A blue and black logo

Description automatically generatedA blue and black logo

Description automatically generated

**Module Name - Cyber Security**

**Assignment No- SE-C2-U14465-Assignment 1**

**Digital Artifact Documentation**

**Table of Contents**

[**Section 01 – Installation Documentation 2**](#_Toc184814773)

[**Installation and Guide 2**](#_Toc184814774)

[**Overview 2**](#_Toc184814775)

[**Prerequisites 2**](#_Toc184814776)

[**Installation Guide 2**](#_Toc184814777)

[**Technical Explanation 3**](#_Toc184814778)

[**Section 02 API Documentation 3**](#_Toc184814779)

[**Section 03 - Server Documentation 10**](#_Toc184814780)

[**Section 04 – Client Documentation 13**](#_Toc184814781)

[**References 19**](#_Toc184814782)

# Section 01 – Installation Documentation

## Installation and Guide

## Overview

The Flashcard Management System is designed for creating, managing, and reviewing flashcards to facilitate efficient learning. This document provides comprehensive guidance on setting up, running, and testing the system.

## Prerequisites

* Ensure your environment meets the following requirements:
* Node.js (Version 14 or higher): [Download](<https://nodejs.org/>)
* npm (Comes with Node.js)
* MongoDB (Hosted(MongoDB Atlas) or Local Instance)

## Installation Guide

1. Step 01 - Clone the Repository

* $ git clone <Repository URL>
* Cd Flashcard System

1. Step 02 – Install Dependencies
   * Navigate to both backend and frontend directories and install dependencies:
   * $ cd backend
   * $ npm install
   * $ cd frontend
   * $ npm install
2. Step 03 - Configure Environment Variables
   * Create `.env` files in both the `backend` and `frontend` directories. Configure the following.
   * Backend `.env`

PORT=5000

MONGO\_URI=<Your MongoDB Connection String>

JWT\_SECRET=<Your JWT Secret>

* + Frontend `.env`

REACT\_APP\_API\_URL=http://localhost:5000

1. Step 04 – Run the Application
   * Start the backend and frontend servers

In the backend folder

$ npm run dev

In the frontend folder

$ npm start

* + Access the application at `http://localhost:3000`.

## Technical Explanation

The Flashcard Management System uses the following stack.

* + Frontend: React with Material-UI for responsive and modern UIs.
  + Backend: Node.js with Express for RESTful API development.
  + Database: MongoDB for scalable, document-based storage.

The system’s architecture is designed to handle user authentication, CRUD operations for flashcards, and secure API integrations.

# Section 02 API Documentation

1. User Authentication APIs

* Register User

Method: POST

Endpoint: `/api/users/register`

Request Body:

```json

{

"name": "John Doe",

"email": "john@example.com",

"password": "123456"

}

```

Response:

```json

{

"\_id": "1234567890abcdef",

"name": "John Doe",

"email": "john@example.com",

"isAdmin": false,

"token": "jwt-token"

}

```

* Login User

Method: POST

Endpoint: `/api/users/login`

Request Body:

```json

{

"email": "john@example.com",

"password": "123456"

}

```

Response:

```json

{

"\_id": "1234567890abcdef",

"name": "John Doe",

"email": "john@example.com",

"isAdmin": false,

"token": "jwt-token"

}

* Get User Profile

Method: GET

Endpoint: `/api/users/profile`

Headers:

```

Authorization: Bearer <token>

```

Response:

```json

{

"\_id": "1234567890abcdef",

"name": "John Doe",

"email": "john@example.com",

"isAdmin": false

}

2. Flashcard APIs

* Create Flashcard

Method: POST

Endpoint: `/api/flashcards`

Headers:

```

Authorization: Bearer <token>

```

Request Body:

```json

{

"title": "JavaScript Basics",

"description": "Learn the basics of JavaScript",

"cards": [

{ "question": "What is JavaScript?", "answer": "A programming language." },

{ "question": "What is a variable?", "answer": "A container for data." }

],

"isHidden": false

}

```

Response:

```json

{

"\_id": "60d5f9c1c2d1c8c45678abc1",

"title": "JavaScript Basics",

"description": "Learn the basics of JavaScript",

"cards": [

{ "question": "What is JavaScript?", "answer": "A programming language." },

{ "question": "What is a variable?", "answer": "A container for data." }

],

"isHidden": false

}

* Get All Public Flashcards

Method: GET

Endpoint: `/api/flashcards`

Response:

```json

[

{

"\_id": "60d5f9c1c2d1c8c45678abc1",

"title": "JavaScript Basics",

"description": "Learn the basics of JavaScript",

"user": { "\_id": "60d5f9c1c2d1c8c45678abc0", "name": "John Doe" }

}

]

* Get Flashcard by ID

Method: GET

Endpoint: `/api/flashcards/:id`

Response:

```json

{

"\_id": "60d5f9c1c2d1c8c45678abc1",

"title": "JavaScript Basics",

"description": "Learn the basics of JavaScript",

"cards": [

{ "question": "What is JavaScript?", "answer": "A programming language." },

{ "question": "What is a variable?", "answer": "A container for data." }

]

}

* Update Flashcard

Method: PUT

Endpoint: `/api/flashcards/:id`

Headers:

Authorization: Bearer <token>

Request Body:

```json

{

"title": "Updated Title",

"description": "Updated description",

"cards": [

{ "question": "Updated Question?", "answer": "Updated Answer." }

]

}

```

Response:

```json

{

"\_id": "60d5f9c1c2d1c8c45678abc1",

"title": "Updated Title",

"description": "Updated description",

"cards": [

{ "question": "Updated Question?", "answer": "Updated Answer." }

]

