



**COLLEGE CODE:0004**

**COLLEGE NAME : MADRAS INSTITUTE OF  
TECHNOLOGY, ANNA UNIVERSITY.**

**DEPARTMENT : INFORMATION TECHNOLOGY.**

**Completed the project named as**

**Phase : 5 – FE**

**NAME : FEEDBACK COLLECTION SYSTEM**

**SUBMITTED BY:**

**NAME :KALIMUTHU K**

# Feedback Collection System-project

## Phase 5 :Project Demonstration & Documentation

### Introduction

The Feedback Collection System is a comprehensive full-stack web application built using the MERN stack (MongoDB, Express, React, and Node.js).

The primary goal of this project is to modernize the process of collecting and analyzing user feedback by providing a digital platform that is both interactive and efficient.

The system enables administrators to create dynamic, customizable feedback forms, and allows users to submit responses effortlessly from any device.

Collected responses are automatically stored in a MongoDB database, and real-time analytics are generated to help organizations interpret feedback effectively.

This project serves as a practical implementation of core full-stack development concepts, including frontend–backend integration, authentication, database management, and visualization.

### Final Demo Walkthrough

The **Feedback Collection System** was successfully developed and demonstrated as a fully functional web application using the **MERN stack**. The system enables users to **submit feedback**, while admins can **view, analyze, and export feedback insights** through a secure dashboard.

### Demonstration Flow

#### Step 1 — Landing Page

- Displays a list of available feedback forms fetched from MongoDB.
- Includes a search bar for quick access to specific forms.
- Provides navigation buttons for “*Give Feedback*” and “*Creator Login*”.

#### Step 2 — Feedback Submission

- Users access a feedback form through a unique public link.
- The form includes customizable fields such as:

- Name, Email
- Rating (1–5 scale)
- Comments (Text Input)
- Optional multiple-choice and checkbox questions
- Data is validated and submitted via Axios to the backend endpoint /api/forms/:id/submit.
- Success confirmation message appears upon submission.

### Step 3 — Creator (Admin) Login & Registration

- Admins register with an email, password, and optional organization name.
- Passwords are hashed using bcrypt.js.
- JWT-based authentication ensures secure login sessions.
- Admin credentials are verified via /api/auth/login.

### Step 4 — Admin Dashboard

- After login, the dashboard provides:
  - **Form Management** – Create, edit, and delete feedback forms.
  - **Analytics Visualization** – View average ratings, comment summaries, and feedback count.
  - **Charts & Graphs** – Displayed using react-chartjs-2.
  - **Export Options** – Download reports in CSV or PDF formats.
- Admin routes are protected by middleware (authMiddleware.js) to ensure secure data access.

### Step 5 — Real-Time Analytics

- All feedback data is fetched using MongoDB aggregation queries.
- Visualization modules display:
  - Rating distribution (bar chart)
  - Feedback frequency (pie chart)
  - Top recurring keywords (word cloud or text summary)

# Project Report

## System Overview

### Admin Features

- Admins can log in using secure authentication credentials.
- Ability to create, edit, and publish feedback forms dynamically.
- View submissions and analytics dashboards instantly.
- Export data in multiple formats for record-keeping.

### User Features

- Access public feedback forms via a unique link.
- Submit responses without the need for login.
- Simple and intuitive form interface optimized for both desktop and mobile.

### Analytics Features

- Real-time visualization of responses using **Chart.js / Recharts**.
- Metrics such as rating averages, option frequencies, and text feedback summaries.
- Interactive charts for administrators to monitor trends.

