HCLTech | Supercharging Progress™

Retail Portal

Retail Website for Pizza & Cold Drinks

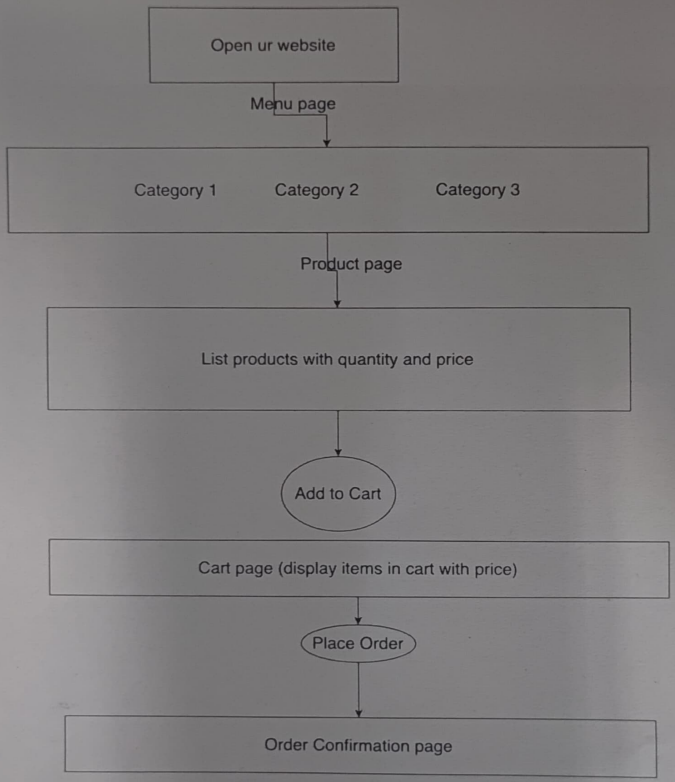
Hackathon

HCL

Business Use Case

- Develop a retail ordering portal for Pizza, Cold Drinks and Breads
- Customers can browse menu, add to cart, and place Carryout or Delivery orders
- Inventory decreases based on order placed
- Security on APIs
- REST endpoints should work from Postman (API tool)
- Code and design committed to GitHub repo
- Email confirmation with order number after successful order (Stretch)
- Order history for authenticated users (Stretch)

Flow


Open ur website
Menu page
Category 1
Category 2
Category 3
Product page
List products with quantity and price
Add to Cart
Cart page (display items in cart with price)
Place Order
Order Confirmation page

Scope for 5-Hour MVP

- GitHub repo with README
- Project design – DB Schema, Collections/Tables design, API design with request and response committed in Git hub
- Menu API: List pizzas & drinks with inventory
- Cart API: Create/update cart and calculate totals
- Order API: Place order, decrement inventory
- Secure endpoints via x-api-key in header
- Logging & Error Handling
- Functional CI/CD pipeline with basic automated tests
- Deployed frontend and backend applications
- Postman collection with working requests
- Order History API: List past orders (Stretch)
- Order confirmation email (Stretch)

Technical Requirements

- Frontend: ReactJS/NextJS
- Backend: Node.js + Express.js or Python FastAPI/ Java Spring Boot
- Database: No SQL Database of choice: MongoDB , Dynamo, Firestore
- API: RESTful API for communication between frontend and backend
- Security: Secure APIs by adding x-api-key in header
- Deployment: Local and Cloud
- CI/CD: GitHub Actions workflow
- Email: Nodemailer with Ethereal or SMTP (Stretch)

Key Features

- Menu Browsing
- Cart Management
- Order Placement
- Secure APIs (add x-api-key in header)
- Inventory Control
- Logging & Error Handling
- Functional CI/CD pipeline with basic automated tests
- Deployed frontend and backend applications
- Order History (Stretch)
- Order email (Stretch)

Sprint wise goals

Iteration 0: Planning and Design

- Finalise the business requirements and scope for the MVP (Minimum Viable Product).
- Design the overall system architecture (frontend, backend, database, API structure).
- Define the database schema and collections/tables.
- Design API endpoints (Menu, Cart, Order, Security, etc.) with request/response formats.
- Prepare a detailed README and documentation in the GitHub repository.
- Plan the CI/CD pipeline and deployment strategy (local and cloud).
- Identify stretch goals (Order History, Email Confirmation) and prioritise core MVP features.

Sprint wise goals

Iteration 1: Development Complete

- Implement the frontend (ReactJS/NextJS) and backend (Node.js/Express.js, Python FastAPI, or Java Spring Boot).
- Develop and test all core RESTful APIs:
  - Menu API (list pizzas, drinks, inventory)
  - Cart API (create/update cart, calculate totals)
  - Order API (place order, decrement inventory)
- Integrate NoSQL database (MongoDB, Dynamo, or Firestore).
- Secure all endpoints using x-api-key in headers.
- Implement logging and error handling.
- Set up a functional CI/CD pipeline with basic automated tests.
- Deploy both frontend and backend applications.
- Commit all code and design artefacts to the GitHub repository.
- Prepare a Postman collection with working API requests.

Sprint wise goals

Iteration 2: Testing, Integration, and Demo

- Conduct end-to-end testing of the entire system (frontend, backend, APIs, database).
- Integrate all components and ensure smooth data flow and user experience.
- Validate security (API key enforcement) and error handling.
- Demonstrate the CI/CD pipeline with automated tests and deployments.
- Prepare and deliver a live demo of the deployed solution.
- (If time permits) Implement and test stretch features:
  - Order History for authenticated users
  - Order confirmation email after successful order
- Finalise documentation and ensure the GitHub repo is up to date.