}

* Delete Flashcard

Method: DELETE

Endpoint: `/api/flashcards/:id`

Headers:

```

Authorization: Bearer <token>

```

Response:

```json

{ "message": "Flashcard deleted successfully" }

* Rate a Flashcard

Method: POST

Endpoint: `/api/flashcards/:id/rate`

Headers:

```

Authorization: Bearer <token>

```

Request Body:

```json

{ "rating": 4 }

```

Response:

```json

{ "message": "Rating submitted successfully" }

```

2.7 Add a Review

Method: POST

Endpoint: `/api/flashcards/:id/reviews`

Headers:

```

Authorization: Bearer <token>

```

Request Body:

```json

{

"comment": "Great flashcard set!",

"rating": 5

}

```

Response:

```json

{ "message": "Review added successfully" }

# Section 03 - Server Documentation

**Server Testing Using Postman**

Postman is used to test API endpoints during development. Below are examples for key functionalities.

1. Testing User Authentication

* Register User

Endpoint:`POST /api/users/register`

Body:

```json

{

"name": "John Doe",

"email": "john@example.com",

"password": "123456"

}

```

Response:

```json

{

"\_id": "60d5f9c1c2d1c8c45678abc1",

"name": "John Doe",

"email": "john@example.com",

"isAdmin": false,

"token": "jwt-token"}

* Login User:

Endpoint: `POST /api/users/login`

Body:

```json

{

"email": "john@example.com",

"password": "123456"

}

```

Response:

```json

{

"\_id": "60d5f9c1c2d1c8c45678abc1",

"name": "John Doe",

"email": "john@example.com",

"isAdmin": false,

"token": "jwt-token"

}

* Testing Flashcard Operations

Create Flashcard:

Endpoint: `POST /api/flashcards`

Headers:

```

Authorization: Bearer <jwt-token>

```

Body:

```json

{

"title": "JavaScript Basics",

"description": "Learn JavaScript fundamentals",

"cards": [

{ "question": "What is JavaScript?", "answer": "A programming language." }

]

}

Response:

```json

{

"\_id": "60d5f9c1c2d1c8c45678abc2",

"title": "JavaScript Basics",

"description": "Learn JavaScript fundamentals",

"cards": [

{ "question": "What is JavaScript?", "answer": "A programming language." }

]

}

* Delete Flashcard:

Endpoint: `DELETE /api/flashcards/:id`

Headers:

```

Authorization: Bearer <jwt-token>

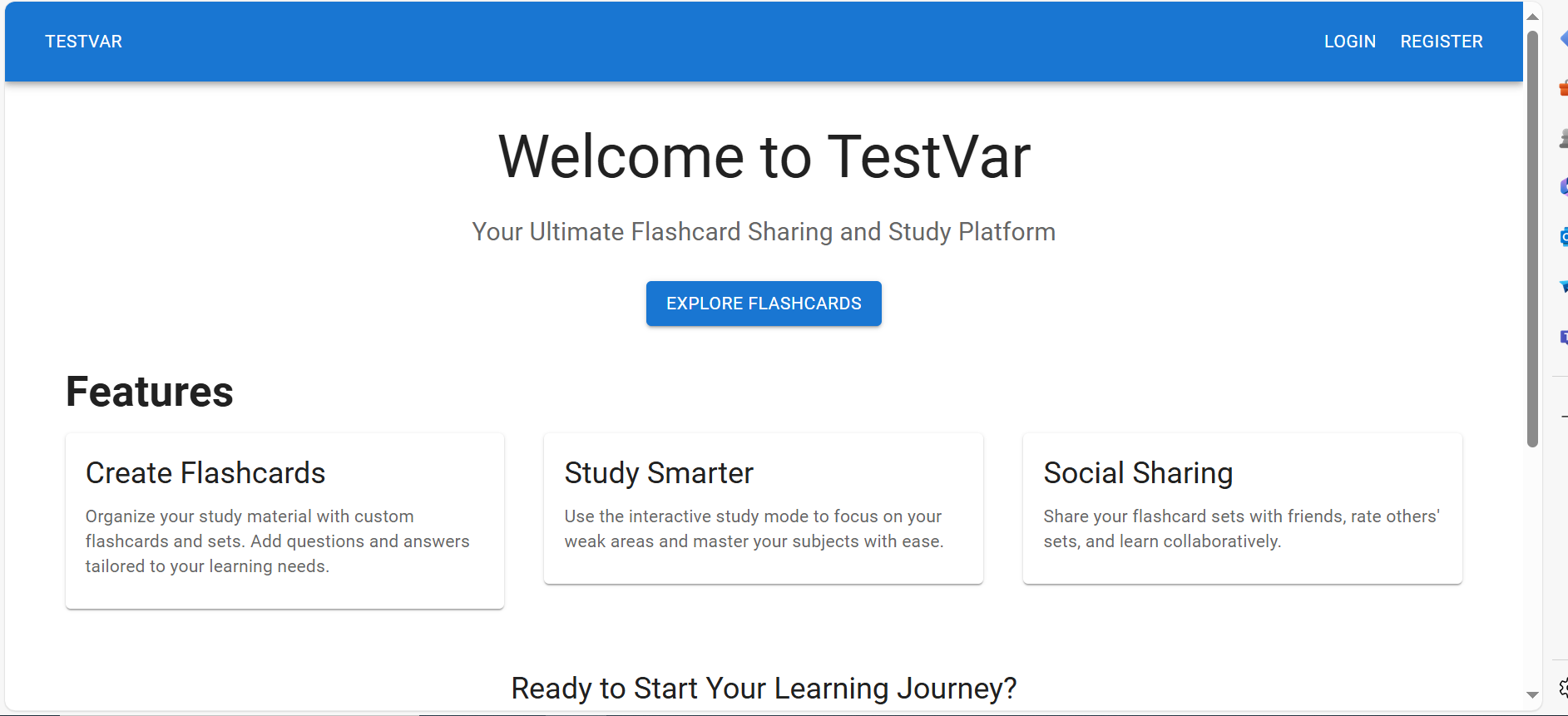
```

