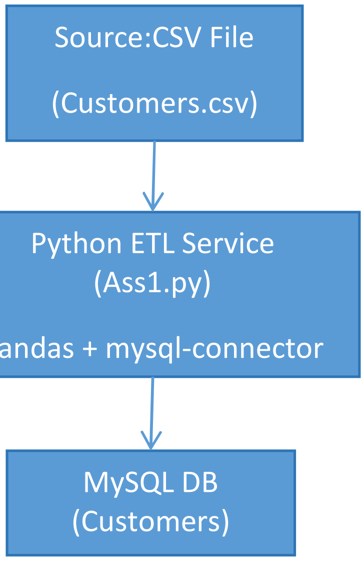
Project Title: Database & Python ETL with Reproducible Infrastructure

1. **Objective :** To design and implement a reproducible, containerized ETL pipeline that extracts data from a CSV source, transforms it using Python, and loads it into a SQL database. The setup to be reproducible with Docker Compose.
2. **Architecture Overview :**



3. **Technology Stack :**

Component	**Technology**	**Purpose**
Programming Language	Python 3.10	ETL logic & data transformation
Libraries	Pandas Mysql-connector	Data processing and DB interaction
Database	MySQL	Target DataBase
Containerization	Docker & Docker Compose	Reproducible Deployment

4. **Data Flow**

Extract

- Reads the input CSV file customers-100.csv using pandas.read_csv().
- Handles missing values by filling them with empty strings (fillna(""))

Transform

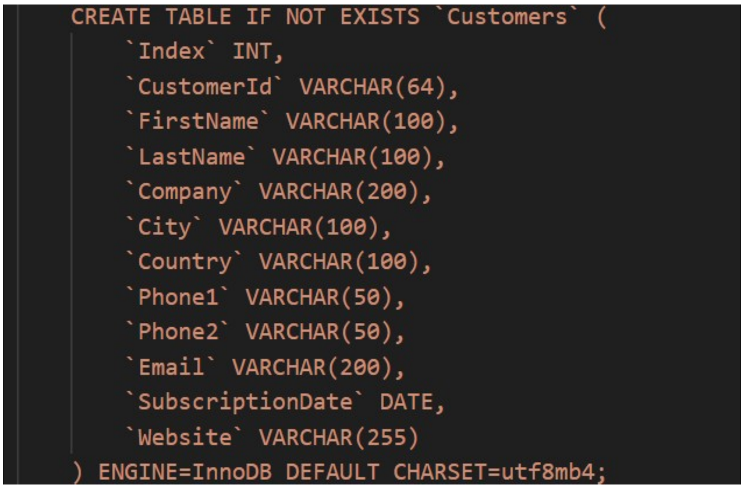
- Converts the Subscription Date column to DATE format using pd.to_datetime(errors="coerce").
- Ensures column consistency even when spaces exist in headers ("First Name", "Customer Id"..).

Load

- Establishes a connection to MySQL using mysql.connector.connect().
- Creates a table Customers (if not exists) with defined schema and UTF-8 charset.
- Iterates over DataFrame rows and executes parameterized INSERT INTO queries.
- Commits transactions and closes the connection gracefully.

