**E-commerce Product Demand**

**Data Collection and Integration (10 Marks)**

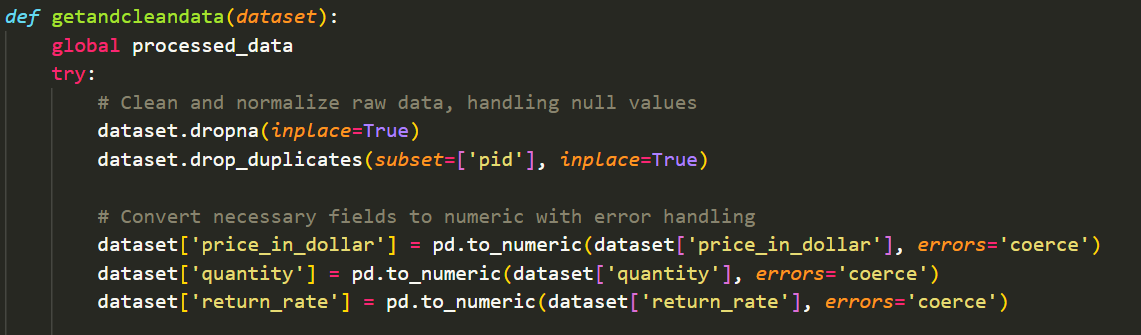
1.Source Diversity (2 Marks): Integrate data from at least five different sources in various formats (JSON, XML, CSV, Excel, HTML).

and

2.Data Fetching and API Integration (3 Marks):Set up APIs, handle connections, and manage data fetching with error handling.

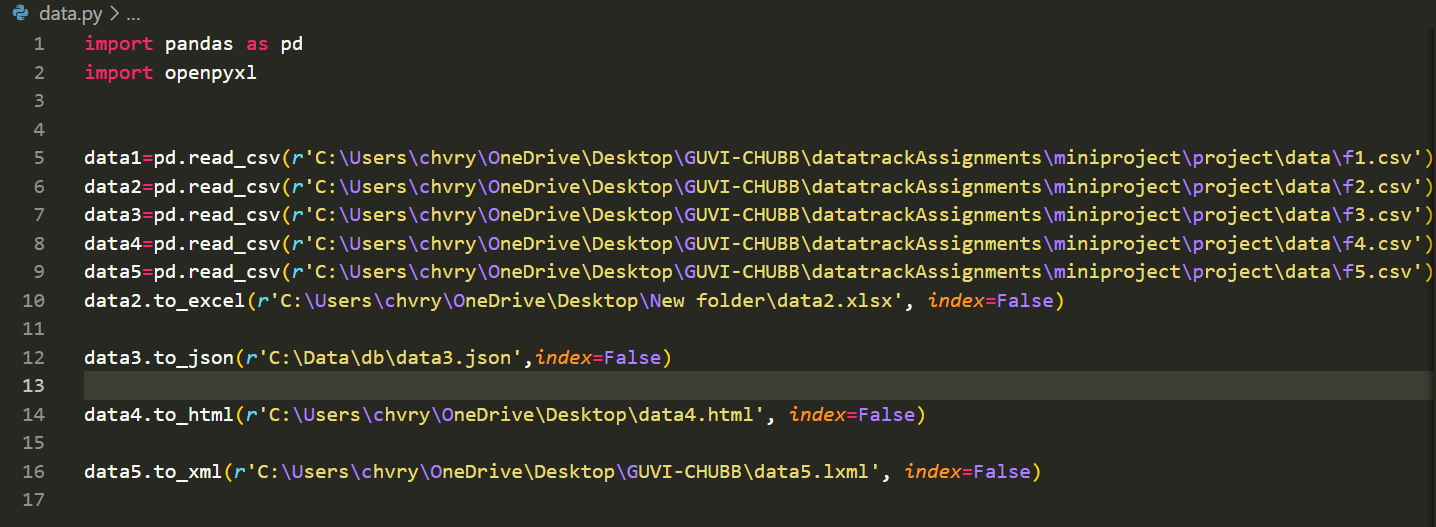


3.Data Pre-processing (2 Marks): Clean and normalize raw data, handling null values, duplicates, and inconsistencies.



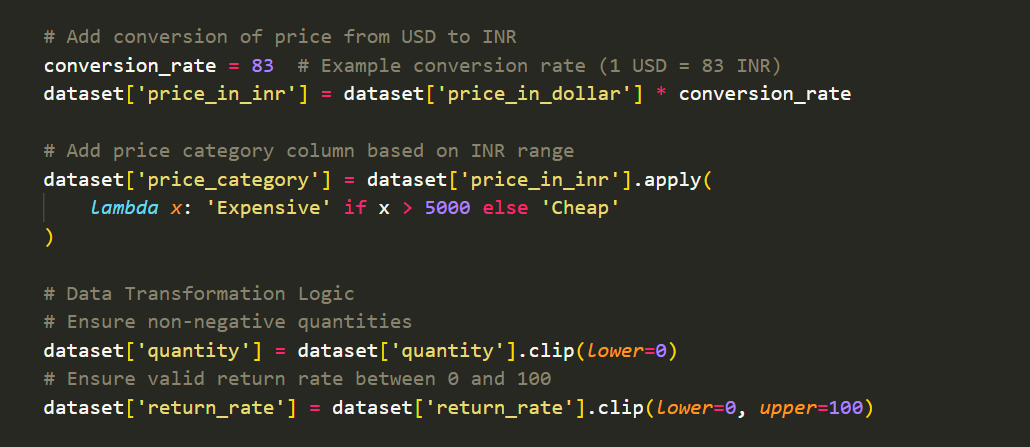
4.Documentation (3 Marks): Document data sources, formats, and integration setup in detail.

