



ECO-CONNECT
AN ECO-FRIENDLY WEB APPLICATION

A PROJECT REPORT

Submitted by

PRITAM ROY CHOUDHURY (27300121046)

BITTU PANDEY (27300121032)

SHUBHAM MAJI (27300121063)

KUMARI RAJ SHILPY SINGH (27300122127)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

NSHM KNOWLEDGE CAMPUS, DURGAPUR

JANUARY 2025

BONAFIDE CERTIFICATE

Certified that this project report “**ECO-CONNECT AN ECO-FRIENDLY WEB APPLICATION**” is the Bonafide work of “**Pritam Roy Choudhury, Bittu Pandey, Shubham Maji, Kumari Raj Shilpy Singh**” who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Mr. Alok Nath Pal

SUPERVISOR

Designation: Assistant Professor

Department of Computer Science Engineering

NSHM KNOWLEDGE CAMPUS

DURGAPUR-GOI

Dr. Bijoy Kumar Mandal

HEAD OF THE DEPARTMENT

Department of Computer Science Engineering

NSHM KNOWLEDGE CAMPUS

DURGAPUR-GOI

Abstract

Eco-Connect is a cutting-edge online platform that aims to revolutionize the management of e-waste by offering a market for the purchase, sale and repair of e-waste while promoting environmentally responsible behavior in order to promote environmental protection, it also has a social function that allows users to exchange feeds and reels. Using the MERN stack (MongoDB, Express, React and Node.js), Eco-Connect makes advantage of contemporary web technologies to provide a user experience that is safe, responsive, and seamless.

The platform uses Node.js and Express.js to manage a robust back-end architecture, MongoDB for secure and scalable data storage, and React.js for an intuitive and dynamic front-end interface. Tailwind CSS enables sure that the design is responsive and occurs on all devices, and Cloudinary allows for efficient reel and image management (JSON Web Tokens) and bcrypt are employed to provide secure user authentication, protecting user data and ensuring privacy.

Eco-Connect stands out as it solves typical problems with e-waste management, such the difficulty in finding repair options, the absence of a community-driven strategy, and the lack of focus on environmental awareness. Through a fully functional eco-friendly marketplace and an engaging social experience, the platform enables both businesses and individuals to play an active role in decreasing e-waste.

In the future, there will be more collaborations with certified recyclers and repair experts, gamified incentives for environmentally responsible contributions, and AI-powered recommendations for sustainable practices. These improvements are intended to expand the platform's functionality and promote the broad adoption of sustainable practices.

With its innovative technological stack, safe infrastructure, and community-focused mission, Eco-Connect is poised to pioneer eco-friendly innovation and encourage people to join the fight for a cleaner, greener world.

LIST OF TABLES

Sl. No	Index	Page No.
1	Software Requirements Specification Table	15
2	Database Schema Design Table	22
3	User Roles and Permissions Table	31
4	Testing Results Table	41
5	API Endpoints Table	41

LIST OF FIGURES

Fig No.	Index	Page No.
1	Project Work Schedule (Gantt Chart)	12
2	Front-end Architecture of Eco-Connect	20
3	Backend Architecture of Eco-Connect	20
4	Database Schema Design (ER Model)	23
5	Flow Chart for User Registration Process	26
6	Flow Chart for User/Admin Login & Change Password	30
7	Flow Chart for Admin Functionality	32

TABLE OF CONTENTS

Sl. No.	Index	Page No.
I	ABSTRACT	3
II	LIST OF TABLES	4
III	LIST OF FIGURES	4
Chapter 1	INTRODUCTION	7 - 12
1.1	Introduction	7
1.2	Aim	7
1.3	Existing System	8
1.4	Proposed System	9-10
1.5	Feasibility Study	11
1.6	Project Work Schedule	12
Chapter 2	HARDWARE AND SOFTWARE SPECIFICATION	13 - 15
2.1	Hardware Requirement	13
2.2	Software Requirement	14 - 15
Chapter 3	DESIGN AND PLANNING	16 - 23
3.1	UI / UX Design Overview	15
3.2	Responsive Design Principles	16
3.3	UX Animation and Smooth Scrolling	19
3.4	API Design and Integration	19

Sl. No.	Index	Page No.
3.5	Database Schema Design	21 - 23
Chapter 4	AUTHENTICATION AND AUTHORIZATION	24 - 32
4.1	Registration and Authentication Flow	24 - 25
4.2	User/Admin Login and Change password Processes	27 - 29
4.3	Admin Dashboard Overview	31
4.4	Admin Controls & Management	31
Chapter 5	TESTING AND DEPLOYMENT	32 - 40
5.1	Unit Testing	33
5.2	Integration Testing	33 – 35
5.3	Configuration and set-up	36 – 37
5.4	Deployment Process	38 - 39
	RESULTS	41
	EXISTING SYSTEM CHALLENGES AND SOLUTIONS	42 - 44
	FUTURE SCOPE	45
	CONCLUSION	46
	REFERENCES	47

1.1 INTRODCUTION

Sustainable solutions are more important than ever in a time when environmental problems are getting worse. Eco-Connect is a cutting-edge online platform created to tackle the urgent problems of recycling, e-waste management, and repair while encouraging a neighborhood-based strategy for environmental preservation. Eco-Connect was developed with the MERN stack (MongoDB, Express.js, React.js, and Node.js) and blends state-of-the-art technology with a user-centric design to give users a smooth and memorable experience.

In addition to being a comprehensive marketplace for the purchase, sale, and repair of e-waste, the platform features a lively social area where users can exchange feeds and reels to encourage environmentally sustainable behavior. Eco-Connect guarantees both functionality and security using tools like Cloudinary for media management, Tailwind CSS for responsive design, and secure authentication methods that use bcrypt and JWT.

The Eco-Connect movement seeks to empower individuals, decrease waste, and encourage sustainable living. There is more to it than just a market. Through the integration of scalable technology and interactive elements, the platform establishes a new benchmark for environmentally conscious innovation in the digital era.

1.2 AIM

To enable consumers to take concrete actions toward environmental sustainability, Eco-Connect seeks to close the gap between e-waste management issues and digital solutions. Through a fun social experience, the platform hopes to establish a complete ecosystem for the purchase, sale, and repair of e-waste while promoting environmentally beneficial behavior.

Eco-Connect makes use of the MERN stack (MongoDB, Express, React, and Node.js) in conjunction with solutions like Tailwind CSS for responsive design and Cloudinary for media management to guarantee a productive, safe, and intuitive experience on all devices. By allowing users to share feeds, reels, and tips, the platform also emphasizes community interaction, encouraging cooperation and environmental awareness.

Eco-Connect's ultimate goal is to establish a standard for environmentally responsible innovation by fusing cutting-edge technology, expandable infrastructure, and a simple design. To encourage people and companies to help create a cleaner, greener world, Eco-Connect aims to make sustainability approachable and manageable.

1.3 Existing System

Features that encourage community involvement and sustainable behavior are frequently overlooked by the current e-waste management systems, which mainly concentrate on fundamental functions like the collecting and recycling of electronic debris. User participation and access to environmentally beneficial solutions are restricted by the majority of systems' lack of an integrated marketplace for the purchase, sale, or repair of e-waste (Fig. 1.0). Additionally, social components that are essential for promoting a sustainable culture—like sharing environmental advice or awareness content—are absent from these platforms.

Current systems' user interfaces lack contemporary design concepts and are frequently generic, which makes them less engaging and intuitive. The whole user experience is further diminished by poor performance in terms of load times and limited responsiveness across devices. Basic user authentication and other security measures are typically insufficient to sufficiently safeguard user data or transactions. Furthermore, present systems rarely provide community-driven features like feeds or reels that could motivate users to adopt eco-friendly practices, nor do they often provide educational content. The importance of sustainability and environmental preservation cannot be sufficiently addressed due to the absence of participatory elements.

Comparatively, Eco-Connect overcomes these drawbacks by combining a social media platform that promotes the exchange of information and experiences regarding sustainable practices with a dynamic marketplace (Fig. 1.1). Its user-centered design makes advantage of the MERN stack to provide smooth performance, and Tailwind CSS and other tools guarantee a clear and responsive user experience. A complete system for managing e-waste and promoting environmental awareness, Eco-Connect also places a high priority on security with strong user authentication. This strategy fosters a thriving, cooperative society dedicated to protecting the environment in addition to advancing sustainability.

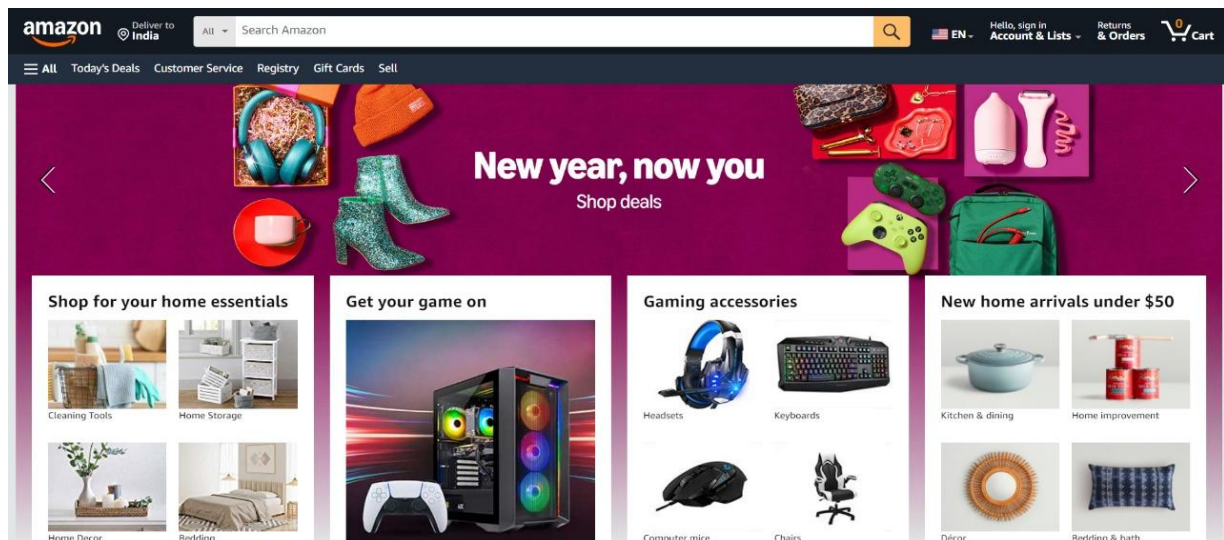
1.4 PROPOSED SYSTEM

By offering a complete, user-friendly solution for purchasing, selling, and repairing e-waste as well as encouraging environmental awareness through community participation, the proposed Eco-Connect system aims to overcome the shortcomings of current e-waste management platforms (Fig. 1.2). As opposed to conventional systems, Eco-Connect has a social aspect that allows users to exchange feeds, reels, and environmentally responsible behaviour, fostering a thriving community dedicated to sustainability.

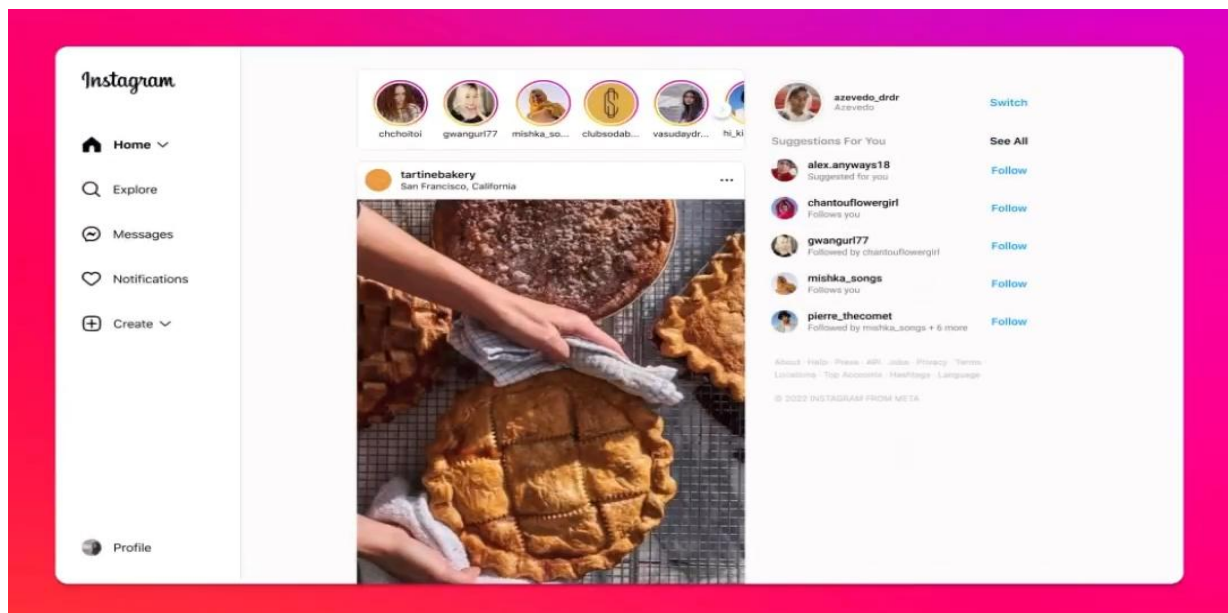
The suggested solution uses a contemporary, responsive UI developed with the MERN stack (MongoDB, Express.js, React.js, and Node.js) to improve usability. While Cloudinary facilitates effective media management for reel uploads and sharing, Tailwind CSS is used for a simple and flexible design (Fig. 1.3). An easy and secure user experience is guaranteed by features including a fluid marketplace interface, dynamic search and filtering options, and a secure user authentication system driven by JWT and bcrypt.

