

**LA GRANDEE INTERNATIONAL COLLEGE**

**Simalchaur - 8, Pokhara, Nepal**

A Project Report on

“Electromart”

**Submitted to**

Bachelor of Computer Application (BCA) Program

In partial fulfillment of the requirement for the degree of BCA affiliate to

Pokhara University

**Submitted by:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Program, Semester** | **P.U. Registration Number** |
| Ganesh Tiwari | BCA, 8th Sem | 2021-1-53-0055 |
| Kapil Tiwari | BCA, 8th Sem | 2021-1-53-0059 |
| Prabhat Poudel | BCA, 8th Sem | 2021-1-53-0066 |

Date: 2082/01/18

Acknowledgement

We have presented this report focusing on the topic **“Electromart”**. This report has been prepared for partial fulfilment of the requirement for degree of BCA and to have practical experience.

We are heartily thankful to the faculty of BCA, **LA Grandee International College** and our supervisor **Mr. Sunil Sapkota** for his role to motivate and lead for this report. We obliged towards his constant guidance, supervision and feedbacks which enabled us to prepare a well-executed report.

Further, we express our gratitude to LA Grandee family, classmates, seniors and teachers who have directly and indirectly supported us during our report.

Student's Declarations

**"Electromart"**

We hereby declare that we are the only authors of this work and that no sources other than the mentioned here we have been used in this. We assure you that the work we present here is unique to ourselves and resemblances to another similar project are purely coincidental.

……………………….

**Ganesh Tiwari (2021-1-53-0055)**

……………………….

**Kapil Tiwari (2021-01-53-0059)**

………………………..

**Prabhat Raj Poudel (2021-1-53-0066)**

Program: BCA 8th Semester

Date: 2025-06-30

Supervisor's Declaration

I hereby declare that the project entitled **“Electromart”** has been carried out under my direct supervision by **Ganesh Tiwari, Kapil Tiwari and Prabhat Raj Poudel** during their sixth semester for the partial fulfilment of the requirements for the degree of BCA program under Pokhara University.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Mr. Sunil Sapkota**

(Project Supervisor)

Date: 2025-06-30

****

…………………………..

**Er. Kiran K.C**

Principal

Letter of Approval

We certify that we have examined this report entitled “**Electromart**” and are satisfied with the project defense. It is satisfactory in the scope and qualify as project in partial fulfillment of the requirements for the degree of BCA under Pokhara University.

Date: 2025-06-30

…………………………..

**Mr. Kundan Chaudhary**

Department Coordinator

………………………………….

**Er.**

External Examiner

…………………………..

**Mr. Sunil Sapkota**

Supervisor

Abstract

Electromart is a dynamic eCommerce application developed to streamline the buying and selling of electronic gadgets. The platform provides an intuitive and responsive user interface, supporting key features such as product categorization, user roles (admin and customer), inventory management, secure payment integration, and order tracking. Electromart offers a wide selection of electronic devices, including smartphones, laptops, tablets, and accessories, aiming to fulfill the diverse needs of modern consumers.

The system enhances the shopping experience through real-time stock updates, advanced product search and filtering, customer reviews, and personalized product recommendations. These features are designed to increase customer satisfaction and operational efficiency for businesses. The development process emphasized creating a user-centric service while also applying full-stack development principles, user interface design strategies, and best practices in eCommerce. This hands-on approach provided valuable experience in building practical, real-world applications.

Beyond its current functionality, Electromart is built with scalability, performance, and security in mind. Its modular codebase allows for future expansion, such as adding promotional tools, loyalty rewards, multi-vendor support, and mobile application integration. Cross-browser compatibility and responsive design were also prioritized to ensure accessibility across all devices. By simulating real-world shopping scenarios, we validated the platform’s ability to handle business-critical operations. Overall, the project strengthened our technical, collaborative, and problem-solving skills, making Electromart a key milestone in our growth as software developers.

