COLLEGE CODE : 9530

COLLEGE NAME : ST.MOTHER THERESA ENGINEERING COLLEGE

DEPARTMENT : COMPUTER SCIENCE AND ENGINEERING

STUDENT NM-ID : BA5644F834F932F9F804D5F59A3AA1AB

ROLL NO : 953023104051

DATE : 29.09.2025

Completed the project named as

Phase_4_ TECHNOLOGY

PROJECT NAME : PRODUCT CATALOG WITH FILTERS

SUBMITTED BY,

NAME : MADHUMITHA.A

MOBILE NO : 9003775802

Additional Features:

Elevating User Choice

- Dynamic Multi-Level Filters :

Implement intuitive filters for categories, price ranges, brands, and ratings, allowing users to drill down with precision.

- Personalised Recommendations :

Leverage user behaviour analytics to offer tailored product suggestions, enhancing discoverability and engagement.

- Real-Time Updates & Urgency :

Display real-time stock levels and promotional badges to create a sense of urgency, driving quicker purchase decisions.

- Wishlist & Compare Functions :

Empower users with tools to save desired products and compare specifications side-by-side for informed choices.

UI/UX Improvements:

Crafting Seamless Experiences

- Intuitive filter panel with collapsible sections and clear labels for ease of use.
- Responsive design ensuring smooth navigation and optimal viewing on all devices, from mobile to desktop.
- Instant feedback with loading skeletons and smooth animations for a delightful user experience during filter changes.
- Accessibility compliance, including keyboard navigation and screen reader support, to cater to all users.

A thoughtfully designed interface is crucial for user engagement and satisfaction, ensuring that finding products is effortless and enjoyable.

API Enhancements:

Types of APIs

1. REST API

- Most common type, uses HTTP methods (GET, POST, PUT, DELETE).

- Data usually in JSON format.

Example: GET https://fakestoreapi.com/products

2. SOAP API

- Uses XML format, heavier and older. Mostly used in enterprise applications.

3. GraphQL API

- Flexible API where clients request exactly what they need.

- Used by Facebook, GitHub.

4. API Methods (CRUD Operations)

- GET: Retrieve data

- POST: Send new data

- PUT/PATCH: Update existing data

- DELETE: Remove data

Example (Product Catalogue project):

- GET /products → Fetch all products

- GET /products/:id → Fetch single product details

- POST /products → Add new product (admin use)

- DELETE /products/:id → Remove a product

5. APIs for Product Catalogue with Filter (Your Project)

In your project, you can use the following APIs:

- Get All Products: https://fakestoreapi.com/products

- Get Categories: https://fakestoreapi.com/products/categories

- Filter by Category: https://fakestoreapi.com/products/category/{categoryName}

- Search Products: https://dummyjson.com/products/search?q={keyword}

Example Output:


```
{
  "id": 1,
  "title": "Men's T-Shirt",
  "price": 299,
  "category": "clothing",
  "image": "https://fakestoreapi.com/img/1.jpg"
}
```


Performance & Security Checks:

Safeguarding Reliability

- Load Testing :

Conduct rigorous load testing using tools like JMeter to guarantee the catalog can handle peak traffic without performance degradation.

- API Security Testing :

Utilise cutting-edge security tools (e.g., Pynt, Jit) to detect and mitigate vulnerabilities such as injection flaws and broken authentication.

- OWASP Top 10 Best Practices :

Enforce OWASP API Security Top 10 best practices, including strict authorisation, rate limiting, and meticulous input validation.

- API Gateways :

Implement API gateways for centralised traffic control, comprehensive logging, and effective threat mitigation, enhancing overall security posture.

Testing of Enhancements:

Ensuring Quality & Stability

- Automated Unit & Integration Tests :

Develop comprehensive automated tests for filter logic and API responses, ensuring every component functions as expected.

- End-to-End UI Testing :

Perform end-to-end UI testing with tools like Cypress or Selenium to simulate real user interactions and validate the entire user journey.

- Security Fuzz Testing :

Employ security fuzz testing to uncover unexpected input handling issues, bolstering the application's resilience against malicious inputs.

- Continuous Integration (CI) Pipelines :

Integrate CI pipelines to automatically run tests on every code push, enabling early detection and resolution of defects, ensuring continuous quality.

Deployment Options:

Choosing the Right Platform

- Netlify & Vercel :

Ideal for static site deployment with serverless functions, offering global CDN, instant rollbacks, and rapid deployment for frontends.

- Cloud Platforms :

AWS, Azure, GCP provide scalable backend hosting, managed databases, API gateways, and advanced monitoring for robust catalog infrastructures.

- Containerization :

Docker and Kubernetes facilitate containerization for complex, microservicesbased catalogs, ensuring portability and efficient scaling.

- CI/CD Integration :

Seamless CI/CD integration enables automated build, test, and deploy workflows, fostering rapid iteration and continuous delivery of enhancements.

Real-World Example:

Deploying on Vercel

- Frontend React App:

Utilising Next.js for server-side rendering (SSR) and static generation, ensuring fast load times and improved SEO.

- API Routes as Serverless Functions:

Implementing API endpoints as serverless functions with built-in caching for enhanced performance and scalability. Preview

- Deployments:

Leveraging Vercel's preview deployment feature for seamless QA and stakeholder reviews before pushing to production.

- Monitoring & Analytics:

Integrating Vercel Analytics and third-party tools for comprehensive performance insights and continuous optimisation.