Comando	Propósito
$ cd /home	Ir al directorio /home
$ mkdir jenkinUser	Crear la carpeta "jenkinUser"

Paso 3. Ubicarse dentro de la carpeta recien creada (jenkinUser) y clonar el repositorio nombrado ánteriormente (Verificar primero que la maquina virtual tiene instalada la librería de git)

Comando	Propósito
$ cd .../jenkinUser	Ir al directorio jenkinUser
$ git clone https://github.com/Bryan100/BryanSegundoParcialOperativos.git	Copiar los archivos del repositorio en la carpeta
$ yum install git	Instalar la librería git, si es necesario

Una vez ejecutado los comandos anteriores, debieron haber quedado guardados los sgts archivos (Almacenados en una carpeta llamada "BryanSegundoParcialOperativos"):

Comando	Propósito
comandos.py	Contiene los algoritmos de la aplicación
test_comandos.py	Codumenta que ejecuta pruebas, por medio del framework de Pytest, para evaluar los algoritmos descritos en el archivo "comandos.py"
test_comandos.pyc	Es el archivo de compilación de "test_comandos.py"
README.md	Informe-Tutorial para ejecutar y probar la aplicación

Los sgtes son los servicios que provee la aplicación, configurados dentro del archivo "comandos.py" y son los algoritmos que vamos a exponer a pruebas, por medio de Pytest

Nombre Algoritmo-App	Parametros	Descripción
darTodosArchivos	---	Muestra todos los archivos que hay dentro de la carpeta "BryanSegundoParcialOperativos"
agregarArchivo	Nombre Del Documentos, Contenido del Documento	Genera, dentro de la carpeta "BryanSegundoParcialOperativos", un archivo con extensión .txt
borrarArchivo	Nombre Del Documentos + Extensión (Ej: .txt)	Elimina el archivo especificado, si existe, de la carpeta "BryanSegundoParcialOperativos"

A partir de aqui, se asume que las librerias necesarias, para configurar entornos virtuales, están instaladas.

Paso 4. Crear y Activar un entorno virtual, dentro de la carpeta "BryanSegundoParcialOperativos"

Comando	Propósito
$ cd .../BryanSegundoParcialOperativos	Ir al directorio "BryanSegundoParcialOperativos"
$ virtualenv miAmbiente	Generar el Ambiente virtual
$. miAmbiente/bin/activate	Activar el Ambiente virtual creado

Paso 5. Con el ambiente virtual activo, instalar la libreria Flask

```
$ pip install Flask
```

Ahora dentro de la carpeta "BryanSegundoParcialOperativos" deberían estár, al menos, 5 archivos importantes

Archivo	Descripción
comandos.py	Contiene los algoritmos de la aplicación
test_comandos.py	Codumenta que ejecuta pruebas, por medio del framework de Pytest, para evaluar los algoritmos descritos en el archivo "comandos.py"
test_comandos.pyc	Es el archivo de compilación de "test_comandos.py"

Archivo	Descripción
miAmbiente	Es el ambiente virtual creado, el cual me va a permitir subir los servicios, configurados en el archivo "URI.py", a la nube
README.md	Informe-Tutorial para ejecutar y probar la aplicación

Hasta este punto, ya están listos los archivos con los servicios de la aplicación, los archivos con los algoritmos que permiten probar la eficacia de dichos servicios y el ambiente virtual que permite ejecutar los servicios definidos, por medio de la consola de python.

Modulo 2: Prueba de La Solución, por medio de pytest

Ya se tienen listos las sgts funciones, de nuestra aplicación:

Nombre Algoritmo-App	Parametros	Descripción
darTodosArchivos	---	Muestra todos los archivos que hay dnetro de la carpeta "BryanSegundoParcialOperativos"
agregarArchivo	Nombre Del Documentos, Contenido del Documento	Genera, dentro de la carpeta "BryanSegundoParcialOperativos", un archivo con extensión .txt
borrarArchivo	Nombre Del Documentos + Extensión (Ej: .txt)	Elimina el archivo especificado, si existe, de la carpeta "BryanSegundoParcialOperativos"

Paso 0. Para ejecutar pruebas unitarias de las funciones anteriores, se requiere el framework de pytest. Para lo anterior, ejecutar los sgts:

Comando	Descripción
$ pip install -U pytest	Instalar Pytest, en caso no de no tenerlo importado en la maquina virtual
pytest --version	Verificar Instalación

Una vez instalado el framework de pruebas, se puede ejecutar el archivo de pruebas "test_comandos.py", el cual permite hacer pruebas a las funciones descritas anteriormente. Dicho archivo contiene los sgts metodos:

Función / Metodo	Descripción
test_agregar	Verifica si el metodo "agregarArchivo()", del archivo "comandos.py", genera y guarda un archivo .txt, dentro de la carpeta "BryanSegundoParcialOperativos"
test_borrar	Verifica si el metodo "borrarArchivo(...)", del archivo "comandos.py", elimina el archivo especificado, de la carpeta "BryanSegundoParcialOperativos"
test_darTodos	Verifica si el metodo "darTodosArchivos()", del archivo "comandos.py", genera un archivo, con el titulo y contenido especificado, dentro de la carpeta "BryanSegundoParcialOperativos"

Las funciones de prueba anteriores utilizan las sgts estrategias, para verificar que sus funciones respectivas cumplen con lo esperado