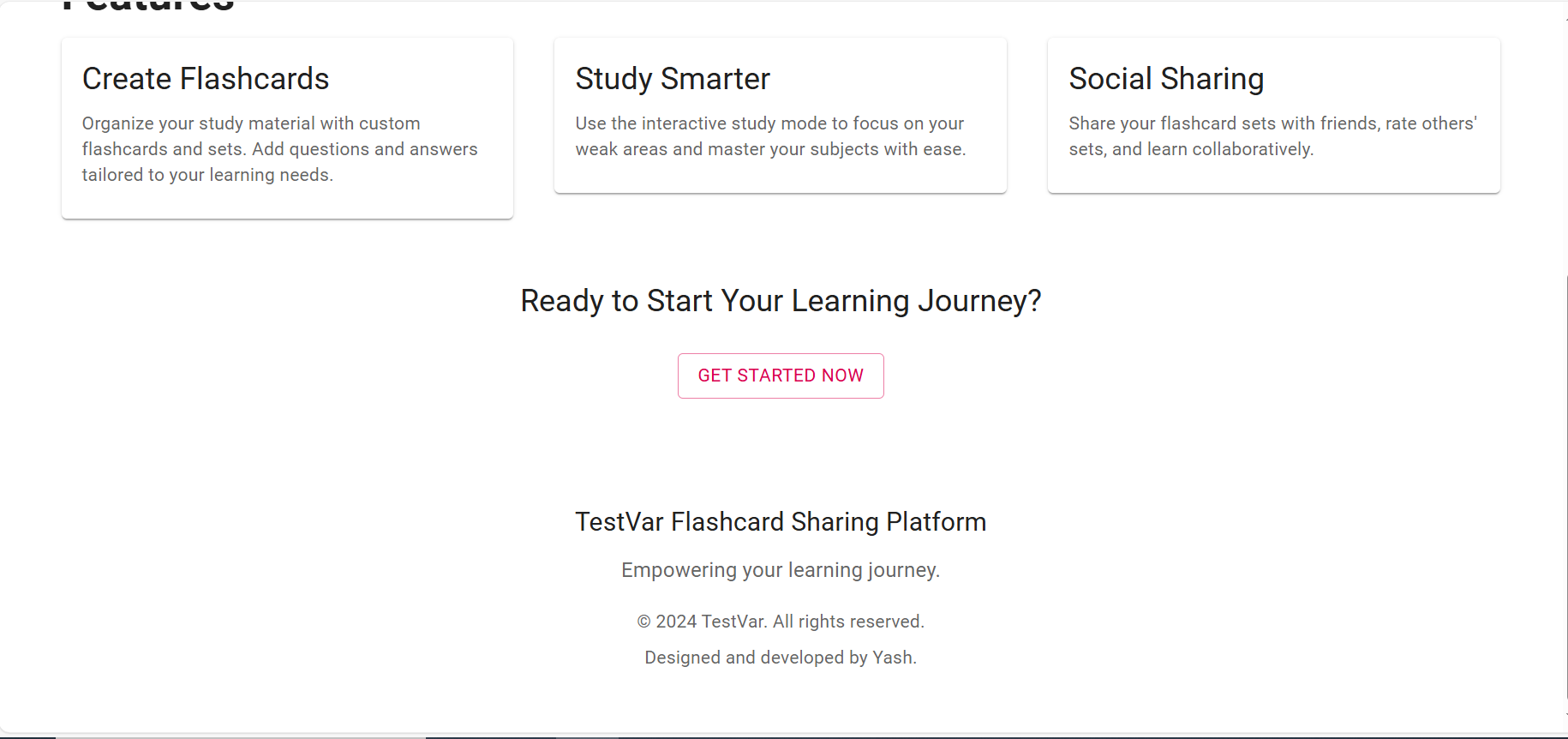
Response:

```json

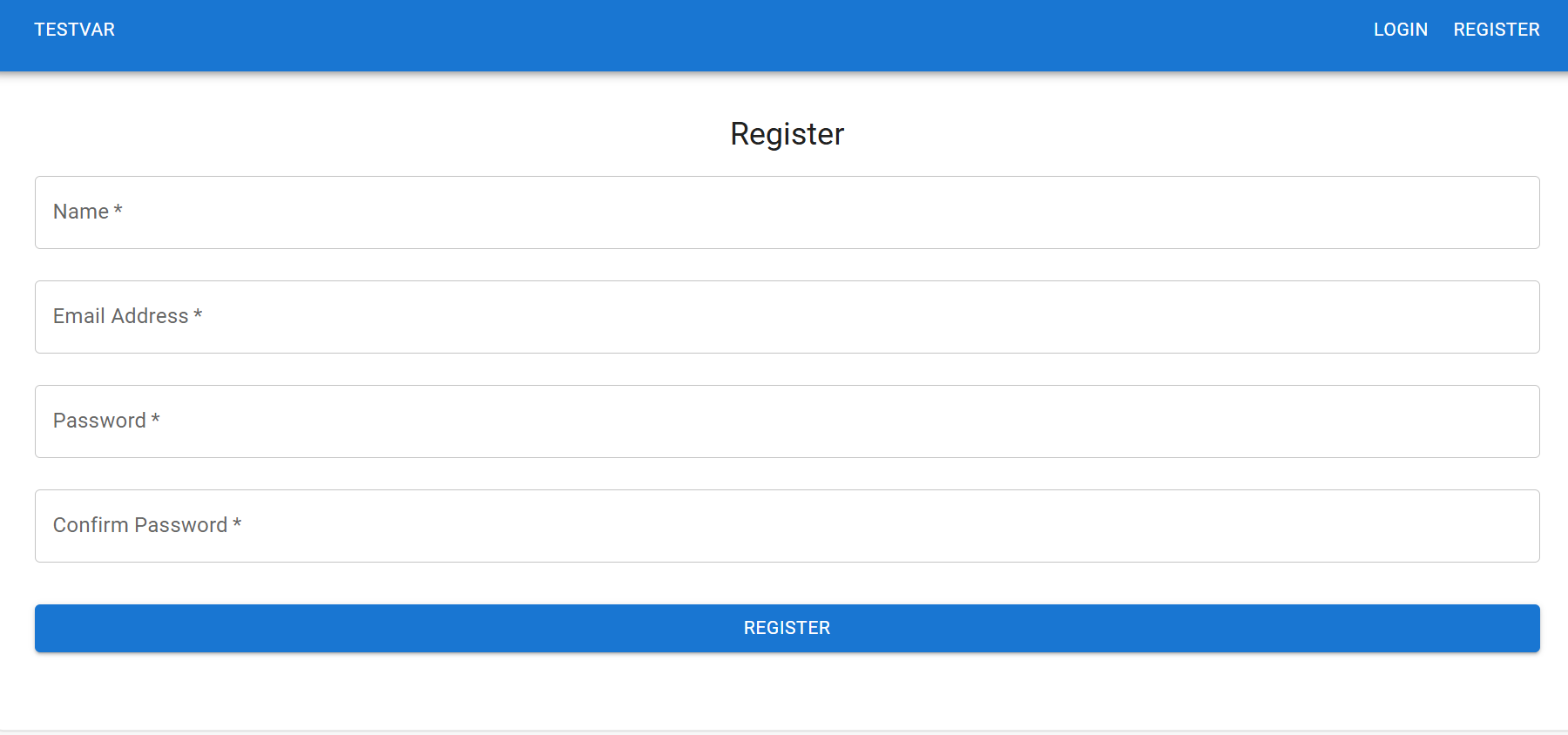
{ "message": "Flashcard deleted successfully" }

