**Database Design and Analysis for TVET College Ecommerce Application Technical Report**

**Introduction**

The purpose of this report is to outline the database design and analysis for an ecommerce application developed for a TVET college. The application aims to provide online services and products to students, staff, and guests across multiple locations within Machakos County. The report discusses the database schema, data flow process, security considerations, and transactional management requirements.

**Database Design**

The database design for the TVET college ecommerce application is structured to efficiently manage various entities such as locations, facilities (cafeteria, tuck shop, pharmacy, bookshop, digital center), products, users, and orders. The normalized schema ensures data integrity and reduces redundancy. Here's a brief overview of the database schema:

* **Locations:** Stores information about different college locations.
* **Facilities (Cafeteria, Tuck Shop, Pharmacy, Bookshop, Digital Center):** Each facility table contains details specific to its type, linked to the Locations table.
* **Products:** Contains information about products available in different facilities.
* **Users:** Stores user credentials and roles.
* **Orders:** Records orders placed by users.

**Data Flow Process**

The data flow process within the application involves various steps such as user authentication, product browsing, order placement, and transaction processing. Here's a simplified overview:

1. **User Authentication:** Users authenticate using their credentials (username and password).
2. **Product Browsing:** Authenticated users can browse products available in different facilities.
3. **Order Placement:** Users select products and place orders, specifying the quantity.
4. **Transaction Processing:** Orders are processed, and transaction records are stored in the database.

**Security Considerations**

PostgreSQL provides robust security features to protect the database from unauthorized access and ensure data confidentiality and integrity. Some security considerations include:

* **Authentication:** Use strong password policies and authentication mechanisms to verify user identity.
* **Authorization:** Implement role-based access control (RBAC) to restrict access to sensitive data and functionalities.
* **Encryption:** Utilize SSL/TLS encryption for secure communication between the application and the database.
* **Parameterized Queries:** Use parameterized queries or prepared statements to prevent SQL injection attacks.
* **Database Auditing:** Enable database auditing to track and monitor user activities and detect suspicious behavior.

**Transactional Management**

The application must ensure transactional integrity to meet the business requirements of the TVET college. Transactional management involves ACID (Atomicity, Consistency, Isolation, Durability) properties to ensure reliable and consistent data processing. Key considerations include:

* **Atomicity:** Transactions should be atomic, ensuring that all database operations within a transaction are either completed successfully or rolled back if an error occurs.
* **Consistency:** Transactions should maintain data consistency by enforcing integrity constraints and business rules.
* **Isolation:** Transactions should be isolated from concurrent transactions to prevent interference and maintain data integrity.
* **Durability:** Completed transactions should be durable, ensuring that changes are permanently stored even in the event of system failure.

**Conclusion**

The database design and analysis presented in this report provide a solid foundation for the development of the TVET college ecommerce application. By implementing appropriate security measures and transactional management techniques, the application can ensure data protection, integrity, and reliability, meeting the business requirements and enhancing the user experience

**References**

* PostgreSQL Documentation: https://www.postgresql.org/docs/
* Flask Documentation: https://flask.palletsprojects.com/en/2.0.x/
* OWASP SQL Injection Prevention Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/SQL\_Injection\_Prevention\_Cheat\_Sheet.html

**Project Set Up**

Step 1. Install PostGre and import the attaches sql file.

Step 2 Install python and install pip

Step 3 Activate virtual environment .venv\Scripts\activate

Step 4. Install flask and pip install Flask Flask-SQLAlchemy Flask-JWT-Extended psycopg2

Step 4. pip install -U flask-cors Install CORS package

Step 5 Download Postman