Converting existing csv to different formats like XML,EXCEL,JSON,HTML

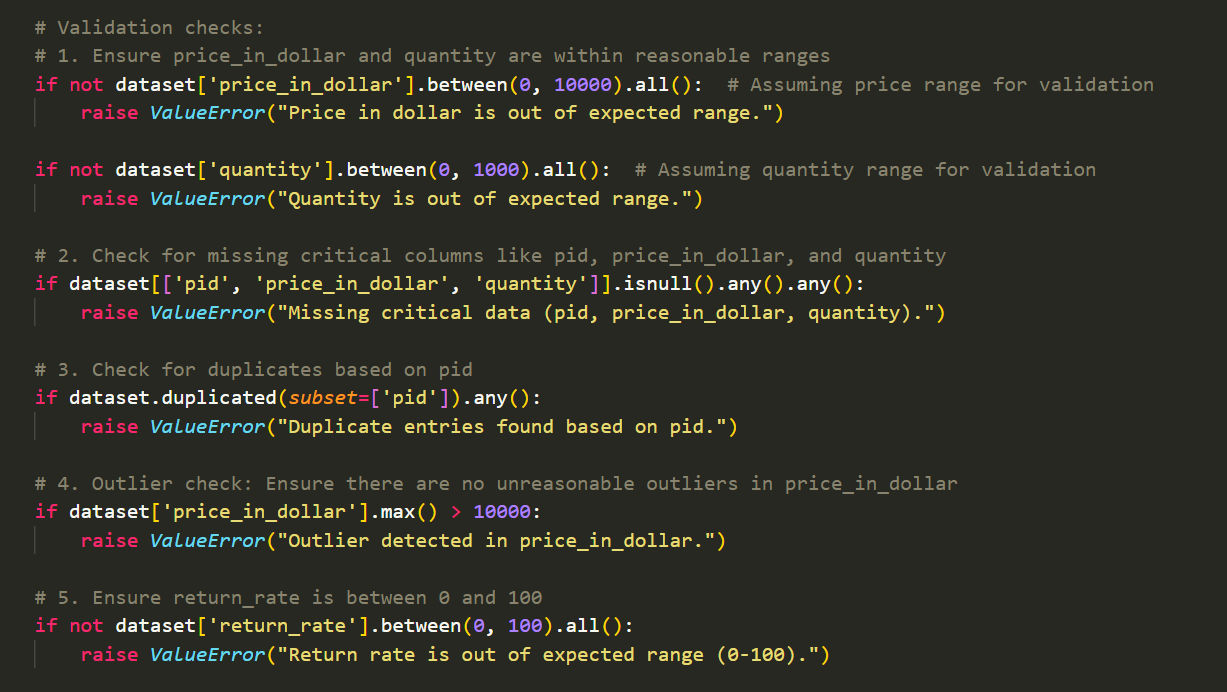


**ETL (Extract, Transform, Load) Pipeline (15 Marks)**

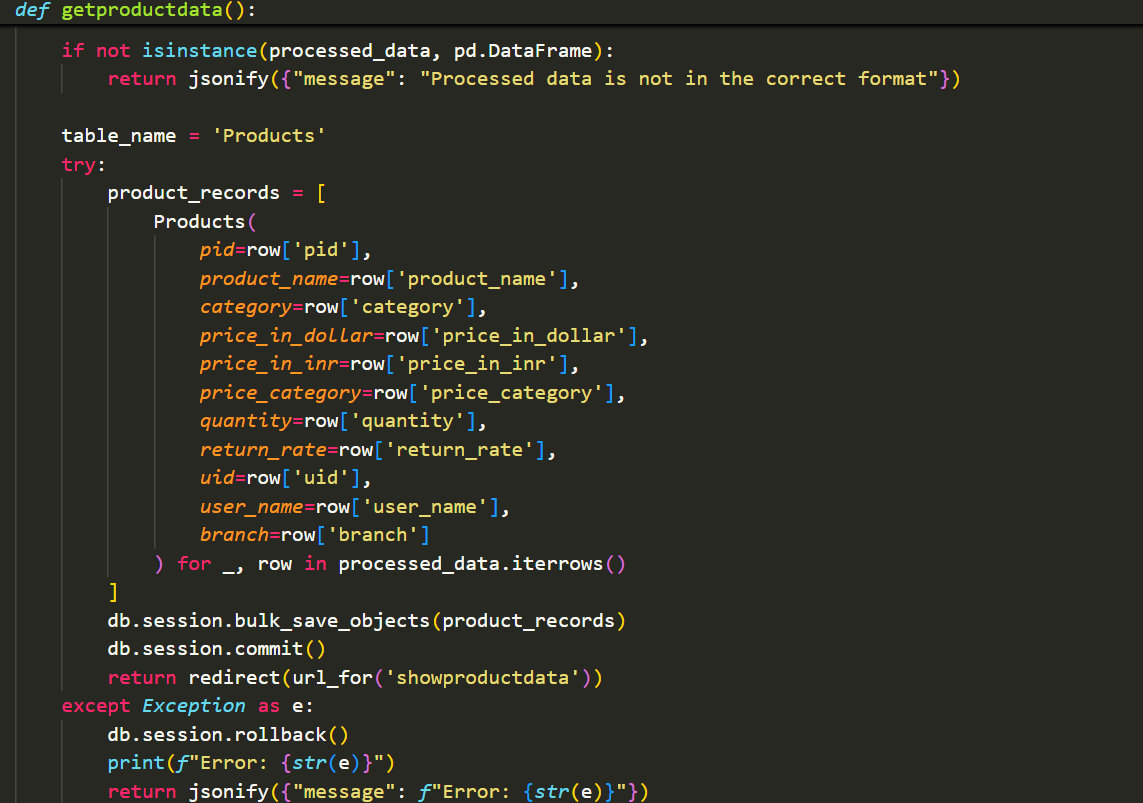
1. Data Validation (3 Marks): Implement five data validation checks for data quality, format checks, range checks, and handling missing values.

****

2. Data Transformation Logic (3 Marks): Apply at least five transformations (e.g., formatting, aggregations, joining datasets).



3.Data Loading (3 Marks): Efficiently load data into the database using ORM with data integrity.



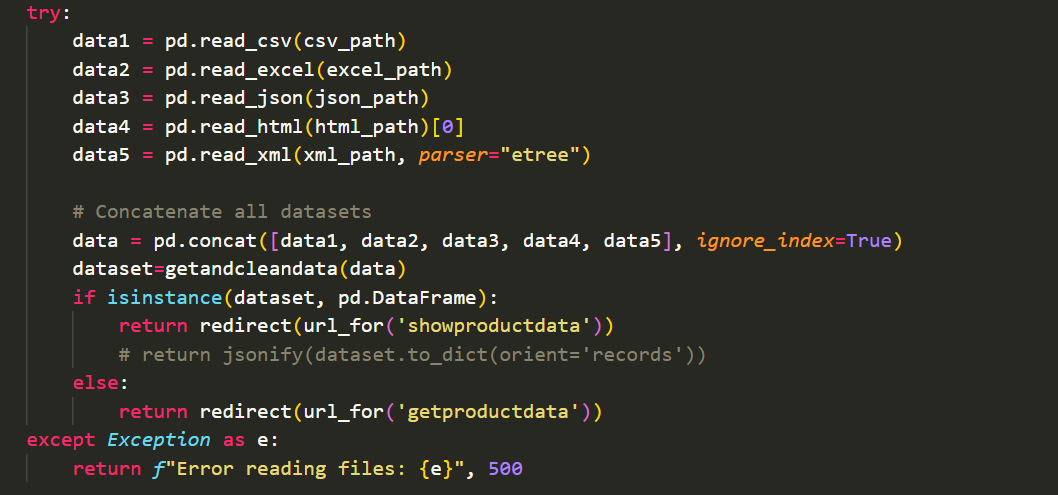
4.Pipeline Automation (3 Marks): Automate the ETL pipeline and document each stage with clear steps.

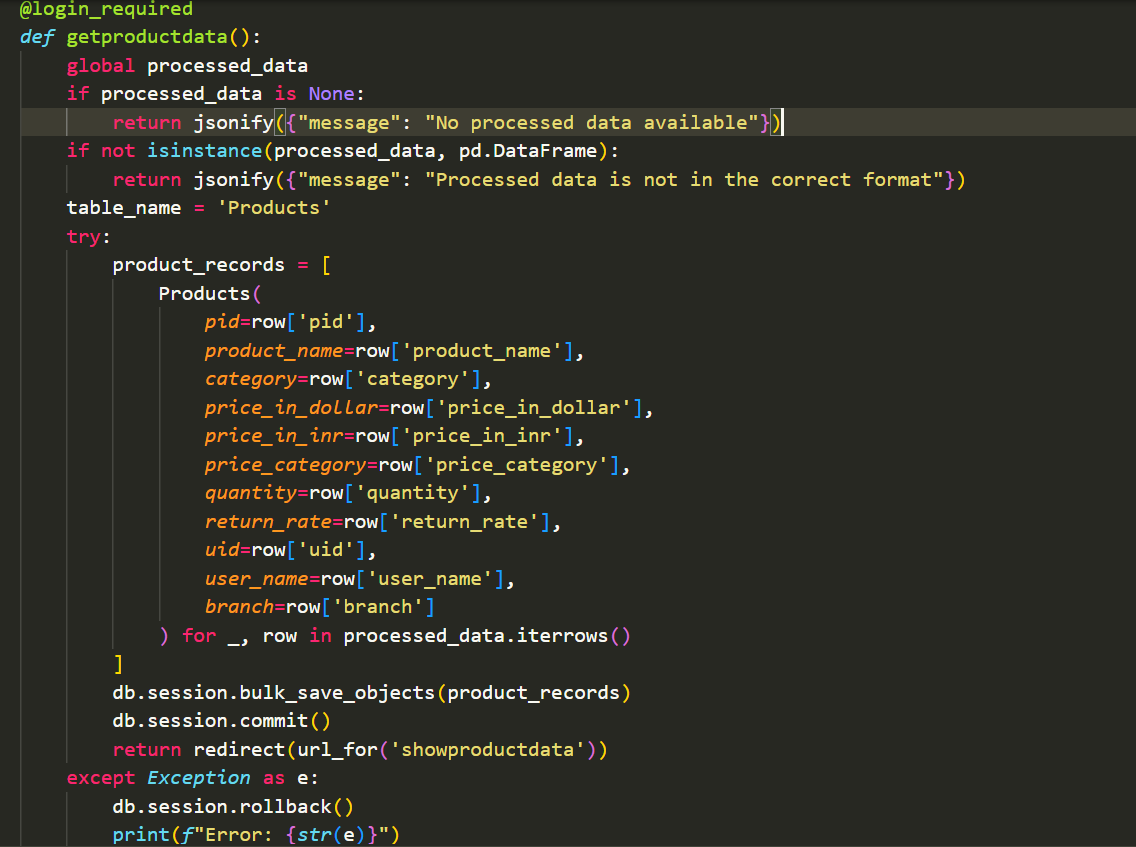
EXTRACTION: is done from extracting 5 different formats from 5 different sources