## Technologies Used

Component	Technology Used	Purpose
Frontend	React.js, Material UI, Axios	User interface & data communication
Backend	Node.js, Express.js	API routing, business logic, validation
Database	MongoDB Atlas	Cloud-based data storage
Authentication	JWT + bcrypt.js	Secure access control
Analytics	Chart.js / Recharts	Real-time visual analytics
Deployment	Vercel, Render	Hosting & deployment

## Major Modules

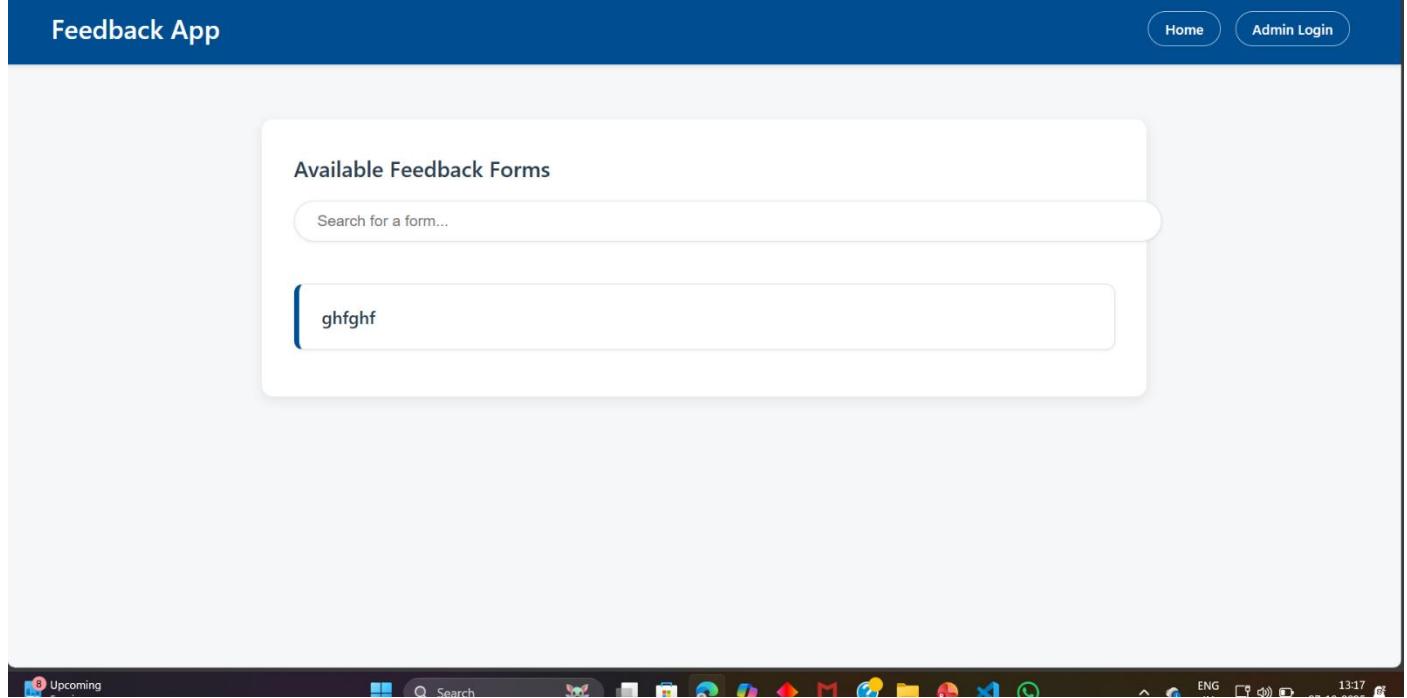
1. Landing Page – Entry for users and creators.
2. Feedback Form Page – Dynamic question rendering and feedback submission.
3. Creator Login/Register Page – Secure admin access and form management.
4. Admin Dashboard – Analytics visualization and report export options.

## Screenshots / API Documentation

### Screenshots

#### Landing Page

Displays available feedback forms and navigation options.



# User Feedback Form

Allows submission of ratings, text, and choices.

A screenshot of a web browser window titled "Feedback App" at "localhost:3000". The page has a dark blue header with "Feedback App" and "Admin Login" buttons. Below the header is a search bar labeled "Available Feedback Forms" with placeholder text "Search for a form...". A vertical scroll bar is visible on the right. A list of five feedback forms is displayed in a grid:

- ghfghf
- Product Quality
- User Experience
- Performance
- Design & Appearance

# Creator Login Page

mail/password-based authentication.

