**E-Commerce Site using MongoDB**

1) Problem Statement

* Developing an efficient MongoDB-based e-commerce platform using Express.js and implementing advanced MongoDB techniques such as sharding, partitioning, and aggregation for optimal data management and analysis, integrating data mining algorithms to extract actionable insights for enhanced business performance.
* The issue statement is significant because it addresses the challenges that e-commerce platforms experience in successfully managing and analysing massive volumes of data.

2) Introduction

* MongoDB's flexible data model and advanced capabilities allow for efficient data management in e-commerce platforms. Partitioning data according to specific criteria (e.g., customer location, product categories) can help to streamline operations and improve data organisation, making it easier to find relevant information.
* Implementing an advanced e-commerce platform might provide you a competitive advantage in the industry. By integrating MongoDB and innovative data approaches, the platform can outperform competitors in terms of performance, scalability, and analytics, attracting new users and driving corporate growth.
* This project aims to address important difficulties faced by e-commerce platforms, such as scalability constraints, performance bottlenecks, and the capacity to extract valuable insights from massive amounts of data. Our goal is to build a high-performing, feature-rich e-commerce platform that not only meets, but surpasses, the expectations of today's online buyers by exploiting MongoDB's strengths and modern data approaches.

3) Different Tools Used

* **MongoDB:** MongoDB is a widely used, open-source NoSQL (non-relational) database management system. It has a document-oriented data format, which means it stores data in flexible, JSON-like documents, making it easier to handle and change data than standard relational databases.
* **Express.js:** Express.js is a Node.js web application framework that helps you construct online applications and APIs. It offers a wide range of functionality for web and mobile applications, including as routing, middleware support, template engines, and more.
* **React.js:** React.js is a widely used JavaScript package for creating user interfaces, notably for single-page apps. Facebook and a development community are responsible for its maintenance. React lets you design reusable UI components that maintain their own state and update quickly when data changes.

4) Literature Review

1. The author in [1], emphasizes MongoDB’s applicability for e-commerce because to its schema flexibility, scalability, and document-oriented approach, which enable efficient management of changing product data, rapid expansion, and adaptable handling of multimedia formats, allowing for quick responses to market changes.
2. The author in [2], compares SQL and MongoDB databases for e-commerce data management, highlighting MongoDB's schema flexibility, horizontal scalability, superior read and write performance, and support for unstructured data, establishing it as a dependable alternative for handling large e-commerce datasets.
3. The author in [3], emphasizes the integration of data mining techniques with MongoDB's e-commerce architecture, allowing for quick data analysis to improve personalized suggestions and marketing strategies.
4. Amazon uses monolithic architecture consisting of search queries for filtering products as per [4], the author in [5] states that sharding are a better technique as it’s more reliable and faster.

5) SWOT Analysis

**Strengths:**

* Interactive & User Friendly UI: The website has a visually appealing and intuitive user interface that makes it easy for customers to navigate and find products.
* Lightweight Database: The website utilizes efficient database techniques like sharding and partitioning to maintain a lightweight and high-performing backend.

**Weakness:**

* Limited Product Variety: The website currently only offers a narrow selection of products, which may limit its appeal to a wider customer base.

**Opportunities:**

* Sustainable on Limited Resources: The website's lightweight architecture and efficient database design allow it to operate effectively with limited resources, creating opportunities for growth and expansion.

**Threats:**

* Cyber Security: The website must prioritize robust security measures to protect against potential cyber threats, such as hacking, data breaches, and online fraud.

6) Objectives

1. Leverage MongoDB's advanced data management capabilities, such as sharding and partitioning, to develop a highly scalable and efficient E Commerce platform.
2. Integrate sophisticated data mining algorithms within the MongoDB ecosystem to analyze large datasets and extract actionable insights**.**

7) Methodology

1. **Define Requirements:**

* Clearly outline the requirements and features of your e-commerce website, such as user authentication, product browsing, cart management, checkout process, etc

2. **Design Database Schema:**

* Design the database schema using MongoDB, considering collections for products, users, orders, and any other necessary data entities. Utilize MongoDB's flexible document structure to store data efficiently.

