**Applied Software Project Report**

By

ASHUTOSH MISHRA

**A Master’s Project Report submitted to Scaler Neovarsity - Woolf in partial fulfillment of the requirements for the degree of Master of Science in Computer Science**

Month of Submission May, 2025



**Scaler Mentee Email ID :** ashu7mishra@gmail.com

**Thesis Supervisor :** Naman Bhalla

**Date of Submission :** DD/MM/YYYY <Date of Submission>

**Certification**

I confirm that I have overseen / reviewed this applied project and, in my judgment, it adheres to the appropriate standards of academic presentation. I believe it satisfactorily meets the criteria, in terms of both quality and breadth, to serve as an applied project report for the attainment of Master of Science in Computer Science degree. This applied project report has been submitted to Woolf and is deemed sufficient to fulfill the prerequisites for the Master of Science in Computer Science degree.

Naman Bhalla

…………………

Project Guide / Supervisor

**DECLARATION**

I confirm that this project report, submitted to fulfill the requirements for the Master of Science in Computer Science degree, completed by me from 1st October 2024 to 31st December 2024, is the result of my own individual endeavor. The Project has been made on my own under the guidance of my supervisor with proper acknowledgement and without plagiarism. Any contributions from external sources or individuals, including the use of AI tools, are appropriately acknowledged through citation. By making this declaration, I acknowledge that any violation of this statement constitutes academic misconduct. I understand that such misconduct may lead to expulsion from the program and/or disqualification from receiving the degree.

**Ashutosh Mishra**

**<Signature of the Candidate> Date: 10 May 2025**

**ACKNOWLEDGMENT**

**I would like to express my heartfelt gratitude to my family for their unwavering support, encouragement, and belief in me throughout this journey. Their patience and understanding have been the foundation of my strength. I am deeply thankful to all the instructors and mentors at Scaler whose expert guidance, insightful feedback, and dedication to teaching have played a crucial role in shaping my skills and understanding. I also want to extend my sincere appreciation to my peers, friends, and everyone who inspired and motivated me during this program. Completing this Master's degree has been a transformative experience, and I am grateful to all who contributed to this achievement.**

**Table of Contents**

[**List of Tables 6**](#_9nnr2lniv90f)

[**List of Figures 7**](#_ju1gc9w3iuai)

[**Applied Software Project 8**](#_b4cf8683b1wd)

[Abstract 8](#_sj7c7bghlznr)

[Project Description 8](#_1z5fx61h0cc)

[Requirement Gathering 9](#_joagy45av5k0)

[Class Diagrams 9](#_nvf4h831fm8o)

[Database Schema Design 9](#_ydqs8nkbe6m9)

[Feature Development Process 11](#_p6mfl8dwb9sy)

[Deployment Flow 12](#_2mk44ad33gi)

[Technologies Used 12](#_wn68bn10ag78)

[Conclusion 13](#_4yf46wt6rx84)

[**References 14**](#_z0iyzog9l959)

## List of Tables

**(To be written sequentially as they appear in the text)**

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Title** | **Page No.** |
| **1** |  |  |
| **2** |  |  |

## List of Figures

**(List of Images, Graphs, Charts sequentially as they appear in the text)**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| **1** |  |  |
| **2** |  |  |

## **Applied Software Project**

### Abstract

### EcommApp is a full-stack, modular, and scalable e-commerce web application built using Django REST Framework (backend) and React.js (frontend), developed as part of the Scaler Neovarsity Master's degree program in Computer Science. This project aims to replicate and simplify real-world e-commerce business processes through software by integrating critical components such as user authentication, product and category management, order and cart handling, inventory tracking, payment gateway integration, and notifications. The system is architected for flexibility, maintainability, and performance. Razorpay is integrated to handle online payments securely, and Redis is employed for caching to enhance responsiveness.

### Objective

* Develop a modular backend that can handle core e-commerce operations.
* Implement clean REST APIs for frontend consumption.
* Integrate Razorpay for secure payments.
* Provide a configurable admin panel and dynamic user frontend.
* Enable caching and optimized query handling.
* Deploy using cloud infrastructure with containerization.
* Ensure secure authentication and authorization using token-based methods (JWT).
* Maintain inventory consistency and transactional integrity across services.
* Implement a scalable notification system for user communication.
* Design a database schema optimized for relational integrity and query performance.
* Ensure code maintainability and scalability through service separation and best practices.
* Facilitate API testing and documentation using Swagger/OpenAPI.