A screenshot of a web browser window titled "Feedback App" at "localhost:3000". The page has a dark blue header with "Feedback App" and "Admin Login" buttons. Below the header is a "Admin Login" form:

Email ID

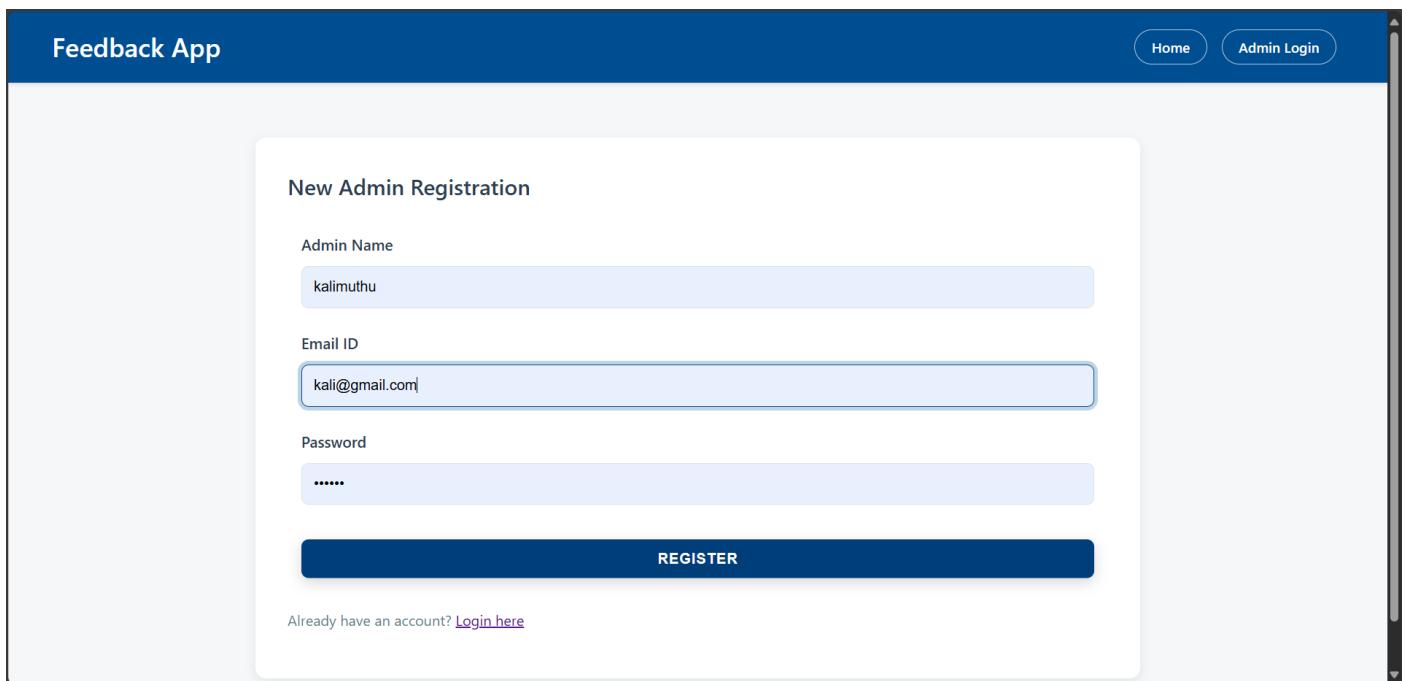
Password

**LOGIN**

Don't have an admin account? [Register here](#)

## Register page

For new admin signup page



The screenshot shows a registration form titled "New Admin Registration". It includes fields for "Admin Name" (containing "kalimuthu"), "Email ID" (containing "kali@gmail.com"), and "Password" (containing "\*\*\*\*\*"). A "REGISTER" button is at the bottom, and a link to "Login here" is at the bottom left.