3. **Backend Development:**

* Set up a backend server using Node.js and Express.js, which will serve as the API layer to interact with the MongoDB database.
* Implement RESTful API endpoints for CRUD operations (Create, Read, Update, Delete) for managing products, users, orders, etc.
* Integrate MongoDB driver or an ODM (Object Data Modeling) library like Mongoose to interact with the MongoDB database from Node.js.

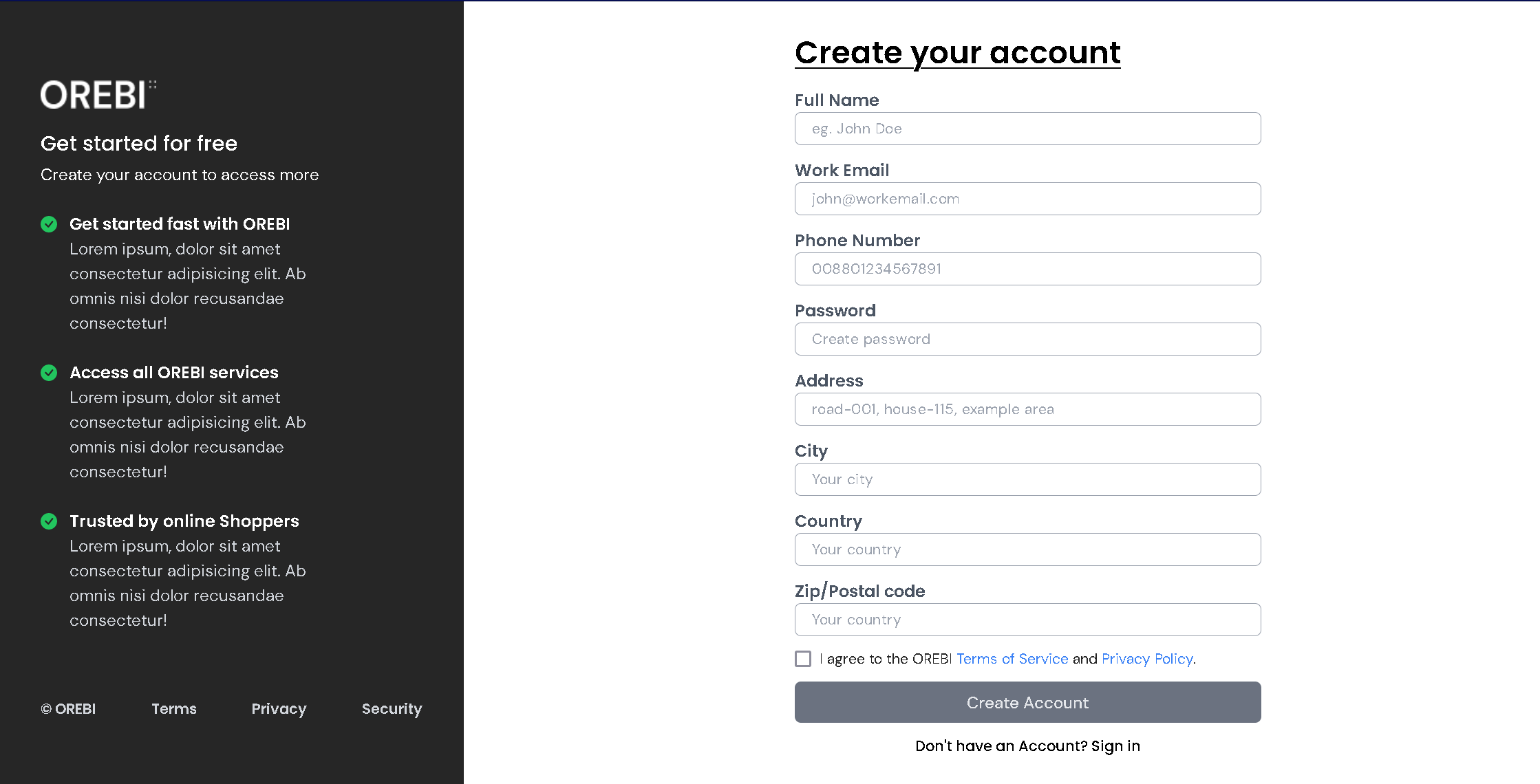
4. **Frontend Development:**

* Set up a React.js project structure using tools like Create React App.
* Design and develop user interfaces for various components of the e-commerce site such as product listings, product details, user authentication, shopping cart, checkout process, etc.
* Utilize React Router for client-side routing to navigate between different pages of the website.
* Implement state management using tools like React Context API or Redux to manage application-level state, such as user authentication status, shopping cart items, etc.