### Literature Review

The design and development of e-commerce platforms have been extensively explored in both academic and industrial domains. Research highlights the importance of modular architecture, RESTful APIs, secure authentication, and database optimization in developing scalable and maintainable systems. Microservice-based architectures are increasingly recommended for large-scale e-commerce systems to improve fault tolerance and service isolation.

Modern web development best practices emphasize decoupling the frontend and backend to allow independent scaling and deployment. Furthermore, integrating third-party services such as payment gateways, SMS/email notifications, and caching layers (like Redis) is a proven approach to enhancing system efficiency and responsiveness.

The Django REST Framework (DRF) is widely acknowledged for its simplicity, robustness, and rapid API development capability. Coupled with React.js for frontend, this combination supports a clean separation of concerns, code reuse, and fast user interfaces. This project is grounded in these proven patterns and principles.

### Market Research

India’s e-commerce market is projected to reach $200 billion by 2026, driven by increasing smartphone penetration, internet access, and digital payment adoption. Consumers today expect fast, intuitive, and secure online shopping experiences, raising the bar for performance and reliability.

Industry leaders like Amazon and Flipkart have set high standards with features such as real-time inventory updates, secure checkout, and intelligent product recommendations. Meanwhile, platforms like Shopify and WooCommerce cater to small businesses with plug-and-play e-commerce features.

EcommApp draws inspiration from both these spectrums—adopting scalable, cloud-ready backend strategies and combining them with admin configurability and modular code that could support plugins or new features in the future. This project is designed with future-readiness in mind, able to adapt to growing user demand, evolving business models, and technological advances.

### Project Description

**EcommApp** is a full-stack e-commerce platform designed to simulate real-world online retail experiences through modular, scalable, and secure backend systems. It supports user authentication, product and category management, shopping cart functionality, order processing, payment integration via Razorpay, inventory tracking, and notification delivery. The backend is developed using Django REST Framework, while the frontend uses React.js. The architecture is built with scalability, performance, and maintainability in mind, aligned with current industry standards.

The project follows agile software development practices with clearly defined phases: **Definition**, **Planning**, **Development**, and **Delivery**. This approach ensures that requirements are gathered iteratively and features are delivered incrementally, with feedback and reflection loops at each stage.

The system is also cloud-ready, containerized using Docker, and deployed on AWS to simulate real-world deployment practices, making it suitable for enterprise use cases or as a foundation for further development.

### Objectives Recap

* Build a modular backend that handles user, product, order, and payment workflows.
* Implement secure, well-documented REST APIs.
* Create a flexible admin dashboard and a responsive user-facing UI.
* Optimize performance using Redis caching and database indexing.
* Ensure scalable deployment via cloud infrastructure and containerization.

### Capstone Project Development Process



**Figure 1.1**: Project Development Process

This diagram represents the four key phases followed during the development of **EcommApp**:

* **Definition:** Project theme selection, user persona studies, and a design sprint.
* **Planning:** Backlog creation, technical setup, and project idea validation.
* **Development:** Conducted across 3 agile sprints, each ending with review and feedback.
* **Delivery:** Final project evaluation, documentation, and report preparation.

### Requirement Gathering

* Describe the requirements - Functional, Non-Functional Requirements
* Describe the Users and Use Cases
  + Include detailed Use Case Diagram / Images to illustrate
* List and detail out the Feature set -
  + Use Tables to list out the feature set

Use the below format as reference when including a table

**Table 1.1:** < Table caption > (Table captions go above tables.)

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

### Class Diagrams

Describe the Low Level Design of the Project…

Provide class diagrams - Provide proper captions and follow the proper format for including diagrams / figures / images

**Tip** - Make images using draw.io and paste here following the guidelines for adding images / figures

### Database Schema Design

Explain the **Low Level Design** of the Project in more detail by providing the **database schema**

**design** description

Provide the schema design textually as well as diagrammatically

Sample Schema Design described textually -

“”

Tables:

Batches

* + Batch\_id
  + Name
  + Start\_month
  + Current\_instructor
  + Batch\_type\_id
  + Primary Key(Batch\_id)

Students

* + student\_id
  + name
  + graduation\_year
  + University\_name
  + email
  + Phone\_number
  + batch\_id
  + Buddy\_id
  + Primary Key(student\_id)