Función	Estrategia
test_agregar	Se le pide a la aplicación la cantidad inicial de archivos, que hay dentro de la carpeta "BryanSegundoParcialOperativos".Luego, se agrega un archivo, por medio de la función a probar.Finalmente, se verifica que: Cantidad final de archivos == La cantidad inicial + 1. Si lo anterior se cumple, la prueba fue exitosa, de lo anterior, fue fallida.
test_borrar	Casi analogo a test_agregar(). Se le pide, a la aplicación, la cantidad inicial de archivos, que hay dentro de la carpeta "BryanSegundoParcialOperativos".Luego, se elimina un archivo, por medio de la función a probar. Finalmente, se verifica que la cantidad final de archivos == Cantidad Inicial - 1. Si lo anterior se cumple, la prueba fue exitosa, de lo anterior, fue fallida.
test_darTodos	Se agregar y eliminan archivos durante toda la ejecución de esta función de prueba. A medida que se agregan/eliminan archivos durante el algoritmo, se hacen muestreos, por medio de la función a probar, de la cantidad de documentos respectiva que hay en la carpeta "BryanSegundoParcialOperativos". Por último, se comparan los muestreos y se verifica que sean coherentes con las acciones de agregar/eliminar que se hicieron a lo largo de todo el algoritmo, para indicar el exito o fallo de la prueba.

Paso 1. Ya se tiene claridad del contenido del archivo "comandos.py", del archivo "test_comandos.py" y de la relación que hay entre ellos. Para ejecutar las pruebas, por medio de pytest, insertar el sgte comando:

```
$ pytest -q test_comandos.py
```

Una vez ejecutado el anterior comando, se puede ver los resultados de las pruebas (Ver Documento "Pantallazo de Pruebas.docx")

Modulo 3: Ejecución de la Solución, por medio de la consola de Python

Otra alternativa, para verificar si las funciones del archivo "comandos.py" están haciendo lo esperado, se puede hacer lo sgte:

Paso 0. Ejecutar la consola de python (Recuerde tener el ambiente virtual activado)

```
$ python
```

Paso 1. Importar los servicios del archivo "comandos.py". En este tutorial se importan todas las funciones al mismo tiempo.

```
$ from comandos import *
```

Paso 2.A Verificar la función darTodosArchivos().

```
$ darTodosArchivos()
```

En la consola de debieron haber desplegado una lista, con los nombre de cada archivo que está dentro de la carpeta "BryanSegundoParcialOperativos". Ahora procedemos a ejecutar los sgts comandos.

Comando	Descripción
exit()	Salir del entorno de python
$ cd .../BryanSegundoParcialOperativos	Ubicarse en la carpeta "BryanSegundoParcialOperativos", en caso de no estár ubicado en ella
$ ls	Enlistar los archivos de la carpeta "BryanSegundoParcialOperativos"

La función darTodosArchivos() y el comando "$ ls" debieron haber arrojado el mismo resultado. Si lo anterior ocurrió, la función darTodosArchivos() está haciendo su tarea correctamente.

Paso 2.B Verificar la función agregarArchivo(String nombreArchivo, String contenido).

Ejecutemos los comandos en el sgte orden (Aquí se asume que la función darTodosArchivos() provee los servicios esperados) :

Precondición: El archivo llamado "miNuevoArchivo.txt" NO existe dentro de la carpeta "BryanSegundoParcialOperativos"

Comando	Descripción
0. Paso 0 y Paso 1	Pasos del modulo 3 (Recordatorio)
1. agregarArchivo("miNuevoArchivo", "...")	Agrega un archivo a la carpeta BryanSegundoParcialOperativos
2. darTodosArchivos()	Entrega todos los archivos
3. exit()	Salir de la consola de python
4. cat miNuevoArchivo.txt	Muestra el contenido de un Archivo

Dentro de la lista, que retorna la función darTodosArchivos(), debe contener un elemento llamado "minuevoArchivo.txt", para poder concluir que el metodo agregarArchivo() es exitoso. Además, la función cat (paso 4) debió haber retornado el contenido que se envió como parametro a la función.

Paso 2.C Verificar la función borrarArchivo(String nombreArchivo).

Ejecutemos los comandos en el sgte orden (Aquí se asume que la función darTodosArchivos() provee los servicios esperados) :

Precondición: El archivo llamado "miNuevoArchivo.txt" YA existe dentro de la carpeta "BryanSegundoParcialOperativos" Recuerde que el parametro de la función debe incluir la extensión del archivo (El nombre es insuficiente)

Comando	Descripción
0. Paso 0 y Paso 1	Pasos del modulo 3 (Recordatorio)
1. borrarArchivo("miNuevoArchivo.txt")	Agrega un archivo a la carpeta jenkinUser

Comando	Descripción
2. darTodosArchivos()	Entrega todos los archivos
3. exit()	Salir de la consola de python

Dentro de la lista, que retorna la función darTodosArchivos(), el archivo llamado "minuevoArchivo.txt" YA NO debe estár, para poder concluir que el metodo borrarArchivo() fue exitoso.

Modulo 4: Servidor de Integración

...

Modulo 5: Anexos

Los sgtes son otros archivos, dentro del repositorio que son importantes

Nombre	Descripción
comandos.py	Codigo Fuente de la Solución
test_comandos.py	Codigo Fuente de las Pruebas de la Solución
Pantallazos de Pruebas.docx	Informe con la descripción de las Pruebas, en diferentes casos y escenarios