New Admin Registration

Admin Name

kalimuthu

Email ID

kali@gmail.com

Password

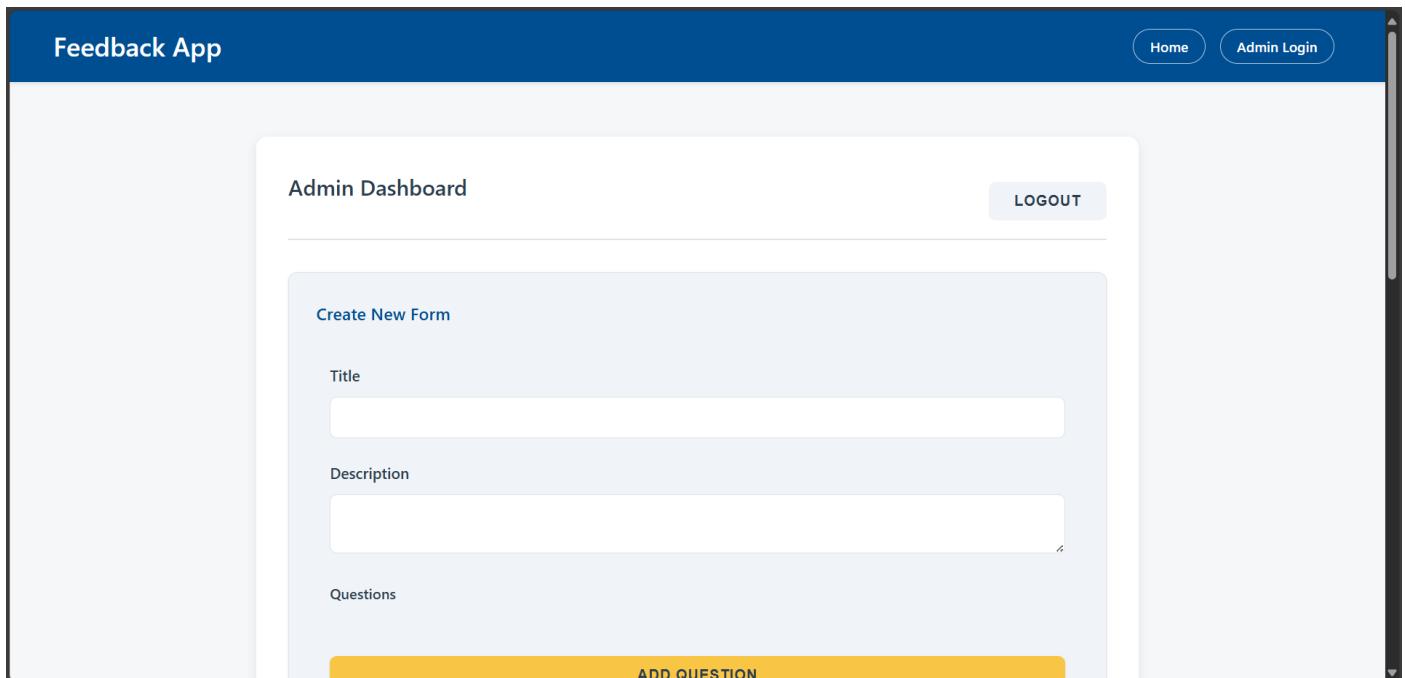
\*\*\*\*\*

REGISTER

Already have an account? [Login here](#)

## Admin Dashboard

Interactive analytics with charts.



The screenshot shows the Admin Dashboard with a "Create New Form" section. It has fields for "Title" and "Description", and a "Questions" section. A yellow "ADD QUESTION" button is at the bottom.