# Section 04 – Client Documentation

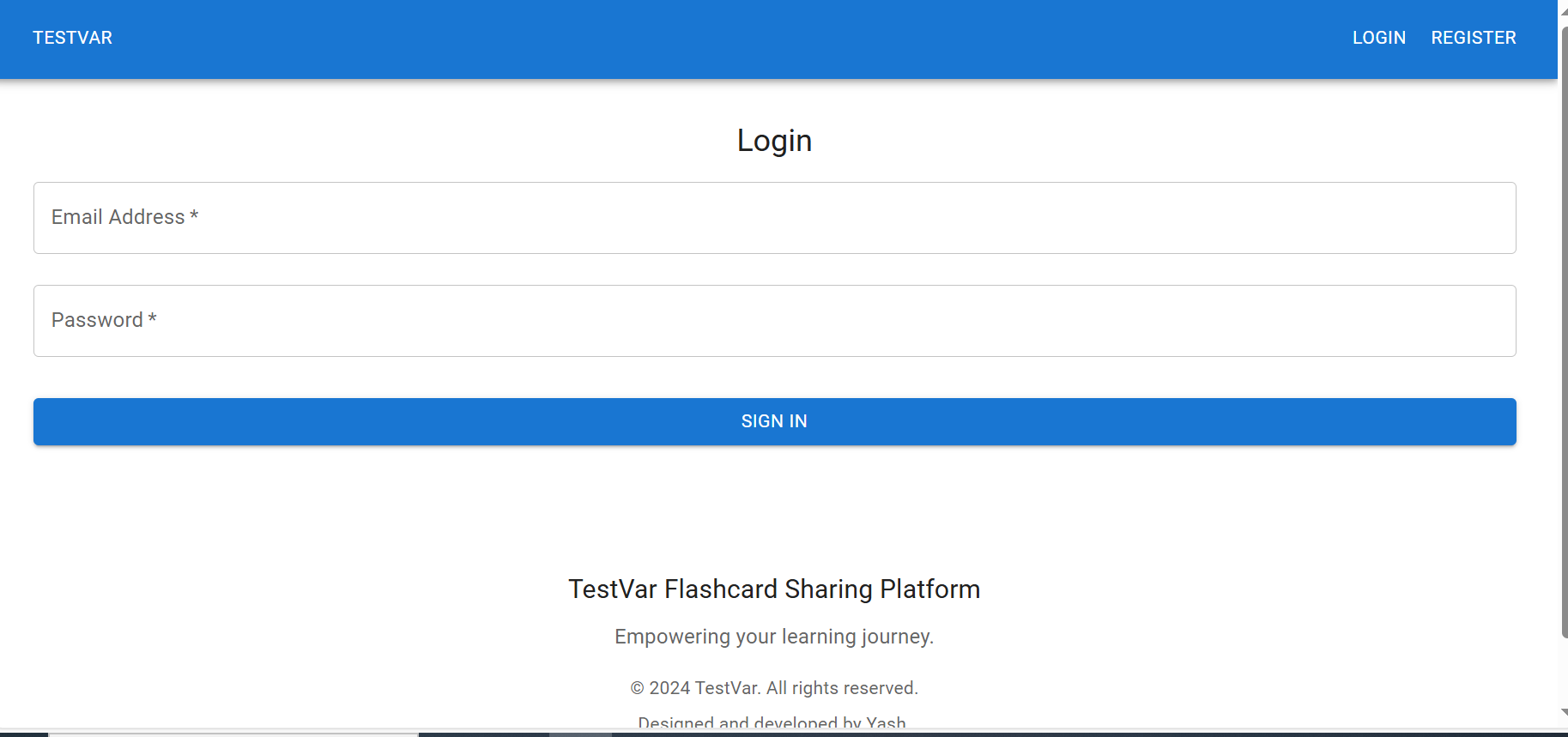
1. Home page



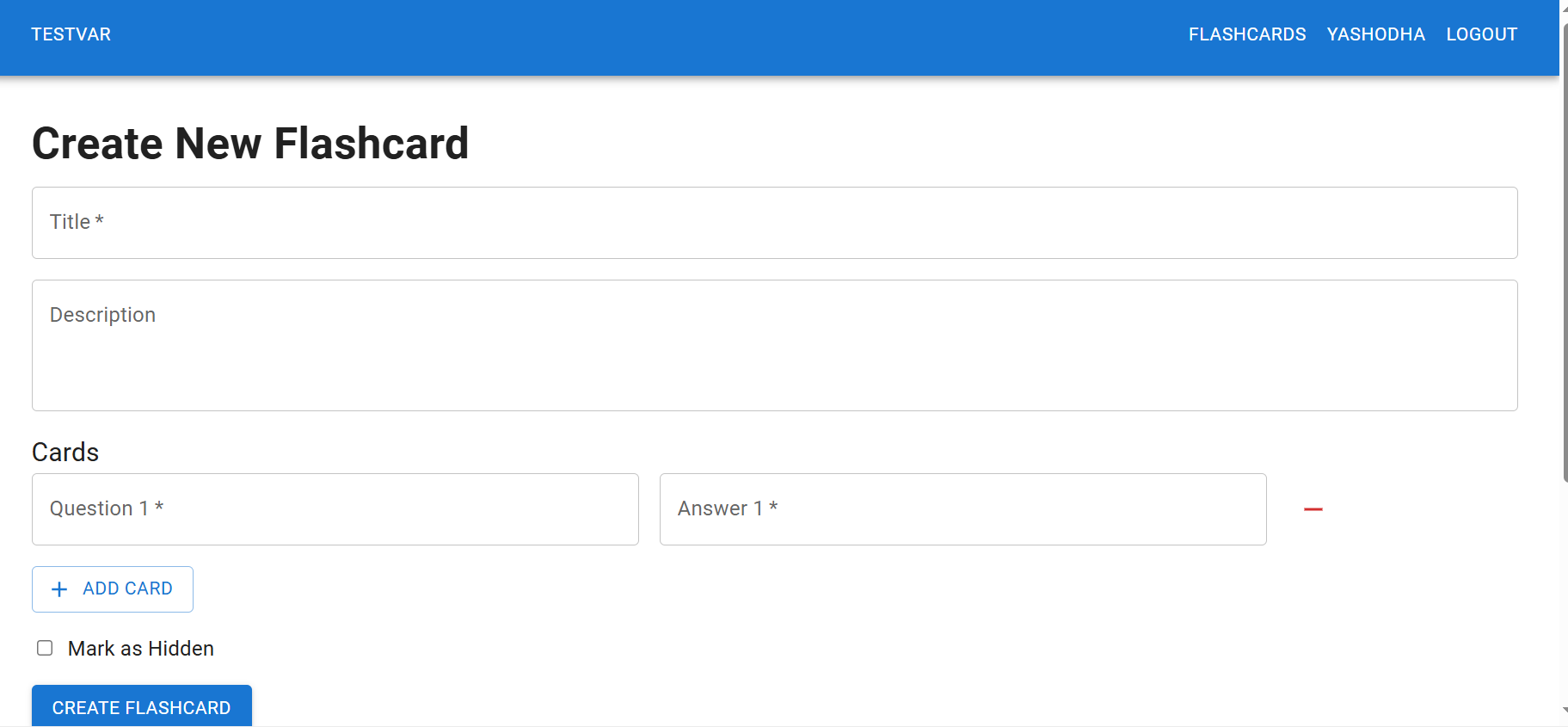