5. Database Schema

Table : Customers



```sql
CREATE TABLE IF NOT EXISTS `Customers` (
    `Index` INT,
    `CustomerId` VARCHAR(64),
    `FirstName` VARCHAR(100),
    `LastName` VARCHAR(100),
    `Company` VARCHAR(200),
    `City` VARCHAR(100),
    `Country` VARCHAR(100),
    `Phone1` VARCHAR(50),
    `Phone2` VARCHAR(50),
    `Email` VARCHAR(200),
    `SubscriptionDate` DATE,
    `Website` VARCHAR(255)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

6. Data Validation & Error Handling

- Invalid or unparsable dates are coerced to Nat and then handled as null.
- Missing optional fields (e.g.,Phone 2, website) default to empty strings.
- MySQL connection details (host, user, password, database) are read from environment variables for portability.

7. Environment Variables

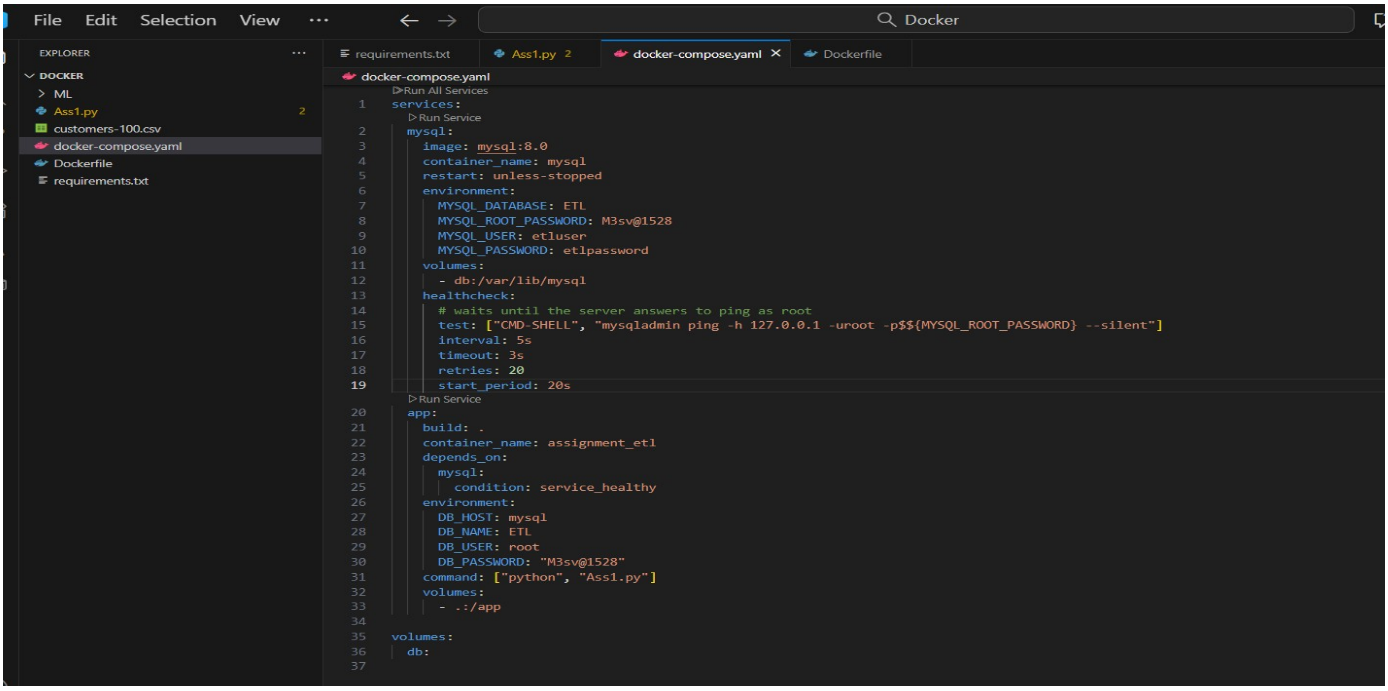
DB_HOST : Database Host - mysql

DB_USER :MySQL Username - root

DB_PASSWORD : MySQL password - password

DB_Name : Target Database - ETL

8. Docker Compose

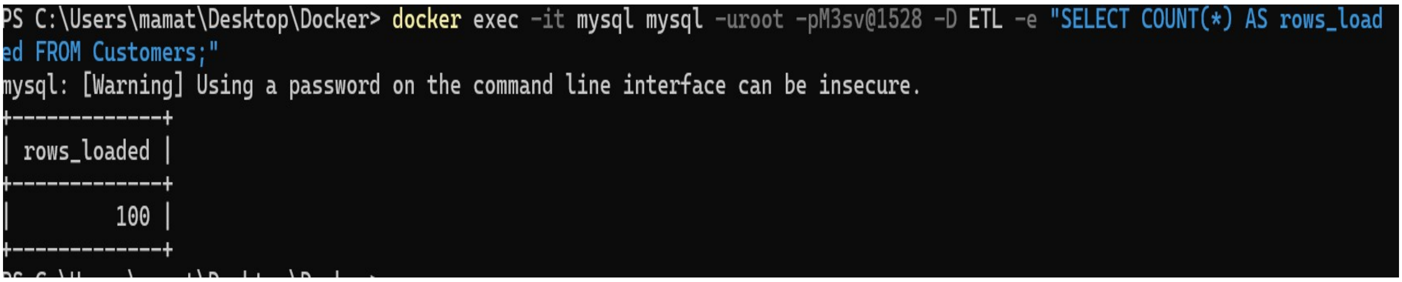


```yaml
services:
  mysql:
    image: mysql:8.0
    container_name: mysql
    restart: unless-stopped
    environment:
      MYSQL_DATABASE: ETL
      MYSQL_ROOT_PASSWORD: M3sv@1528
      MYSQL_USER: etluser
      MYSQL_PASSWORD: etlpassword
    volumes:
      - db:/var/lib/mysql
    healthcheck:
      # waits until the server answers to ping as root
      test: ["CMD-SHELL", "mysqladmin ping -h 127.0.0.1 -uroot -p$${MYSQL_ROOT_PASSWORD} --silent"]
      interval: 5s
      timeout: 3s
      retries: 20
      start_period: 20s
  app:
    build: .
    container_name: assignment_etl
    depends_on:
      mysql:
        condition: service_healthy
    environment:
      DB_HOST: mysql
      DB_NAME: ETL
      DB_USER: root
      DB_PASSWORD: "M3sv@1528"
    command: ["python", "Ass1.py"]
    volumes:
      - .:/app

volumes:
  db:
```

9. Testing & Validation

- Verify that customers-100.csv loads correctly (record count matches DataFrame).
- Validate database connection and table creation.



```
PS C:\Users\mamat\Desktop\Docker> docker exec -it mysql mysql -uroot -pM3sv@1528 -D ETL -e "SELECT COUNT(*) AS rows_load
ed FROM Customers;"
mysql: [Warning] Using a password on the command line interface can be insecure.
+-------------+
| rows_loaded |
+-------------+
|         100 |
+-------------+
PS C:\Users\mamat\Desktop\Docker>
```

10. Reproducibility

- Package ETL script (Ass1.py) and CSV under one folder
- Define .env and docker-compose.yml.
- Runs
  docker compose up --build
- Validate data in MySQL: SELECT COUNT(*) FROM Customers;
- Git : **GitHub**