TRANSFER: transferring data from files concatenating from different sources

LOAD: load data to database

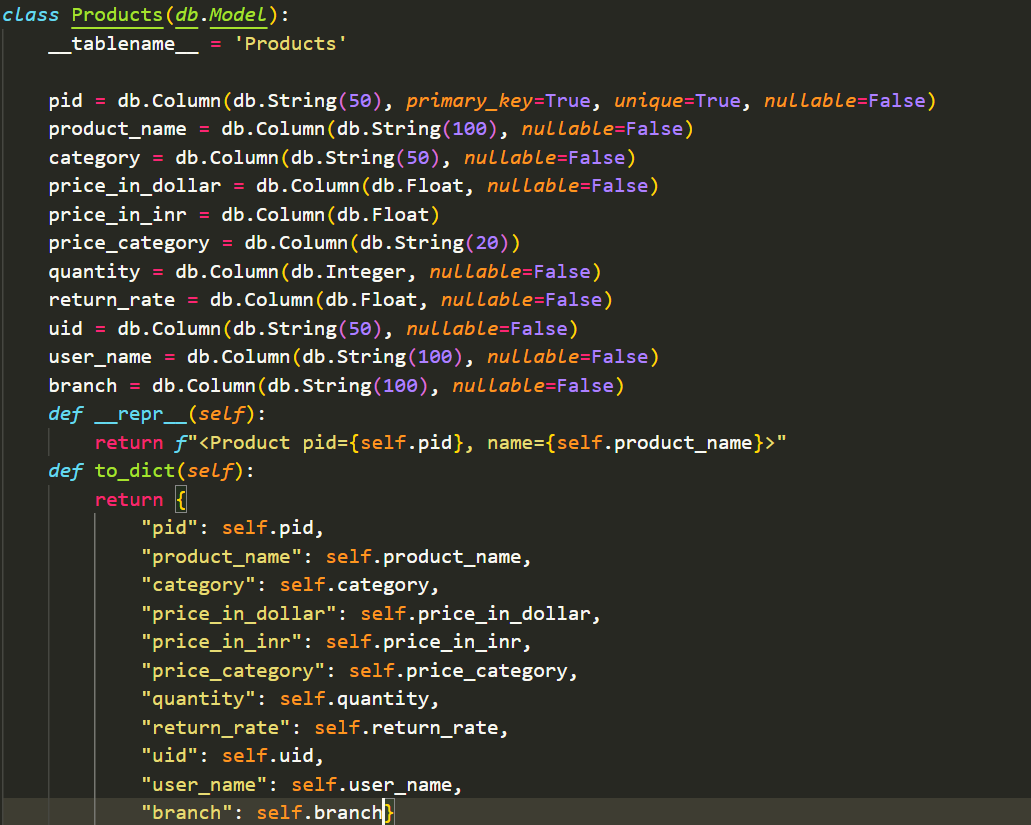
5.Exception Handling (3 Marks): Handle errors gracefully at every stage of the ETL pipeline.

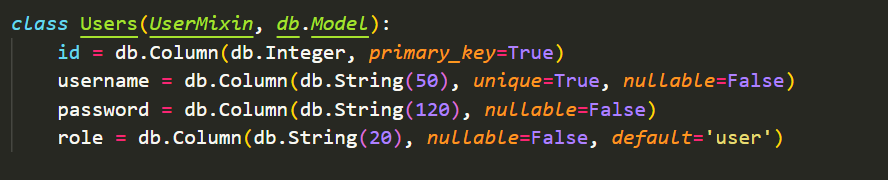
****

****

**Database Management and ORM (10 Marks)**

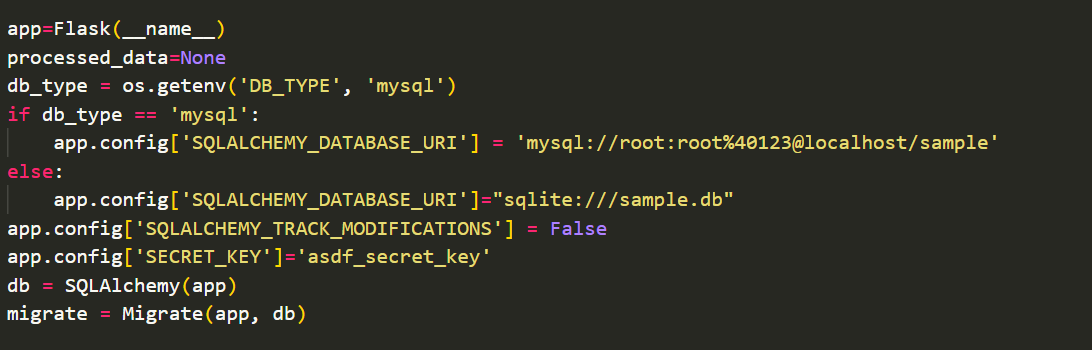
1.Database Schema Design (2 Marks): Design an optimized schema to accommodate ETL needs.

****

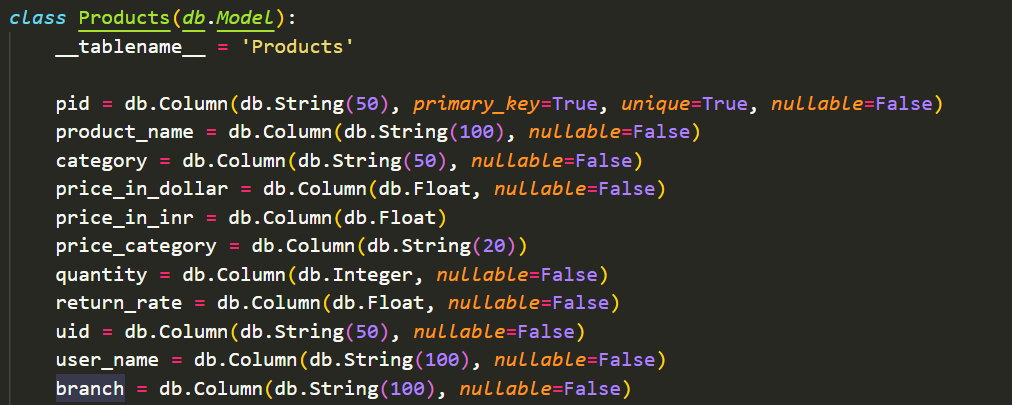
****

2.ORM and Multi-database Switching (3 Marks): Set up ORM models and demonstrate switching databases (e.g., SQLite, MySQL) during the final demo.

Switching databases between SQLite, MySQL

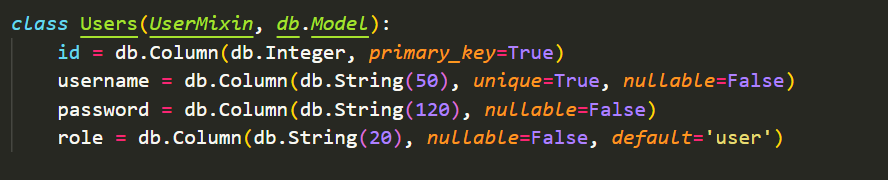


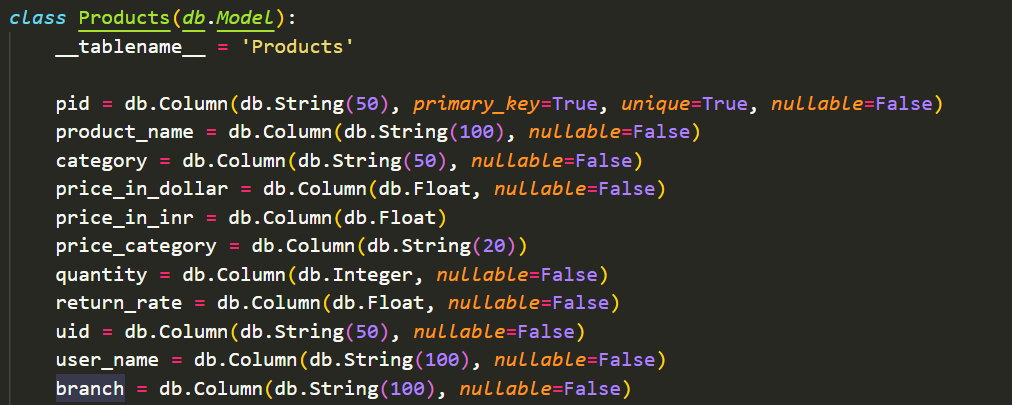
3.Data Integrity Constraints (2 Marks): Define integrity constraints (e.g., primary keys, foreign keys).

