Table of Contents

1. Executive Summary

Brief overview of the project's objectives, technologies used, and the key components like frontend (React), backend (Django), and PHP for email notifications.

2. Introduction

2.1 Background of the Project

Explanation of the rise of e-commerce, importance of the project, and the technologies being used.

2.2 Project Objectives

Key objectives of the project, including creating a user-friendly interface and secure backend, and automating email notifications.

2.3 Project Scope

Description of the core functionalities, including frontend, backend, and mail integration.

2.4 Operating Environment

Technologies and environment in which the project operates, including web server and technologies like React, Django, and PHP.

3. Analysis

3.1 Software Requirement Specification

3.1.1 General Description

Overview of how the system functions (frontend, backend, and email integration).

3.1.2 Problem Statement

The existing challenges of traditional e-commerce systems, and how this project addresses them.

3.1.3 System Objective

Description of the goals such as providing a responsive UI and efficient order processing.

3.2 Software and Hardware Requirements

3.2.1 Software Requirement

Detailed software needs, including frontend, backend, database, and email system.

3.2.2 Hardware Requirement

Specifications for server performance and hardware requirements.

3.3 Existing vs Proposed

Comparison between traditional e-commerce systems and the improved system using modern technologies.

3.4 Software Tools Used

Tools like React.js, Django, PostgreSQL, and PHP used in the project.

4. Design

4.1 Data Flow Diagram (DFD)

Description of how data flows between the user, React frontend, Django backend, and PHP email system.

5. System Implementation

5.1 Module Description

Overview of core system modules like user authentication, product management, order processing, and email notifications.

5.2 Screenshots

Screenshots of important pages like the homepage, product catalog, cart, and order confirmation.

6. System Test

6.1 Integration Testing

Description of tests performed to ensure integration between React, Django, and PHP.

6.2 Test Case for Login Page

Detailed test scenario and expected outcome for user login.

7. Maintenance and Update

7.1 Maintenance Plan

Description of the plans for regular updates, system security, and performance improvements.

7.2 Future Enhancements

Potential enhancements like payment gateway integration, mobile responsiveness, and user analytics.

8. Conclusion & Future Scope

Summary of how the project successfully integrates modern technologies and future updates that could further enhance the system.

---

This table of contents gives a clear structure and detailed overview for the report, covering the different aspects of your e-commerce project with React, Django, and PHP. Each section is designed to delve deep into your project objectives, technologies used, system implementation, testing, and potential future enhancements.

1. Executive Summary

E-commerce online shop is the aim of this project and the design must be new and updated to give the best user experience. Based on React for the front-end and Django for the back-end, consumers can easily search products, put products in their basket and purchase without obstacles. The backend of the application is developed using Django and it deals with various tasks including user authentication, product management and order processing. On the other hand the PHP is used to send automated order confirmation messages to the users, thus enhancing user interaction. The combination of these technologies gives a satisfactory solution to e-commerce system that can easily be managed and expanded.  
  
PHP was used for the creation of the emails because of efficiency, flexibility and light weight as it is one of the best languages for server-side operations such as sending emails. The adoption of PHP in the implementation of the email module enables future changes on confirmational emails and managing the different format without much complications. The combination of such technologies as React, Django, and PHP used in this project makes it possible to have a modern and scalable project with the bonus of leveraging on the strengths inherent in each of the technologies.

2. Introduction

2.1 Background of the Project

Online shopping today has become prominent in the modern world, in which e-commerce is an inseparable component in retail. Consumers require fast, simple and intuitive methods for making a purchase of a product over the internet and firms have to have available robust processes in place. This project seeks to offer solution to the above mentioned needs by establishing a strong e-commerce platform powered by React, Django and PHP. React is used to drive the frontend part and make the working process comfortable for users. Django is the core framework which takes care of business logic and makes sure that the transactions are safe. PHP is responsible for automatic ordering of emails and this is an important feature as it keeps the customers informed of their order status.

2.2 Project Objectives

The project’s main objectives include:

* Providing a modern and responsive user interface: Customers must be able to generate meaningful information from the platform so as to be able to navigate it and purchase products.
* Ensuring secure and efficient backend operations: Django is used on the server-side for user authentication, management of orders and payments processing.
* Automating email notifications: PHP is implemented into the project to notify the customers of the orders they have made as soon as they make the orders to keep them informed.

2.3 Project Scope

The possible scope of this project may involve comes with designing and implementing an e-commerce platform one that has a friendly user interface as well as efficient back end functioning . Key features include:

Frontend: Developed using React for optimal user experience.

Backend: Powered by Django to handle data storage, user authentication, and order processing.

Mail Integration: A PHP script is used to send order confirmation emails to users after successful transactions.

Database: PostgreSQL is used to store product information, user data, and order details.

2.4 Operating Environment

The project runs in a web server environment that supports Django and PHP. The core technologies include:

Backend: Django (Python)

Frontend: React (JavaScript)

Email System: PHP for sending order confirmation emails

Database: PostgreSQL

Operating System: Linux or Windows servers with Apache or Nginx

3. Analysis

3.1 Software Requirement Specification

3.1.1 General Description

The e-commerce platform means that users can explore products, make decisions on which items to purchase and check out the items. The frontend is managed by React ensuring it offers the users a responsive and dynamic interface. Django deals with the back-end functionalities, which deals with data and its processes the request from the browser. After placing an order the system employ PHP to send an email confirming the order to the user.