Personalization and security are important features of the suggested system. Role-based access is used by Eco-Connect to distinguish between features for administrators, merchants, and purchasers. Admins can monitor platform activity using tools, and sellers may manage listings via dashboards. Advanced features on the platform include gamified incentives to promote user engagement and AI-powered suggestions for environmentally responsible behaviour (Fig. 1.4).

Through the integration of scalable technology, engaging social elements, and a secure marketplace, the Eco-Connect system builds a strong ecosystem for the sustainable management of e-waste. up addition to filling up the current gaps in accessibility and community participation, it also establishes a new benchmark for environmentally friendly innovation, motivating people to make a positive impact on the globe.



Existing Websites fig 1.0



Reel Fig 1.1



Eco-Connect Clean UI Fig 1.2

1.5 FEASIBILITY STUDY

Through an analysis of technological, operational, financial, and regulatory factors, the Eco-Connect feasibility study assesses the practicality of creating a platform for e-waste management and community-driven environmental awareness.

Feasibility of Technology

The MERN stack (MongoDB, Express, React, and Node.js) is used in the construction of Eco-Connect to ensure scalability, security, and a dependable development process. React.js enables the creation of a dynamic and responsive user interface, while Node.js and Express.js provide a stable backend infrastructure. MongoDB makes ensuring that user profiles, listings, and transaction histories are stored and managed effectively. Furthermore, the site uses JWT and bcrypt for secure authentication and incorporates Cloudinary for reel and picture management. A smooth user experience is produced by Tailwind CSS's responsive design on all devices.

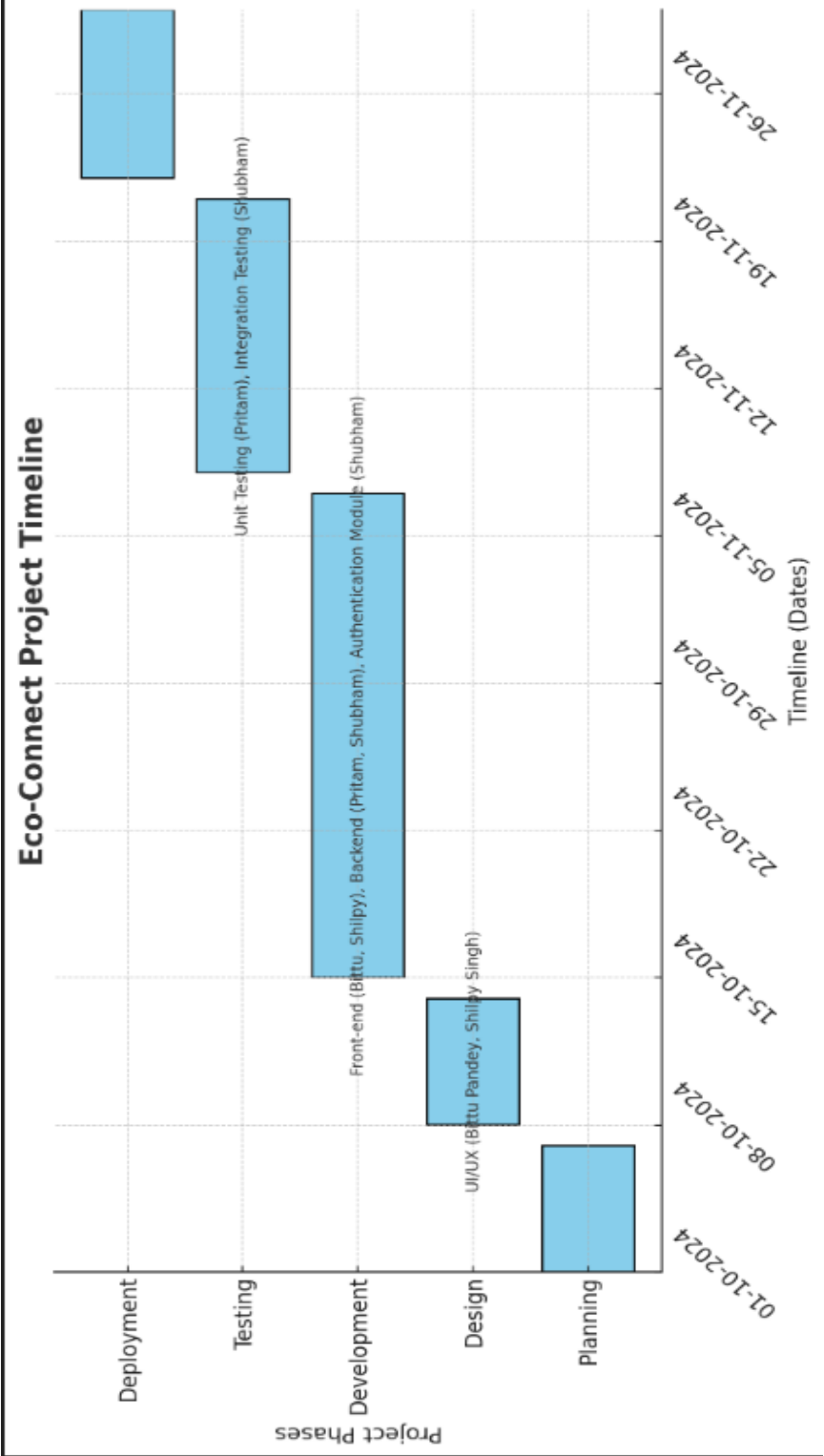
Operational Feasibility

The development team has the resources and know-how to deploy Eco-Connect, including proficiency in security frameworks, UI/UX design, and the MERN stack. Before launching, a systematic testing procedure that includes unit, integration, and user acceptability testing will guarantee platform stability and dependability. Although admin dashboards will enable effective platform administration, including tracking listings and resolving user difficulties, a specialized support system will be set up for user questions.

Legal Feasibility

All applicable laws are followed by Eco-Connect, including those pertaining to intellectual property rights, data protection (such as GDPR and its local counterparts), and e-waste management. Transparency and lower legal risks are ensured by explicit terms of service and privacy policies. Compliance with environmental regulations will be further guaranteed by collaborations with qualified recyclers and repair specialists.

Project Work Schedule (Gantt Chart)



2.1 HARDWARE REQUIREMENTS

In order to guarantee optimum performance, scalability, and a smooth user experience, the Eco-Connect platform has specific hardware requirements. Both server and client specs fall under this category.

1. Server Requirements

Secure user authentication, effective media management, and large-scale transactions must all be supported by the back-end server architecture.

- ◆ Processor (CPU): To handle high concurrency, a multi-core processor (such as AMD EPYC or Intel Xeon) with a minimum clock speed of 3.0 GHz.
- ◆ RAM: A minimum of 32 GB is required to accommodate media uploads, real-time transactions, and heavy traffic.
- ◆ Storage: We need at least 500 GB of SSD storage to store media files, user profiles, and e-waste listings, as well as to enable quick data access. choices for cloud storage (like Google Cloud Storage or AWS S3) for scalability.
- ◆ Network: Fast and dependable data transfer requires a high-speed internet connection with at least 1 Gbps of capacity.
- ◆ Backup Options: Cloud-based backup options that guarantee catastrophe recovery and data redundancy through automated scheduling.

2. Client Requirements

For Eco-Connect users, even on mid-range devices, a configuration that guarantees a responsive and seamless experience is necessary.

- ◆ CPU: A dual-core or quad-core processor (such as an Intel Core i3 or similar model) with a minimum clock speed of 1.8 GHz.
- ◆ Memory (RAM): For basic users, a minimum of 4 GB is required, but for power users, such as administrators or merchants, 8 GB.
- ◆ Storage: To cache and temporarily store media files, there should be at least 1 GB of free disk space.
- ◆ Network: dependable internet access with a minimum 5 Mbps speed for smooth operation, particularly while uploading media and viewing reels.

2.2 SOFTWARE REQUIREMENT

The software requirements for the Eco-Connect application include essential tools and technologies for effective development and operation, categorized as follows:

1. Frontend Requirements

- Framework: React.js for building a responsive user interface.
- Styling: Tailwind CSS for a modern design with utility-first CSS.
- Routing: React Router for seamless navigation.
- State Management: Context API for managing global state.

2. Backend Requirements

- Runtime Environment: Node.js for efficient handling of asynchronous requests.
- Web Framework: Express.js for simplified server-side logic and API development.
- Authentication: JSON Web Tokens (JWT) for secure user authentication.

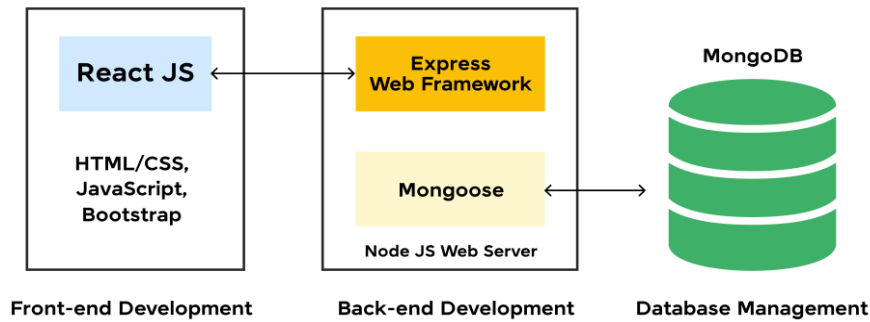
3. Database Requirements

- Database Management System: MongoDB for storing user data and news articles, providing flexibility and scalability.

4. Testing and Deployment Services

- API Integration: MediaStack API for real-time news fetching and Google Maps API for location services.
- Development Tools: Visual Studio Code for coding and Git for version control.
- Testing Framework: Postman for testing API endpoints and ensuring functionality.
- Deployment: Vercel and Render for reliable hosting and performance.

MERN Stack Development



Software Requirements Specification Table	
Section	Details
Key Technologies	MERN (MongoDB, Express.js, React.js, Node.js), Tailwind CSS, Lenis (smooth scrolling)
Core Features	Responsive design, OTP email verification, Admin panel for user management
User Roles	User: Access news, account creation, Admin: User management, blocking, contact review
Security	OTP-based login, secure password Change
Advantages	Enhanced speed, security, and user engagement
Challenges Addressed	Solves responsiveness, speed, and security issues in existing platforms

Table 1

3.1 UI / UX Design Overview

User Interface (UI) Design

- Layout: The layout features a modern, grid-based structure with clear sections for marketplace, feeds, reels, and user dashboards. This organization ensures seamless navigation and content discoverability.
- Color Palette: A refreshing color palette inspired by nature, using shades of green, blue, and earthy tones, conveys the platform's eco-friendly message. A dark mode toggle enhances user comfort in low-light settings.
- Typography: Clean, sans-serif fonts with appropriate font sizes and spacing ensure readability. Bold headings and subtle subtext differentiate content categories effectively.
- Icons and Graphics: Custom-designed eco-themed icons and visuals are used throughout the platform to provide intuitive navigation and reinforce the sustainability theme.
- Media Handling: Reels and images are displayed in a responsive carousel format, optimized for both desktop and mobile users.