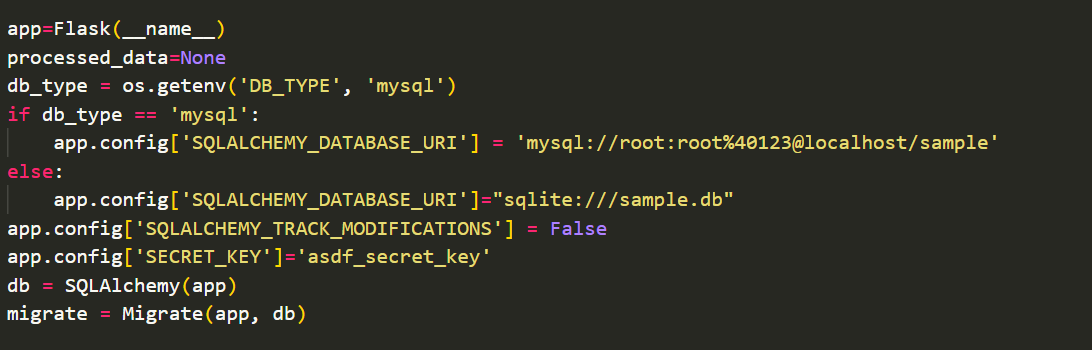


4. Documentation and Code Quality (3 Marks): Include clear explanations for schema, ORM, and database switching code

**Object-Relational Mapping**





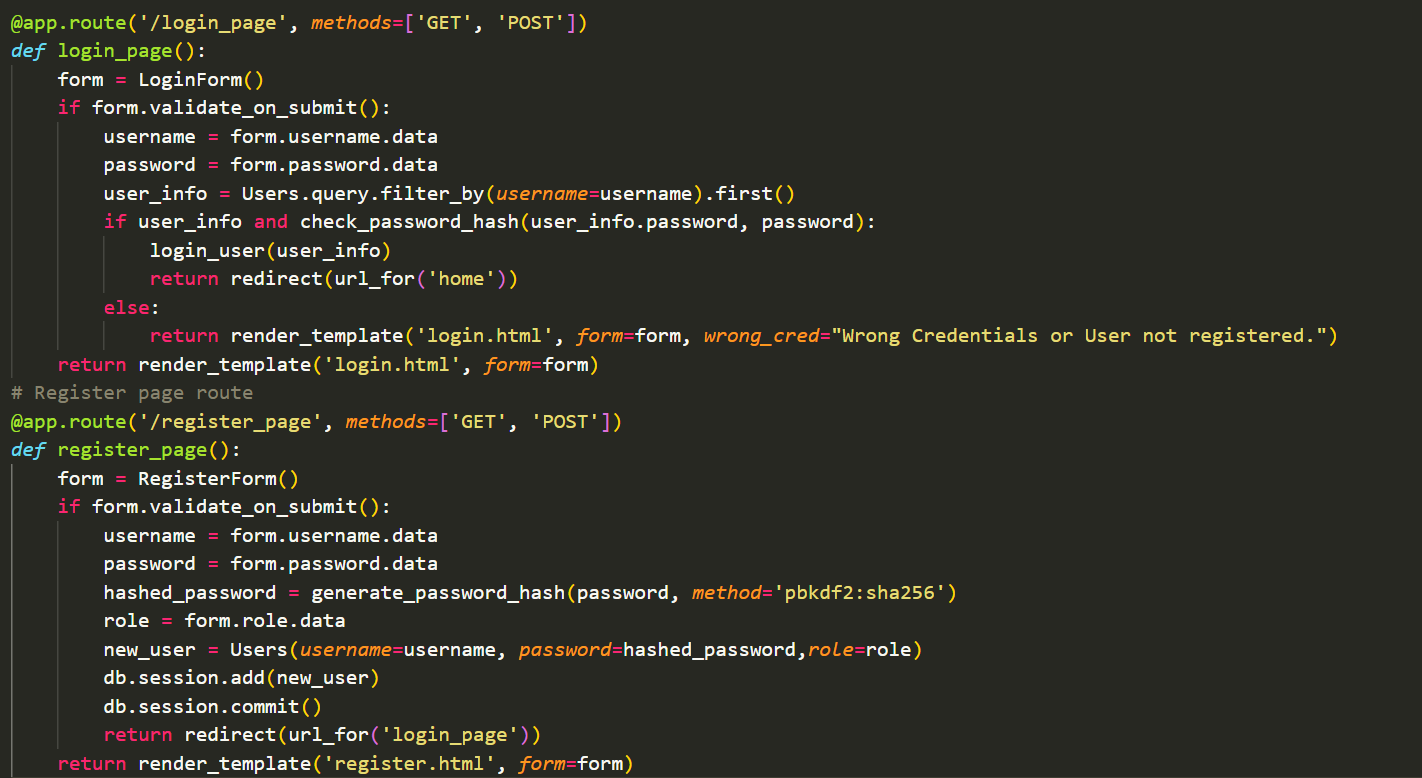


**User Management and Authentication (10 Marks)**

Outputs are shown at the end

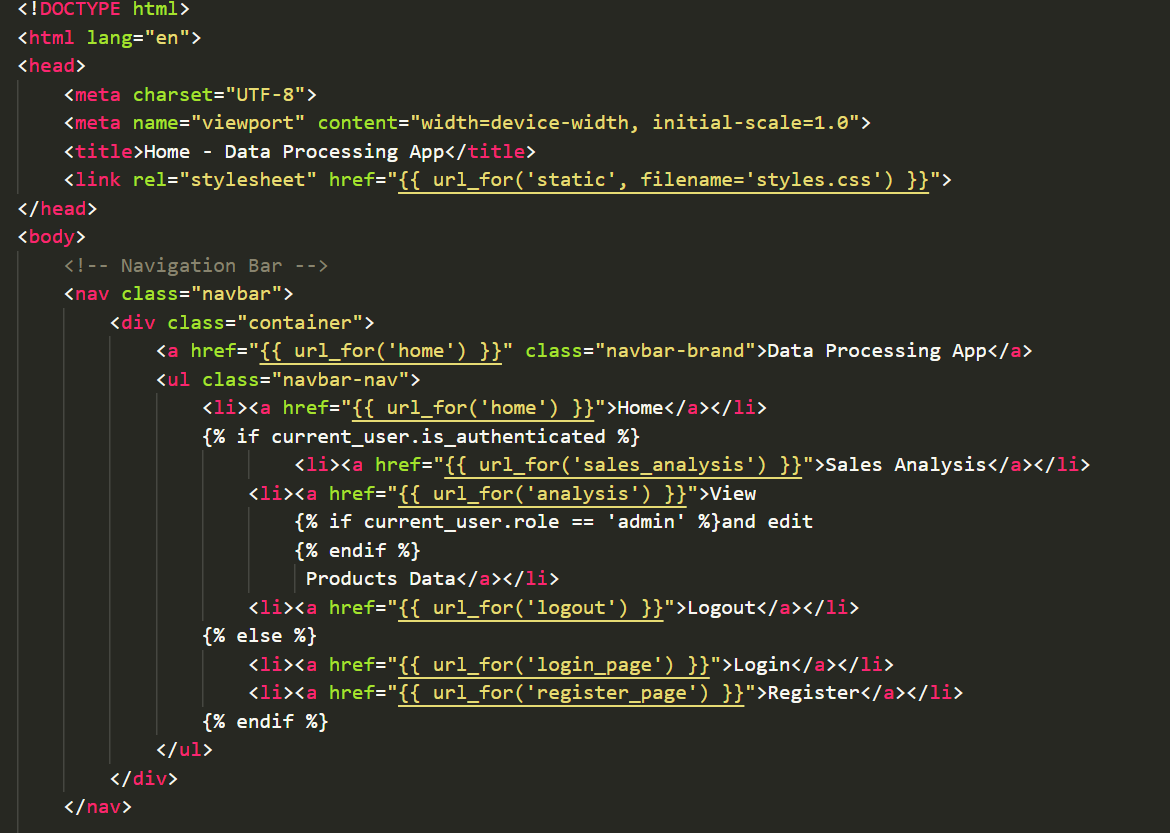
1.User Registration and Authentication (2 Marks): Set up secure user registration, login, and logout





2. Role-based Access Control (3 Marks): Implement distinct user roles with appropriate access levels (e.g., admin, analyst, viewer).

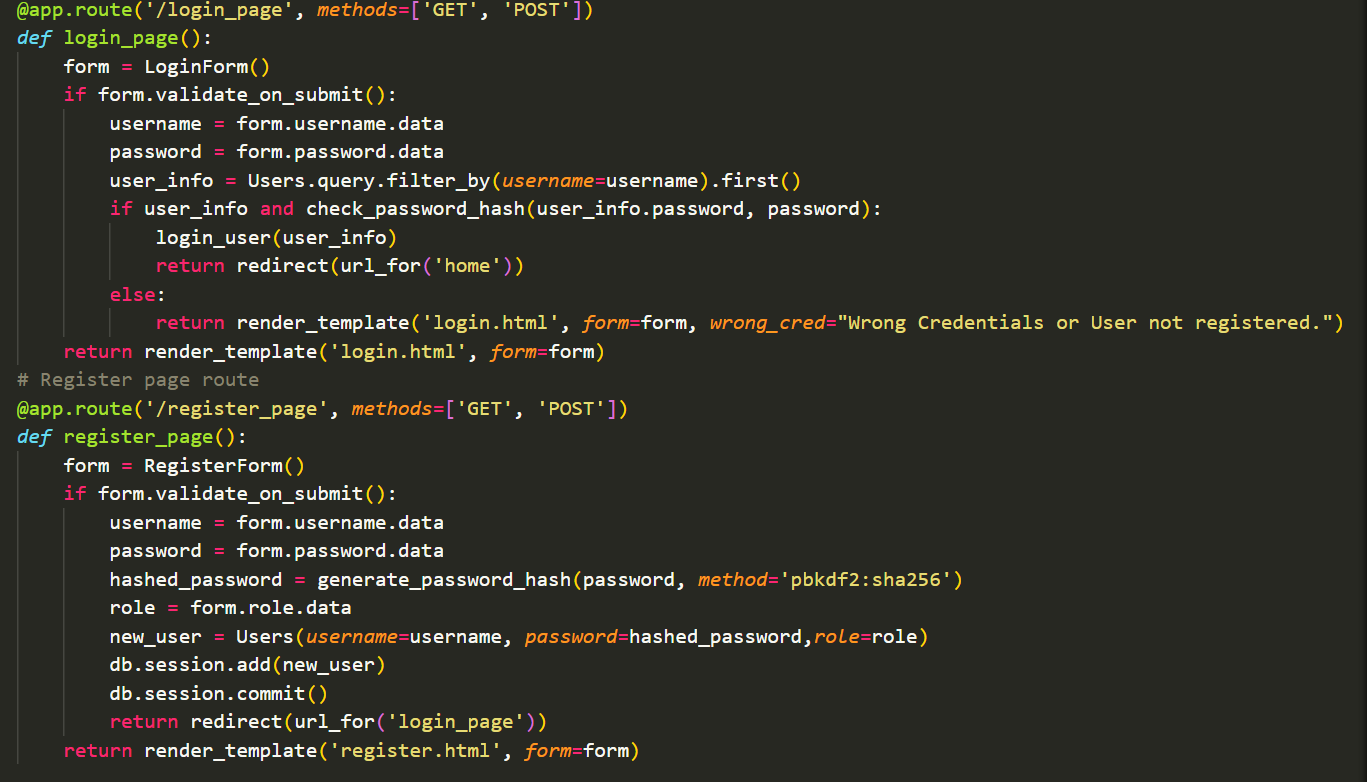




3.Form Validation and Security (2 Marks): Use Flask-WTF for form validation and secure data handling.