1. Register Page



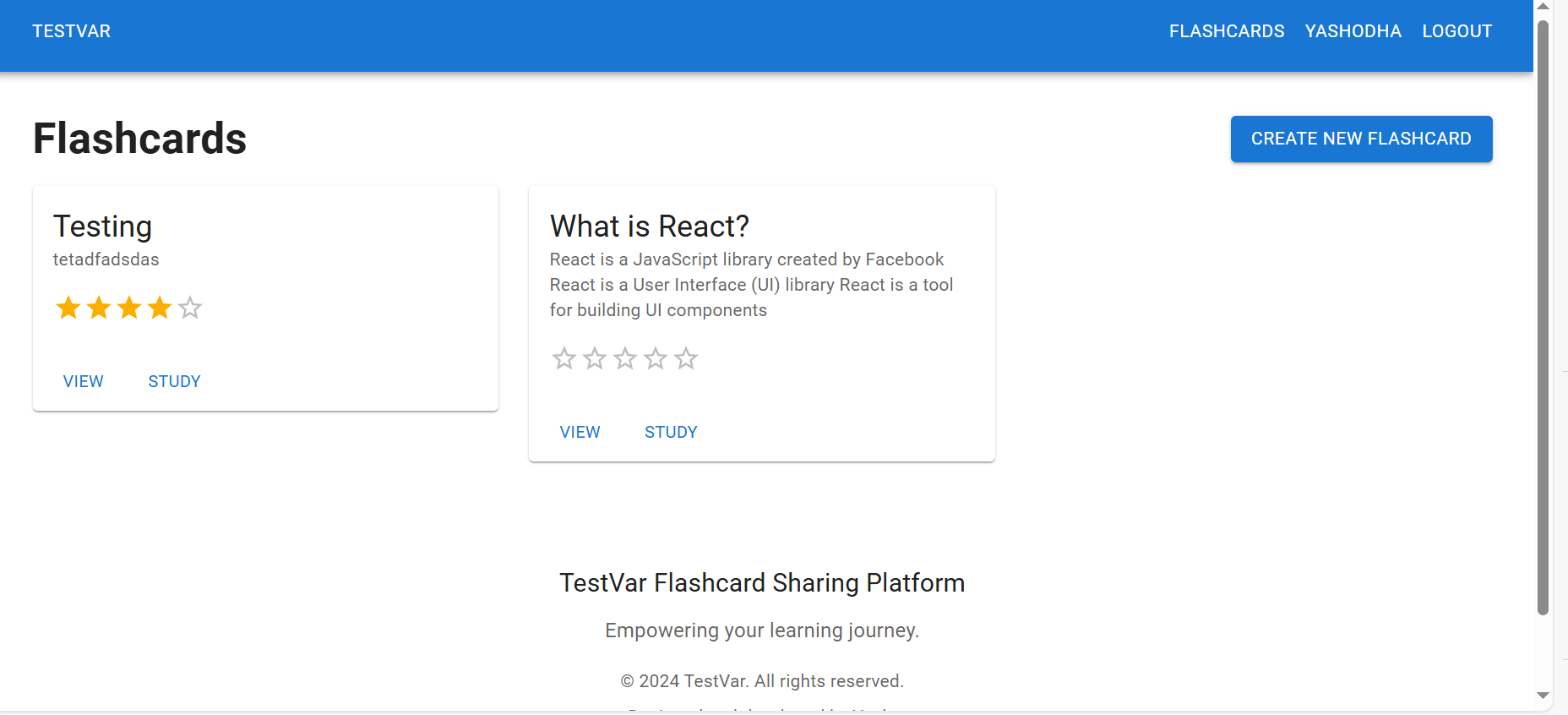
1. Login Page

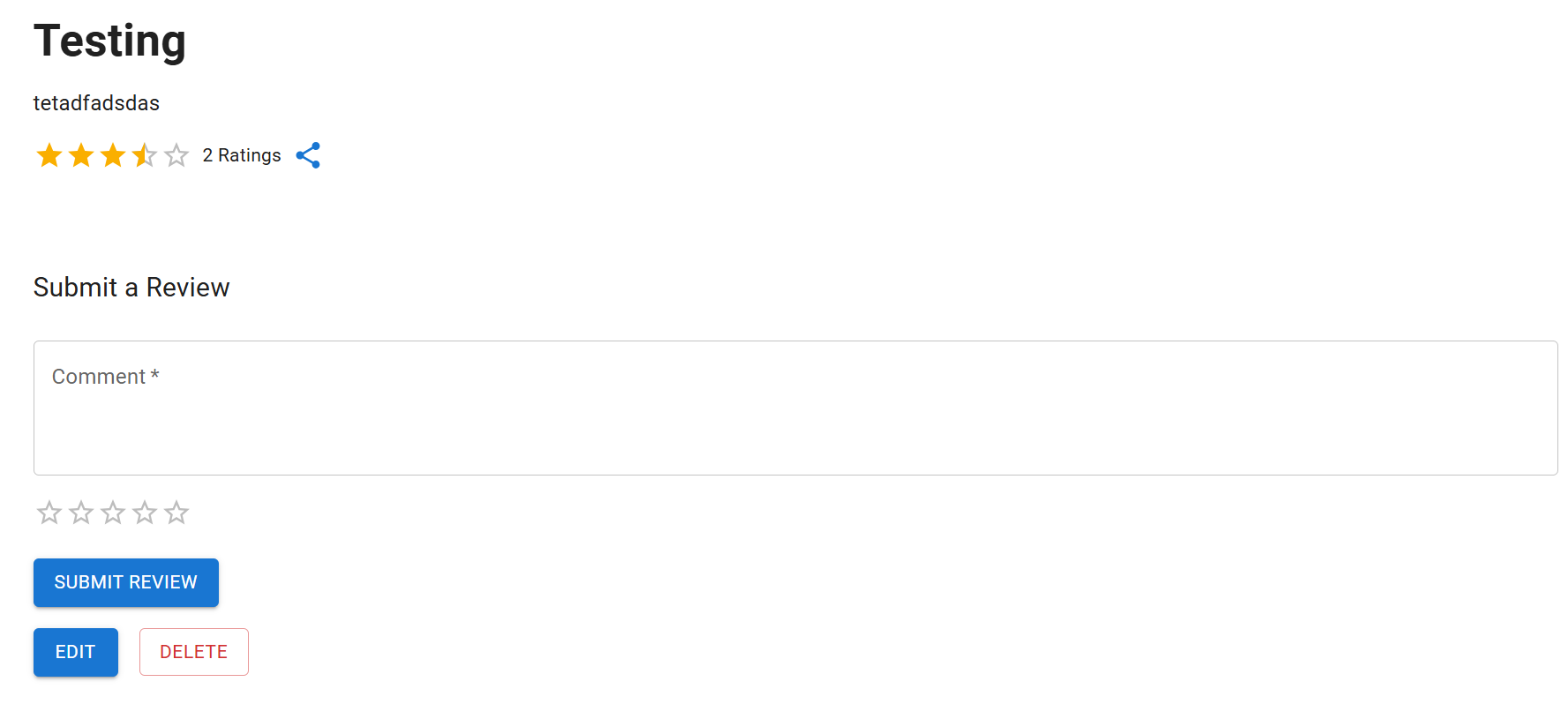


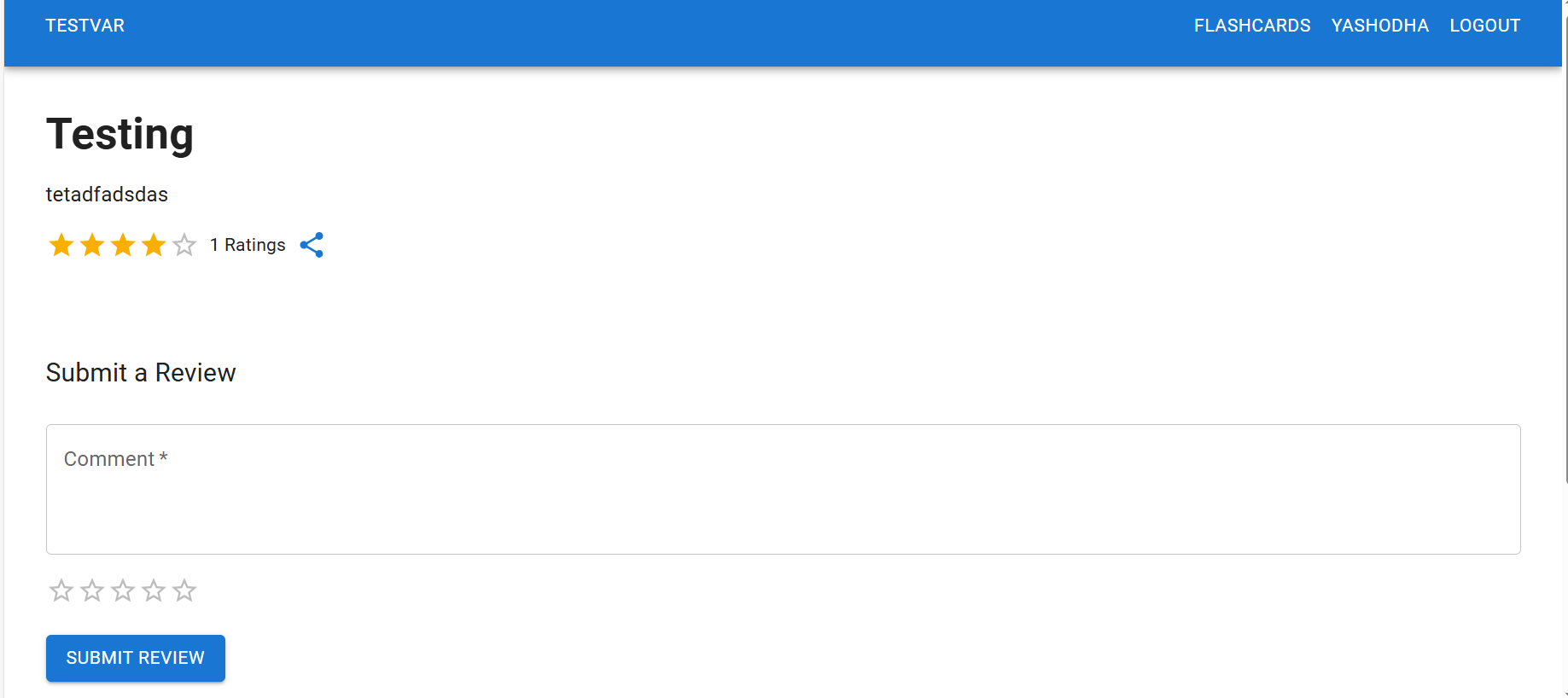
