2.3 TECHNOLOGY

2.3.1 FRONT-END TECHNOLOGIES:-

HTML: Provides the structure and semantics for web pages.

CSS: Used for styling and layout of the user interface.

JavaScript: Enables interactive and dynamic features on the clientside.

2.3.1.1 HTML:- HTML provides the structure and semantics for web pages. It uses tags to define elements like headings, paragraphs, links, and forms, organizing content and conveying meaning. HTML is essential for creating readable, accessible, and search engine-friendly web pages, serving as the foundation of web development.

2.3.1.2 CSS:- CSS (Cascading Style Sheets) is crucial for web development, providing the means to style and layout the user interface of web pages. It defines the presentation of elements, including colors, fonts, spacing, and more, ensuring a visually appealing and cohesive design.

2.3.1.3 JavaScript:- JavaScript is essential for creating interactive and dynamic features on the client-side of web applications. It enables developers to manipulate the Document Object Model (DOM), handle events, and update content dynamically, enhancing user engagement and interactivity.

**2.3.2 BACK-END TECHNOLOGIES**

**Firebase:** Provides real-time database and secure authentication services.

**2.3.2.1 Firebase**

Firebase is a back-end platform developed by Google that offers efficient and scalable solutions for web applications. In the Kiti-Korean Store project, the following Firebase services are used:

**1. Firebase Authentication:**

* Facilitates secure user authentication.
* Supports multiple sign-in methods, including email/password and third-party providers (e.g., Google).
* Ensures secure session management and user identity verification.
* Tracks user sessions for personalized experiences (e.g., wishlists and order history).

**2. Firebase Realtime Database:**

A cloud-hosted NoSQL database that synchronizes data in real-time across all connected clients. In the **Kiti-Korean Store** project, the following information is stored:

* **User Information:**
  + User ID (Unique identifier)
  + Name
  + Email address
  + Contact number
  + Wishlist items
* **Product Details:**
  + Product ID (Unique identifier)
  + Product name and description
  + Price (in INR)
  + Product images (URLs)
  + Available stock and categories
* **Orders:**
  + Order ID (Unique identifier)
  + User ID (Link to customer)
  + Product(s) purchased
  + Order date and time
  + Delivery address
  + Payment status (Pending, Completed, etc.)
  + Order status (Processing, Shipped, Delivered)
* **Wishlist & Gifting:**
  + User ID (to associate with logged-in users)
  + Product ID (for items in the wishlist)
  + Gifting options (Mother, Sister, Wife, Friends)
  + Personalized gift message

**3. PROPOSED SYSTEM**

**3.1 SCOPE**

**✅ User Side:**

* **User Authentication: Secure login with Firebase Authentication (Email/Google).**
* **Product Browsing: View, search, and filter products with detailed pages.**
* **Wishlist & Gifting: Save products and personalize gifts (Mother, Sister, Wife, Friends).**
* **Shopping Cart & Checkout: Multi-step checkout with secure payments.**
* **Order Tracking: View order status and history.**
* **Reviews & Ratings: Submit and view product feedback.**
* **User Profile: Manage account details and addresses.**

**✅ Admin Side:**

* **Dashboard: View sales, orders, and user analytics.**
* **Product Management: Add, edit, and delete products.**
* **Order Management: Track and update customer orders.**
* **User Management: Manage customer accounts and activity.**
* **Reviews Control: Approve, delete, and respond to reviews.**
* **Marketing: Manage discounts and track Meta Ads performance.**
* **Payments: Monitor transactions and handle refunds.**
* **Reports: Generate sales and user engagement reports.**
* **Settings: Configure site settings, shipping, and payment.**

**3.2 AIM & OBJECTIVES**

**Aim:**  
To develop a modern **fashion e-commerce** platform that delivers an intuitive, secure, and feature-rich shopping experience using front-end and back-end technologies.

**Objectives:**

1. **Develop a Responsive Website:** Ensure the site is accessible across all devices (mobile, tablet, desktop).
2. **Implement Secure Authentication:** Use Firebase Authentication to manage user login and protect user data.
3. **Manage Real-Time Data:** Use Firebase Realtime Database to store and synchronize product, user, and order information.
4. **Enhance User Experience:** Include a wishlist feature and personalized gift options to increase customer engagement.
5. **Create an Admin Panel:** Design a robust admin panel for product, user, and order management.
6. **Ensure Smooth Checkout:** Provide a secure and simple checkout process with dynamic shipping and payment options.

**3.3 EXPECTED ADVANTAGES**

1. **Enhanced User Experience:** Smooth navigation with product filters, wishlist, and personalized gifting.
2. **Real-Time Data Management:** Instant updates on inventory, orders, and user actions through Firebase.
3. **Secure User Authentication:** Protect user information with a safe and reliable login system.
4. **Efficient Admin Control:** Easy management of products, orders, and users through a centralized panel.
5. **Increased Engagement:** Special features like "Make It Special" gifting and wishlist options to attract and retain users.
6. **Scalability:** Ability to expand the platform to handle increased product listings and customer traffic.
7. **Accessibility:** Available on multiple devices, ensuring users can shop anytime, anywhere.