And

4.Session Management (2 Marks): Ensure secure session handling and expiration for logged-in users.



5.Documentation of Roles and Access (1 Mark): Describe roles and associated permissions.

There are two types of roles one is the user another is the admin

Where user can only see the sales analysis and see the data, where as the admin can see and delete the data also

**Data Visualization and Dashboard Creation (15 Marks)**

Are presented at the end with insights

**MVC (Model-View-Controller) Architecture (5 Marks)**

**Users products.html**

**Products sales\_analysis.html**

**login.html**

**Model compute data register.html**

**user input Template**

**Design and updates display data**

**'/login\_page',**

**'/register\_page', '/delete\_product/<pid>',**

**'/getproductdata', '/showproductdata'**

**"/analysis","/sales\_analysis"**

**( App.py)**

**View**

**Flask Skills and Code Optimization (10 Marks)**

1.Flask Core Concepts (3 Marks): Show a good understanding of Flask by using Blueprints, URL routing, and request handling.

@app.route("/", *methods*=['GET', 'POST'])

@app.route('/login\_page', *methods*=['GET', 'POST'])

@app.route('/register\_page', *methods*=['GET', 'POST'])

@app.route('/delete\_product/<pid>', *methods*=['POST'])

@app.route('/register', *methods*=['GET', 'POST'])

@app.route('/logout')

@app.route('/getproductdata', *methods*=['GET'])

@app.route('/showproductdata', *methods*=['GET'])

@app.route("/analysis",*methods*=['GET','POST'])

@app.route("/sales\_analysis")

2.Code Optimization (3 Marks): Implement optimized code with efficient querying, looping, and logic to reduce redundancy.

   . total\_sales = sum([product.price\_in\_inr \* product.quantity for product in sales\_data])

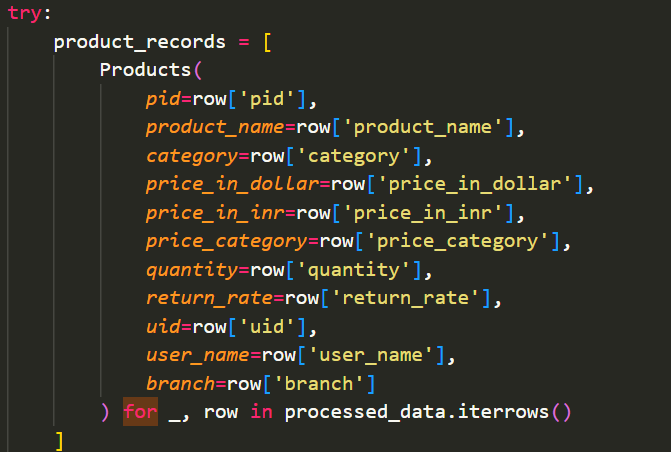
. category\_sales = {}

    for product in sales\_data:

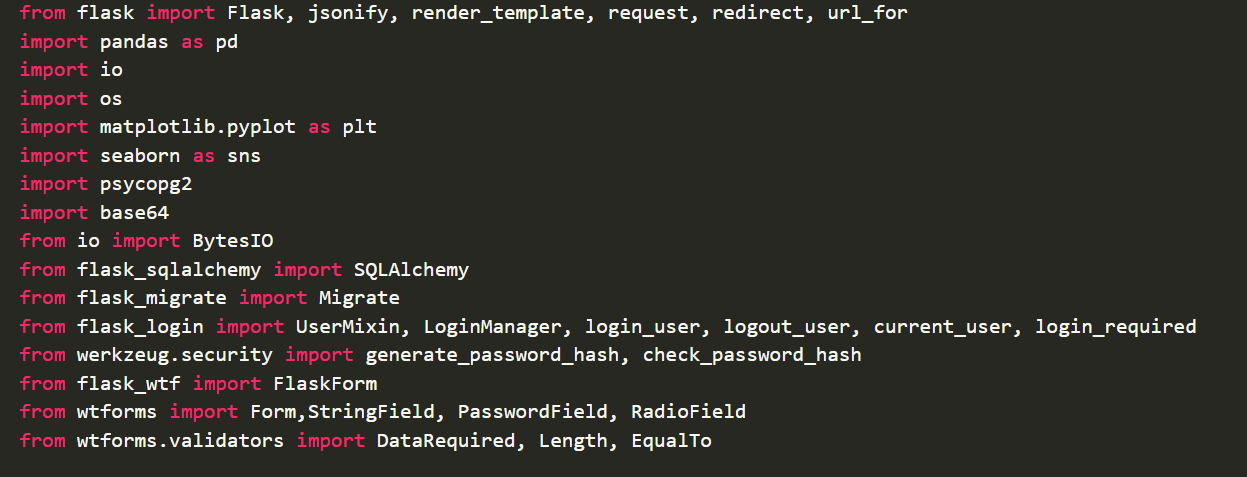
        if product.category not in category\_sales:

            category\_sales[product.category] = 0

        category\_sales[product.category] += product.price\_in\_inr \* product.quantity

. 

3. Library and Package Usage (2 Marks): Use suitable libraries (e.g., Pandas for ETL, SQLAlchemy for ORM) effectively.

****

4. Code Readability and Comments (2 Marks): Ensure readable code with meaningful comments and naming conventions.

Overall code has the required comments and good code readability with meaningful names

**Dynamic Form Handling with Flask-WTF (5 Marks)**

Form Validation (2 Marks): Use Flask-WTF to validate forms, handling all field types effectively.

Dynamic Input Handling (2 Marks): Allow dynamic form handling based on user input or role.

Error Messaging (1 Mark): Implement clear error messages for validation issues.

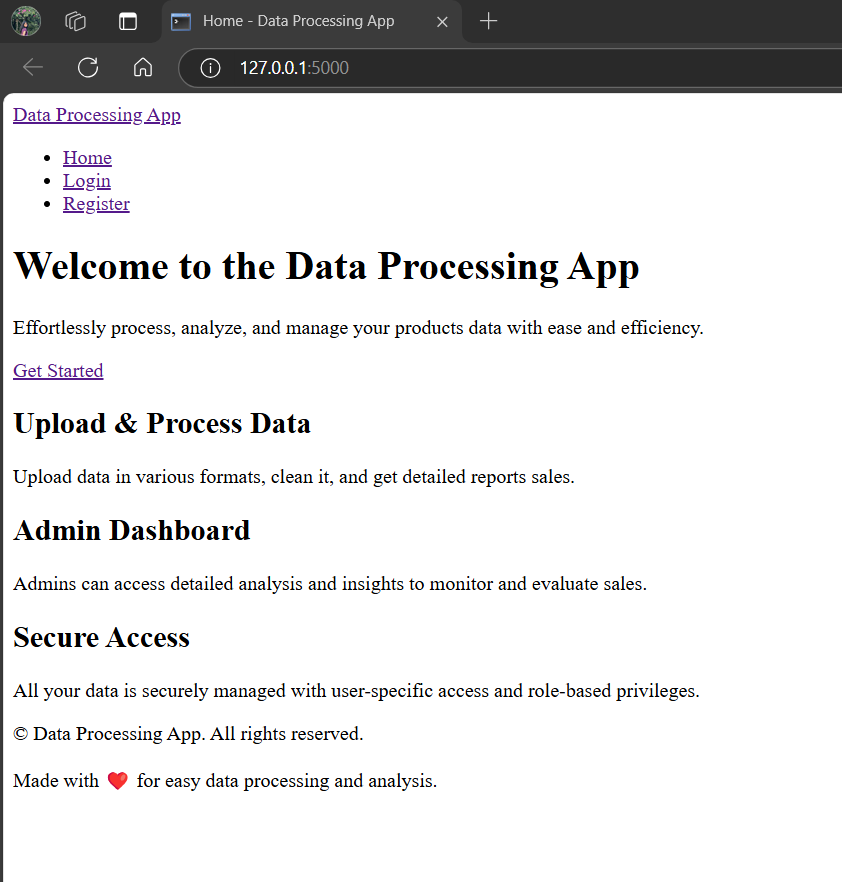
**Presentation and Demo (10 Marks) and Data Model and Design (5 Marks) and Final Deliverables and Evaluation (5 Marks)**