Feedback App

Home Admin Login

Admin Dashboard

LOGOUT

Create New Form

Title

Description

Questions

ADD QUESTION

Admin Dashboard

LOGOUT

Create New Form

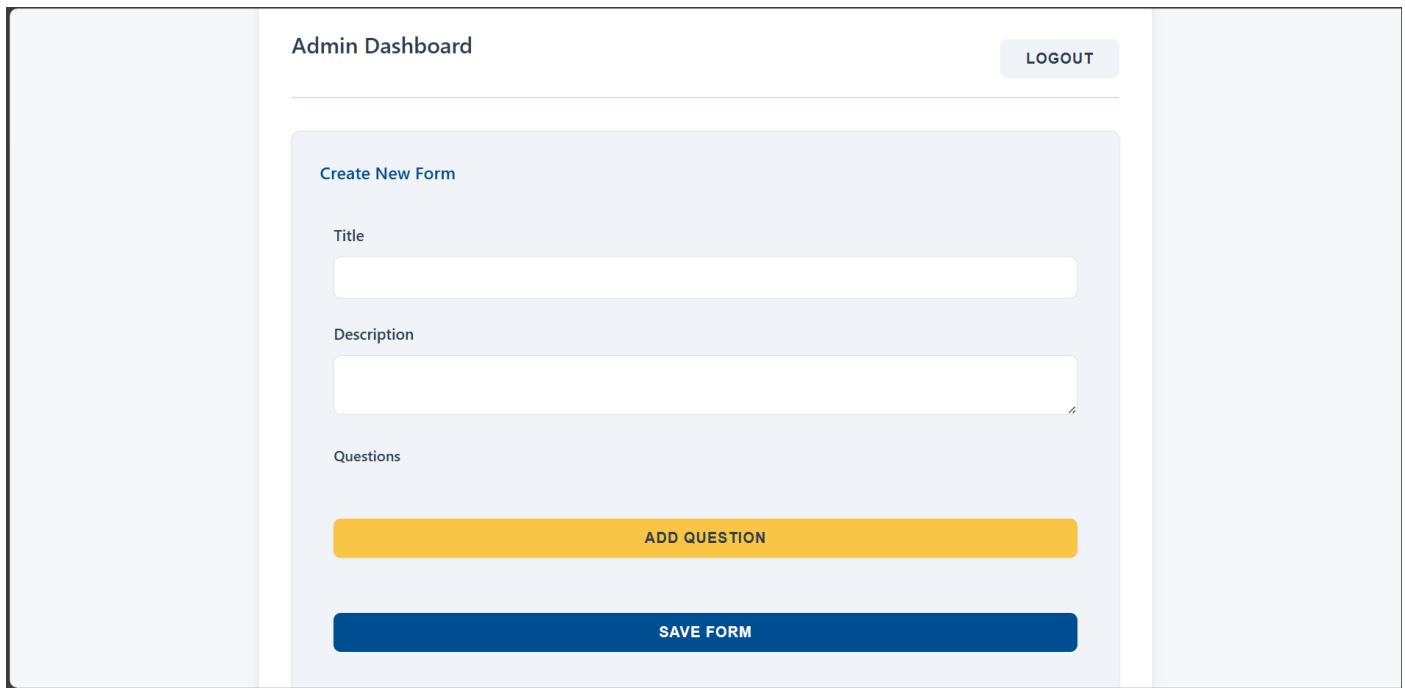
Title

Description

Questions

ADD QUESTION

SAVE FORM

The image shows a wireframe of an 'Admin Dashboard' interface. At the top, there's a 'LOGOUT' button. Below it is a 'Create New Form' section. This section includes fields for 'Title' (with an input field), 'Description' (with a text area), and 'Questions'. A yellow 'ADD QUESTION' button is positioned below the description field. At the bottom of the form creation area is a dark blue 'SAVE FORM' button.