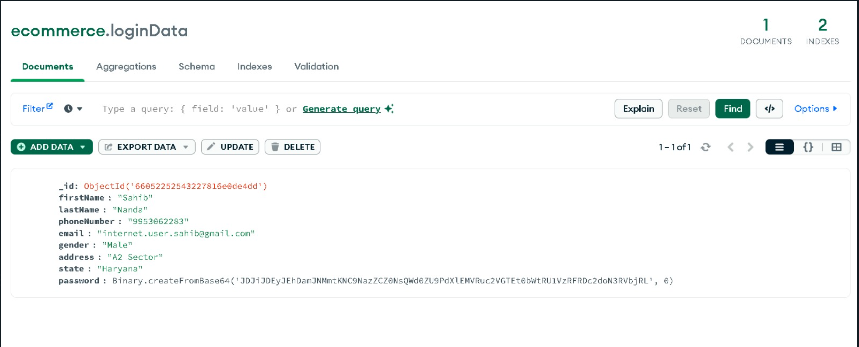
5. **Integration:**

* Connect the frontend React.js application to the backend Node.js server using API requests. Utilize libraries like Axios or the built-in Fetch API for making HTTP requests.
* Ensure proper error handling and validation on both the frontend and backend to provide a smooth user experience

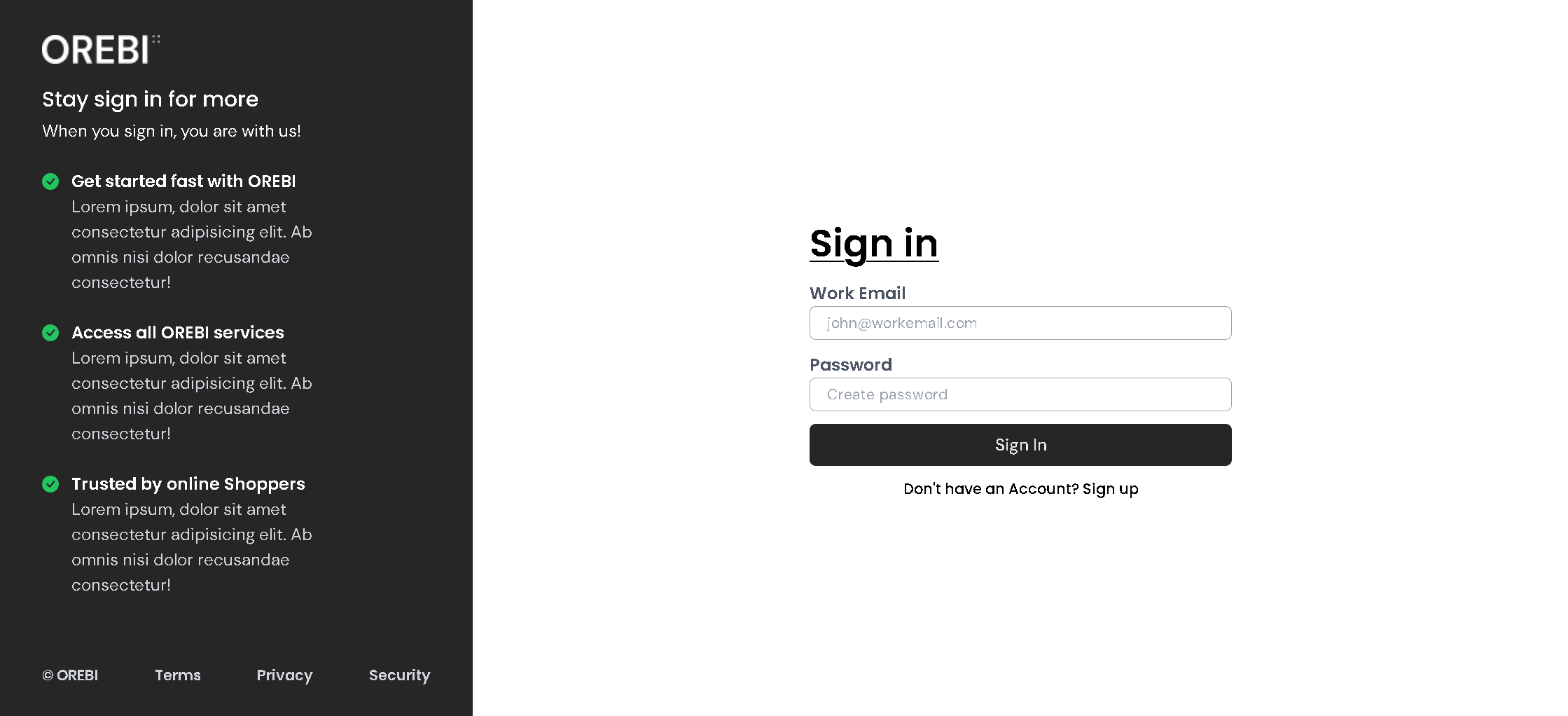
8) Working Model



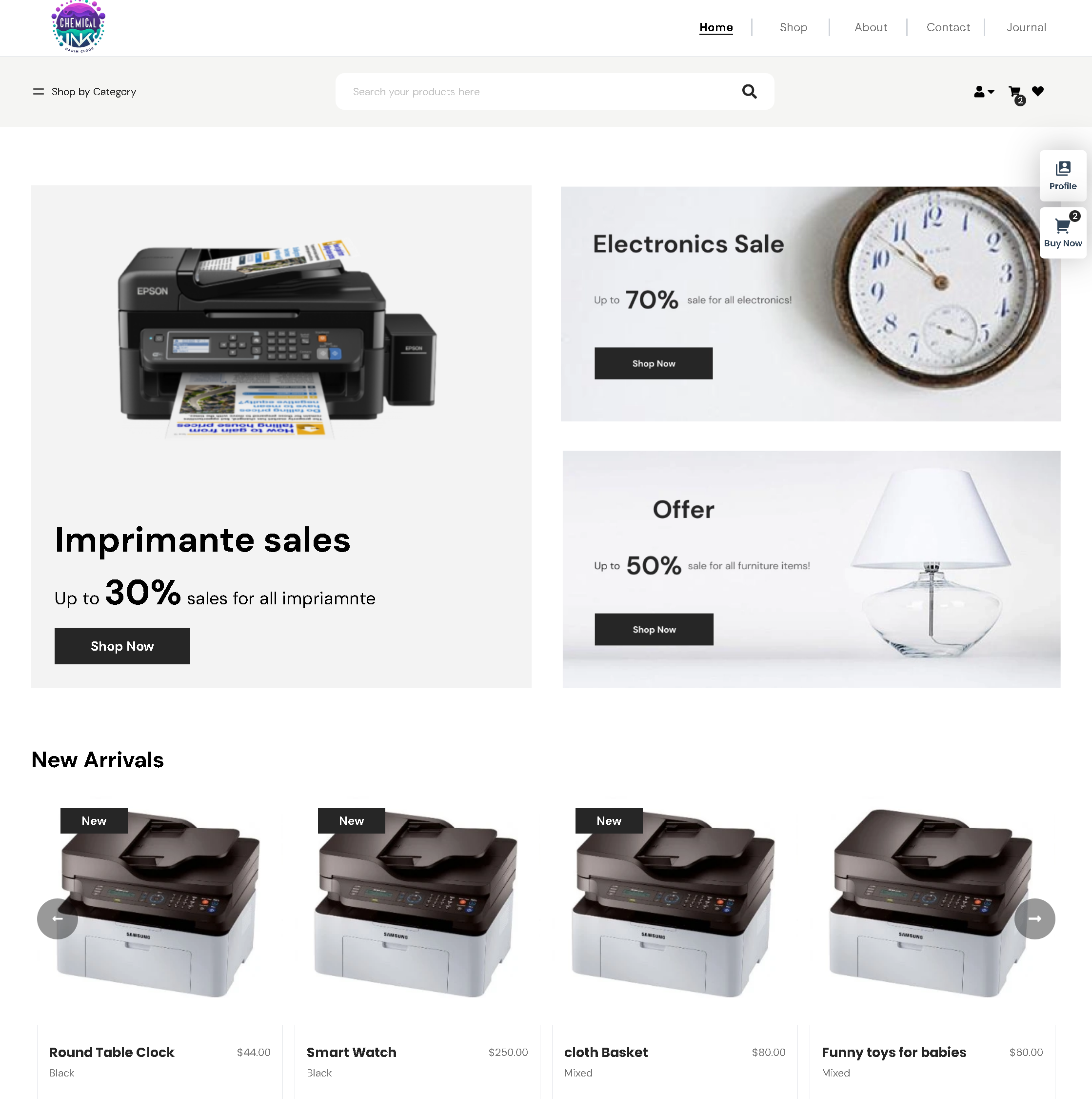
* The above image is the snap shot of our login page, where user will fill the required details and then they will be entering the home page