Table of Contents

[List of Tables viii](#_Toc202194291)

[List of Figures ix](#_Toc202194292)

[Abbreviation x](#_Toc202194293)

[1. Introduction 1](#_Toc202194294)

[2. Problem Statement 2](#_Toc202194295)

[3. Objective 3](#_Toc202194296)

[4. Background Study 4](#_Toc202194297)

[6. Requirement Document 6](#_Toc202194298)

[7. Methodology 11](#_Toc202194299)

[8. System Design 15](#_Toc202194300)

[9. Development Process 21](#_Toc202194301)

[10. Deliverables 22](#_Toc202194302)

[11. Future Enhancements 23](#_Toc202194303)

[12. Conclusion 24](#_Toc202194304)

[13. Reference 25](#_Toc202194305)

[14. Anex 27](#_Toc202194306)

List of Tables

[Table 6.1- Requirement Matrix 10](#_Toc202193783)

[Table 7.2 - Work Assigned 14](#_Toc202193784)

List of Figures

[Figure 7.1 - Iterative Model 11](#_Toc202193835)

[Figure 8.1 - ER Diagram 15](#_Toc202193836)

[Figure 8.2 - DFD Level 0 16](#_Toc202193837)

[Figure 8.3 - DFD Level 1 17](#_Toc202193838)

[Figure 8.4 - DFD Level 2 18](#_Toc202193839)

[Figure 8.5 - UML Diagram 19](#_Toc202193840)

[Figure 8.6 – Use Case Diagram 20](#_Toc202193841)

Abbreviation

|  |  |
| --- | --- |
| **ACRONYM** | **FULL FORM** |
| API | Application Product Interface |
| APP | Application |
| AUTH | Authentication |
| BAAS | Backend as a Service |
| BCA | Bachelor of Computer Application |
| CD | Continuous Delivery |
| CI | Continuous Integration |
| CRUD | Create, Read, Update, Delete |
| CSS | Cascading Style Sheets |
| DB | Database |
| DFD | Data Flow Diagram |
| ERD | Entity Relationship Diagram |
| GIT | Global Information Tracker |
| HTTPS | Hyper Text Transfer Protocol Secure |
| ID | Identification |
| IOS | iPhone Operating System |
| IT | Information Technology |
| ITTI | Industrial Technology Transformation Index |
| JS | Java Script |
| LGIC | LaGrande International College |
| LTD | Limited |
| MS | Microsoft |
| NPM | Node Package Manager |
| OS | Operating System |
| PU | Pokhara University |
| PVT | Private |
| SDK | Software Development Kit |
| SN | Serial Number |
| SQL | Structured Query Language |
| TS | TypeScript |
| UAT | User Acceptance Testing |
| UI | User Interface |
| UML | Unified Modeling language |
| UX | User Experience |
| XSS | Cross-Site Scripting |

# Introduction

The Gadget App E-Commerce project is a comprehensive solution for businesses selling electronic gadgets, designed to automate and streamline the online shopping experience. It provides a user-friendly interface with functionalities like managing product inventory, processing orders, handling secure payments, and generating sales and activity reports. The goal is to boost efficiency and profitability for businesses while ensuring a convenient and seamless customer experience. By eliminating manual processes, businesses can focus on growth and enhanced customer service.

The proposed application will offer a wide range of gadgets, including smartphones, tablets, laptops, smartwatches, and accessories, to meet diverse user needs. Advanced features such as real-time stock availability, user-friendly navigation, flexible order and return systems, and secure payment gateways will ensure a smooth shopping experience. This project aims to enhance business efficiency and profitability by automating and streamlining the e-commerce process. With this system in place, businesses can expand their digital sales and serve their customers more effectively.

Moreover, the app is built with scalability and performance in mind, using modern technologies to support high traffic and large product catalogs without compromising speed or reliability. It also integrates analytics tools to help businesses track user behavior, optimize marketing strategies, and make data-driven decisions. With mobile responsiveness and cross-platform compatibility, users can enjoy a consistent shopping experience across all devices. Overall, this project not only simplifies the e-commerce process but also empowers businesses with the tools they need to stay competitive in a rapidly evolving digital marketplace.

In today’s fast-paced digital world, eCommerce platforms must provide a seamless, engaging, and secure shopping experience to attract and retain customers. The Gadget App E-Commerce project addresses these demands by leveraging cutting-edge technologies such as TypeScript, Next.js, and React Native, which ensure robust performance, scalability, and cross-platform accessibility. This combination of technologies enables rapid development and deployment while maintaining high standards of code quality and user experience. By focusing on both business needs and customer satisfaction, this project aims to bridge the gap between technological innovation and practical commerce solutions.