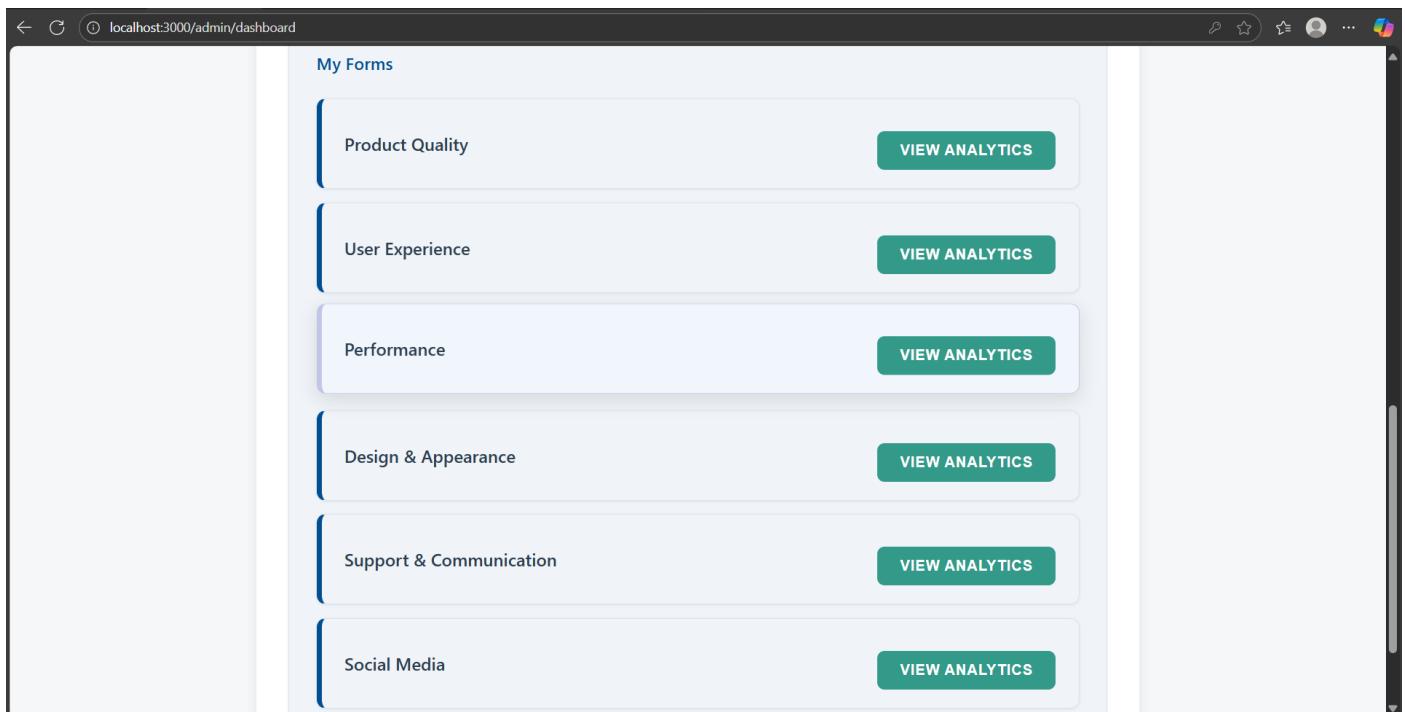
## Admin From

Admin can created the from for many.

localhost:3000/admin/dashboard

My Forms

- Product Quality [VIEW ANALYTICS](#)
- User Experience [VIEW ANALYTICS](#)
- Performance [VIEW ANALYTICS](#)
- Design & Appearance [VIEW ANALYTICS](#)
- Support & Communication [VIEW ANALYTICS](#)
- Social Media [VIEW ANALYTICS](#)

A screenshot of a web browser showing the 'My Forms' section of the admin dashboard. The URL 'localhost:3000/admin/dashboard' is visible in the address bar. The page lists six form categories: 'Product Quality', 'User Experience', 'Performance', 'Design & Appearance', 'Support & Communication', and 'Social Media'. Each category has a green 'VIEW ANALYTICS' button to its right. The entire list is contained within a white box with a thin gray border.

# Users

Users can search the form for feedback and submit.

The screenshot shows the 'Feedback App' homepage at localhost:3000. The title 'Feedback App' is at the top left, and 'Home' and 'Admin Login' buttons are at the top right. A search bar says 'Search for a form...'. Below it is a list of five feedback forms: 'ghfghf', 'Product Quality', 'User Experience', 'Performance', and 'Design & Appearance'. Each form has a blue header and a white body.

The screenshot shows the 'Product Quality' feedback form at localhost:3000/form/68fff6cee312c78d5b1886bc. The title 'Product Quality' is at the top left. It asks 'Rate the quality and reliability of our product/service.' Below are fields for 'Name (Optional)' with input 'syed' and 'Email (Optional)' with input 'syed@gmail.com'. A question 'How satisfied are you with the overall quality of the product?' follows, with a series of radio buttons numbered 1 through 5.

