مکتب شریف

اولین بوتکمپ آموزشی - استخدامی ایران

مـکـتب 126

پروژه نهایی

Online Store (Backend with Django, DRF, and React Frontend)

---

1. Overview of the Project

In this project, students will be tasked with developing the backend for an e-commerce website. They will work on functionalities such as managing products, customer accounts, orders, and discounts, while the frontend (React app) will be provided. The primary focus for students will be creating the backend APIs using Django and Django REST Framework (DRF) to handle the backend logic and connecting the React frontend to the backend.

---

2. Technologies Involved

- **Backend**:
  - **Django** (for the backend framework)
  - **Django REST Framework (DRF)** (to build APIs)
  - **PostgreSQL** (for database management)
  - **Redis** (for OTP generation, caching)
  - **Celery** (for background tasks like sending emails)
  - **RabbitMQ** (optional for managing task queues in Celery)
  - **JWT** (for authentication)
  - **Docker** (for containerizing the application)
  - **Nginx** (for reverse proxy and static file management)
- **Frontend**:
  - **React** (pre-built app for connecting to the backend)
  - **JavaScript, HTML, CSS** (for frontend interactions and styles)
  - **TailwindCSS or Bootstrap** (optional, for styling in React)

3. Project Breakdown (Phases) (۲۰ تیر تا ۲۶ تیر)

Phase 1: Project Setup and Initial Backend Development

- **Tasks**:
  - Analyze the project requirements and create an ERD (Entity Relationship Diagram).
  - Set up the Git repository and configure Django settings.
  - Initialize the backend with Django and DRF.
  - Set up PostgreSQL as the database.
  - Set up Redis for OTP generation and caching.
  - Develop models for:
    - Customer
    - Address
    - Product
    - Order
    - Order Item
    - Store
    - Cart
    - Cart item
    - Payment
    - Reviews (Comment)
    - Category
  - Implement authentication with JWT (Login and Registration).
  - Implement OTP-based login via email or phone using Redis.(one time password)
- **Deliverables**:
  - ERD (PDF)
  - GitHub repository with commit history.
  - Initial backend project structure.

- Sample data in the admin panel.
- Complete customer registration and login APIs.

Phase 2: Developing Core Features (۲۷ تیر تا ۲ مرداد)

- **Tasks**:
  - Implement functionality for managing customer profiles (view, update).
    - User may have multiple address
  - Implement basic API views for CRUD operations on products and store .
    - Implement discount on some product.
  - Create product catalog and product detail APIs.
  - Implement 2 method for find best seller and best price
  - Set up category models and APIs (including hierarchical categories using self relations).
  - The category in the products response returns an array from the leaf to the root of all categories in an orderly manner.
  - Seller Dashbord (only seller can access to this section):
    - Manage Store
    - Msnage Category
    - Manage Product
- **Deliverables**:
  - Product and discount APIs.
  - Category APIs.
  - User profile
  - Working backend for connecting the frontend.

Phase 3: Customer and Order Management (۲ مرداد تا ۹ مرداد)

- **Tasks**:
  - Implement order placement and order item management.
  - Develop the shopping cart API for adding/updateing/removing items.
  - Implement order history and order status tracking for customers.

- Develop API endpoints for managing orders and updating order statuses (e.g., pending, shipped, delivered).
- Integrate background tasks using Celery for order processing (e.g., sending emails for order confirmation).
- Set up Celery with RabbitMQ for task management
- Seller Dashbord (only seller can access to this section):
  - Manage Orders

- **Deliverables**:
  - API endpoints for managing orders and customer data.
  - Email notifications via Celery tasks.
  - Full backend integration with the provided React frontend.

Phase 4: Final Refinements and Deployment (۹ مرداد تا ۱۵ مرداد)

- **Tasks**:
  - Final debugging and testing of the entire system.
  - Test coverage to ensure the backend APIs are well-tested (unit tests, coverage above 80%).
  - Integrate the frontend (React) app with backend APIs, ensuring all interactions work seamlessly.
  - Deploy the application using Docker and Nginx (for reverse proxy and static files).
  - Prepare project documentation and README.
- **Deliverables**:
  - Deployed project (live URL if applicable).
  - Full backend with connected frontend (via React).
  - Well-tested APIs with proper test coverage.
  - Detailed project documentation on GitHub.

---

4. Additional Project Requirements

- **Git Branching**:

  - Use develop and master branches. All code should be developed on the develop branch and merged into master after completing each phase.

- **Testing**:
  - 30% of the grade will be based on testing coverage. Ensure that unit tests are implemented for backend logic.
- **Database and Models**:
  - Models should be normalized and properly related.
  - Implement **soft deletes** (delete_logical) for entities that need deletion.
- **Authentication**:
  - JWT authentication will be used for user login, and the frontend should handle storing and passing the JWT token in HTTP headers for API requests.
- **Deployment**:
  - Dockerize the application using Dockerfile and docker-compose.
  - Deploy using Nginx to serve the backend and handle static files (for production environments).
- **GitHub Setup**:
  - Ensure that the project repository on GitHub is private.
  - Add instructors as collaborators.
    - Mahdizo31
    - Sina-mobarez
    - Mejomba
    - FardinMoghaddamPour
  - Ensure commit history is regular and properly labeled.

---

5. Final Project Submission Details

- Submit the final project by pushing all code to the private GitHub repository.
- Include a detailed README.md that explains the setup, technologies, and instructions for running the backend.
- Submit the ERD in PDF format as part of the GitHub repository.
- Ensure that the frontend and backend are fully integrated and functional.

Key Adjustments for React Frontend:

1. **React Frontend**: The frontend, built in React, will be provided, and students need to ensure the backend APIs integrate seamlessly with it. Students should focus on:
    - Creating and exposing API endpoints that match the data and actions the frontend expects.
    - Handling authentication, including token management and securing routes using JWT.
    - Ensuring the backend can handle user actions like adding to the cart, checking out, viewing orders, and profile management.
2. **Frontend Backend Connection**:
    - Handle API requests in Django views with proper RESTful methods.
    - Return appropriate responses (JSON) for frontend consumption.
    - Allow CORS (Cross-Origin Resource Sharing) if the frontend is served from a different server.