Architecture

Tuesday, November 12, 2024 7:25 PM

https://chatgpt.com/share/6733ab7c-c258-800d-887d-6d06a620e805

1. Architecture Components (Including Payment API Integration)

1. Presentation Layer (Tkinter or Django)

- **Technology**: Django(Python GUI Library)
- **Purpose**: Provides the graphical user interface (GUI) for the e-commerce platform, allowing customers to interact with the system.
- **Features**:
  - **Product Catalog**: Users can browse and search products.
  - **Shopping Cart**: Users can manage items in their cart.
  - **Checkout**: Users can select payment methods and complete purchases.
  - **User Management**: Registration, login, and account management.
  - **Payment Processing**: Collect payment details and trigger API calls for payment authorization.

2. Business Logic Layer (Flask)

- **Technology**: Flask (Python Web Framework)
- **Purpose**: Manages core business logic, handles API requests, and interacts with databases and external payment services.
- **Components**:
  - **User Authentication**: Manages user login and registration securely.
  - **Order Management**: Validates and processes orders, calculates totals, and updates order status.
  - **Loyalty Program**: Calculates reward points and manages voucher redemptions.
  - **Payment Processing**: Integrates with MB WAY and credit card payment APIs.
  - **Analytics & Monitoring**: Collects data on user interactions and provides insights.

Data Layer

- **Primary Database (Relational - Oracle)**
  - **Purpose**: Manages structured data, including user accounts, products, orders, and payment records.
- **Secondary Database (Non-Relational - MongoDB)**
  - **Purpose**: Stores semi-structured data, such as user interactions and feedback.

4. Workflow for Payment Processing

1. **Checkout Process in Tkinter**:
   - Users select items and proceed to checkout.
   - They choose between MB WAY and credit card payment options.
   - Tkinter collects payment details and makes API requests to Flask.
2. **Flask Handles Payment API Calls**:
   - **MB WAY**: Flask sends a payment request to the MB WAY API using the provided phone number and order amount. It then checks the payment status and updates the order accordingly.
   - **Credit Card**: Flask sends credit card details to a payment gateway securely. The payment status is checked, and the order is updated.

3. **Order Status Update**:
    - If the payment is successful, the order status is updated to "Accepted" in the Oracle database.
    - If the payment fails, the user is notified, and the order status is set to "Suspended."
4. **Data Storage**:
    - **Payment Records**: Stored in the Oracle database for transaction history and auditing.
    - **User Interaction Logs**: Stored in MongoDB for analytics.

5. Security Considerations

- **Data Encryption**: Ensure sensitive data, such as credit card details, are encrypted and never stored directly in the database.
- **SSL/TLS**: Use secure communication protocols for all API calls to payment gateways.
- **PCI Compliance**: If dealing with credit card information, ensure the application follows PCI DSS standards.

6. Technology Stack Summary

- **Frontend**: Tkinter for GUI
- **Backend**: Flask for API and business logic
- **Databases**: Oracle (relational) and MongoDB (non-relational)
- **Payment Gateways**: Integration with MB WAY and credit card payment providers

Tables

17 de setembro de 2024 08:46

1. Warehouse - SQL

- **Attributes**:
  - warehouse_id (Primary Key)
  - name
  - address
  - location (WGS84 coordinates)

2. Zone SQL

- **Attributes**:
  - zone_id (Primary Key)
  - warehouse_id (Foreign Key referencing Warehouse)
  - Capacity

3. Item SQL

- **Attributes**:
  - item_id (Primary Key)
  - name
  - description
  - brand
  - type (Product or Service)
  - primary_supplier_id (Foreign Key referencing Supplier)
  - purchase_price
  - sales_price

4. Product (For item type "product") SQL

- **Attributes**:
  - product_id (Foreign Key referencing Item)
  - Warehouse_id (Foreign Key to warehouse) **Justification:**
    it includes information on the quantity in stock and the minimum
    required stock levels, which vary by warehouse.
  - quantity_in_stock
  - minimum_stock (secondary key composed from primsry + warehouse_ID)
  - Price