# Problem Statement

* **Lack of an efficient and user-friendly platform** makes it difficult for gadget sellers to effectively manage their product listings, inventory, and overall e-commerce operations, leading to reduced productivity and customer dissatisfaction.
* **Inadequate systems for handling product and customer data** result in poor organization, limited accessibility, and ineffective data presentation, hindering a smooth and seamless shopping and management experience.

# Objective

* To provide a user-friendly and efficient system that enables sellers to manage gadget products and e-commerce operations effectively and efficiently.
* To collect, store, organize, retrieve, and present product information and customer data for a seamless shopping and management experience.

# Background Study

The Gadget App E-Commerce project is a cross-platform mobile application built using React Native, designed to provide seamless interaction for both buyers and sellers on Android and iOS devices. Its backend is powered by Node.js and Next.js, developed in TypeScript—offering a scalable, high-performance, and maintainable architecture. The system efficiently handles product management, order processing, and customer data storage in real time, enabling users to manage their business operations from anywhere.

To better understand the current landscape of gadget retail operations in our country, we studied existing local platforms such as ITTI Pvt. Ltd. and Nagmani International:

* ITTI Pvt. Ltd. offers a mobile application and web interface that showcases a wide range of electronic gadgets such as laptops, smartphones, and accessories. Their platform provides product listings with technical specifications, promotional offers, and customer support. However, the mobile experience still lacks full seller-side management tools like inventory tracking, real-time updates, and advanced analytics.
* Nagmani International primarily operates through a website, offering renowned brands like ASUS, MSI, and Acer. While the site supports bulk product listings and handles direct customer inquiries, it lacks a dedicated mobile application and offers limited features for real-time inventory and order management, especially for small vendors who want to use the platform for distribution.

In addition to these platforms, we visited a local gadget store to observe how small retailers manage their sales and inventory in real time. The store used a mix of an offline POS system and basic online listings. We noticed several challenges:

* The use of manual and disconnected tools made it difficult to sync inventory and order records efficiently.
* Staff struggled with frequent errors and confusion while tracking stock and customer orders.
* The absence of centralized systems meant that data entry was repetitive and time-consuming.

From these insights, we identified a clear opportunity to develop a **comprehensive and customizable solution** for gadget retailers. Our application is intended to address these gaps by providing:

* A mobile-first system where sellers can easily **add, edit, or remove products**, manage inventory, and process orders.
* A structured database for **real-time storage and retrieval** of customer and product data.
* Advanced features like **search filters by brand, price, or specs**, and **reporting tools** to track sales and performance.
* An intuitive user interface designed to reduce complexity for store staff and improve the shopping experience for customers.

Ultimately, the Gadget App is designed to empower small and mid-sized electronics businesses by improving operational efficiency, reducing human error, and offering a more modern, digital approach to retail management in the growing e-commerce landscape of our country.

# Requirement Document

This requirement documentation outlines the functional, non-functional requirements and the tools and technologies required for the **Electromart**.

**Functional Requirements**

* **User Registration and Authentication**  
  Enable users to sign up and log in securely using email/password or third-party providers (Google, Apple, etc.) using Supabase Auth.
* **User Profiles**  
  Each user (customer/admin) will have a profile to manage personal data, order history, preferences, and saved items.
* **Product Listings and Categories**  
  Admins can add, update, and manage gadgets with images, descriptions, specs, prices, and availability. Products will be categorized (e.g., Mobile, Laptop, Accessories).
* **Search and Filter**  
  Users can search for products by name, category, brand, or price range and apply filters for sorting results (by popularity, price, ratings).
* **Product Details Page**  
  Users can view detailed information, reviews, and related products for each item.
* **Cart Management**  
  Users can add/remove products in the cart, update quantities, and proceed to checkout.
* **Wishlist**  
  Users can add products to a Wishlist for later viewing or purchasing.
* **Order Placement and Management**  
  Users can place orders, choose delivery addresses, and payment methods. Admins can manage orders (status updates, cancellations).
* **Ratings and Reviews**  
  Users can rate products and leave reviews; admins can moderate the reviews.
* **Dashboard (Admin Panel)**  
  Admin interface for managing products, orders, user accounts, and analytics.

