

**PROYECTO PARCIAL**

**DATAOPS**

**INTEGRANTES:**

Gamero Chávez Fabian Alberto

**Docente:**

Pamela Pimentel Huerta

**Año:**

2024

**PROYECTO PARCIAL**

El presente proyecto tiene como objetivo descargar un dataset desde plataforma Kaggle usando Python, el dataset es cargado a SQL Server y este proceso debe ser automatizado con Jenkins.

La base de datos escogida es tesla-stock-price de henryshan, la base de datos nos brinda información diaria del precio del mercado de las acciones correspondientes a la empresa Tesla,

1. **Beneficios de Jenkins para la automatización.**

Jenkins nos permite automatizar la compilación de nuestros códigos, lo cual permite la ejecución de tareas rutinarias una vez sea configurada la ejecución dentro de la plataforma Jenkins

Permite relacionarse con distintos tipos de software, lo cual permite integrar procesos independientemente si las etapas de estos se encuentren distribuidos en distintos programas que puedan o no compartir lenguajes de programación entre sí.

Jenkins es de código abierto, lo cual permite encontrar en internet gran cantidad de información desarrollada previamente por otros usuarios, lo cual permite aprender de la plataforma.

Configuración sencilla, la cual permite, en base a ejemplos que brinda la plataforma, poder adaptarse a las necesidades que uno presenta como usuario

Cuenta con un sistema capaz de identificar y solucionar los fallos muy rápidamente, lo que contribuye a que el software se mantenga en un estado donde tenga la posibilidad de liberar cuando quiera de manera segura.



1. **Despliegue del Pipeline en Jenkins.**
   1. **Creación del programa Script\_python.py**

El propósito del programa es definir en que lugar se va a guardar el dataset de nuestro proyecto y cargar la información en dicha dirección.

* Una vez definida la dirección, se comprueba su existencia y en caso contrario, se procede a crearla.
* Este programa esta pensado para ser ejecutado periódicamente, por lo cual, en cada ejecución, elimina los archivos creados a fin de cargar la data en una carpeta vacía.
* Finalmente, utiliza las herramientas de kaggle para encontrar el archivo que necesitamos en su repositorio y proceder a descargarlo en nuestra dirección dentro de nuestra computadora.

import os

from kaggle.api.kaggle\_api\_extended import KaggleApi

location = "C:/Users/FABIAN/Documents/DevOps/proyecto\_parcial/python/dataset"

###Validar que la carpeta exista###

if not os.path.exists(location):

    ##En caso mi carpeta no exista, voy a crear una nueva##

    os.mkdir(location) ##mkdir -> make directory

else:

    ##Si la carpeta ya existe, entonces borramos el contenido##

    for root, dirs, files in os.walk(location, topdown=False):

        for name in files:

            os.remove(os.path.join(root,name)) ##elimino todos los archivos

        for name in dirs:

            os.rmdir(os.path.join(root,name)) ##rmdir -> remove directory / elimino todas mis carpetas

api = KaggleApi()

api.authenticate()

#print(api.dataset\_list(search='henryshan'))

api.dataset\_download\_file('henryshan/tesla-stock-price','TSLA.csv',path=location,quiet = False)

* 1. **Creación del programa Script\_python.py**

El presente programa carga la data descargada desde kaggle hacia un tabla definida en Sql Server

* Primero se debe de crear la base de datos Kaggle, donde será cargada la informacion

Texto

Descripción generada automáticamente

* En caso no exista la tabla dbo.TSLA, se creara dicha tabla con los atributos correspondientes.
* Se eliminan los datos cargados en una ejecuacion previa
* Finalmente se procede a cargar la data proveniente del archivo .csv descargado en el programa anterior

USE Kaggle

GO

IF NOT EXISTS (SELECT \* FROM SYS.TABLES WHERE object\_id = OBJECT\_ID(N'dbo.TSLA') AND TYPE = 'U')

CREATE TABLE dbo.TSLA (

FECHA DATE,

Open\_DB FLOAT,

High\_DB FLOAT,

Low\_DB FLOAT,

Close\_DB FLOAT,

Adj\_Close FLOAT,

Volume FLOAT

)

GO

TRUNCATE TABLE dbo.TSLA;

GO

BULK INSERT dbo.TSLA

FROM 'C:\Users\FABIAN\Documents\DevOps\proyecto\_parcial\python\dataset\TSLA.csv'

WITH

(

FIRSTROW = 2,

FIELDTERMINATOR = ',',

ROWTERMINATOR = '0x0a'