3.1.2 Problem Statement

The conventional e-commerce infrastructures are characterized by slow response time, lack of a good integration platform, and rudimentary manner of dealing with email. Another common problem that comes to many platforms’ interest is managing the conflict of interest between an application’s user interface and performance at the back end. The first problem that this project solves is the combination of using the latest technologies such as React and Django with relatively basic yet efficient email system with PHP.

3.1.3 System Objective

Responsive User Interface: A modern UI using React ensures users have an intuitive and interactive experience while browsing and purchasing products.

Efficient Order Processing: Django handles server-side operations such as user authentication, product management, and order fulfillment.

Automated Email System: PHP ensures users receive prompt order confirmations, providing transparency in the purchase process.

3.2 Software and Hardware Requirements

3.2.1 Software Requirement

Backend: Django for handling server requests and database operations.

Frontend: React for building dynamic web pages.

Database: PostgreSQL for storing product and user data.

Mailing System: PHP for sending order confirmation emails.

Version Control: Git

3.2.2 Hardware Requirement

Processor: Intel i5 or higher

RAM: 8GB or higher

Storage: 500GB SSD or HDD

3.3 Existing vs Proposed

The traditional systems are most of the time characterized by slow page loading time and outdated design. With regards to the frontend, the utilization of React means that we develop a more effective and faster web application that results in improved user experience. In the backend, Django takes care of the work since all the operations including product management and user authentication are made secure and possible with scalability. Email confirmation of the order placed is a crucial part of the e-commerce business that PHP has a simplistic and, at the same time, powerful solution to implement.

3.4 Software Tools Used

Frontend: React.js (JavaScript framework)

Backend: Django (Python-based framework)

Database: PostgreSQL

Mail Integration: PHP for sending emails via PHPMailer

Version Control: Git for tracking changes and collaboration

4. Design

4.1 Data Flow Diagram (DFD)

Frontend, backend, and even individual elements of a third-party software like the email server will be able to maintain simple and coherent communication with the system. Level 0 DFD identifies the basic input and output between the users, the React frontend, Django backend, and PHP email system.

1. Users interact with the React frontend, which sends requests to Django.

2. Django handles server-side operations, interacting with the PostgreSQL database to retrieve or update information.

3. Upon successful order completion, Django triggers PHP to send an order confirmation email to the user’s email address.

5. System Implementation

5.1 Module Description

User Authentication: Django handles user login, registration, and session management.

Product Management: The product catalog is stored and managed through Django’s admin interface.

Order Processing: Django handles order placement, ensuring the cart is processed and stock levels are updated.

Email Notification: Once the order is placed, PHP sends an order confirmation email using PHPMailer.

5.2 Screenshots

homepage,

|  |
| --- |
| A close up of food  Description automatically generated |

|  |
| --- |
| A screenshot of food  Description automatically generated |

product,

|  |
| --- |
|  |

cart,

|  |
| --- |
| A screenshot of a food ordering  Description automatically generated |

Checkout page

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

Login page

|  |
| --- |
|  |

Register

|  |
| --- |
| A screen shot of a login form  Description automatically generated |

Forgot password

|  |
| --- |
| A screenshot of a login page  Description automatically generated |

Send activation link page

|  |
| --- |
| A screenshot of a login form  Description automatically generated |



Confirmation mail

|  |
| --- |
|  |

6. System Test

6.1 Integration Testing

Various tests were conducted to ensure smooth integration between React, Django, and PHP. The following were thoroughly tested:

Login and Registration: Testing Django’s authentication system.

Order Placement: Ensuring orders are processed and stored correctly in the database.

Email Sending: Ensuring PHP’s email system sends confirmation emails promptly.

6.2 Test Case for Login Page

Test Scenario: User inputs valid login credentials.

Expected Outcome: User is redirected to the homepage, and a session is created.

7. Maintenance and Update

7.1 Maintenance Plan

Maintenance: The application is expected to be updated frequently to enhance on its security and functionality. This also entails fixing the PHP mail system and ensuring the compatibility of the system with email servers and updating firefox and chrome django and React for security kills.

7.2 Future Enhancements

Payment Gateway Integration: Incorporating payment methods like PayPal or Stripe.

Mobile Responsiveness: Further improving the design for mobile users.

User Analytics: Implementing analytics to track user behavior and optimize product offerings.

8. Conclusion & Future Scope

The online store effectively applies the up-to-date technologies such as React and Django to enhance consumers’ satisfaction. One of the vital uses of PHP in the application is to notify the users of the various orders thanks to the order confirmation emails. Future updates could therefore centre on payment methods, mobile optimisation and the integration of user data for optimum customer insight.

**Group members**

* K C Udugamakorala
* MH Asara 32412
* DA DIAS 32502
* P.N.P.Mannage 33068
* B.S.L.R.Senarathna 32425
* P.Dusanthan 32969
* HVN Sithmini 33055
* N.Nijith 32816
* S.Naveendran 32791
* S.Danushkan 32405