**1.INRODUCTION**

In In today’s digital age, e-commerce has transformed how footwear is showcased and sold, enabling shoe stores to reach a global audience beyond physical locations. Managing products, brands, and customer interactions manually, however, can be challenging —which is where the **KickCouture (E-Commerce) Website** comes in.

This platform provides an interactive and user-friendly environment for browsing and purchasing footwear online. It streamlines store operations, connecting customers and administrators seamlessly.

For shoppers, the platform offers a smooth, engaging experience with features like category browsing, detailed product descriptions, and brand profiles. For store administrators, it includes powerful tools to manage products, categories, brands, and customer data efficiently.

Key features include advanced search, category filtering, and detailed product views — all designed to enhance navigation and increase user engagement.

Built with **React (frontend), Node.js (backend),** and **MongoDB (database)**, the platform expands the reach of footwear collections while improving operational efficiency, offering a modern, seamless online shopping experience.

## **2. SYSTEM STUDY**

* 1. **EXISTING SYSTEM**

In the existing system, managing a shoe store is often carried out manually or through basic offline methods. Footwear collections are displayed only in physical store locations, restricting the reach to local customers. Shoppers who wish to browse or purchase shoes must visit the store in person.

**The following are the disadvantages of the existing system:**

* Requires more manpower to manage products, brand information, and customer interactions.
* Customers must physically visit the store to view or buy shoes.
* Product details, pricing, and stock availability are maintained manually, which can lead to errors and inconsistencies.
* Time-consuming for both customers and store staff
  1. **PROPOSED SYSTEM**

The new KickCouture Website aims to make footwear browsing, purchasing, and management easier and more efficient for both customers and store administrators.

**The following are the advantages of the proposed system:**

* Customers can browse and purchase footwear online from anywhere, anytime.
* Improves convenience by removing the need for physical visits just to explore product options.
* Reduces paperwork by digitizing product records, brand profiles, and sales history.
* Displays product details such as price, description, available sizes, and brand information in real-time.
* Allows easy searching and filtering of products based on category, size, or brand.
* Expands the store’s reach to a wider audience through an online presence.

**2.3 PROBLEM DEFINITION AND PROJECT DESCRIPTION**

The project titled KickCouture Website is a web-based application designed to showcase and manage various footwear collections, brands, and related services. In the shoe shopping portal, users can explore different categories of shoes, view product details, and make purchases online.

Shoe brands or store managers can register on the platform to display and sell their products, while the admin manages and oversees the overall system operations.

This project aims to provide a convenient and engaging platform for shoe shoppers, enabling them to discover and purchase footwear from anywhere, while giving store owners and brands a space to reach a wider audience and manage their collections efficiently.

**Modules**

* **Admin Module**
* **User Module**

**ADMIN MODULE**

* Add and manage footwear categories.
* Approve or reject brand/store registrations.
* View and manage the list of registered brands/stores and users.
* View all uploaded products.
* Manage sales and purchase records.
* Generate reports on sales, users, and brands.

**USER MODULE**

* Browse footwear by category, brand, size, or search options.
* View detailed product information including images, price, sizes available, and brand details.
* Purchase shoes securely through the platform.
* View order history and purchase details.

**3.SYSTEM ANALYSIS**

**3.1 REQUIRMENTS SPECIFICATION**

**HARDWARE REQUIREMENTS**

Processor : AMD Ryzen 5 5500U with Radeon Graphics i3 or higher  
 RAM : Minimum 4 GB (8 GB recommended)  
 Hard Disk : Minimum 40 GB free space222  
 Monitor : 15’’ LED/LCD monitor or higher  
 Keyboard : Standard 104-key keyboard  
 Mouse : Optical mouse

**SOFTWARE REQUIREMENTS**

Operating System : 64-bit Windows 10 / 11 (or equivalent Linux/Mac)

Local Host Server : Node.js with Express framework

Front End : React.js(HTML , CSS , Javascript )

Back End : Node.js

Database : MongoDB Compass

Database Connectivity : Mangoose

Browser : Google Chrome / Firefox

Other Tools : Visual Studio Code, Postman

**3.2 FEASIBILITY STUDY**

A feasibility analysis evaluates whether it's more viable to enhance the current manual shoe store operations or develop a new automated system. It identifies challenges and estimates the practicality of implementing a solution. The study covers three key areas:There are three major aspects covered in the feasibility study during the preliminary investigation:

* **Technical Feasibility**
* **Operational Feasibility**
* **Economic Feasibility**

**TECHNICAL FEASIBILITY**

Technical feasibility assesses whether the required technology is available, manageable, and if the development team has the necessary skills. It examines system inputs, processes, outputs, and overall architecture to ensure efficient operation.