  - category
  - Subcategory
  - Technical information (e.g., EAN, model) & physical atributes (e.g., color, weight, height) [JSON key]
  - pstart_date
  - end_date
  - is_current

5. Service (For item type "service") SQL

- **Attributes**:
  - service_id (Foreign Key referencing Item)
  - max_execution_hours
  - execution_time

- responsible_employee_id (Foreign Key referencing Employee)
- price
- start_date
- end_date
- is_valid

6. Customer SQL

- **Attributes**:
  - customer_id (Primary Key)
  - name
  - address
  - postal_code
  - nif
  - email
  - account_id (Primary Key)
  - password_hash
  - gdpr_terms (Text of GDPR terms accepted)
  - accepted_dateas
  - points_balance
  - last_points_redeemed_date
  - status (new, active, blocked, prohibited)

8. Order SQL

- **Attributes**:
  - order_id (Primary Key)
  - customer_id (Foreign Key referencing Customer)
  - delivery_address
  - status (Enum for in transit, delivered)
  - checkout_total
  - payment_status (Enum for accepted, suspended)
  - shipping_status (Enum for in transit, delivered)

9. OrderItem SQL

- **Attributes**:
  - order_item_id (Primary Key)
  - order_id (Foreign Key referencing Order)
  - item_id (Foreign Key referencing Item)
  - quantity
  - Price

10. Voucher SQL

- **Attributes**:
  - voucher_id (Primary Key)
  - customer_id (Foreign Key referencing Customer)
  - amount
  - valid_until

11. Rating NoSQL (MongoDB)

- **Attributes**:

- rating_id (Primary Key)
- customer_id (Foreign Key referencing Customer)
- item_id (Foreign Key referencing Item)
- rating (1-5 scale)
- Anexex(image/videos)
- comment

12. BrowsingHistory NoSQL (MongoDB)

- **Attributes**:
  - action_id (Primary Key)
  - customer_id (Foreign Key referencing Customer)
  - page
  - action
  - visit_date
  - visit_time

13. Supplier SQL

- **Attributes**:
  - supplier_id (Primary Key)
  - name
  - contact_info
  - best_selling_item_id (Foreign Key referencing Item)

14. Employee SQL

- **Attributes**:
  - employee_id (Primary Key)
  - Name
  - email
  - account_id (hexadecila AD000000)
  - password_hash
  - role

15. SupplierItems SQL

- **Attributes**:
  - supplier_item_id (Primary Key)
  - supplier_id (Foreign Key referencing Supplier)
  - item_id (Foreign Key referencing Item)
  - price (Price at which supplier sells the item)
  - availability (Boolean or Enum to track availability status)

Requirements

Thursday, November 14, 2024 2:19 PM

1. Unregistered User

- **US1**: As an unregistered user, I want to register in the system.

2. System User

- **US2**: As a system user, I want to view all the information about a particular product.

3. Customer

- **US3**: As a Customer, I want to buy items.
  - **Acceptance Criteria**:
    - The total amount in the shopping cart must not exceed €2000, which is the maximum billing limit for each cart.
    - Product ratings/comments should be available wherever possible.
- **US5**: As a Customer, I want to track the current status of a specific purchase.

4. Warehouse Manager

- **US6**: As a Warehouse Manager, I want to know which suppliers provide the best-selling items.
- **US7**: As a Warehouse Manager, I want to know which suppliers provide the best-selling items.
- **US8**: As a Warehouse Manager, I want to know the most-voted items and their suppliers.
- **US9**: As a Warehouse Manager, I want information about products that have reached their minimum stock level and the suppliers who supply these items.
- **US10**: As a Warehouse Manager, I want information about the warehouse aisles that currently store products with the highest number of purchase orders that received a discount greater than 20%.

