**Project kết thúc môn học Thực hành tin học ứng dụng**

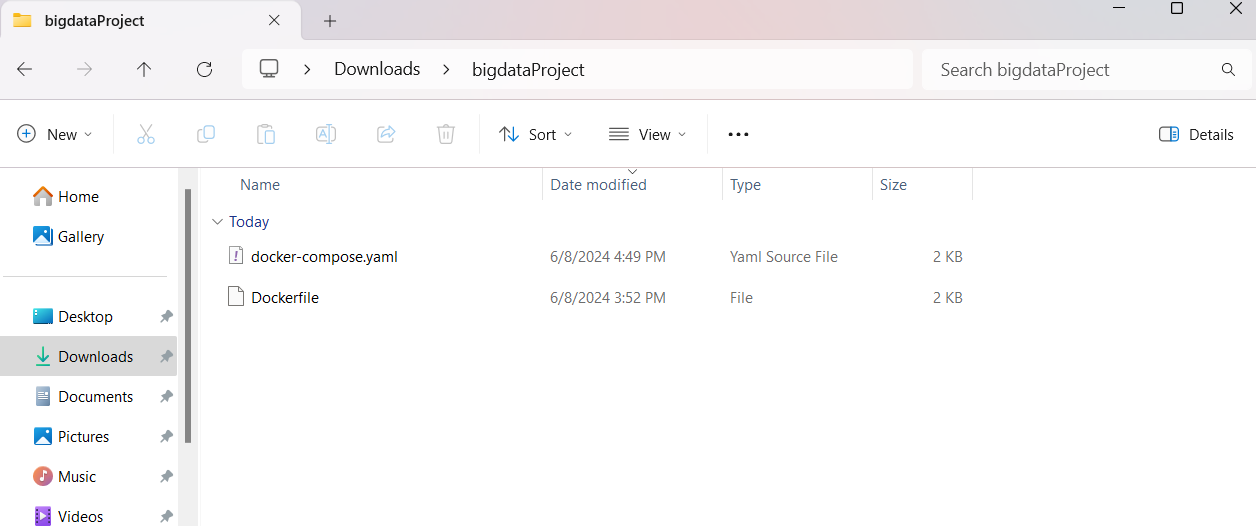
**1. Docker Setup**

* Thực hiện tải xuống Docker Desktop cho hệ máy Window

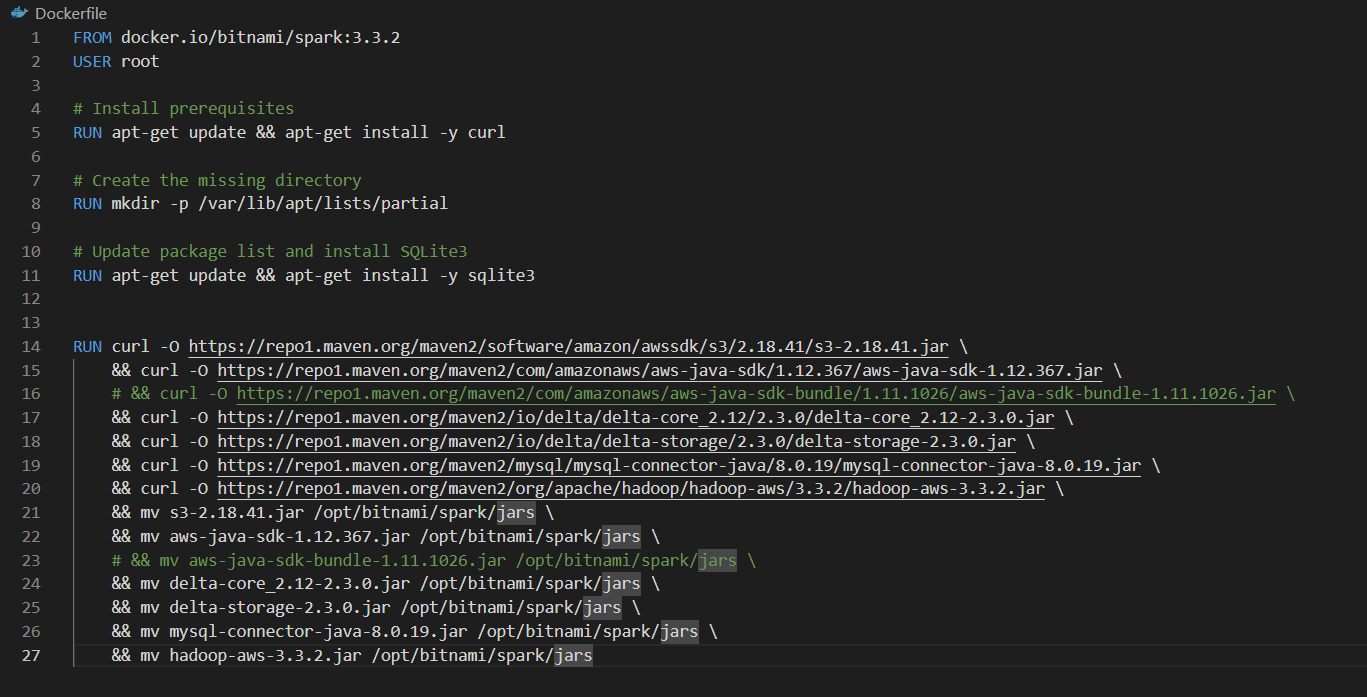
****

*Hình 1: Docker Desktop download*

* Tạo một thư mục có tên là bigdataProject, bên trong tạo một file Dockerfile và 1 file docker\_compose.yaml để phục vụ cho quản lý đa container.