**4.1 TASK LIST**

**4.1.1 Requirement Gathering & Analysis**

* Collect and analyze functional and non-functional requirements.
* Define system requirements (Firebase for Authentication & Realtime Database).
* Document **System Requirement Specification (SRS)**.

**4.1.2 Planning**

* Create a detailed **Project Plan** with timelines and milestones.
* Conduct a background study of front-end (HTML, CSS, JS) and **Firebase**.
* Perform a **Feasibility Study** to ensure technology compatibility.

**4.1.3 Modeling**

* Identify and map the **Project Workflow** (User & Admin side processes).
* Recognize system **constraints** (real-time updates, user authentication).
* Define the **Scope** for both user and admin functionalities.

**4.1.4 System Design**

* Design **System Flow** (User navigation, data handling).
* **Database Design** (Structure for storing products, users, orders in Firebase).
* **Interface Design** (UI for product pages, wishlist, checkout, and admin panel).

**4.1.5 Coding**

* Develop the **User Interface** (Product pages, Wishlist, Checkout).
* Implement **Firebase Authentication** (Login/Sign-up).
* Set up **Realtime Database** for storing user data, product details, and orders.
* Build the **Admin Panel** for managing products, users, and orders.

**4.1.6 Testing**

* Test all web pages for functionality and performance.
* Ensure **Error Handling** and resolve identified bugs.
* Validate real-time synchronization and user authentication.

**4.3 EFFORT DESCRIPTION**

**1. Analysis**

**2. Design**

**3. Programming/Unit testing**

**4. System test**

**5. Acceptance test**

**6. Manual procedures**

**7. User training**

**8. Conversion**

**9. Technical support**

**10. Project management**

4.4 TIMELINE CHART

A timeline chart, commonly known as a Gantt chart, is a visual tool that represents the sequence and duration of project activities along a time axis. In the context of the Kiti project, this chart illustrates how different phases—such as requirement gathering, design, development, and testing—are scheduled across the project timeline.

Each task or activity is visually represented by a horizontal bar, with its placement indicating the start and end dates. The length of each bar corresponds to the estimated duration of the activity. This graphical approach helps track the flow of the project, ensuring that each task aligns with the planned schedule.

In the development of the Kiti online store, the timeline chart supports efficient project management by clearly showing the dependencies between stages such as analysis, designing, coding, testing, and deployment. It allows for timely allocation of resources and helps identify potential delays early in the process.

Moreover, timeline charts improve communication across the project team by offering a shared visual reference of milestones and deadlines. They provide transparency in tracking progress and help stakeholders stay informed about upcoming tasks and completed phases.

Beyond project management, timeline charts can also be applied in research or strategic planning to evaluate time-based data and spot patterns or trends. For Kiti, the timeline chart serves as an essential planning tool that ensures each module—like user login, wishlist, cart management, and product updates—is developed and delivered on time.

**5. SYSTEM DIAGRAM**

**5.1 UML Diagram**

UML (Unified Modeling Language) diagrams serve as the visual blueprints for the Kiti project. They illustrate how different parts of the system interact and work together. For Kiti, UML diagrams help represent key components such as the user interface (for browsing products, adding items to the wishlist or cart, etc.), Firebase Authentication for secure login, and Firebase Realtime Database for managing product and order data. These diagrams ensure that every module—from customer interactions to administrative functions—is clearly mapped out.

**5.2 Why Use UML?**

Using UML diagrams brings several benefits:

* **Clear Communication:** They provide a common language for developers, designers, and stakeholders, making it easier to understand system functionalities and design choices.
* **Visual Clarity:** UML diagrams break down complex processes into simple, visual representations. This helps identify potential issues early and streamlines the development process.
* **Documentation:** They serve as a detailed record of the system’s structure, which is useful for future maintenance and enhancements.
* **Efficient Planning:** By mapping out interactions and dependencies, UML helps ensure that all components of Kiti (like product listings, user authentication, and order management) are integrated effectively.

**5.3 Types of UML Diagrams**

For the Kiti‑Korean Store project, the following UML diagrams are especially useful:

* **Use Case Diagrams:**  
  Illustrate how different users (e.g., customers and administrators) interact with the system. They help define the functionality that the system must provide, such as browsing products, managing wishlists, or processing orders.
* **Activity Diagrams:**  
  Map out the workflow for key processes. For instance, an activity diagram can show the steps a user takes when adding a product to the cart or during the checkout process, ensuring smooth navigation and process clarity.
* **Sequence Diagrams:**  
  Show the order of interactions between various system components. For example, a sequence diagram can depict the process of user login—from entering credentials to Firebase authentication—demonstrating how data flows between the user interface and the backend services.
* **Class Diagrams (Optional):**  
  Represent the system’s data structure by showing the relationships between different classes (such as user profiles, product details, and order records). This is especially useful for planning the database schema within Firebase.