SAHYADRI
COLLEGE OF ENGINEERING & MANAGEMENT
An Autonomous Institution
MANGALURU

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Hands-on Laravel Project Report

ON

Customer Feedback System

Name	Rohith M	Name	Karthik Acharya
USN	4SF23CS094	USN	4SF23CS082
Section	A	Section	A
Department	CSE	Department	CSE

Faculty In charge: Ms. Chaithra S

Assistant Professor

Department of Computer Science & Engineering

Sahyadri College of Engineering & Management

Adyar, Mangalore

2024-2025

SAHYADRI
COLLEGE OF ENGINEERING & MANAGEMENT
An Autonomous Institution
MANGALURU

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RUBRICS TO EVALUATE THE HANDS-ON PROJECT AND REPORT

Sl.No	Criteria	Excellent (4 Marks)	Good (3 Marks)	Average (2 Marks)	Needs Improvement (1 Mark)	Marks
1	**Functionality & Features**	Complete and flawless implementation of all required features; robust error and edge case handling	Most features implemented correctly; minor bugs or missing small features	Basic features implemented; some bugs or missing important features	Key features missing; frequent bugs and errors	6
2	**Code Quality & Security**	Code is clean, well-organized, follows Laravel best practices, secure and maintainable	Generally clean code with minor issues; basic security implemented	Code works but has poor structure or inconsistent style; minimal security	Poorly structured, insecure code; lacks validation and best practices	4
3	**User Interface & Experience**	UI is intuitive, fully responsive, user-friendly with clear feedback and error messages	UI mostly clear and functional; minor usability issues or inconsistencies	UI usable but clunky, not fully responsive; some missing feedback	UI is confusing, unresponsive, lacks feedback, or hard to navigate	3
4	**Database Design & Efficiency**	Database schema is well-designed with proper relationships; queries efficient and optimized	Schema mostly correct with minor inefficiencies or missing relations	Basic schema with some design flaws; queries may be inefficient	Poor schema design; incorrect or slow queries affecting functionality	3
5	**Report Quality**	Well-structured, clear, detailed report with screenshots, challenges, solutions, and conclusions	Complete report with good clarity; some minor details missing	Basic report covering main points but lacks depth or visuals	Incomplete, unclear, or poorly written report missing key sections	4

Total Marks Awarded

Student name	1	2	3	4	5	Total
Rohith M						
Karthik Acharya						

Signature

CHAITHRA S

TABLE OF CONTENTS

INTRODUCTION

In today’s fast-paced digital world, customer feedback has become a cornerstone for shaping better products and delivering exceptional services. One of the most intuitive and user-friendly ways to gather this feedback is through a star rating system, where users can quickly rate their experiences and share comments. This simple yet powerful tool empowers users to voice their opinions with just a few clicks, while offering valuable insights to businesses striving to grow.

A star rating-based feedback system not only fosters transparency but also helps build trust within the user community. By allowing people to see what others think, it encourages informed decision-making—whether it's choosing a product, booking a service, or exploring a new offering.

The feedback collected through this system can be analyzed to gauge overall customer satisfaction and pinpoint specific areas that need improvement. It becomes a living pulse of user sentiment, guiding product teams, service providers, and decision-makers in enhancing the customer experience.

This project aims to create a robust feedback platform where users can submit ratings and written reviews. On the backend, an intuitive moderation panel gives admins control to manage, review, and curate feedback as needed. Additionally, the system will calculate and display average ratings for each product or service, providing a clear snapshot of public sentiment at a glance.

Ultimately, this feedback system not only encourages user engagement but also supports a continuous cycle of improvement and innovation.

OBJECTIVES

The primary aim of this project is to build an intuitive, user-centric feedback management system that leverages star ratings to gather and display valuable user opinions about products or services. In today's digital ecosystem, customer feedback is a vital resource for maintaining quality, improving offerings, and ensuring user satisfaction. With that in mind, the following objectives were defined for the system:

1. Provide an interactive and responsive platform for feedback submission. The system should allow users to easily provide feedback through a clear, visually engaging star-rating interface accompanied by an optional comment section. Whether accessed on desktop or mobile devices, the interface must be responsive, accessible, and intuitive to encourage participation and reduce friction in the feedback process.

2. Accurately calculate and display average ratings for each product or service. Once feedback is collected, the system should process and display an aggregate rating—typically an average—based on all approved feedback submissions. This metric offers new users a quick, visual summary of how well a product or service is performing in the eyes of past users.

3. Equip administrators with powerful moderation tools. To maintain the integrity and quality of feedback, the system should give administrative users the ability to moderate content. This includes the ability to approve, reject, or delete submissions, ensuring that the feedback displayed publicly is constructive, respectful, and relevant.