Will be considered as this whole document

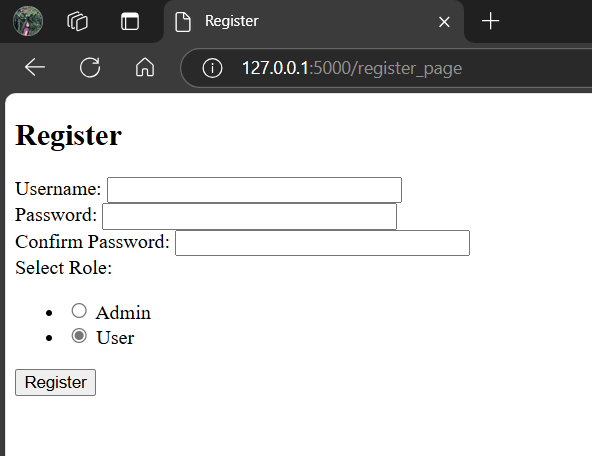
Database Switching Demo ,Project Presentation, Screenshots and Documentation, Self-reflection

Complete Working Code, Output Screenshot Accuracy------all these are already mentioned above

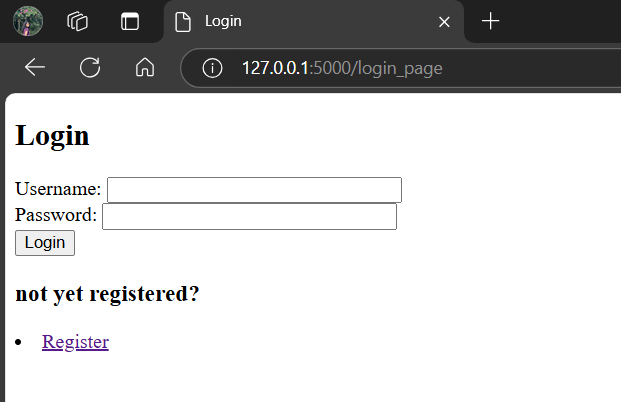
‘/’



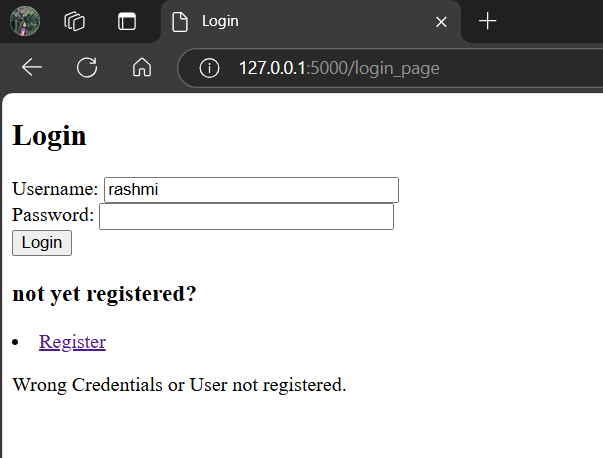
'/register\_page'



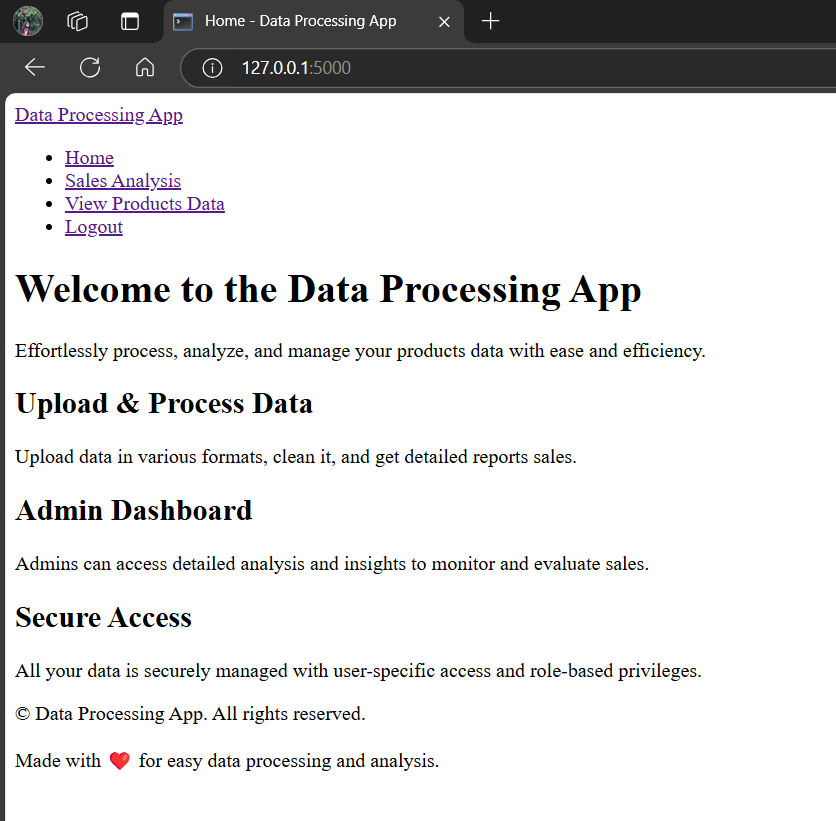
'/login\_page'