For the proposed **Online Shoe Shopping Website**, both hardware and software environments are considered. The system will use **ReactJS (frontend), Node.js (backend),** and **MongoDB (database) —**all modern, scalable, and industry-standard technologies.

The application runs smoothly on basic setups with minimum requirements: **Intel i3 processor, 4 GB RAM,** and **40 GB disk space.** Software needs include a **64-bit OS, Node.js environment,** and **a MongoDB server.**

**OPERATIONAL FEASIBILITY**

1. Problem-Solving: The system addresses current challenges by enabling online shoe browsing, order placement, and catalog management—eliminating in-store limitations.
2. User Adoption: With a user-friendly interface, the platform is intuitive for both customers and administrators, requiring minimal training for users familiar with online shopping.
3. Admin Support: It supports essential admin tasks like inventory tracking, user management, and product updates, making operations more efficient.

**ECONOMIC FEASIBILITY**

The system offers a cost-effective solution by reducing manual work, increasing sales potential, and improving customer satisfaction. Initial development costs are justified by long-term benefits like higher revenue and streamlined operations. Ongoing maintenance is minimal, making it economically feasible.

**4. SYSTEM DESIGN**

**4.1 ARCHITECTURE DESIGN**

ONLINE SHOE SHOPPING WEBSITE

Admin

User

Login/Register

Login

Add & Manage

Product

View Product

Update Profile

Place Order

Mange Users

Manage Order

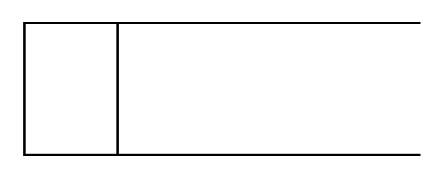
### **DFD Level 0**

ADMIN

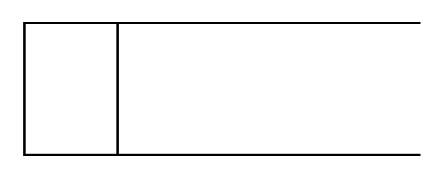
DATABASE

USER

**DFD Level 1**

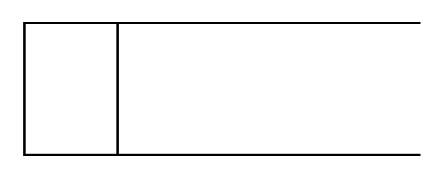


Collection\_Items

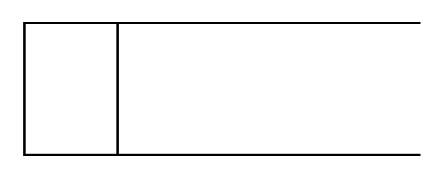


Collection\_Orders

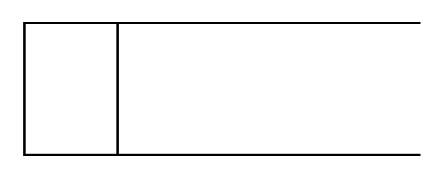
**ADMIN**



Collection\_Users

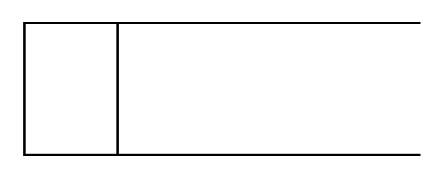


Collection\_User



Collection\_Items

**ADMIN**



Collection\_Users

* 1. **DATA DICTIONARY**

**Collection Name:** Admin

**Purpose:** To store login details of admin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **FieldName** | **DataType** | **Constraint** | **Description** |
| 1 | Id | ObjectId | Primary Key | Unique identifier |
| 2 | AdminName | String | Not null | Admin’s name |
| 3 | Email | String | Not null | User name |
| 4 | Password | String | Not null | Password |