## View Analytics

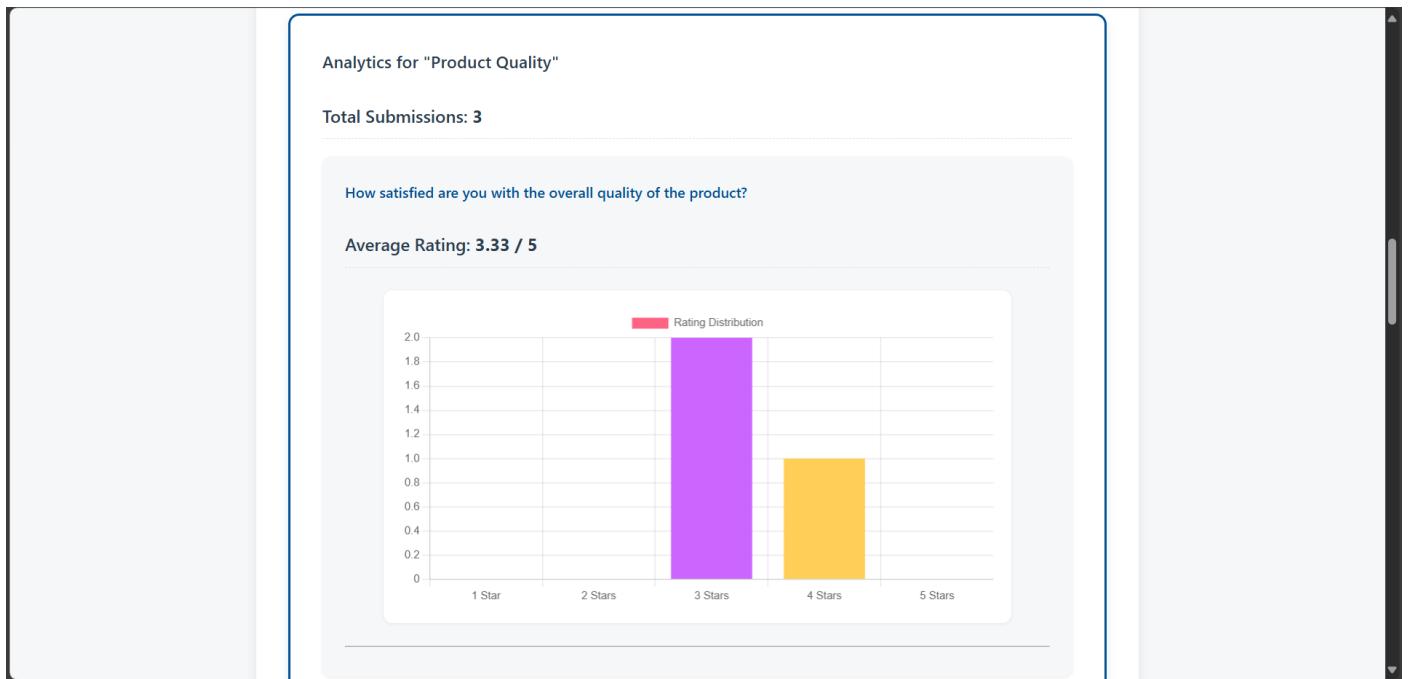
Admin can view the analytics how many user feedback and are we change will analyes.

The screenshot shows a web browser window with the URL `localhost:3000/admin/dashboard`. The main content area is titled "My Forms". It lists six form categories, each with a "VIEW ANALYTICS" button:

- Product Quality
- User Experience
- Performance
- Design & Appearance
- Support & Communication
- Social Media

The screenshot shows the "Analytics for 'User Experience'" page. It displays the following information:

- Total Submissions: 0
- How easy is it to use the system?
- Average Rating: 0.00 / 5
- Rating Distribution chart (empty)



## API Documentation

Method	Endpoint	Description
POST	/api/auth/register	Register a new admin
POST	/api/auth/login	Login admin and issue JWT token
GET	/api/forms	Retrieve all forms created by admin
POST	/api/forms	Create a new feedback form
GET	/api/forms/:id	Retrieve a specific form
POST	/api/forms/:id/submit	Submit user feedback
GET	/api/analytics/:id	Fetch analytical data for a form

## Challenges & Solutions

Challenge	Description	Implemented Solution
<b>Data Validation</b>	Invalid or incomplete user inputs during submission.	Used express-validator for input sanitization and server-side validation.
<b>Authentication Security</b>	Prevent unauthorized dashboard access.	Implemented JWT authentication with role-based middleware.
<b>Real-Time Chart Updates</b>	Delay in analytics rendering after new feedback submission.	Integrated frontend re-render triggers via React state management (useEffect hooks).
<b>Cross-Origin Issues</b>	CORS errors between frontend and backend servers.	Configured CORS middleware in Express and proxy setup in React.
<b>Deployment Integration</b>	Backend and frontend communication on cloud.	Deployed backend using Render & frontend using Vercel with environment-based API URLs.