5. Delivery Order Manager

- **US11**: A
- s a Delivery Order Manager, I want the system to provide the location of orders on a specific day and time.
- **US12**: As a Delivery Order Manager, I want to know the route taken by a particular order.

6. Manager

- **US13**: As a Manager, I want to know all the products purchased by the customer who used the highest number of vouchers purchased.
- **US14**: As a Manager, I want to know which purchases were made between June and August 17, with a preparation time of less than 10 hours and a delivery date more than 10 days after the purchase date.
- **US15**: As a Manager, I want information about the monthly purchases volume for products stored in warehouses where stock is at least 50% above the minimum, for the year 2018.

7. CIO (Chief Information Officer)

- **US16**: As a CIO, I want the system to keep track of product stock in real-time to provide accurate product availability information to customers.
- **US17**: As a CIO, I want to know the monthly sales trends for the 5 best-selling products over the past 6 months.
- **US18**: As a CIO, I want to know the number of site visits per day and week during the current year.

- **US19**: As a CIO, I want to know the geographic location of visitors and the most popular pages on the site.
- **US20**: As a CIO, I want to know which pages have the highest user abandonment rates and which pages have the most users clicking the help button.

B. Index Performance Analysis

- **Select two scenarios** to demonstrate when and under what circumstances indexes can or cannot improve query performance.
- **Justification**: Provide reasons to support the chosen scenarios.

Non-Functional Requirements

Thursday, November 14, 2024 2:25 PM

- **Scalability**:
  - The system must be capable of handling many customers simultaneously, especially during peak periods.
- **Security**:
  - Customer data and payment information must be stored securely, with encryption for sensitive data.
- **Availability**:
  - The system should be operational 24/7, ensuring high availability for purchases at any time.
- **Performance**:
  - Product searches and checkout processes should be fast, even when dealing with a large number of items.

Capacity estimations

Thursday, November 14, 2024 2:59 PM

Active Users: 100k per month, each searching 20 products per month

=100k/per_month * 20searches/per_month

=2000k searches / 30 * 24 * 60 *60

=2000k searches / 2592k seconds

=0.772 search/second. Giving a slack interval considering peak hours lets round to 10.

Scalability/Performance: Application should be able to handle 10 searches per second with low latency.

Table space:

*Items

TADBD - Oracle Masterclass

Friday, November 8, 2024 7:35 PM

*Full table scan, what is
-Till high-water-mark

Dive deep on Indexes.

Autonomous oracle databases.

Tables an fragemt as well

-Execution plan
-Indexes
-Materialized Views
-Partitioning(very important tool to optimize, along with INdexing)
-Parallel Exceuton

*local indexes

FrontEnd Views

Thursday, November 28, 2024 5:08 PM

-Home page - Catalog(customer and unregistered)- best selling items
-Login
-Register
-Product
-Cart
-Payment
-Account details:
    -Customer Account
    -Warehouse Manager window de consultas
    -Delivery Order Manager window de consultas
    -Manager window de consultas
    - CIO (Chief Information Officer) window de consultas

-

BackEnd Functions

Thursday, November 28, 2024 5:22 PM

-Home page - Catalog(customer and unregistered)best selling items

- -GetBestSellingProduct()
- -Filter()

-Login

- -Login()

-Register

- -Register()

-Product

	-GetProduct()
	-AddToCart()

-Cart

- removeCart()
- addCart()
- addVoucer()
- calculateCheckout()

-Payment

- -choosemethod
- -creditcard()
- -mbway()

-Account details:

- -Customer Account
  - Getcustomer
  - GetActiveOrders

GetOrders

For each query analisis have a materialized view and have x functions Get'MaterializedView()

-Warehouse Manager window de consultas
-Delivery Order Manager window de consultas
-Manager window de consultas
- CIO (Chief Information Officer) window de consultas