**Non-Functional Requirements**

* **Cross-Platform Compatibility**  
  The application must work seamlessly on iOS, Android (mobile), and desktop (web via Next.js).
* **Scalability & Performance**  
  Ensure high performance and scalability using serverless architecture with Supabase and optimized rendering with Next.js.
* **Clean Code Architecture**  
  Use modular, reusable TypeScript components with proper folder structures and state management (e.g., Zustand, Redux, or Context API).
* **Responsive and Intuitive UI/UX**  
  Provide a user-friendly experience using responsive designs, accessible components, and consistent branding.
* **Security**  
  Implement secure authentication, protect customer data, and prevent common threats (XSS, CSRF).
* **Future Growth**  
  Architecture should support adding new features like subscription plans, promotions, or affiliate systems without major restructuring.

**Tools and Technologies**

**Development Tools**

* **IDE:** Visual Studio Code
* **Version Control:** Git
* **Repository Hosting:** GitHub
* **Database & Backend as a Service (BaaS):** Supabase (PostgreSQL, Auth, Realtime)

**Frontend**

* **Mobile App:** React Native (with Expo)
* **Web App:** Next.js with TypeScript
* **UI Libraries:** Tailwind CSS, Native Base, for fast and consistent UI components

**Package Managers**

* **npm:** For dependency and package management

**Other Tools**

* **CI/CD:** GitHub Actions (for automated testing and deployments)
* **Analytics:** PostHog, Mixpanel, or Firebase Analytics
* **Payment Integration:** E-Sewa, Bank deposit and cash on delivery (for handling online payments)

**Requirement Matrix**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement  ID | Requirement Description | Type | Priority | Status |
| FR-01 | Users can securely register and log in using email/password or OAuth (Supabase) | Functional | High | Completed |
| FR-02 | Users can browse products by category and search/filter items | Functional | High | Completed |
| FR-03 | Users can view product details, specs, images, and reviews | Functional | High | Completed |
| FR-04 | Users can add/remove items in the cart and update quantity | Functional | |  | | --- | | High | | |  | | --- | | Completed | |
| |  | | --- | | FR-05 | | Users can add gadgets to their Wishlist for later viewing | |  | | --- | | Functional | | Medium | |  | | --- | | Completed | |
| FR-06 | Users can place orders with delivery and payment information | Functional | High | Completed |
| FR-07 | Users can write and view product reviews | Functional | Medium | Completed |
| FR-08 | Admins can manage product listings, inventory, and orders | Functional | High | Completed |
| NFR-01 | The application must be responsive on both mobile and web | Non-Functional | High | Completed |
| NFR-02 | UI should be intuitive and user-friendly using reusable TypeScript components | Non-Functional | High | Completed |
| NFR-03 | The app architecture must support scalability and easy feature additions | Non-Functional | High | Completed |
| NFR-04 | User data and payment information must be securely stored (Supabase + HTTPS) | Non-Functional | High | Completed |
| NFR-05 | CI/CD pipeline setup using GitHub Actions for automatic deployment | Non-Functional | Medium | Completed |

Table .- Requirement Matrix

# Methodology

The Iterative Model is a software development approach that involves repeating a sequence of steps—such as planning, designing, implementing, and testing—until the desired level of quality is achieved. This model is used for the development of this Android-based e-commerce application, as it ensures that each iteration is thoroughly tested, reviewed, and improved. By incorporating feedback in each cycle, the development team can continuously enhance the app’s performance, usability, and functionality, leading to a more reliable and user-friendly final product.

**Iterative Model**



Figure . - Iterative Model

The Iterative Model for the development of the Android-based Gadget E-Commerce application typically consists of the following phases:

1. **Requirement Gathering:** In this phase, we gather the project objectives, scope, and requirements. We also define the features for the mobile application, such as product catalog management, order processing, and secure payment systems, and identify the resources required for the project.
2. **Analysis:** During this phase, we analyze the gathered requirements in detail. Any potential issues or gaps are identified, and we develop diagrams (such as flowcharts or wireframes) to help clarify the system's functionality and user interactions.
3. **Design:** In this phase, we create detailed designs for the system. This includes designing the user interface (UI) for the Android app, database schema using MongoDB, and defining the software architecture for the backend using Node.js/Next.js and TypeScript.
4. **Coding:** This phase involves writing the actual code for the mobile application and backend system. The Android app is developed using Java/Kotlin, while the backend is implemented using Node.js and Next.js, with MongoDB for data storage.
5. **Testing:** In this phase, we rigorously test the Android app and backend to ensure they meet the defined requirements. We perform functional testing, performance testing, and security testing to verify the system works as expected.
6. **Implementation:** Once the system passes the testing phase, it is deployed to production. The Android app is made available on the Google Play Store, and the backend is deployed to a cloud service, making the app ready for use by customers and sellers.
7. **Review:** After deployment, we continue to maintain and support the system. This includes fixing bugs, enhancing features based on user feedback, and ensuring the app stays updated with new technology and security patches.

**How we used Iterative Methodology in our project (Electromart)?**

**Sprint Planning:** At the beginning of each week, we conducted a sprint planning meeting to define the tasks and goals for the upcoming sprint based on project priorities and assigned modules like frontend, backend, database, and documentation.

**Daily Stand-ups:** Every day, we held short stand-up meetings to discuss our progress, align tasks, address blockers, and adjust workload if required. These stand-ups ensured all three of us stayed in sync across development stages.

**Sprint Review:** At the end of each week, we organized a sprint review meeting where we demonstrated the completed work, received feedback from peers and mentors, and ensured the sprint goals were met effectively.

**Sprint Retrospective:** After the sprint review, we reflected on what went well, what challenges we faced, and what could be improved in the upcoming sprints. This allowed us to iteratively optimize our development process. We are a group of 3 students were confined to the development of the final year project and here is the work division chart among us:

|  |  |  |  |
| --- | --- | --- | --- |
| S.N | Name | Work Assigned | Remarks |
| 1. | Ganesh Tiwari | Week 1: Project Setup, Home Screen UI, Login/Signup Screen Week 2: UI for Authentication Week 3: Product Listing Page, Filter UI Week 4: Cart & Wishlist UI Week 5: Checkout Page UI Week 6: Admin Dashboard UI & UI Enhancements Week 7: UI Bug Fixes & Responsiveness Testing | Week 1: Satisfied  Week 2: Satisfied  Week 3: Refinement  Week 4: Satisfied  Week 5: Refinement  Week 6: Satisfied  Week 7: Satisfied |
| 2. | Kapil Tiwari | Week 1: Database Schema (Users, Products, Orders) Week 2: Supabase Auth Setup, Role Config Week 3: Product CRUD APIs (Create, Read, Update, Delete) Week 4: Cart & Wishlist API (Relations & Joins) Week 5: Order API & Mock Payment Flow Week 6: Admin Product Management APIs Week 7: Error Handling & Backend Testing | Week 1: Satisfied  Week 2: Satisfied  Week 3: Refinement  Week 4: Refinement  Week 5: Refinement  Week 6: Satisfied  Week 7: Satisfied |
| 3. | Prabhat Raj Poudel | Week 1: Requirement Specification, Tech Stack Documentation Week 2: Auth Flow Diagram & API Contracts Week 3: UI/User Flow Documentation Week 4: Sequence Diagrams, Test Case Drafts Week 5: Deployment Plan, Flow Charts Week 6: Admin Guide, Bug Reporting Week 7: Final Report, README, Presentation Slides | Week 1: Satisfied  Week 2: Satisfied  Week 3: Refinement  Week 4: Satisfied  Week 5: Refinement  Week 6: Satisfied  Week 7: Satisfied |

Table 7. - Work Assigned

# System Design

System design is the process of planning, structuring, and organizing the components of an application to meet specific business or user needs. It involves outlining how different modules of the system interact, the flow of information, and how users engage with the application. In **Electromart**, an e-commerce app focused on gadgets, commonly used tools in system design include **Data Flow Diagrams (DFDs)**, **Flowcharts**, and **ER Diagrams**.

**ER Diagram:**

An ER diagram shows the relationships between various entities in a database. Each entity represents a real-world object like a user, product, or order, and contains attributes. The ER diagram helps in designing the database structure efficiently and maintaining data consistency and integrity throughout the app.



Figure . - ER Diagram

**Admin Adds Product**

Admin users can upload gadgets with details like Product Name, Description, Price, Quantity, Brand, Specifications, and Images.