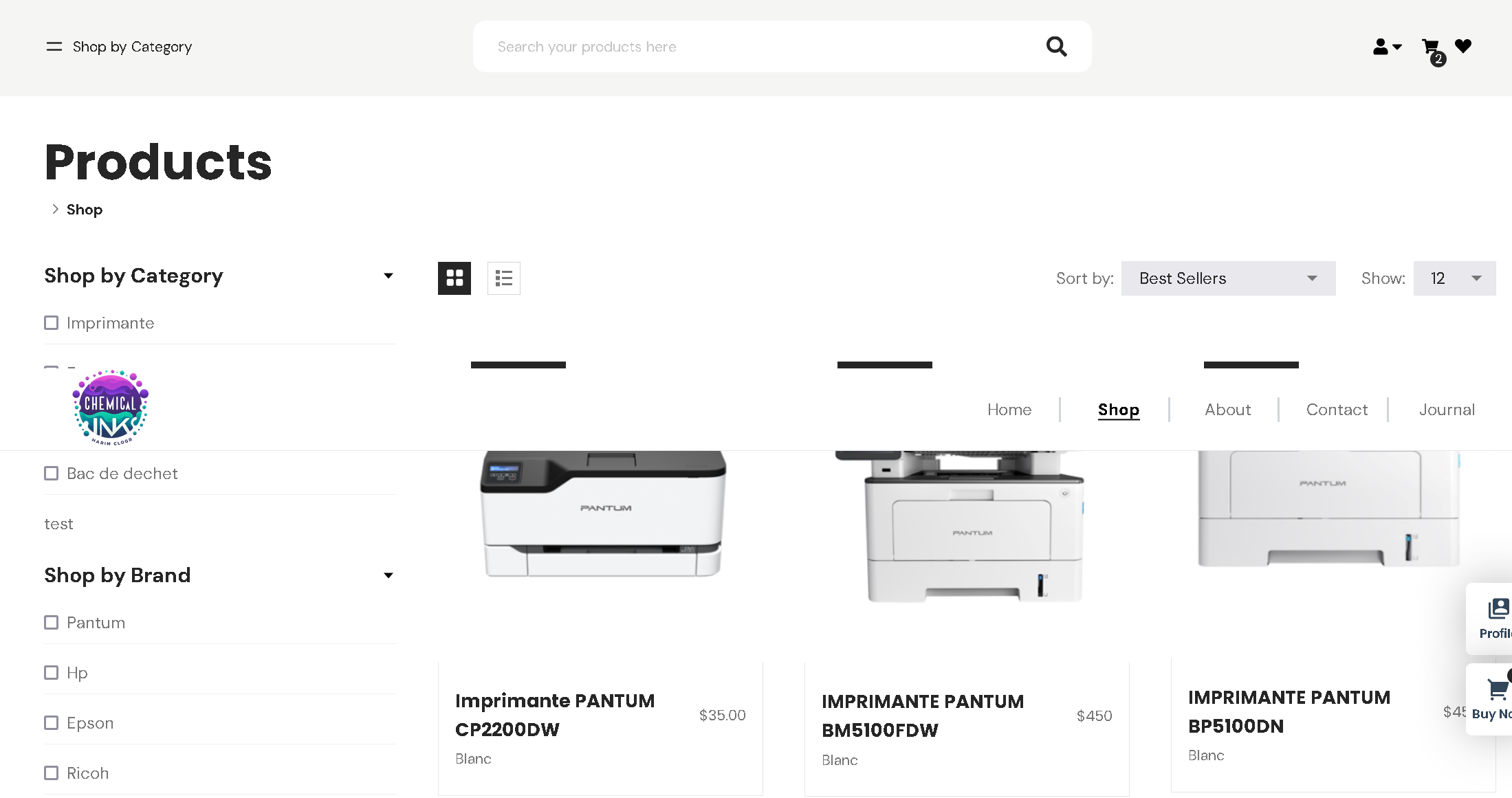
* This is where the login data is stored.