4. Implement full CRUD (Create, Read, Update, Delete) operations for feedback records. Both users and administrators should be able to manage feedback data effectively. This includes not only the ability to submit feedback but also to view, edit, or remove entries depending on their role and permission level. This functionality ensures transparency, flexibility, and user control over submitted content.

5. Create a user-friendly experience that supports scalability and long-term usability. The entire system should be designed with simplicity and efficiency in mind. Users should find it effortless to submit feedback, while administrators should be able to moderate content without confusion. The architecture must also support future enhancements like image attachments, user responses, or data analytics without significant rework.

TOOLS AND TECHNOLOGIES USED

To build a robust and user-friendly feedback management system, the following technologies were used:

1. **Laravel**

   A PHP framework following the MVC pattern. It simplifies backend development with built-in features like routing, authentication, and database migrations.
2. **Blade**

   Laravel's templating engine that allows embedding PHP in HTML. It supports layout inheritance and clean separation of logic and UI.
3. **MySQL**

   An open-source relational database used to manage structured data like user info and feedback. Laravel's Eloquent ORM makes database interaction seamless.
4. **Bootstrap**

   A front-end framework that ensures responsive and consistent UI design across devices using pre-built components.
5. **JavaScript**

   Enhances interactivity, especially for dynamic elements like star ratings, improving user engagement.
6. **XAMPP**

   A local development environment that includes Apache, MySQL, and PHP—ideal for testing Laravel apps before deployment.

SYSTEM ARCHITECTURE

The architecture is based on a Model-View-Controller (MVC) pattern as provided by Laravel.

User Flow:

1. Users visit the product/service page.
2. Users input rating (1-5 stars) and textual feedback.
3. Feedback is stored in the MySQL database.
4. The average rating is calculated per product.
5. Admin reviews feedback through a moderation panel.

Component Overview:

- Frontend: Blade templates, star-rating UI, form submission.
- Backend: Laravel Controllers and Models for logic and DB operations.
- Database: Tables for users, products/services, feedback.
- Admin Panel: Route-protected UI for moderation (approve/delete feedback).

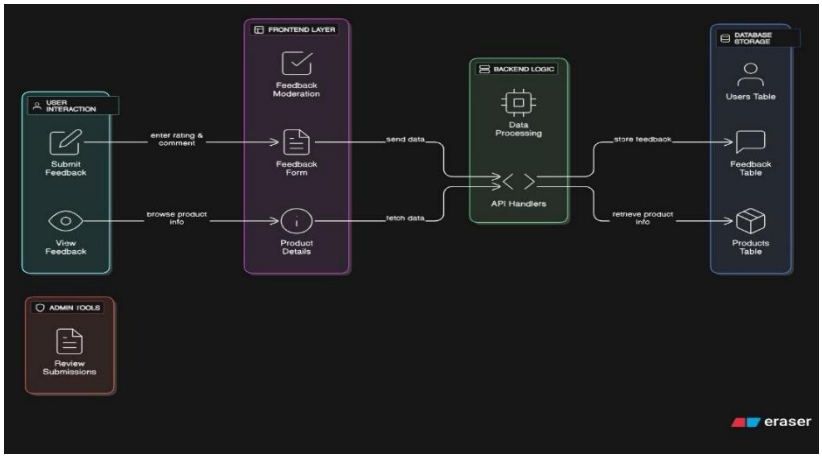

Fig:4.1 System architecture

DATABASE DESIGN

The feedback management system uses a relational database structure implemented in MySQL. The design ensures efficient storage, retrieval, and association with user feedback on specific products or services.

Key Tables:

1. Users

   Stores information about each user (user_id, name, email, password). Users are linked to the feedback they submit.

2. Products

   Contains product or service details (product_id, name, description). Each product can have multiple feedback entries associated with it.

3. Feedback

   This central table stores individual feedback records, including user_id, product_id, rating (1–5 stars), comment, status (pending, approved, rejected), and timestamps. It connects users to products via foreign keys.