User Experience (UX) Design

- Navigation: Marketplace listings, reels, feeds, and user settings are all easily accessible through categorized links on a sticky top navigation bar. Usability is further improved with breadcrumb navigation and drop-down menus.
- Search Functionality: Users may easily locate e-waste listings or community postings thanks to a large, AI-powered search bar with filtering options.
- Interactive Elements: The interface is made more engaging by smooth animations (driven by libraries like GSAP) and hover effects that give feedback on user activities. Fast performance is ensured by lazy loading for reels and photos.
- Community Engagement: Interaction and the sharing of environmentally friendly practices are encouraged by areas specifically designed for user-generated feeds and reels. Buttons for sharing, like, and commenting improve social interaction.
- Accessibility: All users may utilize the design since it complies with WCAG guidelines and includes features like keyboard navigation support, appropriate color contrast, and alt text for pictures.

3.2 RESPONSIVE DESIGN PRINCIPLES

Delivering a smooth and simple user experience across many devices is the main goal of Eco-Connect's responsive design principles, which guarantee usability and accessibility for all users on desktop and mobile platforms.

1. A Flexible grid layout (Figure 1.3)

Using a fluid grid pattern instead of set dimensions, Eco-Connect scales items proportionately. This guarantees that the content is consistently organized and aesthetically pleasing across all screen sizes.

Whether on a small mobile device or a large desktop, listings, feeds, and reels are presented in an orderly fashion.

2. Adaptable Picture and Media (Figure 1.4)

The flexible design of images, video and other media components allows them to seamlessly adapt to their containers without sacrificing quality. This guarantees ideal display with no loss of clarity or overflow problems.

improved loading speeds by using technologies like Cloudinary for picture and video optimization and dynamic scaling.

3. Media Queries (Figure 1.5)

The platform's appearance may be adjusted to fit various screen sizes and orientations by using CSS media queries. This makes it possible for customized font sizes, spacing, and layouts according to device requirements. An uniform experience across breakpoints, including desktops at 1200px, tablets at 768px, and tiny displays at 320px.

4. Mobile-First Approach (Figure 1.6)

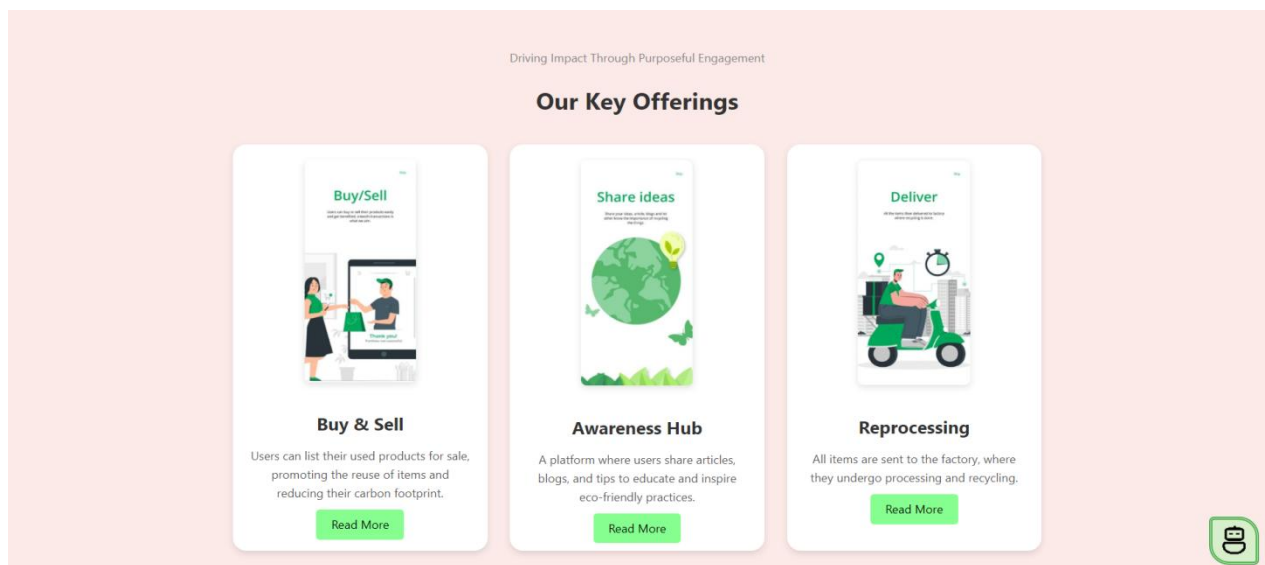
Mobile displays are given top priority from the start of the design process, guaranteeing smaller devices can access essential capabilities including media upload, search, and navigation. A streamlined user experience that adapts to larger screens with ease.

5. Touch-Friendly Elements(Figure 1.7)

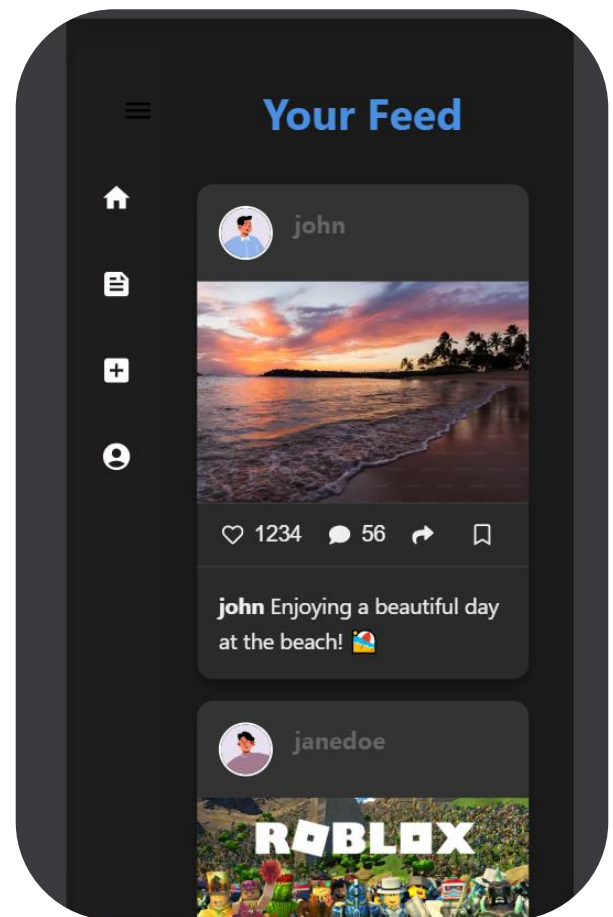
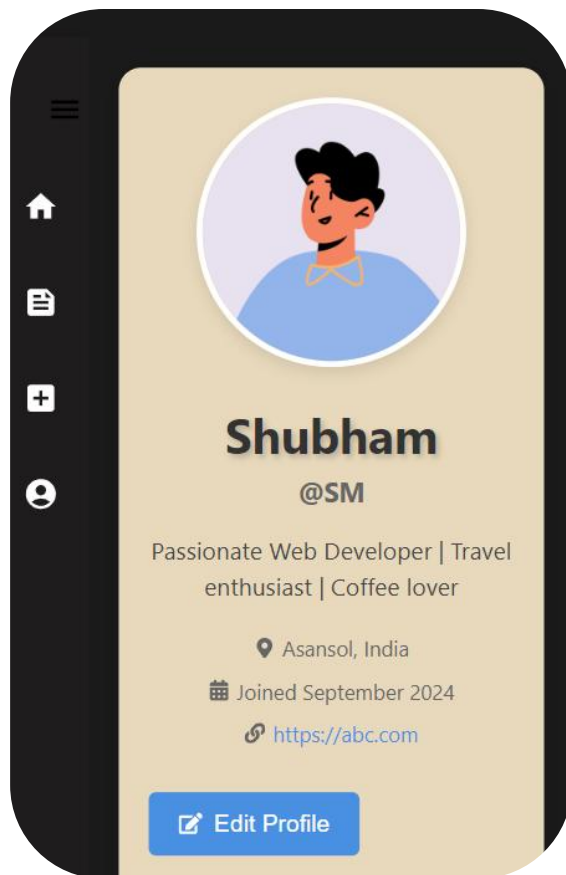
In order to accommodate touch-based devices, interactive components such as carousels, sliders, and buttons are made with Proper spacing is necessary to avoid unintentional clicks. Click and hover states that are responsive to give actions visual feedback.



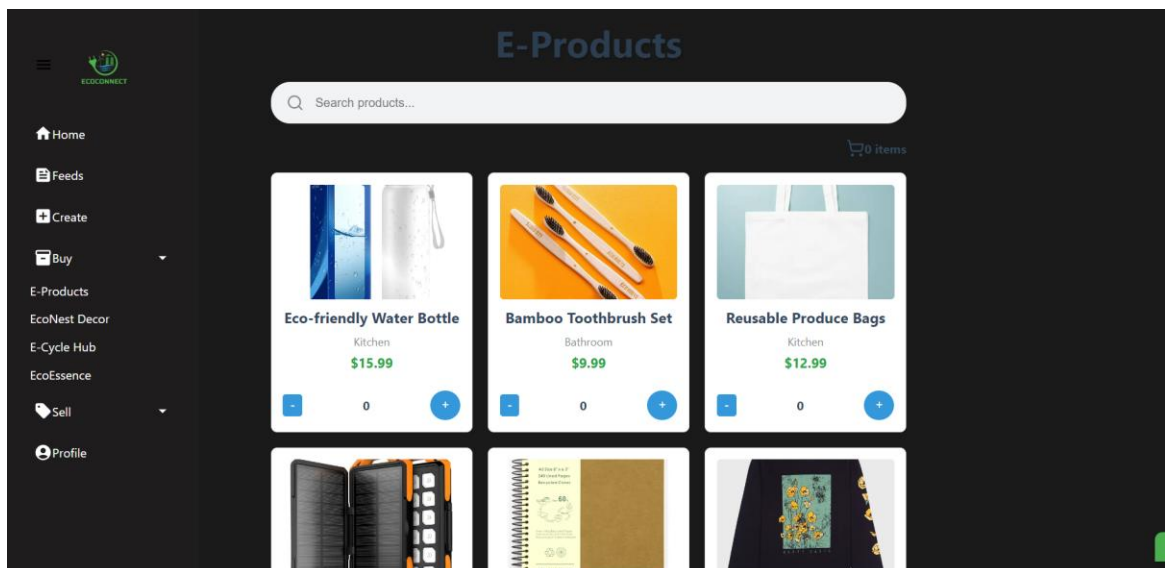
Flexible Grid Layout (Figure 1.3)



Flexible Images and Media (Figure 1.4)



Mobile-First Approach (Figure 1.6)



Media Queries (Figure 1.5)

3.3 UX ANIMATION AND SMOOTH SCROLLING

- **Interactive Feedback:** By graphically highlighting actionable components, buttons, links, and icons with hover and click animations help users navigate.
- **Page Transitions:** With delicate fade-ins and slide-ins while navigating between areas like the marketplace, reels, and user profiles, seamless page transitions guarantee a seamless surfing experience.
- **Reel Playback Effects:** Smooth play/pause animations are incorporated into reel elements to guarantee that users may interact with material in a natural manner.
- **Loading Animations:** To give users a visual representation of system activities, spinners and skeleton loaders are employed during data retrieval.
- **Scroll Optimization:** To prevent sudden leaps and provide a smooth flow, the platform makes use of libraries like Lenis or Smooth Scroll to make gentle transitions between parts.
- **unlimited Scrolling:** Users may read material continually without having to reload pages thanks to feeds and listings that support unlimited scrolling.
- **Scroll indications:** Reels and feeds with progress indications let users keep track of their navigation and promote exploration.
- **Performance considerations:** To ensure a flawless experience across all platforms, smooth scrolling is performance-optimized to avoid latency or stuttering.