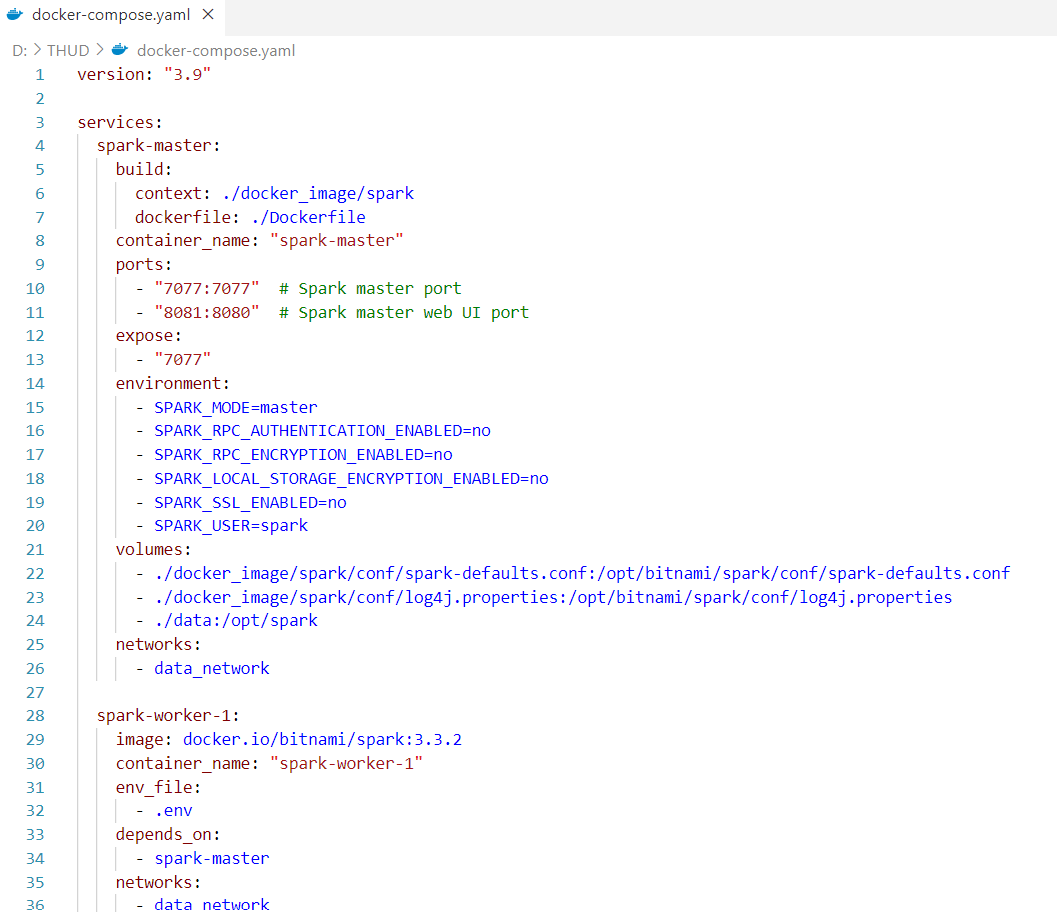


*Hình 2. Thư mục của project*

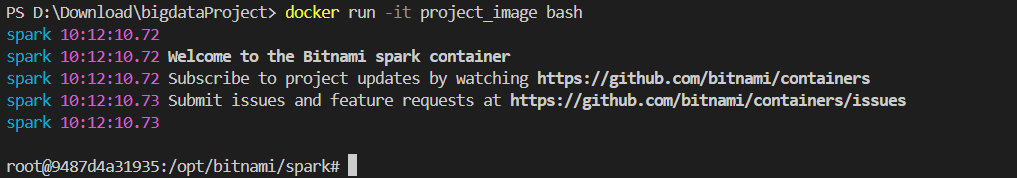
* 

*Hình 3. File Dockerfile*

* Mở terminal của thư mục trên hình 2, thực hiện tạo một images bằng lệnh: "**docker built –t project\_image .**"
* Sau đó chạy images đó để chạy container bằng lệnh: "**docker run -it project\_image bash**"