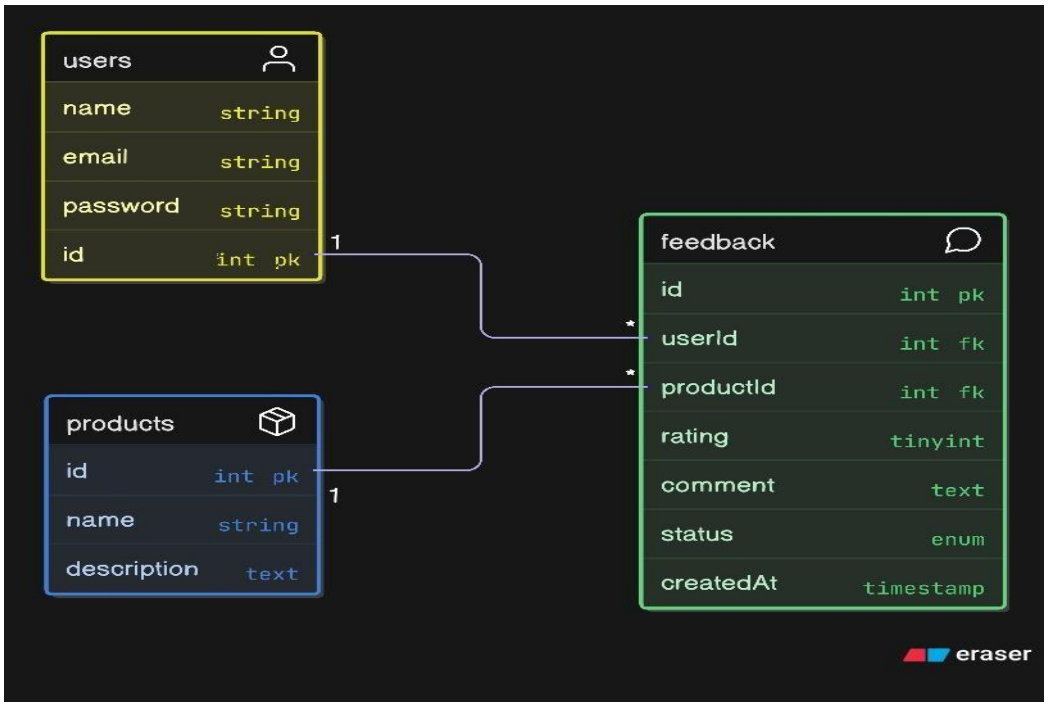

Fig5.1 Database Design

IMPLEMENTATION

The feedback management system was implemented using Laravel as the backend framework and Blade as the templating engine for the front-end. The goal was to create a user-friendly and functional system where users can easily submit reviews and administrators can manage them efficiently.

Frontend (Blade Templates)

The interface uses Laravel Blade templates, allowing users to submit feedback via a star-rating form with optional comments. Submitted feedback and average ratings are displayed to help users quickly assess product quality.

Backend (Laravel Logic)

Laravel handles all routing, data processing, and business logic. Key controllers include:

- FeedbackController: Manages feedback submission, listing, and average rating calculation.
- AdminFeedbackController: Provides moderation tools for admins to approve, reject, or delete feedback.

Eloquent ORM is used to interact with the MySQL database. Relationships are defined between users, feedback, and products to ensure smooth data operations.

Admin Panel

A separate admin interface allows moderators to manage feedback. They can filter feedback by product, view full comments, and perform actions like approve, reject, or delete—all from a single dashboard.

CONCLUSION

This project set out to create a platform where users could easily share their thoughts and experiences through feedback, and businesses or administrators could manage that feedback with confidence. From collecting star ratings to moderating user comments, the system provides a complete and practical solution for gathering insights.

One of the standout achievements of the system is its ease of use. The interface is clean and intuitive—allowing users to rate products or services quickly, add optional comments, and view overall ratings briefly. This simplicity encourages more engagement, helping service providers get a clearer picture of what users think.

On the backend, Laravel's powerful framework enabled a smooth and efficient development process. With organized routes, controllers, and Eloquent ORM, data flows cleanly from form inputs to database entries and back to the user view. MySQL provides the structure for storing feedback securely, while admin tools ensure only meaningful and appropriate feedback is displayed to others.

Perhaps most importantly, the system brings value to both users and administrators. Users feel heard. They can voice opinions that influence others and help shape product decisions. Administrators, on the other hand, get a flexible moderation panel to maintain quality and transparency.

In the end, the project not only meets its goals, but lays a solid foundation for future growth—whether through analytics, user profiles, or even AI-assisted sentiment tracking. It's a strong first step toward building a smarter, more responsive relationship between users and the services they engage with.

FUTURE ENHANCEMENT

1. Add user authentication with roles (admin/user).
2. Email notifications to admins on new feedback.
3. AJAX-based dynamic rating submission.
4. Add charts and analytics for admin dashboard.
5. Allow image uploads with feedback.
6. Enable replying to user feedback (admin responses).
7. Mobile app version using Flutter or React Native.

References

Laravel Documentation - *https://laravel.com/docs*

Blade Templating Engine - *https://laravel.com/docs/blade*

MySQL Official Documentation - *https://dev.mysql.com/doc/*

Bootstrap Documentation - *https://getbootstrap.com/docs/*

Stack Overflow - *https://stackoverflow.com/*