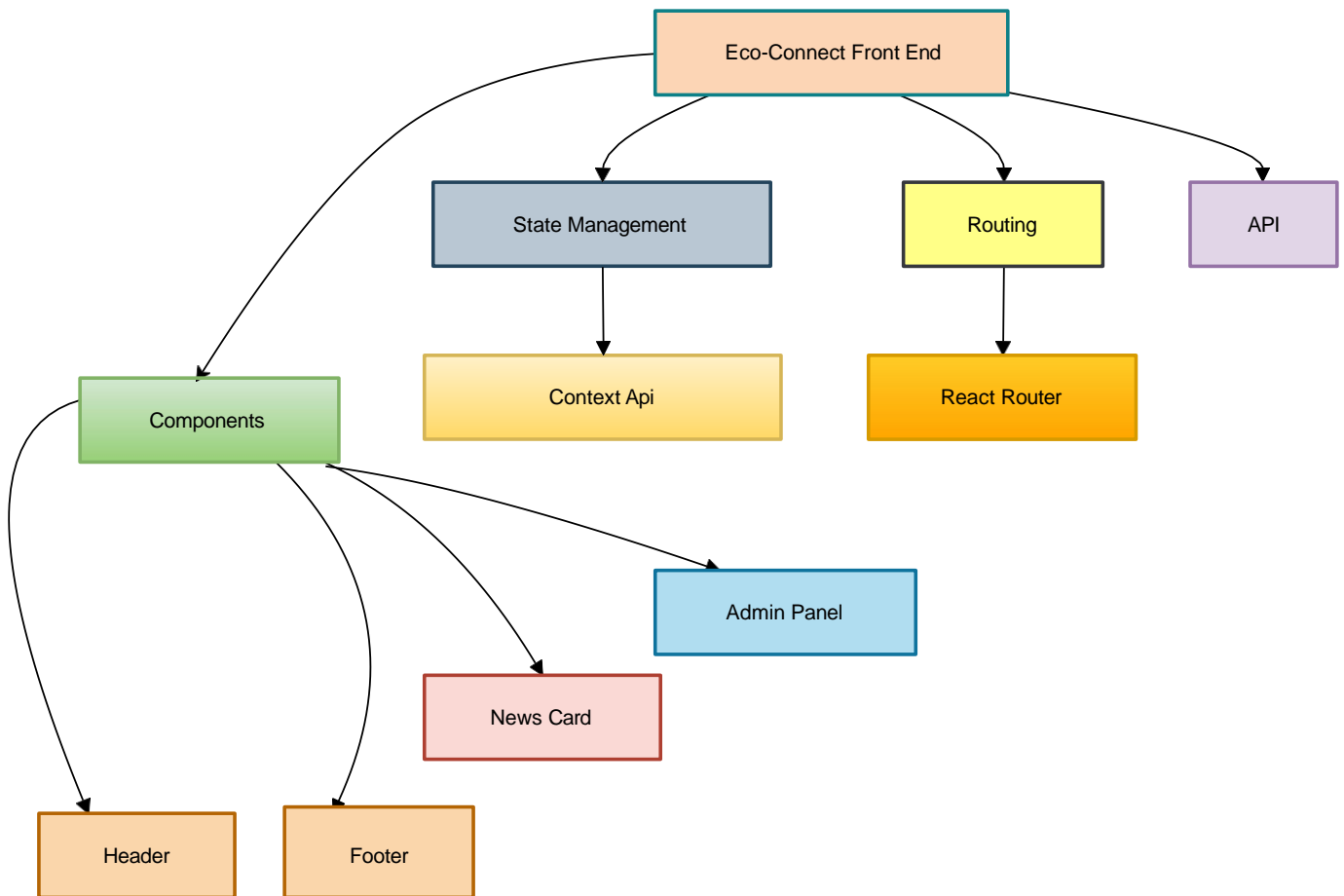
3.4 API DESIGN AND INTEGRATION

The API design and integration for the Eco-Connect application are essential for facilitating seamless communication between the frontend and backend components. This architecture enables the application to efficiently retrieve and display news articles while maintaining user interactions and data integrity.

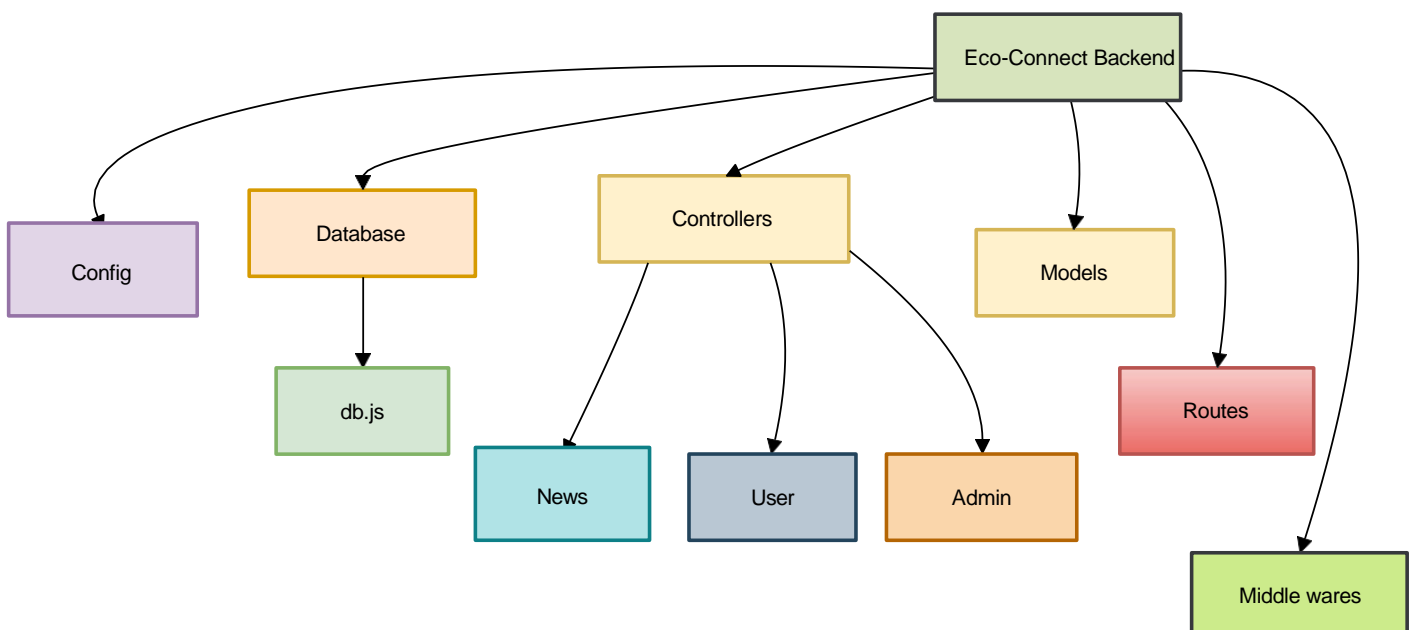
User Management Routes

- **POST /api/auth/register:** Handles user registration by accepting necessary user data and creating a new account.
- **POST /api/auth/login:** Authenticates users and returns a token for session management.
- **GET /api/auth/user:** Fetches details for the currently authenticated user.
- **PATCH /api/auth/user/update:** Updates user profile information for the authenticated user.

Front-End Architecture of Eco-Connect



Backend Architecture of Eco-Connect



3.5 DATABASE SCHEMA DESIGN

The "Eco-Connect" database schema is made to effectively handle user, eco-related content, project, category, and admin data. The schema ensures scalability and dependability while organizing data into organized entities. Referring to the Entity-Relationship Diagram (ERD), the main entities and their connections are broken down in depth below.(Fig. 1.8)

Key Entities in Eco-Connect:

1. **User:** Manages user information and roles.

- **id** (Primary Key): Unique identifier for each user.
- **fullName, email (unique), phoneNumber:** Contact information.
- **password:** Securely hashed password for authentication.
- **isAdmin** (Boolean): Indicates if the user is an admin.
- **isVerified** (Boolean): Tracks email/phone verification status.
- **createdAt, updatedAt:** Timestamps for record management.

2. **Project:** Stores details about eco-friendly projects.

- **id** (Primary Key): Unique identifier for each project.
- **title, description:** Basic project details.
- **categoryId** (Foreign Key): Links to the Category table.
- **createdBy** (Foreign Key): Links to the User table.
- **createdAt, updatedAt:** Track project creation and updates.

3. **Category:** Defines the classification of eco-related content and projects.

- **id** (Primary Key): Unique identifier for each category.
- **name (unique):** Name of the category (e.g., Recycling, Renewable Energy).
- **description:** Additional details about the category.

4. **Contact:** Manages eco-related blogs, articles, or resources.

- **id** (Primary Key), **fullName, email, text:** Basic message details.
- **createdAt:** Date when the message was created.

2. **Admin**: Holds information for admin users who manage the system.

- **id** (Primary Key), **username**, **password**: Admin login details.
- **createdAt**: Timestamp for record creation.

Relationships:

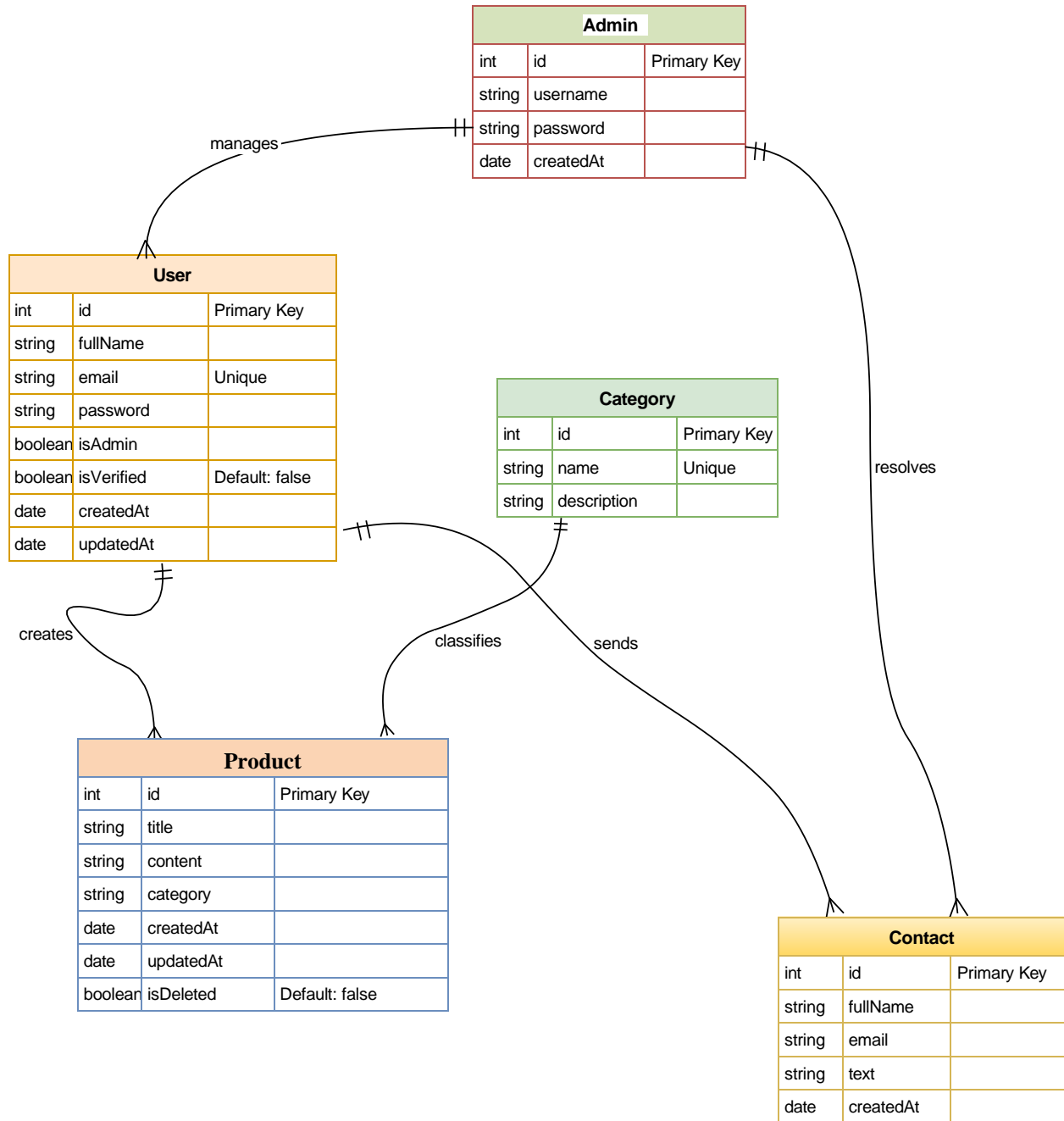
- **User–Project**: A one-to-many relationship where a user can create multiple eco-friendly projects.
- **User–Admin** : Admins are a subset of users, distinguished by the **isAdmin** Boolean in the User table.
- **Project–Category**: The **categoryId** foreign key in the Content table links to the Category table.
- **User–Contact**: A one-to-many relationship where a user can send multiple messages or inquiries.
- **Admin–Project**: Admins oversee and manage the approval process for projects and content, ensuring they meet platform guidelines.

The ERD in Fig. 1.8 visually depicts these entities and relationships, showing how data is structured and interlinked within the Eco-Connect database.