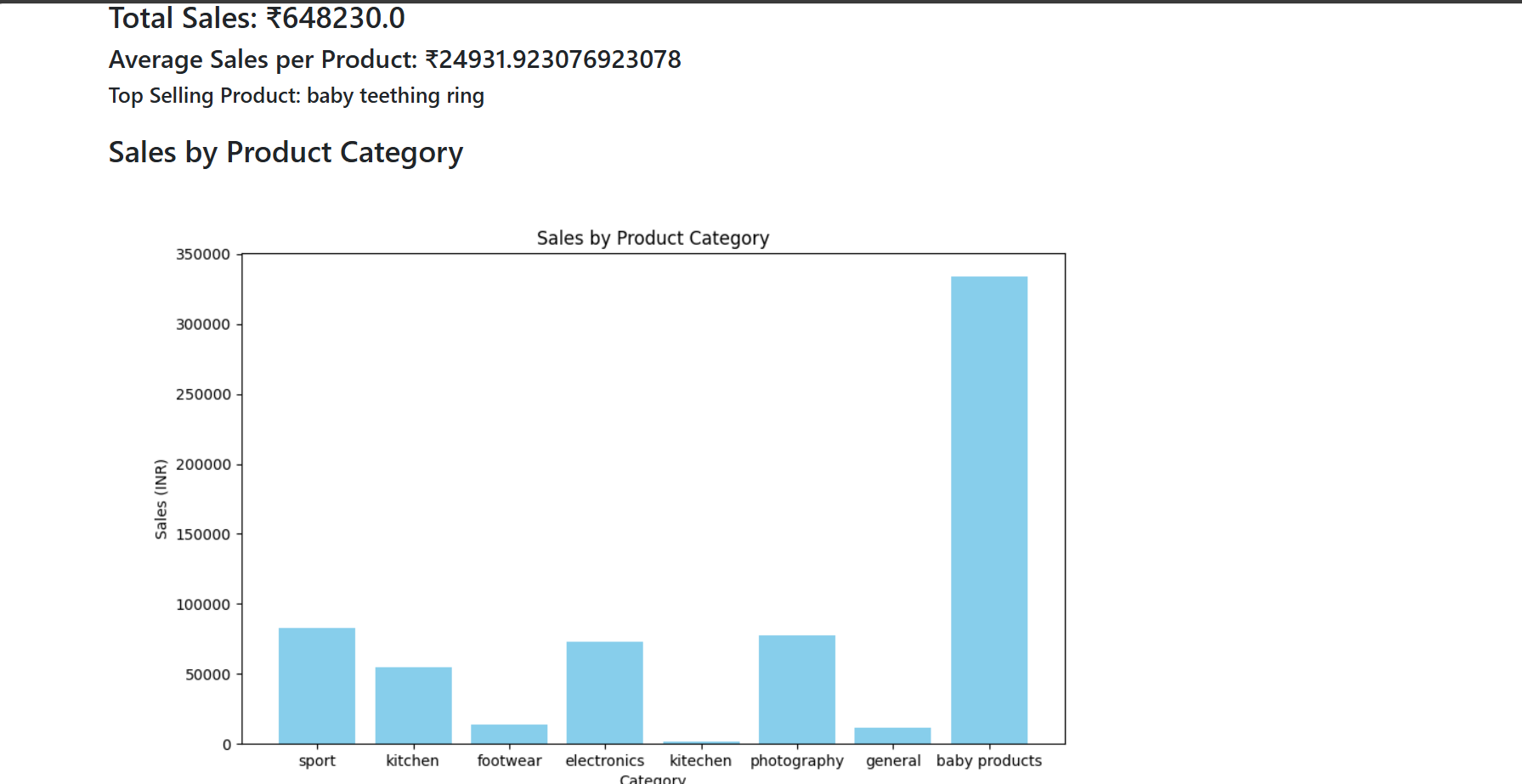
For invalid username and password

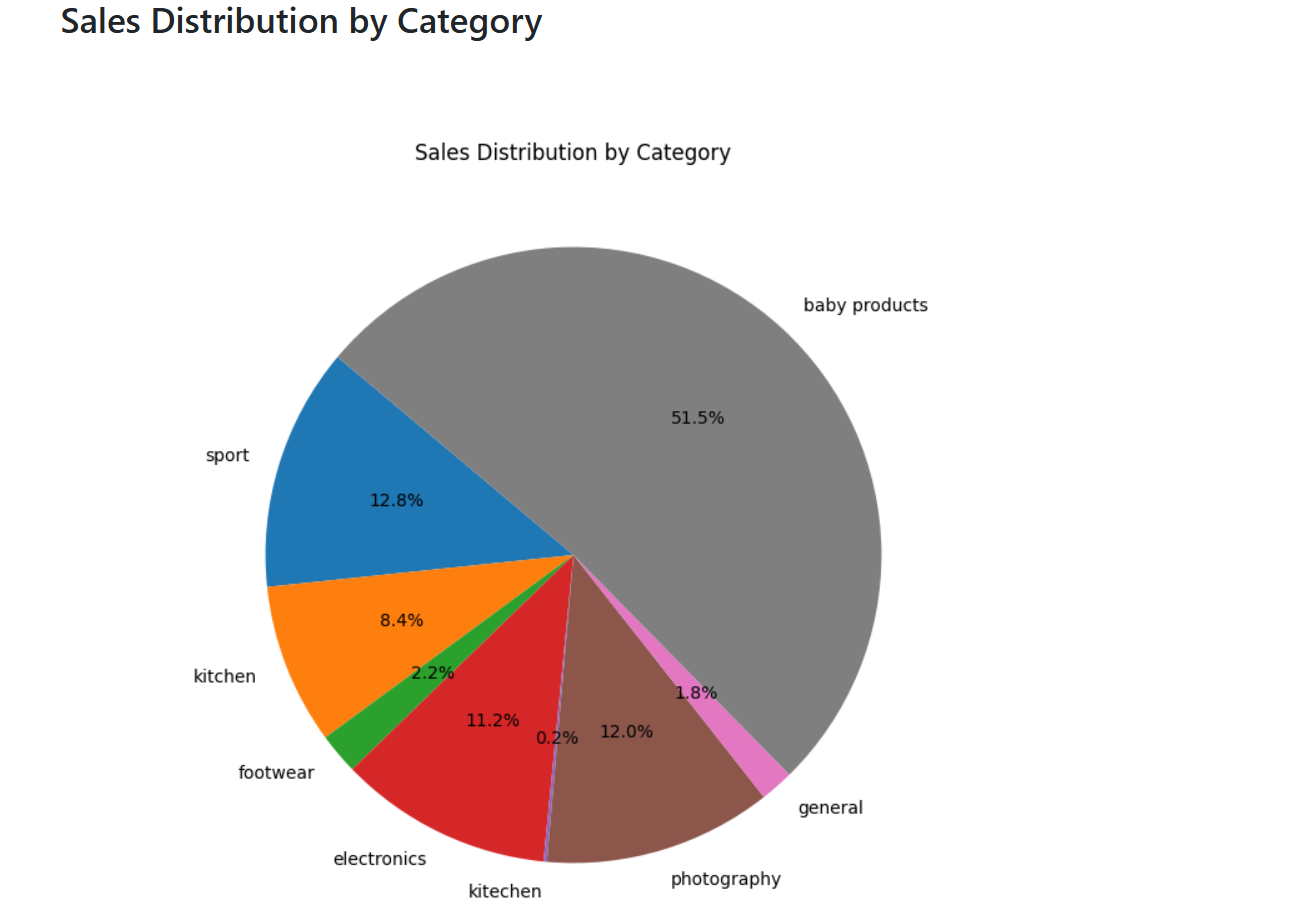


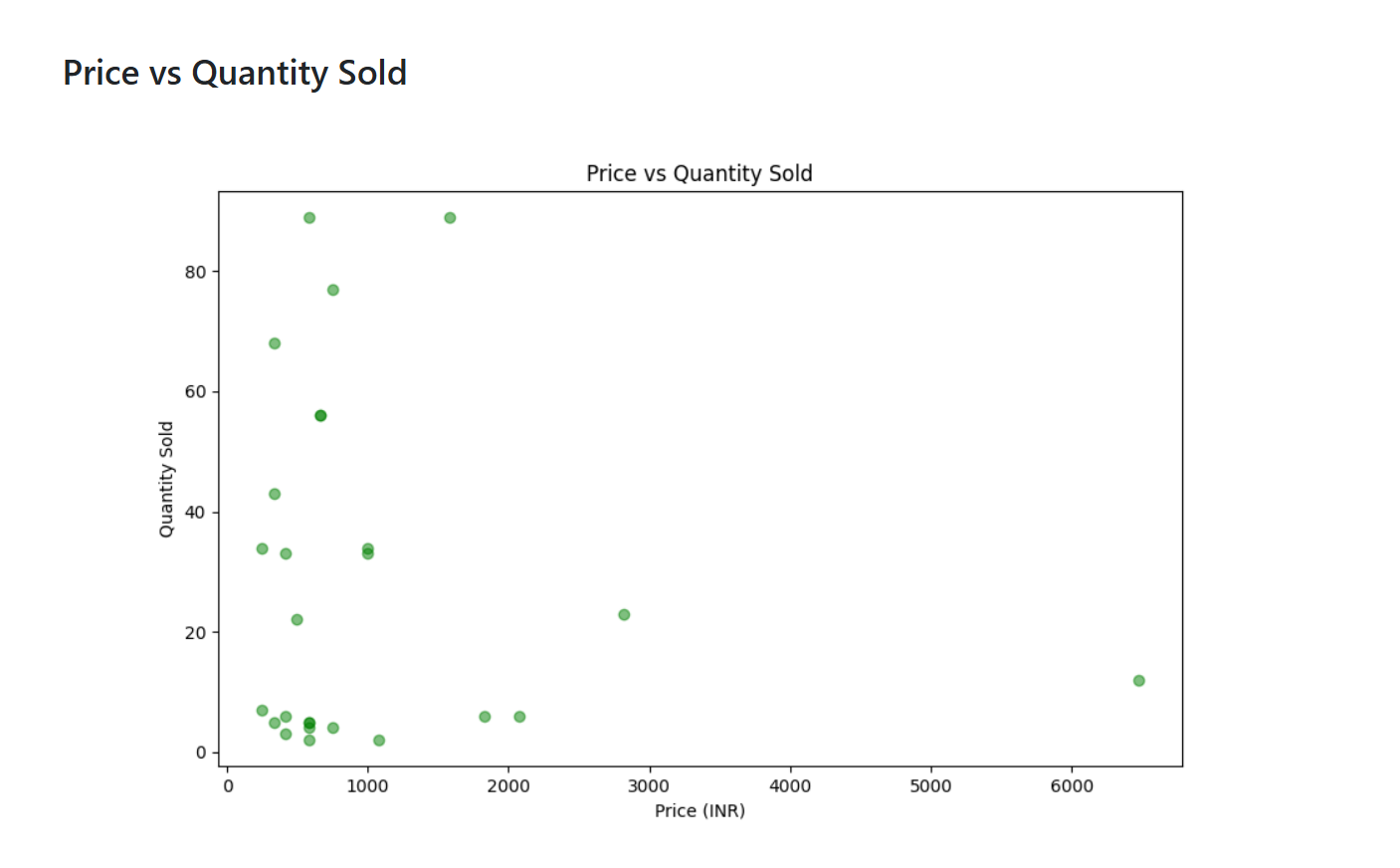
USER LOGIN



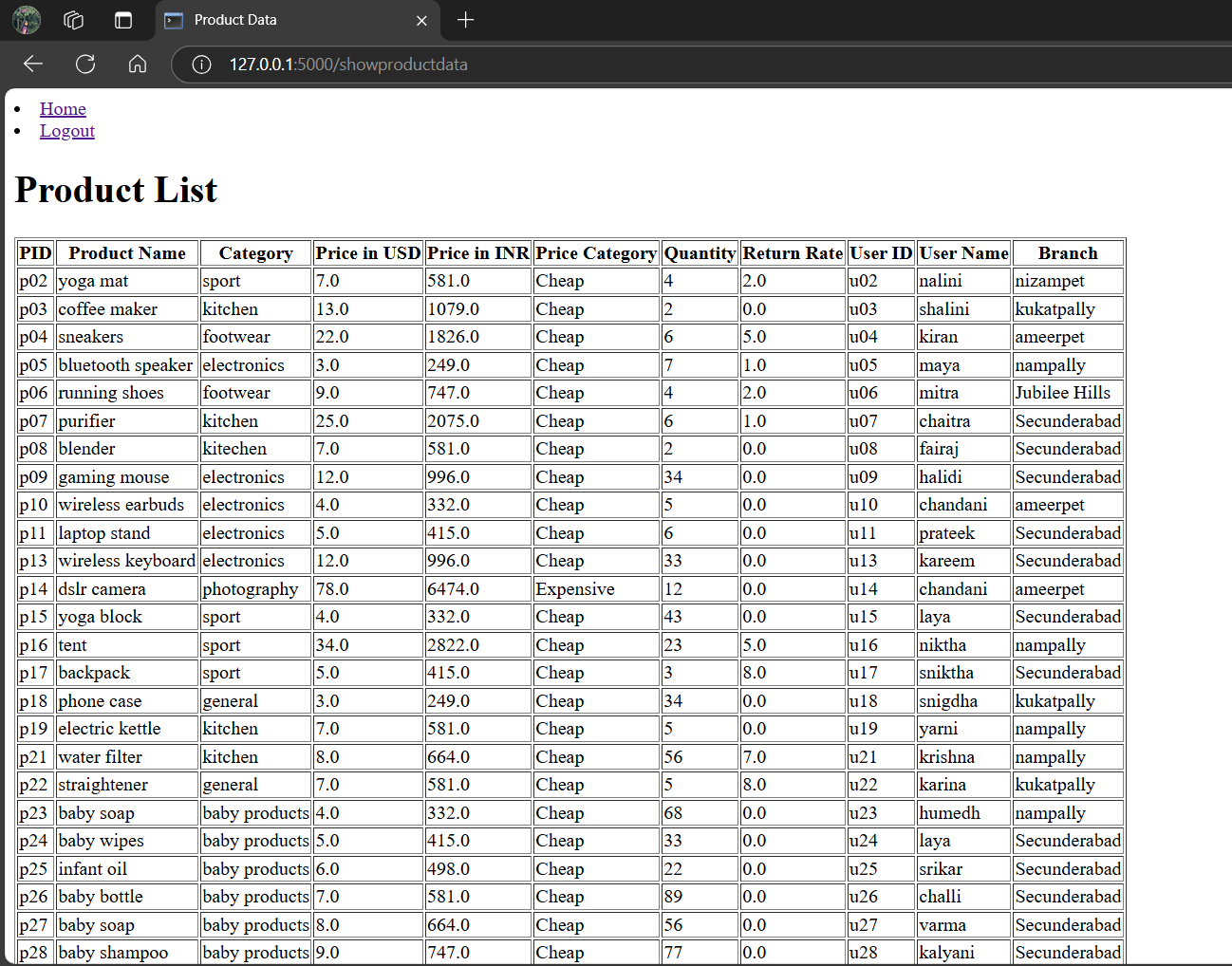
Sales analysis same for user and admin



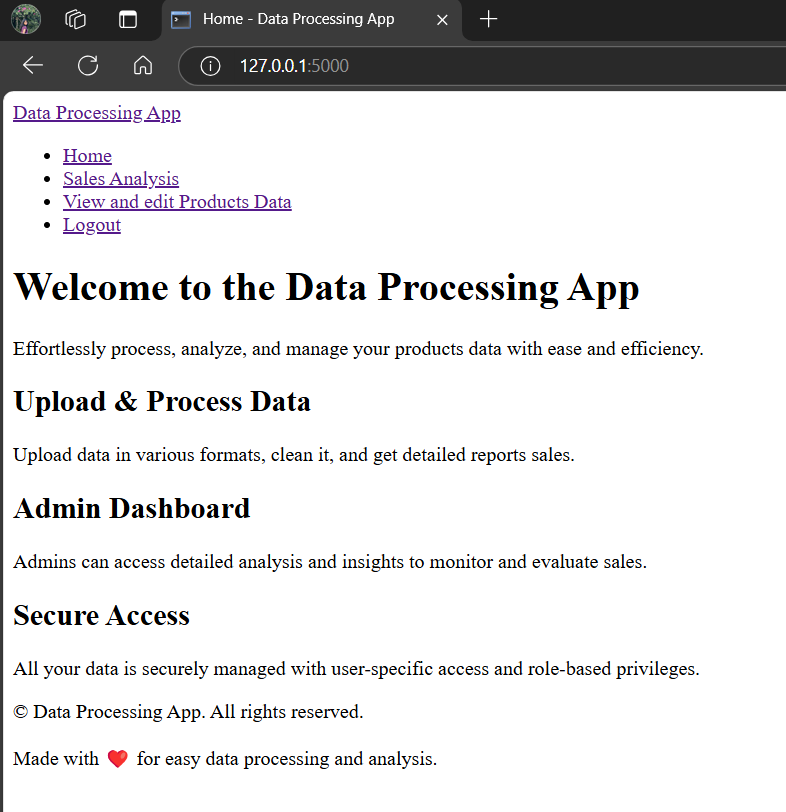




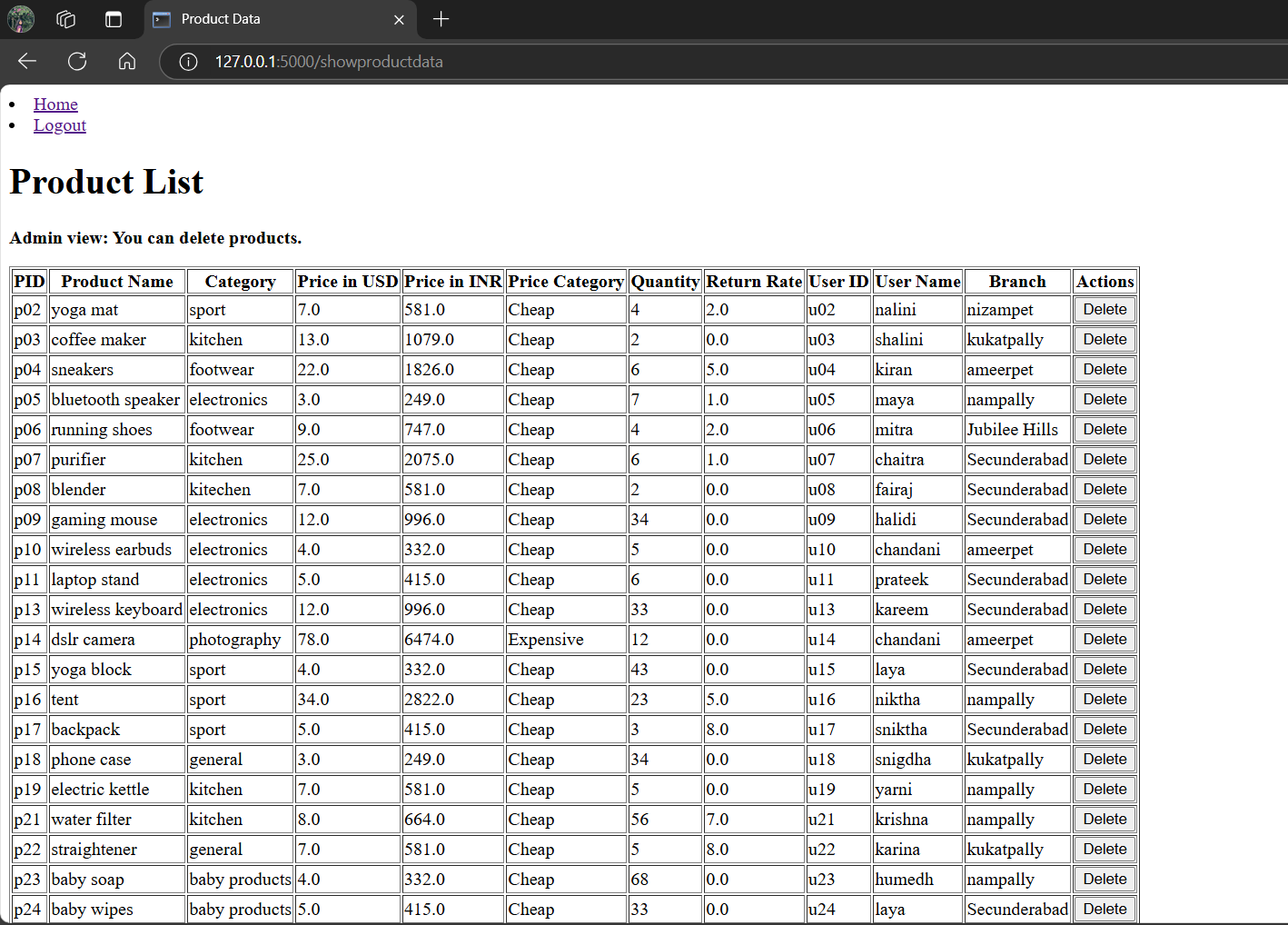
View products details

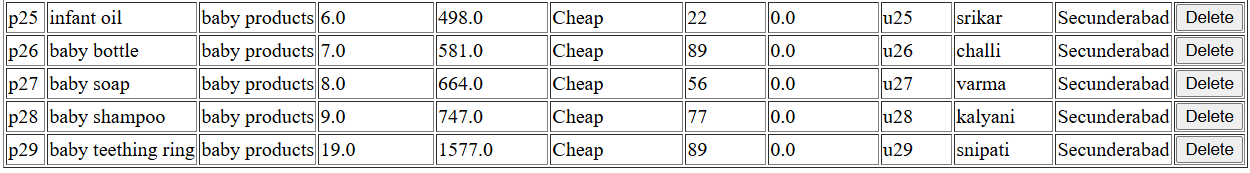


ADMIN LOGIN



View and edit product data





**INSIGHTS:**

Baby products are sold in high amount at Secunderabad, so to increase sale variety of baby products toys could be sold well.

From price vs quantity graph: Low cost items are taken in large quantity when compared with high cost items.

Majority of sales products are baby products.

High demand product is baby teething ring.

Theres only one expensive thing—dslr camera.

Return rate is higher for backpack and straightener so change in manufacturer would help and change in brand.