**Collection Name:** User

**Purpose:** To Store User details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **FieldName** | **DataType** | **Constraint** | **Description** |
| 1 | Id | ObjectId | Primary Key | Unique identifier |
| 2 | Name | String | Not Null | User’s full name |
| 3 | Email | String | Not Null, Unique | User’s login email |
| 4 | Password | String | Not Null | Hashed password |

**Collection Name:** items

**Purpose:** To Store Product details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **FieldName** | **Datatype** | **Constraint** | **Description** |
| 1 | Id | ObjectId | Primary Key | Unique Identifier |
| 2 | Productid | String | Unique | Custom product code |
| 3 | Name | String | Not Null | Product’s Name |
| 4 | Category | String | Not Null | Category of the Product |
| 5 | Size | String | Not Null | Size of the product |
| 6 | Desc | String | Not Null | Detailed description |
| 7 | Image | Object | Not Null | Product’s image data in binary  format |

**Collection Name:** Order

**Purpose:** To Store order details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **FieldName** | **Datatype** | **Constraint** | **Description** |
| 1 | Id | ObjectId | Primary Key | Unique identifier |
| 2 | orderId | String | Unique | Custom order code |
| 3 | userId | ObjectId | Foreign Key | The customer who placed the order |
| 4 | Items | Array of Objects | Not Null | List of products in the order |
| 5 | totalAmount | Number | Not Null | Total price for all items |
| 6 | paymentStatus | String | Not Null | Status of payment |
| 7 | orderStatus | String | Not Null | Current progress of the order |
| 8 | shippingAddress | Object | Not Null | Full delivery address for the order |
| 9 | orderDate | Date | Not Null | Date when the order was placed |
| 10 | deliveryDate | Date | Not Null | Date of delivery |

4.4. **User Interface Design**

**HOME PAGE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
| **Shop Name** | Home Products | About Contact | Login |  |
| Image 1 | **Products Collection**      Image 2 | Image 3 | |  |
| Copyright Reserved | Designed by Pream | | | | |

**Login Page**

**Email:**

**Password:**

Don’t have an account Register here

**Login**

**Register page**

**Full Name:**

**Email:**

**Address:**

**Phone No:**

**Password:**

**Already have an account Login here**

**Register**

**Admin’s Dashboard**

Manage Product

Manage Orders

Manage User

Total User

Total Order

Welcome Admin

**Manage Products:**

**Shop Name**

**Add New Product**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ProductID** | **Name** | **Images** | **Category** | **Size** | **Actions** |
|  |  |  |  |  | Edit  Delete |
|  |  |  |  |  |  |

**Manage Orders**

**Shop Name**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Order ID** | **Customer Name** | **Total Amount** | **Payment Status** | **Order Status** | **Order Date** | **Delivery Date** | **Actions** |
|  |  |  |  |  |  |  | Edit Delete |
|  |  |  |  |  |  |  |  |

**Manage Users:**

**Shop Name**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Email** | **Phone** | **Actions** |
|  |  |  |  | Edit Delete |

**Products Page:**

Name of the Product Select:

Price: 1,500

**Shop Name**

Name of the Product Select :

Price: 2,000

Heels

Image of the Product

View Details

shoe

Image of the Product

View Details

**FIRST NORMAL FORM (1NF)**

For a collection to be in 1NF:

1. Atomic values only – Each field must contain a single value. For example, in the users collection, the phoneNumber field should store only one phone number, not a list of numbers.
2. Same data type in a field – All values in a field should be of the same data type. For example, the price field in the products collection should always be a number.
3. Unique field names – Each attribute (field) must have a unique name.
4. Order of data does not matter – The sequence of records should not affect the meaning of the data.

**SECOND NORMAL FORM (2NF)**

For a collection to be in 2NF:

1. The collection must be in 1NF.
2. No Partial Dependency – Non-key attributes should depend on the whole primary key, not just a part of it.

**Example in project:**

If the primary key in the orderItems subdocument is a combination of orderId and productId, then attributes like productName should not be stored there (since it depends only on productId). Instead, we link to the products collection to retrieve the product details.

**THIRD NORMAL FORM (3NF)**

For a collection to be in 3NF:

1. Must be in 2NF.
2. No Transitive Dependency – Non-key attributes should not depend on other non-key attributes.

**Boyce Codd Normal Form (BCNF)**

Conditions:

1. Must be in 3NF.
2. For every functional dependency (X → Y), Xshould be a superkey.