Database Schema Design Table	
Entity	Attributes
Users	user_id, email, password
Posts	post_id, user_id, title, description, media_url, category, created_at
Comments	comment_id, post_id, user_id, content, created_at
Likes	like_id, post_id, user_id, created_at
Categories	category_id, name, description
Admin_Actions	action_id, admin_id, action_type, timestamp
Contact_Requests	contact_id, user_id, fullname, email, message, created_at

Table 2

Entity-Relationship Model of Eco-Connect



4.1 REGISTRATION AND AUTHENTICATION FLOW

The application's registration and authentication process is made to provide a smooth user experience while maintaining strong security. New users can safely register, log in, and efficiently manage their session using this flow, which protects their personal data all along the way.

Registration Flow

1. User Registration:

- **Username:** Chosen by the user for their profile.
- **Email Address:** Unique identifier for communication and login.
- **Password:** Must meet security requirements (e.g., minimum length, special characters, etc.).
- **Confirm Password:** Ensures that users entered their password correctly.

2. Data Validation:

- All required fields are filled.
- The email format is correct.
- Passwords meet strength criteria (length, special characters, etc.).
- Confirm password matches the initial password.

3. Account Creation:

If the validation is successful, the application creates a new user account in the database, assigning a unique user_id and setting the isAdmin field to false by default.

Authentication Flow

1. User Login:

- Registered users can log in by providing their **email** and **password**.
- The application ensures that the credentials are validated against the stored records in the database.

2. Credential Verification:

- Upon submission, the system compares the entered credentials with the stored ones.
- If the credentials match, the system grants access and generates a **session token**.

3. **Token Generation:** Upon successful authentication, the application generates a **JWT** (JSON Web Token) and sends it back to the user. This token is used for maintaining the user's session and accessing protected routes.

4. **Session Management:** The token is stored securely (e.g., in HTTP-only cookies) to prevent unauthorized access. Users are then redirected to their personalized dashboard or the home page.

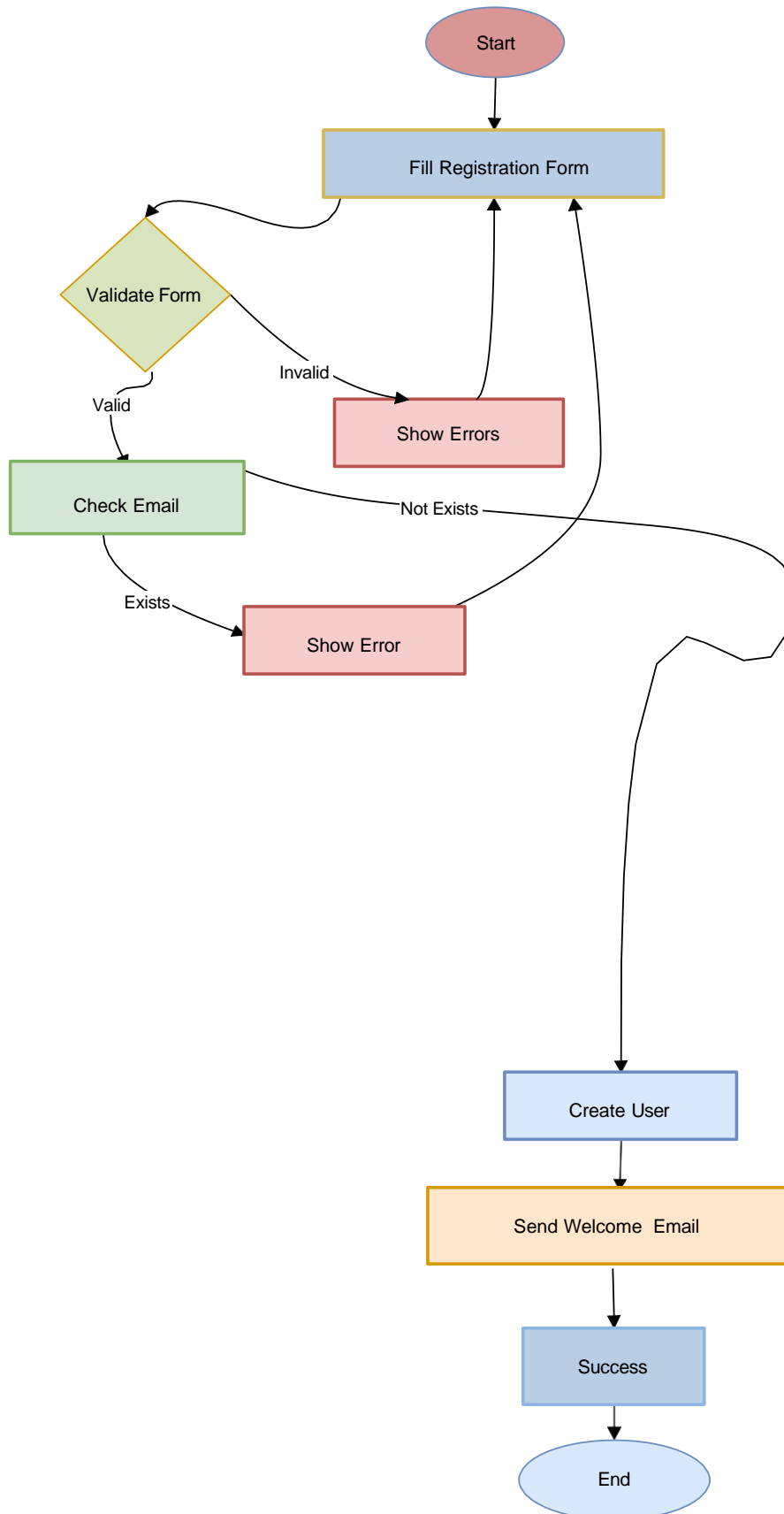
5. **Logout:** Users can log out, which clears their session and invalidates the token, ensuring that their account remains secure.

6. **Password Hashing:** User passwords are hashed using a strong hashing algorithm (e.g., bcrypt) before storing in the database.

7. **Email Verification:** An email verification step adds an extra layer of security to confirm the authenticity of the registered email address.

8. **Encryption:** Sensitive user data, including session tokens and personal information, is encrypted during transmission (using SSL/TLS) and in storage.

Flow Chart For Registration Process



4.2 USER/ADMIN LOGIN PROCESSES

User Login Process

1. **Login Interface:**
 - Users navigate to the login page, where they are prompted to enter their **registered email** and **password**.
 - The login page is designed to be intuitive, with clear labels for each field and a “Forgot Password” link for users who have trouble remembering their login credentials.
2. **Credential Verification:** Upon submission, the application checks the entered credentials against the database. If the credentials are correct, the user is authenticated.
3. **Token Generation:**
 - After successful authentication, the application generates a **JSON Web Token (JWT)**.
 - This token acts as proof of the user’s identity and is required for accessing protected routes and services within the application.
4. **Session Management:** The generated JWT is securely stored in an **HTTP-only cookie** to mitigate cross-site scripting (XSS) attacks.
5. **Access Control:** Users can access routes intended for them, including the news feed, their profile, and the contact page.

Admin Login Process

1. **Admin Login Interface:** Administrators submit their admin-specific email address and password to access the same login screen as users.
2. **Credential Verification:** The application checks the database for the administrator's credentials. The administrator is authenticated if it is legitimate.
3. **Token Generation:** Upon successful authentication, the application generates a JWT specifically for the admin.
4. **Session Management:** The admin’s JWT is also stored in an HTTP-only cookie, ensuring a secure session.
5. **Access Control:** Regular users cannot access privileged routes like user management, content moderation, and site statistics; only administrators can.

4.3 USER/ADMIN CHANGE PASSWORD PROCESS

Login Requirement:

- To begin the process, users or admins must first log in with their current credentials (email and password).
- Once logged in successfully, the user or admin is directed to their respective dashboard.

Accessing Password Management:

- From the dashboard, the user or admin navigates to the **top-right corner** where a drop-down menu is available.
- In this drop-down menu, users select the "**Change Password**" option, which opens the **Change Password page**.

Change Password Page:

- On the password management page, a form is presented with the following fields:
 - **Current Password:** The user or admin must enter their current password for verification.
 - **New Password:** The new password that the user/admin wants to set. The password must meet security requirements (e.g., length, complexity).
 - **Confirm New Password:** To ensure accuracy, the new password must be re-entered in this field.

Password Validation:

- Upon submission, the application validates the entered information:
 - The current password is checked against the stored password in the database.
 - The new password and confirmation password are compared to ensure they match.
 - The new password must also meet predefined security standards (e.g., length, special characters).
- If any validation fails, an error message is displayed, and the user/admin is prompted to correct the issues.

Password Update:

- If the new password is successfully validated, it is securely hashed and updated in the database, replacing the old password.
- The user/admin is then notified with a success message, confirming that the password has been changed successfully.

Session Management:

- After the password change, the user or admin is usually logged out to ensure security, requiring them to log in again with the new credentials.
- Alternatively, the user can remain logged in, depending on the application settings.