Hình 4: Docker-Compose



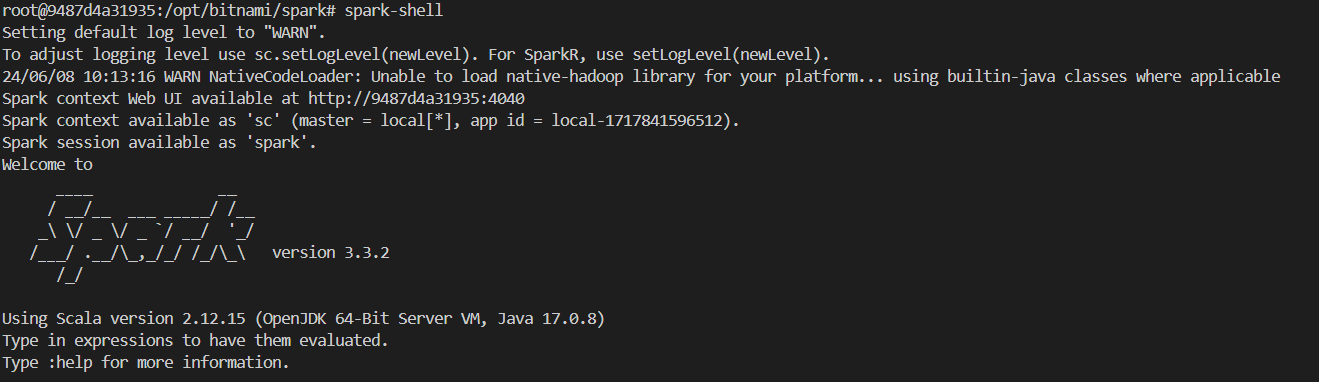
Hình 5: Xác nhận build thành công

**2. Spark Installation**

Trong Dockerfile chứa các câu lệnh cho việc cài đặt các công cụ cần thiết cho việc sử dụng và chạy Spark trên container. Ngoài ra có một số công cụ và thư viện để dễ dàng hơn trong quá trình làm việc với Docker như sau:

* SQLite3: Một hệ quản trị cơ sở dữ liệu nhúng (embedded database) nhẹ, không cần server, thường được sử dụng cho các ứng dụng nhỏ hoặc phát triển.
* AWS SDK for Java (s3-2.18.41.jar): Bộ công cụ phát triển phần mềm của Amazon cho phép Spark tương tác với các dịch vụ lưu trữ Amazon S3.
* AWS Java SDK (aws-java-sdk-1.12.367.jar): Bộ công cụ toàn diện hơn của Amazon cho Java, bao gồm nhiều dịch vụ AWS khác ngoài S3.
* Delta Lake (delta-core\_2.12-2.3.0.jar & delta-storage-2.3.0.jar): Một framework mã nguồn mở giúp xây dựng hồ dữ liệu (data lake) trên nền tảng Apache Spark, cung cấp các tính năng ACID và khả năng xử lý dữ liệu theo thời gian thực (real-time).
* MySQL Connector/J (mysql-connector-java-8.0.19.jar): Trình điều khiển JDBC để kết nối Spark với cơ sở dữ liệu MySQL.
* Hadoop AWS (hadoop-aws-3.3.2.jar): Thư viện Hadoop cung cấp tích hợp giữa Hadoop và các dịch vụ AWS, cho phép Spark đọc và ghi dữ liệu từ các dịch vụ lưu trữ AWS như S3.

Sau đó, thực hiện **build** lại images với tên my-task và chạy container và cuối cùng, nhập lệnh "**spark-shell**" để xác minh Spark đã chạy

****

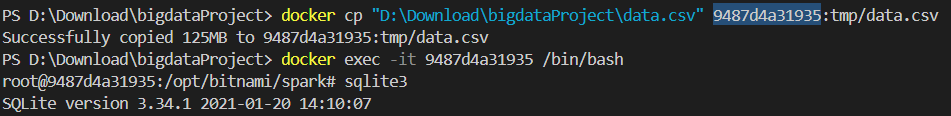
Hình 6: Xác nhận Spark thành công

**3. Database Setup**

Trong Dockerfile, chúng ta đã có câu lệnh thực hiện việc update và install Sqlite để đảm bảo việc Sqlite được cài đặt khi chúng ta thực hiện tạo Image và chạy Container

Tiếp theo, chúng ta thực hiện việc copy 1 file có tên 1000000 Sales Records.csv được lấy từ Excel BI Analytics, sau đó ta đổi tên thành data.csv.

Truy cập vào terminal của container bằng câu lệnh docker exec -it <spark-id> /bin/bash , sau đó tạo và chạy cơ sở dữ liệu sqlite 3. <spark-id> hiện ở vị trí root@<spark-id> ở màn hình build.



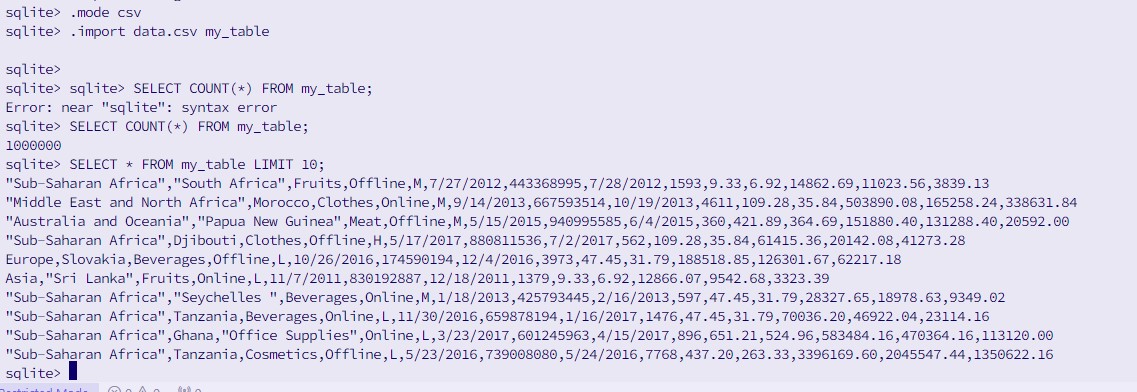
**Ảnh 7: Tạo file data.csv và cơ sở dữ liệu sqlite 3**