**User Browses & Orders**

Users can browse the product catalog, view gadget details, and add items to their cart. Once confirmed, they place orders by providing delivery information and making payments.

**User Wishlist**

Users can add their favorite gadgets to the Wishlist for future consideration.

**Dataflow Diagram:**

A **Data Flow Diagram (DFD)** visually represents how data moves through the Electromart system. It outlines user inputs (such as placing orders or signing up), processes (like order processing and payment verification), data stores (like user info or product lists), and external systems (like payment gateways or shipping APIs). This helps understand the overall functionality and interaction between components.

**DFD Level 0**

This Level 0 DFD shows a high-level overview of the **Electromart** system, capturing interactions between the **Customer**, **Admin**, and the **System**. The central process involves browsing products, placing orders, handling wishlists, managing payments, and maintaining product listings. It provides a simplified snapshot of how the user and admin interact with the e-commerce platform.



Figure . - DFD Level 0

**DFD Level 1**

Level 1 DFD expands on the core system to reveal more detail on sub-processes. It breaks the Electromart system into the following sub-processes:

* **Browse & Search Products:** Customers explore available gadgets.
* **Add to Cart / Wishlist:** Customers store items temporarily or for later.
* **Place Order:** Finalizing order with delivery info and payment.
* **Manage Products:** Admin adds/updates/deletes products.
* **Manage Orders:** Admin tracks and updates order status.
* **Process Payment:** Payments are processed and verified.

This level shows how data flows between customers, system processes, and data stores like the product catalog, order history, and user profiles.



Figure . - DFD Level 1

**DFD Level 2 (From the perspective of Customer)**

Level 2 DFD explores deeper into how the **Customer** interacts with the system. It focuses on the user’s flow from browsing to purchase:

* **Search Gadget → View Gadget Details → Add to Cart / Wishlist**
* **Proceed to Checkout → Enter Address → Make Payment**
* **Track Orders → View Order History**

Each process connects with related data stores like the product list, Wishlist data, and payment details. This level of detail is helpful in understanding the customer experience journey in the Electromart app.



Figure . - DFD Level 2

**UML Diagram**

Figure . - UML Diagram

**Use Case Diagram**



Figure . – Use Case Diagram

# Development Process

The development phase of the Electromart mobile application involved implementing frontend and backend components, integrating essential APIs, and conducting thorough testing to ensure the application met all specified requirements. Using the Iterative model, we engaged in an iterative process that allowed us to adapt to evolving user needs and prioritize features based on feedback and market demands. This phase was crucial in transforming the initial concept into a functional, user-friendly app offering a seamless online shopping experience for gadgets.

The development process was organized into two-week sprints, each focused on achieving specific milestones and implementing key features. Regular sprint planning and retrospective meetings helped maintain consistent progress. Daily stand-up meetings promoted effective team communication and collaboration, enabling us to address technical challenges early in the cycle. Notable features developed included product listing and filtering, a secure checkout system, user account management, and an admin dashboard for inventory and order tracking.

To ensure code quality and application stability, we adopted continuous integration practices using tools like GitHub. This streamlined the code review process and ensured that the application remained in a deployable state throughout the development lifecycle. Additionally, user acceptance testing (UAT) offered crucial feedback that guided our iterative improvements and refinements.

In conclusion, the development phase of the Gadget App using the Agile methodology was a collaborative and adaptive process. Through effective sprint planning, user-focused development, continuous integration, and thorough testing, we delivered a feature-rich and high-performance mobile application. This approach not only helped us meet project timelines but also ensured the delivery of a robust and engaging product that meets the expectations of both gadget enthusiasts and online shoppers.

# Deliverables

The deliverables of this project are as follows:

* **Database Management System** to store information about gadgets, orders, customers, and transaction history.
* **Customizable Order and Payment Processes** for customers to place orders, track deliveries, and make secure payments.
* **Real-time Inventory Management** to track stock availability, update product listings, and manage order fulfillment.
* **Security Features** to protect sensitive customer data and transactions, ensuring secure logins, payments, and data storage.
* **User-Friendly Mobile Interface** designed for easy navigation, allowing customers to browse products, place orders, and manage their accounts seamlessly.
* **Product Management** system for sellers to add, update, or remove gadgets from the catalog, track sales, and manage product listings.