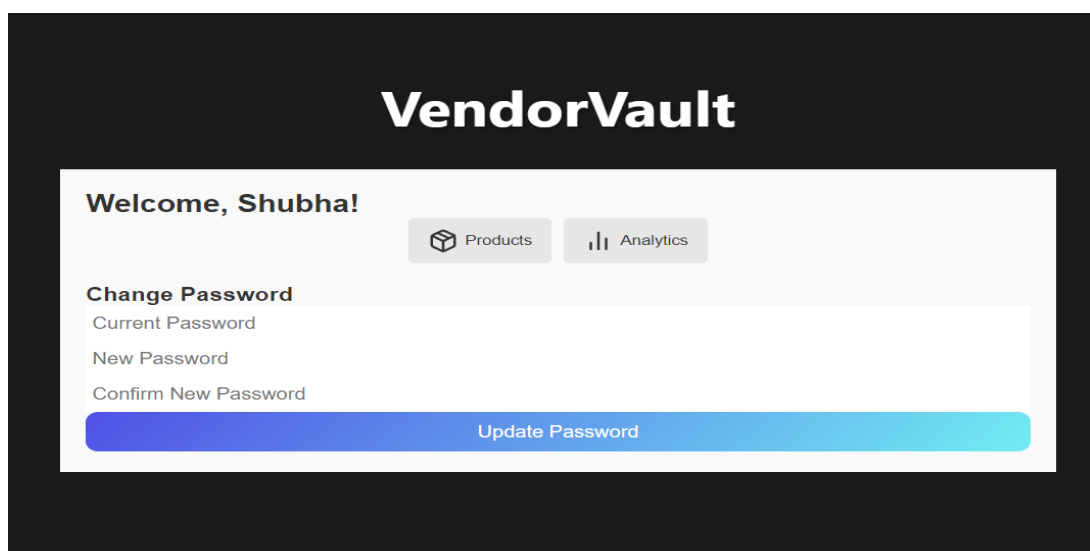
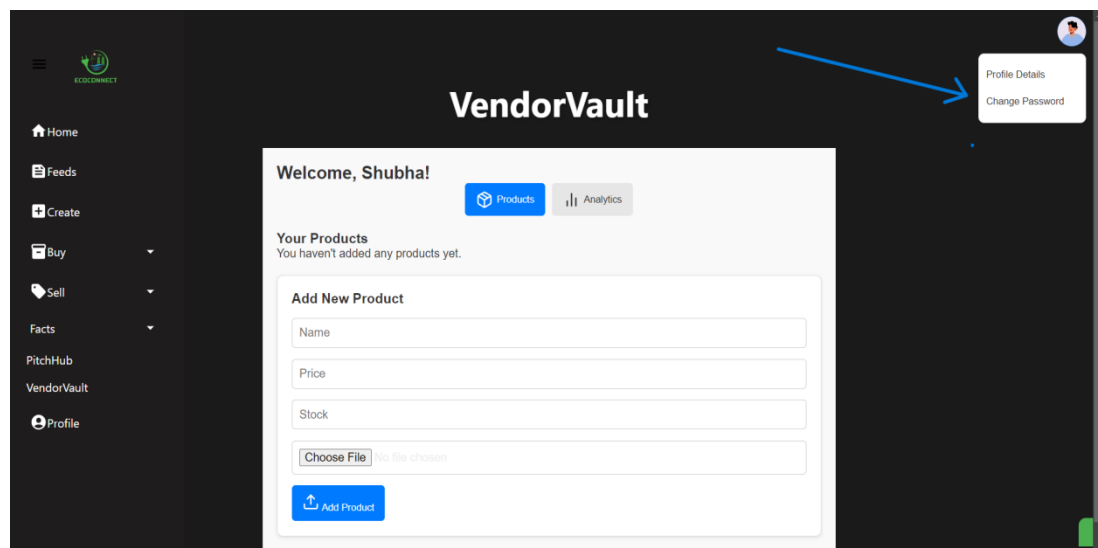
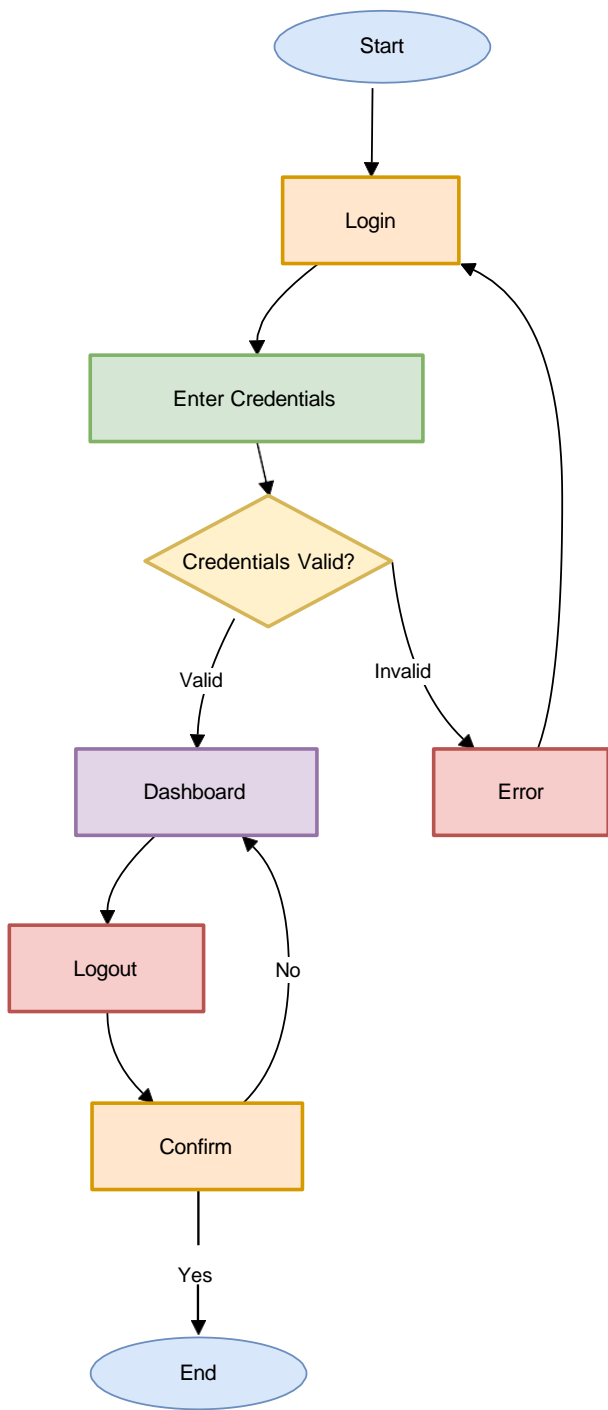
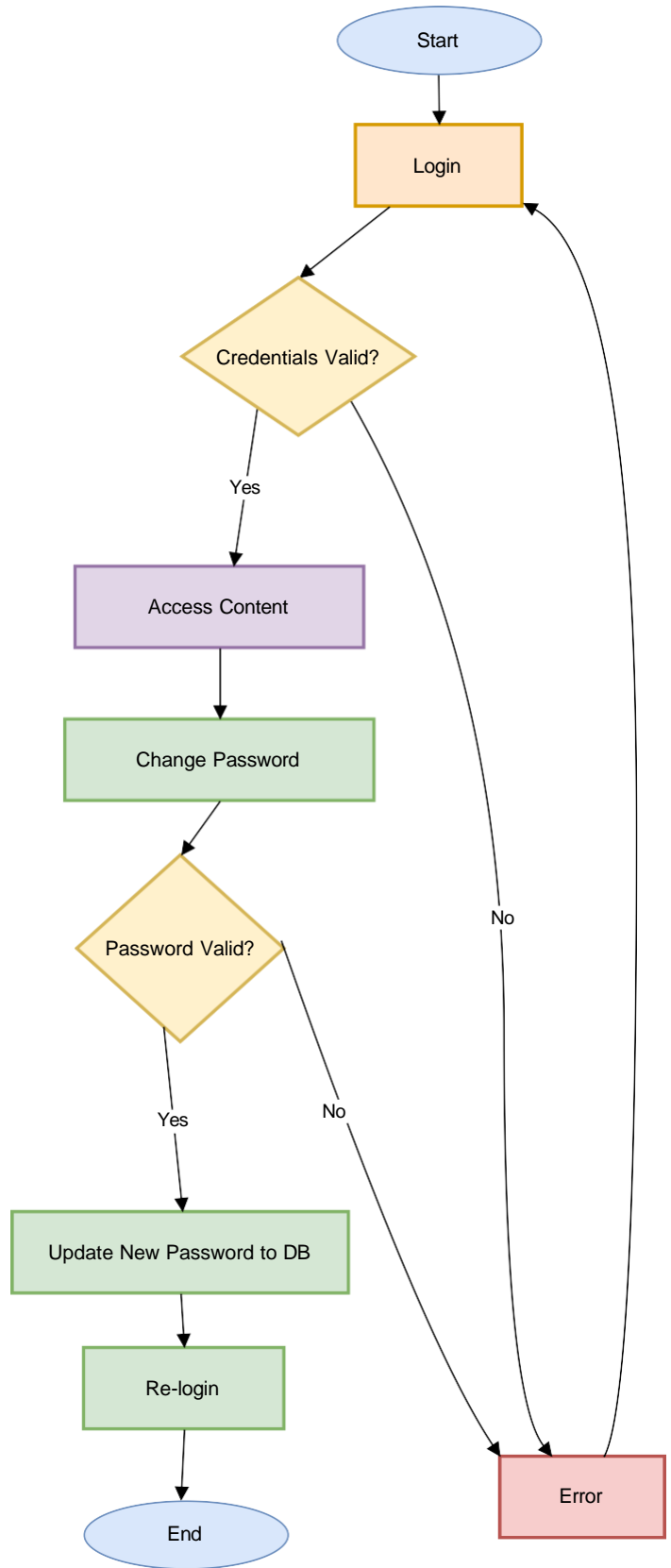


Fig.1.7

Flow Chart for Login Process



Flow Chart for Change Password Functionality



4.4 ADMIN DASHBOARD OVERVIEW

The Admin Dashboard in Eco-Connect application is a consolidated interface made to give administrators the resources they need to effectively manage the program. Admins may examine, approve, or suspend members based on their compliance with community norms using this user-friendly interface. To preserve the platform's quality and integrity, administrators may examine, approve, alter, or remove user-generated material using the dashboard's robust content moderation options in addition to user management. Along with comprehensive analytics and performance measurements, the dashboard provides information on system health, content performance, and user engagement. Role-based access guarantees that only authorized administrators have access to sensitive areas, and notifications alert them to significant occurrences like user activity or flagged content.

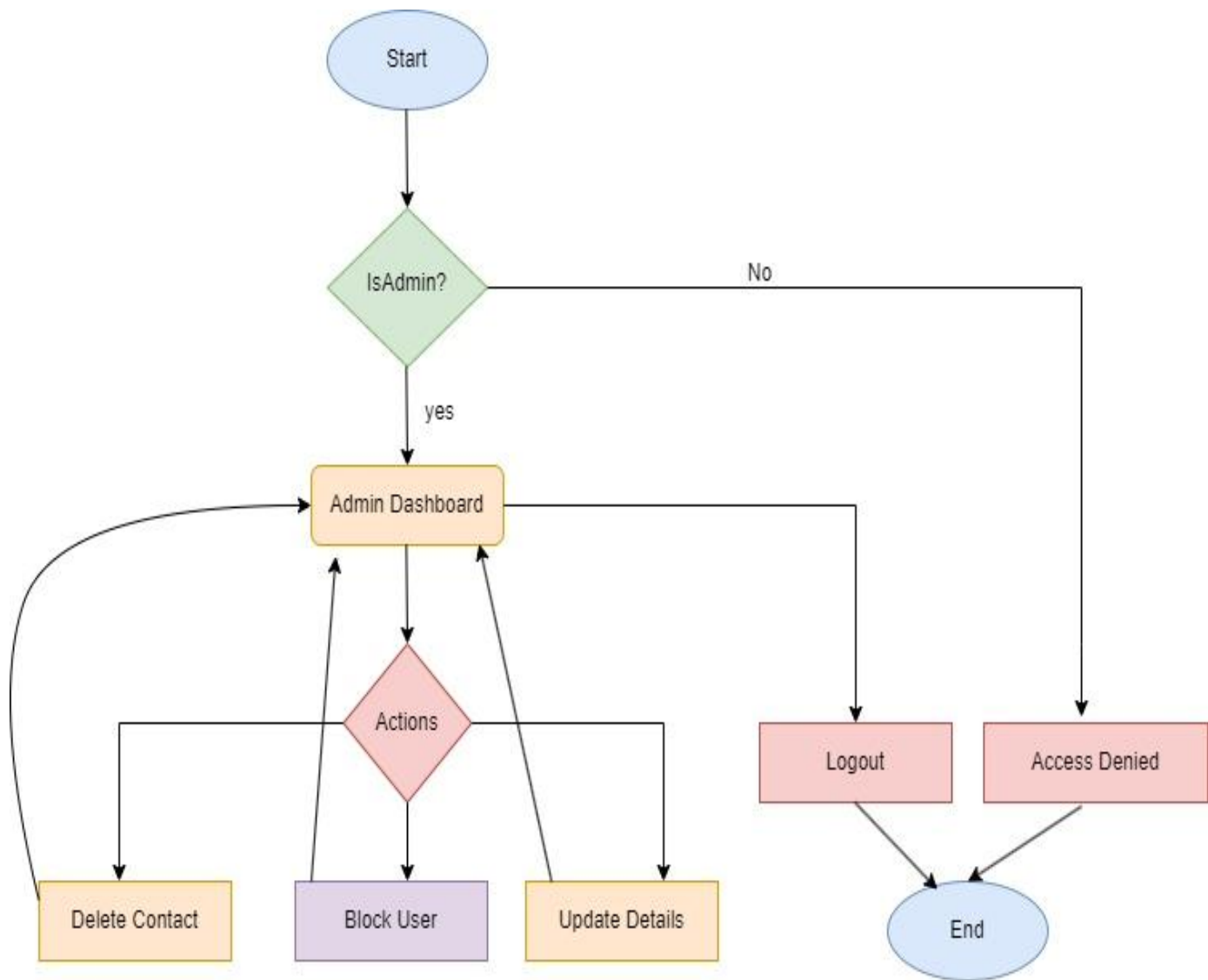
4.5 ADMIN CONTROLS & MANAGEMENT

The platform's admin controls are intended to give administrators complete control and supervision over the platform, guaranteeing its seamless and safe operation. With the use of these controls, administrators may carry out crucial duties like approving, amending, or deleting information before it is made public. To make sure that user submissions, news items, and other material adhere to platform standards, administrators can examine them. The admin panel's robust analytics capabilities, in addition to content management, let administrators monitor site traffic, analyze user interaction, and spot popular content—all of which facilitate data-driven decision-making. Role-based access control (RBAC) is another feature that the platform uses to make sure that only authorized users may access sensitive features and carry out important administrative duties.

User Roles and Permissions Table	
Role	Permissions
User	View Reels, View Blog, Upload Blog, Pitching Ideas, Create Account, Change Password, Buy Products, Sell product
Admin	Manage Users, Edit/Delete Content, Security and Audit logs, System Settings

Table 3

Flow Chart For Admin



5.1 UNIT TESTING

Unit testing is essential to guaranteeing the platform's dependability and performance. It emphasizes on confirming that every part of the application—from frontend features to backend operations—functions as intended. In addition to making it simpler to find and address defects before they become more serious, this enables developers to discover problems early on and keeps changes made to one portion of the program from unintentionally impacting other areas.

For unit testing, the application makes use of the Jest and React Testing Libraries. The React Testing Library makes it possible to simulate user interactions and render React components in a virtual document object model (DOM), while Jest acts as the assertion library and test runner. Critical tasks like rendering components, managing user authentication, verifying form submissions, and examining UI responsiveness across various devices are all covered by unit tests.