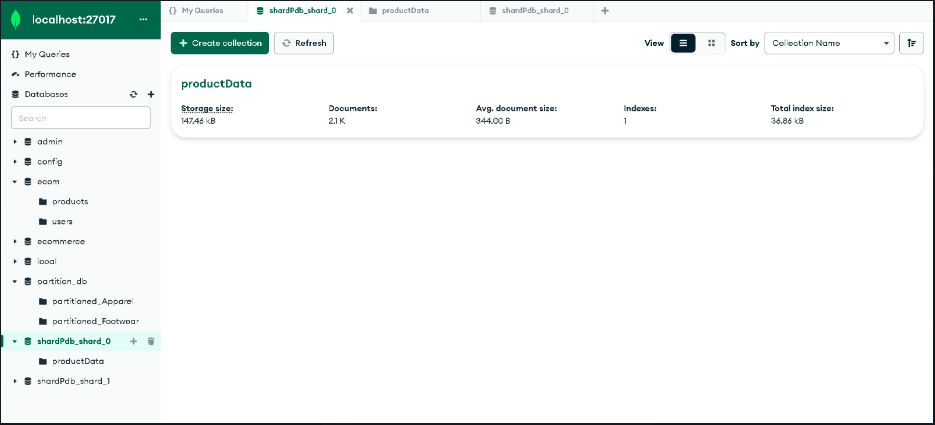
* If it is the second time the user is accessing the site, they will have to sign in by entering their email and password



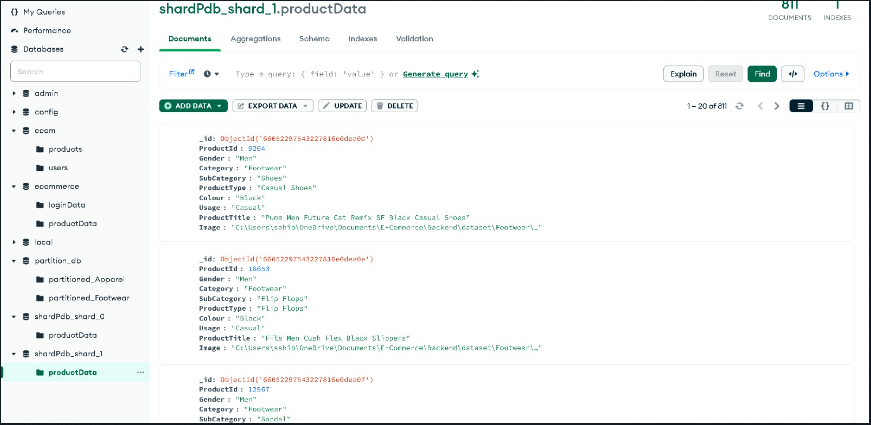
* This the home page of our site.
* It shows different products along with their prices.
* There is an option to add to cart for future.



* This is where one can see different products.
* Here you can filter on basis of categories, brand and colour.



* This is where in the backend the shards are created.
* As it is seen, a new database named shardPdb is created where shards of the product data is created



* According to the condition selected, the shards are created as shown above.

9) References

1. Inclusion of e-commerce workflow with NoSQL DBMS: MongoDB document store: <https://ieeexplore.ieee.org/document/7919652>
2. Performance Evaluation of SQL and MongoDB Databases for Big E-commerce Data: <https://ieeexplore.ieee.org/document/7369245>
3. E-Commerce with Backbone of Data Mining: https://www.researchgate.net/publication/328628430\_E-Commerce\_With\_Backbone\_Of\_Data\_Mining
4. AWS Monolithic Architecture: <https://aws.amazon.com/blogs/architecture/architecting-a-highly-available-serverless-microservices-based-ecommerce-site/>
5. Database Sharding: https://www.dbta.com/Editorial/Trends-and-Applications/Database-Sharding--The-Key-to-Database-Scalability-55615.aspx