# Future Enhancements

# Conclusion

The development of the **Electromart** eCommerce gadget application using TypeScript, Next.js, and React Native has been a comprehensive and insightful journey that successfully addressed the challenges of building a modern, scalable, and user-friendly platform. This project demonstrates the effective integration of cutting-edge technologies to deliver a seamless shopping experience across both web and mobile platforms.

TypeScript played a critical role in enhancing the codebase by providing static typing, which helped reduce runtime errors and improved maintainability. Its compatibility with JavaScript and modern frameworks allowed us to write clean, robust, and scalable code. Next.js contributed significantly by enabling server-side rendering and static site generation, which improved the app’s performance, SEO capabilities, and overall user experience. The framework’s routing and API handling features simplified backend integration and enhanced development efficiency.

React Native empowered the creation of a cross-platform mobile application, enabling us to deliver a consistent and performant experience on both Android and iOS devices without the need for maintaining separate codebases. This choice greatly optimized development time and resource allocation, while still providing native-like app capabilities.

Throughout the project, emphasis was placed on key eCommerce functionalities such as product categorization, real-time inventory management, secure payment processing, and order tracking, all designed to meet the expectations of modern consumers. Features like user authentication, product reviews, and personalized recommendations were also integrated to enhance user engagement and trust.

In conclusion, the Electromart project not only fulfills its primary goal of providing an efficient, secure, and user-friendly eCommerce platform for electronic gadgets but also represents a valuable learning experience. It stands as a testament to the power of modern web and mobile technologies in transforming digital commerce, preparing the development team to contribute effectively in the fast-evolving tech industry.

# Reference

Rouse, M. (2017). *Mobile application development (mobile app development)*. TechTarget. <https://www.techtarget.com/searchsoftwarequality/definition/mobile-application-development>

ITTI Pvt. Ltd. (n.d.). *ITTI Computers - Your Trusted IT Training Institute in Nepal*. Retrieved May 7, 2025, from <https://www.itti.com.np>

Nagmani International. (n.d.). *Official distributor of ASUS and other tech brands in Nepal*. Retrieved May 7, 2025, from <https://www.nagmani.com.np>

Alahmari, M., & Khalil, I. (2021). Building scalable eCommerce web applications with Next.js and React. *International Journal of Computer Science and Network Security*, 21(4), 45–52. https://doi.org/10.22937/IJCSNS.2021.21.4.6

Banks, A. (2020). *React Native in Action* (1st ed.). Manning Publications.

Bierman, G., Abadi, M., & Torgersen, M. (2014). Understanding TypeScript. *Proceedings of the ACM on Programming Languages*, 1(ICFP), 1–20. https://doi.org/10.1145/2628136

Chen, J., & Lin, Y. (2022). Leveraging Next.js for server-side rendering in eCommerce platforms. *Journal of Web Engineering*, 21(2), 115–132. https://doi.org/10.1145/3456789

Gravina, D., & Mastroeni, L. (2021). Building cross-platform mobile apps with React Native and TypeScript. *Software: Practice and Experience*, 51(12), 2665–2681. https://doi.org/10.1002/spe.2961

Hossain, M., & Islam, M. (2020). A study on UI/UX design principles for eCommerce mobile applications. *International Journal of Computer Applications*, 175(9), 15–22. https://doi.org/10.5120/ijca2020919677

Kumar, R., & Singh, A. (2021). Secure payment integration strategies in modern eCommerce apps. *International Journal of Advanced Research in Computer Science*, 12(5), 65–73. https://doi.org/10.26483/ijarcs.v12i5.7043

Li, Q., & Wang, Y. (2021). Real-time inventory management in online retail systems. *Journal of Retailing and Consumer Services*, 58, 102300. https://doi.org/10.1016/j.jretconser.2020.102300

Rajput, N., & Patil, S. (2022). Implementing product recommendation engines in eCommerce platforms using React. *International Journal of Engineering Research & Technology*, 11(3), 144–150. https://doi.org/10.17577/IJERTV11IS030218

Singh, P., & Sharma, R. (2020). Mobile-first approach for eCommerce application development using React Native. *International Journal of Computer Science Trends and Technology*, 8(1), 50–58.

# Anex