1. Create FlashCards



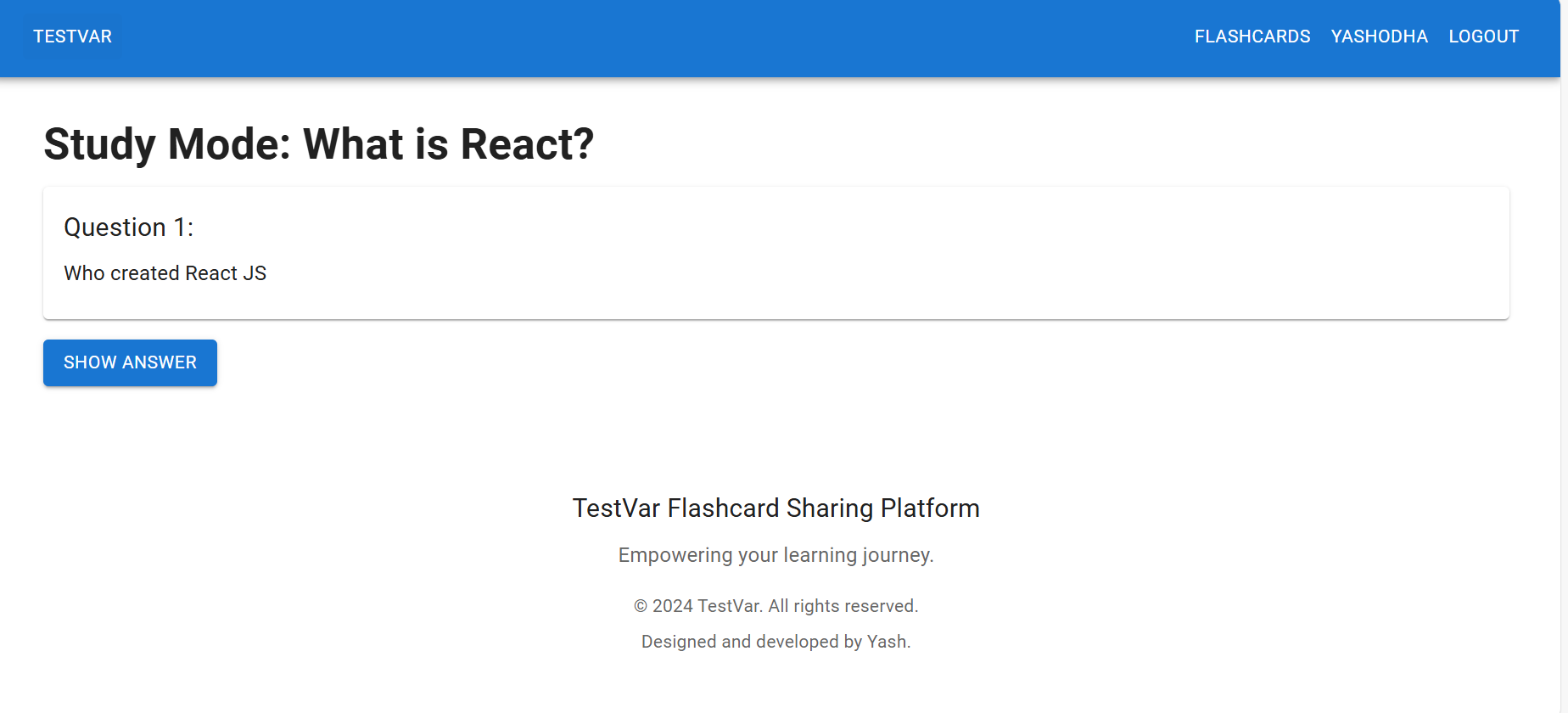
1. FlashCard View



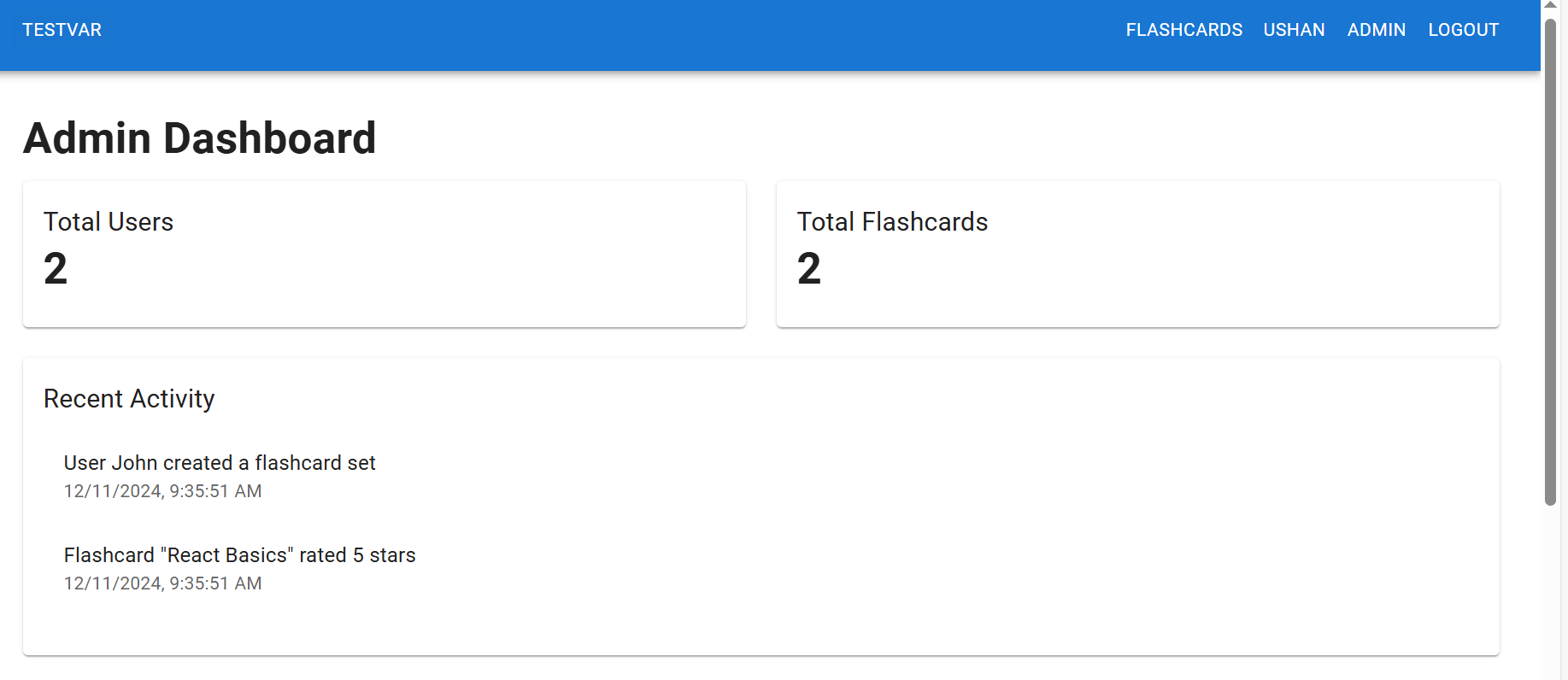




1. Study Mode



1. Admin Dashboard



# References

*Index | Node.js v23.4.0 Documentation*. (n.d.). https://nodejs.org/docs/latest/api/

*Quick Start – React*. (n.d.). https://react.dev/learn

*MongoDB Documentation*. (n.d.). MongoDB Documentation. https://www.mongodb.com/docs/