## GitHub README & Setup Guide

### Project Overview

The Feedback Collection System is a full-stack web application built using the MERN (MongoDB, Express, React, Node.js) stack.

It enables organizations, educators, and developers to create, distribute, and analyze custom feedback forms — similar to Google Forms — with real-time analytics and an intuitive user experience.

Admins can dynamically design feedback forms (A–Z categories, rating, text, multiple-choice), while users can easily respond via a clean, responsive interface.

All submissions are securely stored in MongoDB and instantly reflected on the admin dashboard with analytical insights.

## Key Features

### Dynamic Form Builder

Create fully customized forms with drag-and-drop simplicity.

### Supports multiple field types:

#### Text input:

The screenshot shows a web-based form builder interface. At the top, there's a title field containing "Comprehensive Feedback Form". Below it is a description field with a large empty text area. Under the "Questions" heading, a question card is displayed with a "Question Text" input field and a dropdown menu. The dropdown menu is open, showing "Text Input (Short Answer)" as the selected option, along with other choices like "Rating (1-5 Stars)", "Multiple Choice (Radio)", and "Checkboxes". To the right of the dropdown is a "REMOVE" button. Below the dropdown is a yellow "ADD QUESTION" button. At the bottom of the screen is a large blue "SAVE FORM" button.

### Secure Admin Panel

Admin login & registration using Email ID and Password.

Role-based access control to manage forms and view submissions securely.

### Real-time Analytics

Instant data visualization with charts and tables:

Rating distributions

Choice frequency

Comments and textual responses

### Public Form Access

Published forms can be accessed publicly via links or searched by title.

Smooth, mobile-friendly UI for user submissions.

## **Aesthetic UI/UX**

Modern and clean design with professional color themes.

Responsive layout for desktops, tablets, and mobile devices.

Simple, intuitive navigation inspired by Google Forms.

## **Tech Stack:**

### **Frontend:**

React.js

React Router

Axios

Chart.js / D3.js (for analytics)

### **Backend:**

Node.js

Express.js

MongoDB

JWT Authentication

Bcrypt for password hashing

## **Database:**

MongoDB Atlas (Cloud)

## **Installation & Setup**

### **Clone the repository**

```
git clone https://github.com /feedback-collection-system.git
```

```
cd feedback-collection-system
```

## **Install dependencies**

### **For backend:**

```
cd backend  
npm install
```

### **For frontend:**

```
cd ../frontend  
npm install
```

## **Set up environment variables**

Create a .env file in the backend directory and add:

```
PORT=5000  
MONGO_URI=mongodb://localhost:27017/feedbackapp  
JWT_SECRET=your_super_secret_jwt_key
```

## **Run the application**

### **Run backend:**

```
cd backend  
npm start
```

### **Run frontend:**

```
cd ../frontend  
npm start
```

Frontend runs on(ie. http://localhost:3000)

Backend runs on (ie.http://localhost:5000)

## **Future Enhancements**

- Role-based multi-admin support
- Email notifications for submissions
- AI-based feedback summary generation

Export analytics reports (PDF, Excel)

Dark mode toggle

## License

This project is licensed under the MIT License — you're free to use, modify, and distribute it with attribution.

## Conclusion

The **Feedback Collection System** successfully demonstrates a scalable, secure, and user-friendly solution for collecting and analyzing user opinions.

It bridges the gap between traditional survey tools and modern analytics by providing **real-time visualization, cloud-based storage, and intuitive design**.

Through this project, advanced concepts of **MERN full-stack development, RESTful API design, authentication, and deployment** were effectively implemented and validated.

## Final Submission (Repo):

**Github Repository:** [Feedback-collection-system](#)