**做题时间：预计1.5-2个小时**

**目标：**

设计并采用Python实现一个端到端的数据解决方案，将数据其转化并加载到一个 NoSQL (譬如 MongoDB) 数据库，并通过受保护的 GraphQL API 提供数据访问。最后将完整的系统dockerize以方便我们操作。

**情境：**

设想一个大型电商平台希望通过 GraphQL API 向其合作伙伴展示其产品目录。原始的产品数据存储在几个 CSV 文件中：

products.csv：包含列 product\_id、name、description、price 和 category\_id。

categories.csv：包含列 category\_id 和 category\_name。

解决方案应使 API 使用者能够根据各种属性（譬如 name、category 等等）查询产品，并检索所需的产品信息以及其类别名称。

**任务：**

**第一步：数据提取、转换和加载 (ETL)**

从给定的 CSV 文件中提取数据。

规范和清理数据（编写逻辑处理不一致性、缺失值等）。

将数据转换成适合 NoSQL 数据库的模式，并考虑未来的查询模式。

将转换后的数据加载到 NoSQL 数据库中。

**第二步：NoSQL 数据库设置**

选择一个适合存储产品数据的 NoSQL 数据库，并说明你的选择理由。

定义一个数据库模式，以支持常用用例的高效查询（例如，按类别、价格范围等检索产品）。

确保数据完整性并实现必要的索引。

**第三步：GraphQL API**

开发一个NoSQL 数据库中产品数据的 GraphQL API。

API 应支持基于 token 的身份验证。

确保 API 支持：

* 根据不同属性（例如，name、price、category\_name）查询产品。
* 检索相关的产品详情。

实施适当的错误处理和数据验证。

**需要输出以下内容：**

1. ETL 过程、NoSQL 数据库设置和 GraphQL API 的代码库，结构清晰且易于维护。
2. 包括以下内容的文档：

* 设置和安装说明。
* 数据库模式和数据模型详情 (字段等等）。
* 关键设计决策的解释和理由。
* API 使用示例。
* 有关可扩展性、数据一致性和错误处理的注意事项说明。

1. Dockerfile。
2. 展示解决方案正确功能的测试用例 （如API访问的例子）。