Sau khi thực hiện bước trên, giao diện sẽ hiển thị tương tác với database, lúc này ta sẽ thực hiện 2 lệnh:

* + .mode csv
  + .import data.csv my\_table

Quá trình này import file data.csv gồm 1 triệu dòng vào my\_table

-Thử một câu truy vấn hiển thị 10 dòng dữ liệu đầu

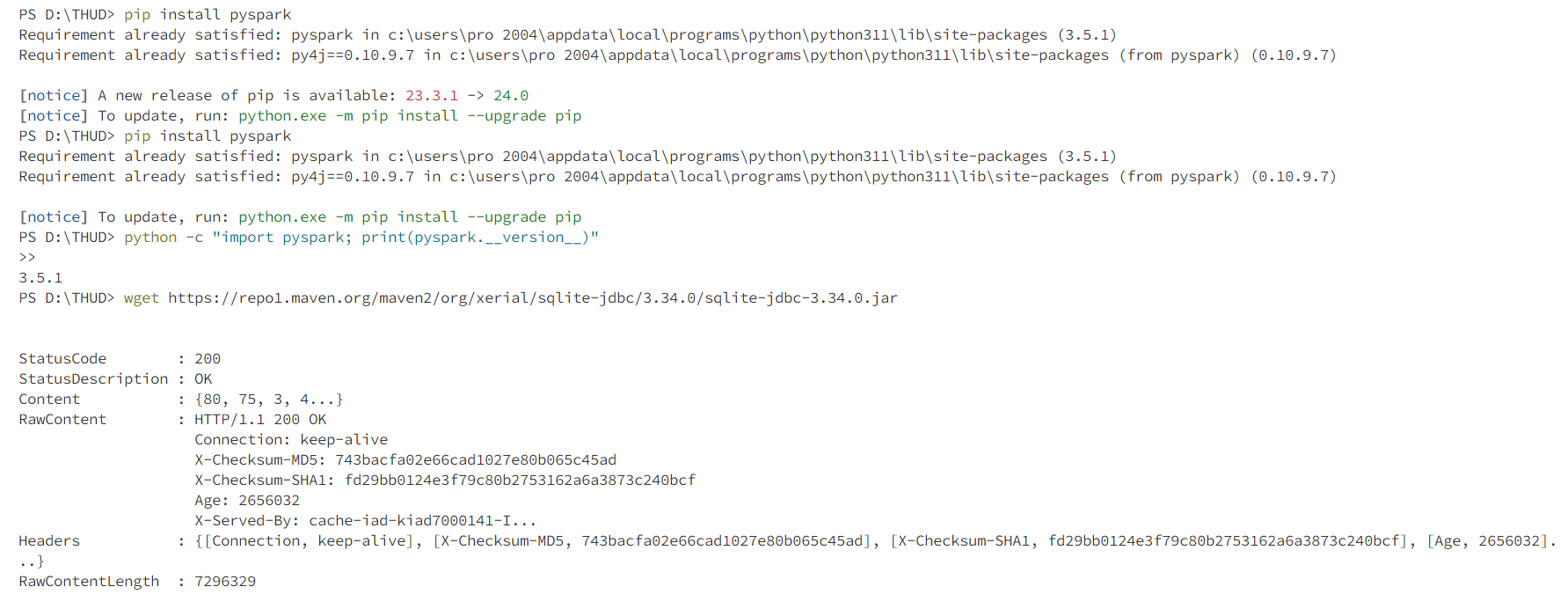


**4. Dependencies**

Chúng ta sẽ thêm 2 thành phần cần thiết là PySpark và JDBC cho sqlite để giúp kết nối và thực hiện truy vấn từ Spark đến cơ sở dữ liệu. Để thực hiện được tôi đã sử dụng 3 câu lệnh sau:

**pip install --upgrade setuptools**  
 **pip install pyspark**

**wget** [**https://repo1.maven.org/maven2/org/xerial/sqlite-**](https://repo1.maven.org/maven2/org/xerial/sqlite-) **jdbc/3.34.0/sqlite-jdbc-3.34.0.jar**

****

**5. Configuration**