5.2 INTEGRATION TESTING

The Eco-Connect web application's integration testing makes sure that many modules and components function as a unit. Verifying the interactions between the frontend, backend, and external services is the main objective in order to guarantee proper system operation and seamless data flow. The flow of data between the frontend and backend, API integration to guarantee proper data exchange between the frontend and server, and user authentication—where the interaction between the login system and backend is validated—are important testing areas.

Updates to user profiles and data are also tested to make sure that changes are reflected correctly across the platform; external service integrations, like weather data and carbon footprint calculators, are tested for proper data retrieval and error handling; automated tests, which mimic real user interactions and validate that the application works as intended in a variety of scenarios, are conducted using tools like Jest, Supertest, and Cypress.

VendorVault

Sign up for VendorVault

[Sign Up](#)

Already have an account?
[Login](#)

VendorVault

Login to VendorVault

[Login](#)

Don't have an account?
[Sign up](#)

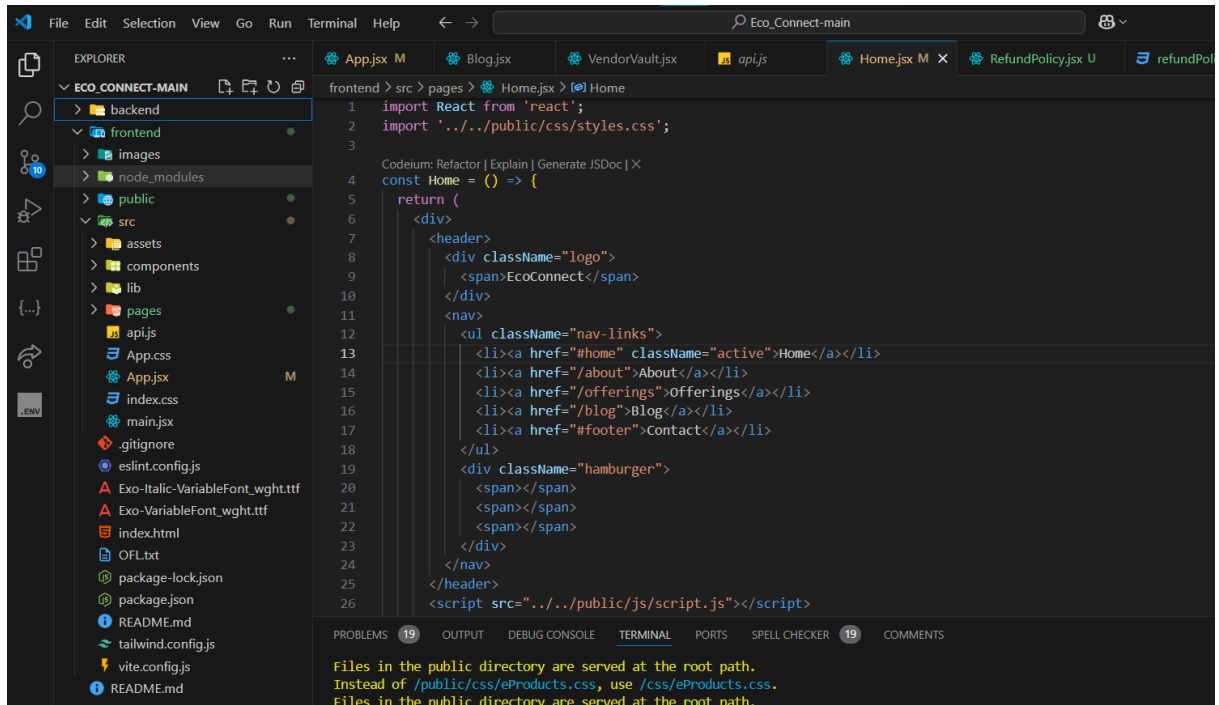
User Registration Integration Testing with React Framework

5.3 CONFIGURATION AND SET-UP

For the Eco-Connect web application to develop, deploy, and function smoothly, configuration and set-up are essential. The platform is built on the MERN stack, which consists of MongoDB, Express.js, React.js, and Node.js. This combination gives the platform a scalable and dynamic base. The first step requires cloning the project repository from a version control system like GitHub and installing essential dependencies using Node Package Manager (NPM). Developers may install all necessary packages listed in the **package.json** files by going to the frontend and backend folders and using **npm** install.

In order to configure the backend, developers need to connect to a MongoDB database. Sensitive environment variables, like the database connection string and authentication secret keys, must be safely stored in a **.env** file created in the backend directory. This guarantees that private information is kept apart from the source code and managed appropriately. In addition, the backend server needs to be configured to manage a variety of API requests, such as those for eco-related data retrieval and user authentication.

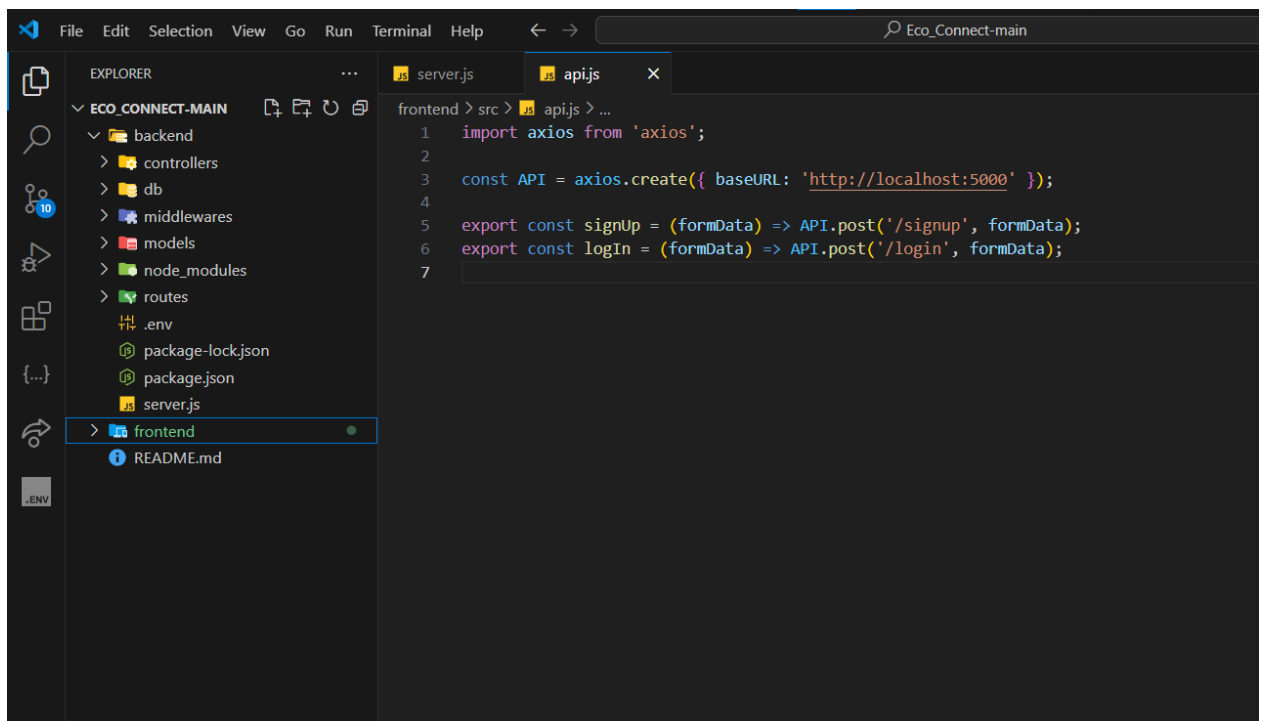
Configurations on the frontend side include utilizing Axios to connect to the backend API endpoints for HTTP queries. In order to guarantee seamless communication between the frontend and backend, developers must also resolve CORS (Cross-Origin Resource Sharing) concerns. The frontend is delivered using services like **Netlify** or **Vercel**, and the backend is hosted on cloud platforms like **Heroku** or AWS. This thorough set-up and configuration procedure guarantees ongoing deployment and integration, giving Eco-Connect users a dependable and effective experience.



```
1 import React from 'react';
2 import '../public/css/styles.css';
3
4 const Home = () => {
5   return (
6     <div>
7       <header>
8         <div className="logo">
9           <span>EcoConnect</span>
10        </div>
11        <nav>
12          <ul className="nav-links">
13            <li><a href="#home" className="active">Home</a></li>
14            <li><a href="#about">About</a></li>
15            <li><a href="#offerings">Offerings</a></li>
16            <li><a href="#blog">Blog</a></li>
17            <li><a href="#footer">Contact</a></li>
18          </ul>
19          <div className="hamburger">
20            <span></span>
21            <span></span>
22            <span></span>
23          </div>
24        </nav>
25      </header>
26      <script src="../../public/js/script.js"></script>
```

Files in the public directory are served at the root path.
Instead of /public/css/eProducts.css, use /css/eProducts.css.
Files in the public directory are served at the root path.

Front-end Code File



```
1 import express from 'express';
2 import dotenv from 'dotenv';
3 import axios from 'axios';
4
5 const app = express();
6
7 app.use(express.json());
8 app.use(express.urlencoded({ extended: true }));
9 app.use(express.static('public'));
10
11 const API = axios.create({ baseURL: 'http://localhost:5000' });
12
13 export const signUp = (formData) => API.post('/signup', formData);
14 export const login = (formData) => API.post('/login', formData);
```

Back-end Code File

5.4 DEPLOYMENT PROCESS

The Eco-Connect application's deployment procedure is methodical in order to guarantee a smooth roll-out and peak performance. In order to provide effective scalability and resource management, the backend is implemented on Heroku, while the frontend is hosted on Vercel. Deploying the frontend on Vercel is the initial step. First, developers connect the Vercel dashboard to the GitHub repository that has the frontend code. The React framework is immediately detected by Vercel, which also sets up the required build parameters. Through the Vercel dashboard, developers can also easily set up environment variables, guaranteeing that customizations like API endpoints are implemented appropriately. Vercel provides a preview of the website prior to the final deployment, which is started when the set-up is finished.

Render is used to deploy the Node.js application on the backend. On Render, developers build a new web service and link it to the frontend's GitHub repository. The Node.js environment is automatically detected by Render, which then asks developers to set up necessary environment variables like the database connection string and authentication secrets. The backend is deployed after the configurations are complete, and Render offers performance monitoring tools and real-time logs to guarantee the application functions properly. Render creates a public URL for the backend API when deployment is complete.

The final step in the deployment process is consolidating the URLs for both the frontend and backend. This ensures the Eco-Connect application is fully functional and accessible to users. The deployed URLs for the Eco-Connect application are as follows:

- **Frontend:** <https://eco-connect-frontend.vercel.app>
- **Backend:** <https://eco-connect.onrender.com>

This streamlined deployment process guarantees that Eco-Connect is both accessible and performant, delivering a reliable experience for users seeking eco-friendly content and features.