Classes

* Class\_id
* Name
* Date
* Time
* Instructor
* Primary Key(Class\_id)

Mentors

* + Mentor\_id
  + Name
  + Current\_company
  + Primary Key(Mentor\_id)

Mentor\_Sessions

* mentor\_session\_id
* time
* Duration
* Student\_id
* Mentor\_id
* Student\_rating
* Mentor\_rating
  + Primary Key(mentor\_session\_id)

Batches\_Classes

* + Batch\_id
  + Class\_id
  + Primary Key(Batch\_id, Class\_id)

Student\_batch\_history

* + student\_id
  + batch\_id
  + Shift\_date
  + Primary Key(student\_id, batch\_id)

Batch\_type

* + Batch\_type\_id
  + Batch\_type
  + Primary Key(Batch\_type\_id)

**Foreign Keys:**

* Batches(batch\_type\_id) refers Batch\_type(batch\_type\_id)
* Students(batch\_id) refers Batches(batch\_id)
* Mentor\_Sessions(Student\_id) refers Students(Student\_id)
* Mentor\_Sessions(Mentor\_id) refers Mentors(Mentor\_id)
* Batches\_Classes(Batch\_id) refers Batches(batch\_id)
* Batches\_Classes(student\_id) refers Students(Student\_id)
* Student\_batch\_history(student\_id) refers Students(Student\_id)
* Student\_batch\_history(batch\_id) refers Batches(batch\_id)

**Cardinality of Relations:**

* Between Batches and Batch\_type -> m:1
* Between Students and Batches -> m:1
* Between Batches and Classes -> m:m

“”

### Feature Development Process

Pick One key feature - Talk about its development process, implementation and performance optimisation / metric optimisation achieved…

For example, ‘Book a seat’ feature in developing ‘BookMyShow’ app

Elaborate the request flow to backend

* 1. API Request Payload
  2. Service which picks the request
  3. Flow of MVC architecture

Explain the performance improvement / metric optimization achieved.

For example,

* Used Cache to reduce API Response time by X seconds…
* Optimized Query Response time by using Indexing…

Benchmarking of response time without the optimisation and post the optimisation

### Deployment Flow

Explain how the deployment will work via AWS (Describe the below) -

* EC2
* VPC
* Security Groups
* RDS
* Cache
* Managed Infra / Elastic Beanstalk

Use diagrams, images to explain better

### Technologies Used

Kafka, MySQL, Springboot, Cloud etc…

* For each key technology used in building the project,
  + Detail and describe each of them
  + Elaborate how they can be used in real life
  + Provide example of real-life applications using them

Use diagrams, images to explain better

**Tip** - Use the internet to improve your project but DO NOT PLAGIARIZE - Include proper references if you are quoting articles from the internet

### Conclusion

The Conclusion should include some key points as elaborated below -

* Key Takeaways: Highlight the important concepts and technologies learned from doing the Project
* Practical Applications : Significance of technologies with their real-world applications
* Limitations : Limitations of the technologies, cost implications and suggestions for improvement

## References

Include the websites or works or the list of works referred to in a text or consulted by you for writing this report

1. Name of the Website, Date and time of referring to the Website, Name of the Author, Title/Topic
2. Author Name, Title / Topic, Research Paper Name / Book Name, Year of Publication

Format Guidelines

1. Detailed and Elaborate report of 40 pages at least is expected
2. Margins - Every page of your document must meet the margin requirements of 1.25 inches on the left and right, and 1 inch on the top and bottom.
3. Font:
   1. Style: Times New Roman,
   2. Font Size:14 (For Headings), 12 (For body text) in black colored text.
   3. All text must be the same justification, like left justified or fully justified.
4. Line Spacing:
   1. Body of the text: 1.5
   2. List of Tables/graphs/charts/bibliography: single line.
5. Alignment:
   1. Title page: Centre
   2. Chapter Heading: Centre
   3. Subheading: Left
   4. Body of the text: Justify
6. Titles: All titles and subtitles should be in bold. All tables/graphs/charts/figures should have appropriate titles.
7. Numbering of the tables, charts, graphs should be in the following fashion: Second table/graph/chart in the second chapter should be numbered as Table/graph/chart no. 2.02; where the first digit stands for chapter no. and digits after (.) stands for number of table/graph/charts in that chapter. Same numbering should be followed for all other chapters.