)

GO

1. **Código del Pipeline.**

El código en su primera versión cuenta con dos Stage (Etapas):

* La primera etapa se encarga de ejecutar el programa en Python para obtener el archivo con la data procedente de kaggle
* La segunda etapa realiza la conexión con la base de datos en Microsoft SQL server, y procede a cargar la data en un tabla “dbo.TSLA”

pipeline {

agent any

stages {

stage('Python') {

steps {

bat label:'import\_Kaggle', script: 'python C:/Users/FABIAN/Documents/DevOps/proyecto\_parcial/python/Script\_python.py'

}

}

stage('bd'){

steps {

bat label:'import\_sql', script:'"C:/Program Files/Microsoft SQL Server/Client SDK/ODBC/170/Tools/Binn/SQLCMD.EXE" -S "DESKTOP-78LVVQL" -d Kaggle -E -i "C:/Users/FABIAN/Documents/DevOps/proyecto\_parcial/SQL/script\_db.sql"'

}

}

}

}

1. **Monitoreo del Pipeline.**

Se procedió a construir el Pipeline dentro de Jenkins, se visualiza que ejecuta la primera etapa, descargando al 100% el archivo .csv desde el repositorio kaggle



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Al momento de ejecutar la siguiente etapa, se visualiza la conexión que realiza con Microsoft SQL Server y como procede a añadir las 3383 filas del archivo TSLA.csv a nuestra base de datos

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Visualizamos la data cargada en la tabla dbo.TSLA

Tabla

Descripción generada automáticamente

1. **Conexión Jenkins con Github.**

Como primer paso se tiene que contar con un repositorio dentro de GitHub, en el cual se deben de cargar los archivos a ser ejecutados

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Se debe de instalar las herramientas git en la computadora.

Texto

Descripción generada automáticamente

Al momento de configurar el pipeline, se debe indicar el Url del proyecto en GitHub

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Se procede a modificar el código con la finalidad que los archivos de los programas sean extraídos desde GitHub

Se define dos etapa, la primera Checkout, que nos permite confirmar la conexión con el repositorio github y la segunda Build que ejecuta los programas provenientes de github

pipeline {

agent any

stages {

stage('Checkout') {

steps {

checkout scmGit(branches: [[name: '\*/main']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/FabianGa1998/Proyecto\_Parcial\_DevOps.git']])

}

}

stage('Build') {

steps {

git branch: 'main', url: 'https://github.com/FabianGa1998/Proyecto\_Parcial\_DevOps.git'

bat label:'import\_Kaggle', script: 'python Script\_python.py'

bat label:'import\_sql', script:'"C:/Program Files/Microsoft SQL Server/Client SDK/ODBC/170/Tools/Binn/SQLCMD.EXE" -S "DESKTOP-78LVVQL" -d Kaggle -E -i "script\_db.sql"'

}

}

}

}

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. **Envío de notificaciones mediante mail.**

Para realizar este envio de notificaciones, se deben realizar las siguientes configuraciones.

Añadir una credencial, que incluya tanto el correo como la contraseña del correo desde el cual se va a enviar el correo.

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

En el pipeline, se deberá añadir un stage Email, donde se detalle el correo a ser redactado

stage('Email') {

steps {

emailext body: '''El proceso a sido terminado con éxito.

Atentamente,

Jenkins''', subject: 'Proceso terminado trabajo Parcial', to: '74225091@certus.edu.pe'

}

}

Se confirma él envió del correo:

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. **Conclusiones y Recomendaciones.**

* El uso de Jenkins permite automatizar la ejecución de programas independientemente del lenguaje y software a utilizarse mientras se realice la correcta configuración
* El grado de automatización que se pueda lograr a través de Jenkins depende de la habilidad y conocimiento del usuario, es una herramienta flexible y adaptable a diversas necesidades
* El uso de Python con Jenkins requerirá configurar correctamente el Path donde se localice el ejecutable de Python
* El uso de SQL Server requiere la versión de desarrollador del software, en caso contrario, no ejecutara el programa por una limitante en la sintaxis del nombre del servidor.
* El uso de github como repositorio de los códigos de los programas, permite guardar copias de seguridad de las distintas versiones de los programas a ejecutarse.
* Es de tener cuidado al momento de configurar el correo electrónico para Jenkins debido a que estará solicitando tanto usuario y contraseña, lo que si no se tiene el cuidado suficiente, puede provocar un problema de seguridad del usuario y de la institución que proporciono el correo.