Eco-Connect / frontend /			Add file	...
shubhammaj1 Update api.js			de50113 · 8 hours ago	History
Name	Last commit message	Last commit date		
..				
images	Add files via upload	9 hours ago		
public	Add files via upload	9 hours ago		
src	Update api.js	8 hours ago		
Exo-Italic-VariableFont_wght.ttf	Add files via upload	9 hours ago		
Exo-VariableFont_wght.ttf	Add files via upload	9 hours ago		
OFL.txt	Add files via upload	9 hours ago		
README.md	Add files via upload	9 hours ago		
eslint.config.js	Add files via upload	9 hours ago		
index.html	Add files via upload	9 hours ago		
package-lock.json	Add files via upload	9 hours ago		
package.json	Add files via upload	9 hours ago		
tailwind.config.js	Add files via upload	9 hours ago		

Github Repository of Eco-Connect

Dashboard
Eco-Connect
Events
Settings
MONITOR
Logs
Metrics
MANAGE
Environment
Shell
Docs
Changelog
Community
Feedback
Invite a friend
Compliance and documents
Contact support
Render Status

WEB SERVICE

Eco-Connect Node Free Upgrade your instance →

Connect Manual Deploy

shubhammaj1 / Eco-Connect main

https://eco-connect.onrender.com

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more. Upgrade now

January 15, 2025 at 2:07 AM Live

cc429b7 Add files via upload

All logs Search Jan 15, 2:06 AM - 2:11 AM GMT+5:30

```

Jan 15 02:11:12 AM ● reason: TopologyDescription {
Jan 15 02:11:12 AM ●   type: 'Unknown',
Jan 15 02:11:12 AM ●   servers: Map(1) { 'localhost:27017' => [ServerDescription] },
Jan 15 02:11:12 AM ●   stale: false,
Jan 15 02:11:12 AM ●   compatible: true,
Jan 15 02:11:12 AM ●   heartbeatFrequencyMS: 10000,
Jan 15 02:11:12 AM ●   localThresholdMS: 15,
Jan 15 02:11:12 AM ●   setName: null,
Jan 15 02:11:12 AM ●   maxElectionId: null,

```

Backend Deployment On Render



Frontend Deployment On Vercel

Results

Testing Result Table			
Metric	Target Value	Description	Frequency
Page Load Time	$\leq 1.5s$	Time for full page load	Monthly
API Response Time	$\leq 1s$	Time for API calls to respond	Weekly
Authentication Time	$\leq 1s$	Time to log users in	Monthly
Data Refresh Rate	Every 1800s	Updates for data or content	Real Time Real Time
Error Rate	$< 1\%$	Occurrences of API/page/UI errors	Weekly
System Uptime	99.9%	Availability of the platform	Monthly
Mobile Responsiveness	$\leq 2s$	Mobile page load time	Monthly

Table 4

Testing Result Table				
Route	Method	Description	Authenti- cation	Approx. Response Time
/api/auth/register	POST	Registers a new user	No	100-200 ms
/api/auth/login	POST	Logs in a user and returns a token	No	100-150 ms
/api/auth/logout	POST	Logs out the user	Yes	50-100 ms
/api/data/fetch	GET	Fetches user-specific data	Yes Yes	150-250 ms
/api/data/refresh	GET	Refreshes data periodically	No	200-300 ms
/api/user/details	GET	Fetches user profile details	Yes	100-150 ms
/api/user/update-password	POST	Updates user password	Yes	150-200 ms

Table 5

EXISTING SYSTEM CHALLENGES AND SOLUTIONS

1. Responsiveness and User Experience

- **Challenges:** Many web applications face limited mobile optimization, leading to poor usability and inconsistent navigation across various devices.
- **Eco-Connect:** Utilizes **Tailwind CSS** and **Framer Motion** to ensure a seamless, responsive design.

2. Performance and Load Times

- **Challenges:** High latency and slow load times negatively impact user engagement, particularly in regions with low bandwidth.
- **Eco-Connect:** The platform, which is based on the MERN stack and has Redux, lazy loading, and optimized APIs, guarantees quick load times even in low-network locations.
-

3. Security and User Authentication

- **Challenges:** Weak authentication mechanisms can lead to data breaches and compromised accounts.
- **Eco-Connect:** Implements **secure password hashing**, **JWT-based authentication**, and **email-based verification**.

4. Administrative Control and User Management

- **Challenges:** Inefficient admin tools make it challenging to manage users and content effectively.
- **Eco-Connect:** Offers a robust admin panel with tools for role-based access control, content moderation, and user account management, simplifying administration.

5. Environmental Impact Awareness

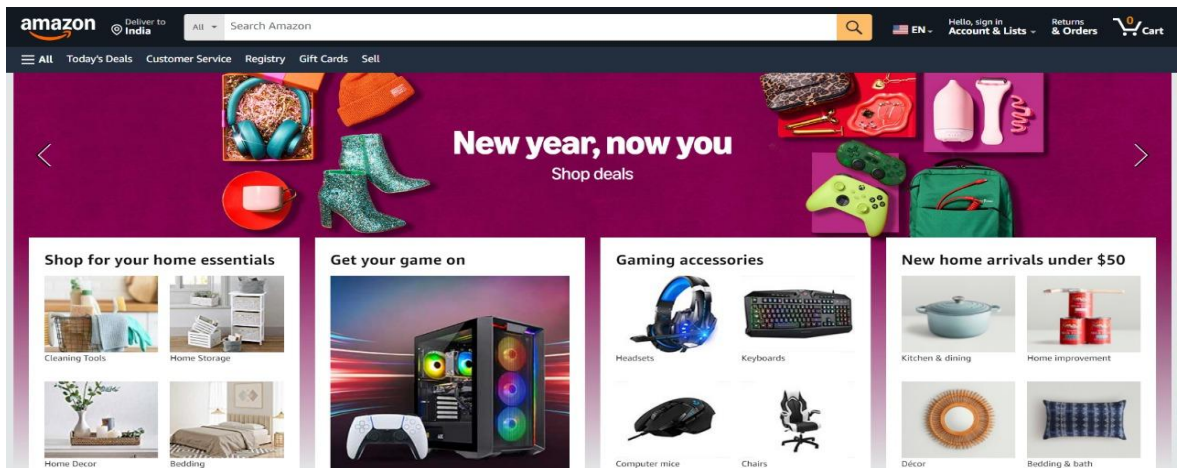
- **Challenges:** Many platforms lack features that educate users about their environmental footprint or actions they can take.
- **Eco-Connect:** Provides interactive tools such as carbon footprint calculators and personalized eco-friendly tips, encouraging sustainable practices among users.

6. AI Driven Chat-bot

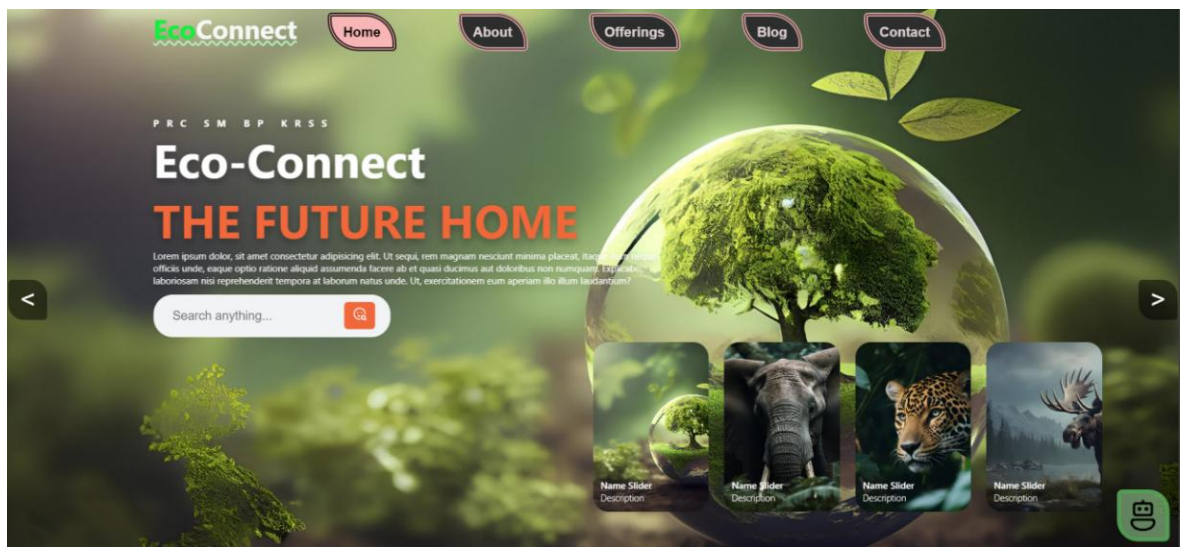
- **Challenges:** Existing platforms often lack interactive and personalized support, leaving users to navigate complex interfaces without guidance.
- **Eco-Connect:** The chat-bot assists users by answering frequently asked questions, guiding them through the platform, and providing eco-friendly tips tailored to their interests.

7. Content Discoverability and Search

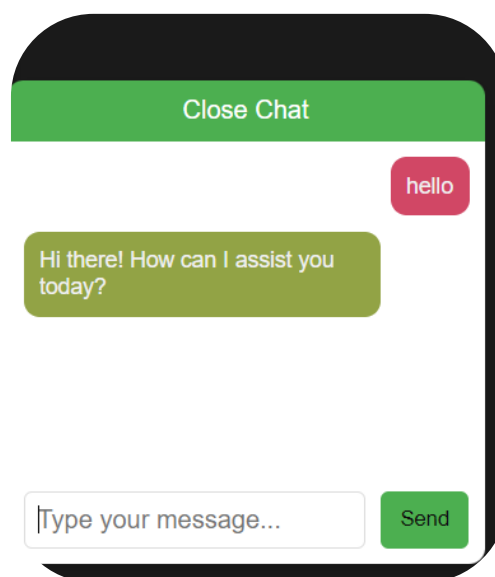
- **Challenges** Inefficient search capabilities make it difficult for users to find relevant content.
- **Eco-Connect:** Introduces an advanced tag-based search system, making it easier for users to discover content based on their interests and preferences, thereby improving content visibility.



Poor UI with Advertisements



Clean UI with Smooth Scroll



24/7 Chat Support

FUTURE WORK AND ENHANCEMENTS

The Eco-Connect project offers tremendous potential for future developments to enrich user experience and strengthen platform capabilities. Integrating OTP based authentication may streamline the login process, boosting simplicity and security while attracting more users.

A number of cutting-edge features might be added to the Eco-Connect project to increase user usefulness and engagement. By including an image tag recommendation system, user-uploaded photographs may be automatically tagged. With the use of sophisticated picture recognition algorithms, this feature will increase search functionality, improve content classification, and save users time by eliminating the need to manually tag their uploads.

Additionally, installing a robust AI-driven chat-bot will give consumers with fast aid and direction. The chat-bot may respond to frequently asked questions, offer tailored eco-tips, and assist users in efficiently navigating the platform. The platform would become more engaged and user-friendly with this functionality, which would guarantee ongoing support and involvement.

Lastly, Eco-Connect may become a more engaging and powerful platform by including AI-powered suggestions for tailored eco-friendly advice and actions depending on user activity. With these improvements, Eco-Connect will be positioned as a major force in encouraging environmentally conscious behavior and sustainable practices.

Conclusion

An inventive step in advancing environmental awareness and sustainability is the Eco-Connect platform. It has been painstakingly created to provide an easy-to-use and captivating user experience while addressing contemporary issues in the eco-awareness space. Based on a strong MERN stack design, the platform effectively manages user demands by integrating cutting-edge technologies like Redux-based state management, streamlined APIs, and lazy loading, guaranteeing flawless performance even in low-bandwidth settings. The project's present features, such as an interactive user interface, secure authentication, and responsive design, provide a strong basis for future expansion.

Eco-Connect uses creative and useful solutions to address urgent problems including content management, user engagement, and security. An environment that is safe and well-managed is offered by features like email verification, JWT-based authentication, and a separate admin panel. In the meanwhile, a favorable user experience is guaranteed by its intuitive design, enhanced speed, and features like real-time APIs. However, the platform understands that in order to remain influential and relevant, it must be continuously improved.

In conclusion, Eco-Connect is well-positioned to become a leader in the eco-awareness space. By addressing existing challenges and continuously innovating with user-centric features, the platform can provide a transformative experience that encourages environmental responsibility. With its planned enhancements, Eco-Connect will not only improve functionality and engagement but also strengthen its position as a trusted, forward-thinking platform. These developments will empower users to take actionable steps toward sustainability, ensuring a meaningful impact on both individual behaviors and global environmental efforts. Through its commitment to innovation and user satisfaction, Eco-Connect has the potential to drive significant positive change in the years to come. .

REFERENCES

- I. Kadam, Y., Goplani, A., Mattoo, S., Gupta, S. K., Amrutkar, D., & Dhanke, J. (2023). *Introduction to MERN Stack: Comparison with Previous Technologies* [Research paper]. https://www.researchgate.net/publication/371459805_Introduction_to_MERN_Stack_Comparison_with_Previous_Technologies.
- II. GitHub. (2023). Express.js: Fast, unopinionated, minimalist web framework for Node.js. Retrieved from <https://github.com/expressjs/express>.
- III. **Express.js Documentation**
<https://expressjs.com/>
- IV. W3C. (2022). **Web Content Accessibility Guidelines (WCAG) Overview**. World Wide Web Consortium (W3C). Retrieved from <https://www.w3.org/WAI/WCAG21/quickref/>.
- V. **MediaStack API**: http://api.mediastack.com/v1/news?access_key=e7487077363c7
- VI. **MongoDB Atlas Documentation**: <https://www.mongodb.com/products/platform/atlas-database>
- VII. **Node.js Documentation**: <https://nodejs.org/>
- VIII. **React Documentation**: <https://react.dev/>
- IX. **Tailwind CSS Documentation**: <https://tailwindcss.com/docs/installation>
- X. **Lenis. (2022). Lenis Smooth Scroll**: A JavaScript library for smooth scrolling and scroll-based animations. Retrieved from <https://github.com/darkroomengineering/lenis>.