Memories Mirror React Application

A Project Report Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

Computer Science and Engineering

By

Kshitij Gupta

(181500337)

Under the Guidance of

Mr. Pankaj Kapoor

Department of Computer Engineering and Applications

Institute of Engineering and Technology

GLA University

Mathura- 281406, India

Dec, 2020

Declaration

I hereby declare that the work which is being presented in the FullStack Project "**Memories Mirror**", in fulfillment of the requirements for FullStack project in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of our own work carried under the supervision of **Mr. Pankaj Kapoor (Technical Trainer).**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Kshitij

Sign
Name of Candidate: Kshitij Gupta
University Roll No.: 181500337

Certificate

This is to certify that the above statements made by the candidate are correct to the best of my/our knowledge and belief.

_____

Supervisor
Mr. Pankaj Kapoor
Technical Trainer

Date : 10 May, 2020

ACKNOWLEDGEMENT

It gives me a great sense of pleasure to present the report of the B. Tech FullStack Project undertaken during B. Tech. Third Year. This project in itself is an acknowledgement to the inspiration, drive and technical assistance contributed to it by many individuals. This project would never have seen the light of the day without the help and guidance that we have received.

Our heartiest thanks to Dr. (Prof). Anand Singh Jalal, Head of Dept., Department of CEA for providing me with an encouraging platform to develop this project, which thus helped me in shaping our abilities towards a constructive goal.

I owe a special debt of gratitude to Mr. Pankaj Kapoor, Technical Trainer, for his constant support and guidance throughout the course of my work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. He has showered me with all his extensively experienced ideas and insightful comments at virtually all stages of the project & has also taught me about the latest industry-oriented technologies.

I also do not like to miss the opportunity to acknowledge the contribution of all instructors who are available on YouTube and Stack overflow. I would like to thank all my friends who helped me in making this project.

Last but not the least; I would like to express our deep sense of gratitude and earnest thanks giving to our dear parents for their moral support and heartfelt cooperation during the project.

ABSTRACT

I have used React that is the best library for UI development for the frontend part and NodeJS, ExpressJs for the backend part. For database, I am going to use MongoDB that is document-oriented database and stores data in BSON (JSON) format.

I have built a web application pertaining where users can save their memories of their trips. My website will be built using web languages and frameworks. It will provide an interface to the users that provide them to easily save their precious moments in card format.

In this application, every user can see other user's memories that allow other users to know more about different places and get correct feedbacks about these places. Users can also like each other posts and also able to see the date of creation.

This application that I have built using front-end as well as back-end web technologies and I have provided a user-friendly interface so that users get never stuck.

CONTENTS

1. Introduction

1.1 Overview and Motivation

I am going to build a web application pertaining where users can save their memories of their trips. My website will be built using web languages and frameworks. It will provide an interface to the users that provide them to easily save their precious moments in card format.

In this application, every user can see other users memories that allows other users to know more about different places and get correct feedbacks about these places. Users can also like each other posts and also able to see the date of creation. Every user can delete and update their posts.

I am going to use React that is best library for UI development for the frontend part and NodeJS, ExpressJs for the backend part. For database, I am going to use MongoDB that is document-oriented database and stores data in BSON (JSON) format.

Problem Statement:

There are many websites on the internet which provide the users to save their memories in different forms on all over the internet but my idea is to create a web application through which users can share their thought about places freely that will help others users to know about different places more precisely.

Objective:

The aim of my project is to target thousands of users to use and engage with my web application. I have decided to work on this project because most of the people do not get correct feedback about the places they want to visit but through this application they will get correct feedback about the places by the users who will share their memories in this application.

2. Software and Requirement Analysis

2.1 Software Requirement

Visual Studio:

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug- ins that enhance the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new tool sets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic

.NET, C#, F#,JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python,Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past.

Web Browser:

A **web browser** (commonly referred to as a **browser**) is a software application for accessing information on the World Wide Web. Each individual web page, image, and video is identified by a distinct Uniform Resource Locator (URL), enabling browsers to retrieve these resources from a web server and display them on the user's device.

A web browser is not the same thing as a search engine, though the two are often confused. For a user, a search engine is just a website, such as google.com, that stores searchable data about other websites. But to connect to a website's server and display its web pages, a user needs to have a web browser installed on their device.

The most popular browsers are Chrome, Firefox, Safari, Internet Explorer, and Edge.

2.2 Language and Framework Requirements

HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web.

Web Browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web
page semantically and originally included cues for the appearance of the document.

HTML Elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <img/> and <input/>directly introduce content into the page. Other tags such as
<p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. Without HTML, a browser would not know how to display text as elements or load images or other elements. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance. One could think of HTML as the bones (structure) of a web page, and CSS as its skin (appearance).

CSS (Cascading Style Sheets)

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

Bootstrap

Bootstrap is a free and open front-end framework for designing websites and web applications. It contains HTML - and CSS -based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many earlier web frameworks, it concerns itself with front end development only.

Bootstrap is the second most-starred project on GitHub, with more than 129,000 stars. Bootstrap comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.

Java Script (JS)

JavaScript, often abbreviated as **JS**, is a high-level, interpreted programming language that conforms to the ECMAScript specification. It is a programming language that is characterized as dynamic, weakly typed, prototype-based and multi-paradigm.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it, and major web browsers have a dedicated JavaScript engine to execute it.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has APIs for working with text, arrays, dates, regular expressions, and the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities. It relies upon the host environment in which it is embedded to provide these features.

Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

The terms *Vanilla JavaScript* and *Vanilla JS* refer to JavaScript not extended by any frameworks or additional libraries. Scripts written in Vanilla JS are plain JavaScript code.

Although there are similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design. JavaScript was influenced by programming languages such as Self and Scheme.

jQuery

jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax.It is free, open-source software using the permissive MIT License. As of May 2019, jQuery is used by 73% of the 10 million most popular websites.[5] Web analysis indicates that it is the most widely deployed JavaScript library by a large margin, having at least 3 to 4 times more usage than any other JavaScript library.

jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, themeable widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

The set of jQuery core features—DOM element selections, traversal and manipulation—enabled by its *selector engine* (named "Sizzle" from v1.3), created a new "programming style", fusing algorithms and DOM data structures. This style influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo, later stimulating the creation of the standard *Selectors API*. Later, this style has been enhanced with a deeper algorithm-data fusion in an heir of jQuery, the D3.js framework.

Microsoft and Nokia bundle jQuery on their platforms. Microsoft includes it with Visual Studio for use within Microsoft's ASP.NET AJAX and ASP.NET MVC frameworks while Nokia has integrated it into the Web Run-Time widget development platform.

React

React (also known as React.js or ReactJs) is an open-source, front end, JavaScript library for building user interfaces or UI components. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications. However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

MongoDB

It is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL). MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in MongoDB.

2.3 Software And Hardware requirement:

Following are the hardware and the software requirements for our project:

1. Hardware:

- Laptop/Desktop
- 1.8 GHz or faster processor. Quad-core or better recommended
- 4 GB of RAM and core i3 processor
- Hard disk space: Minimum of 500MB

2. Software:

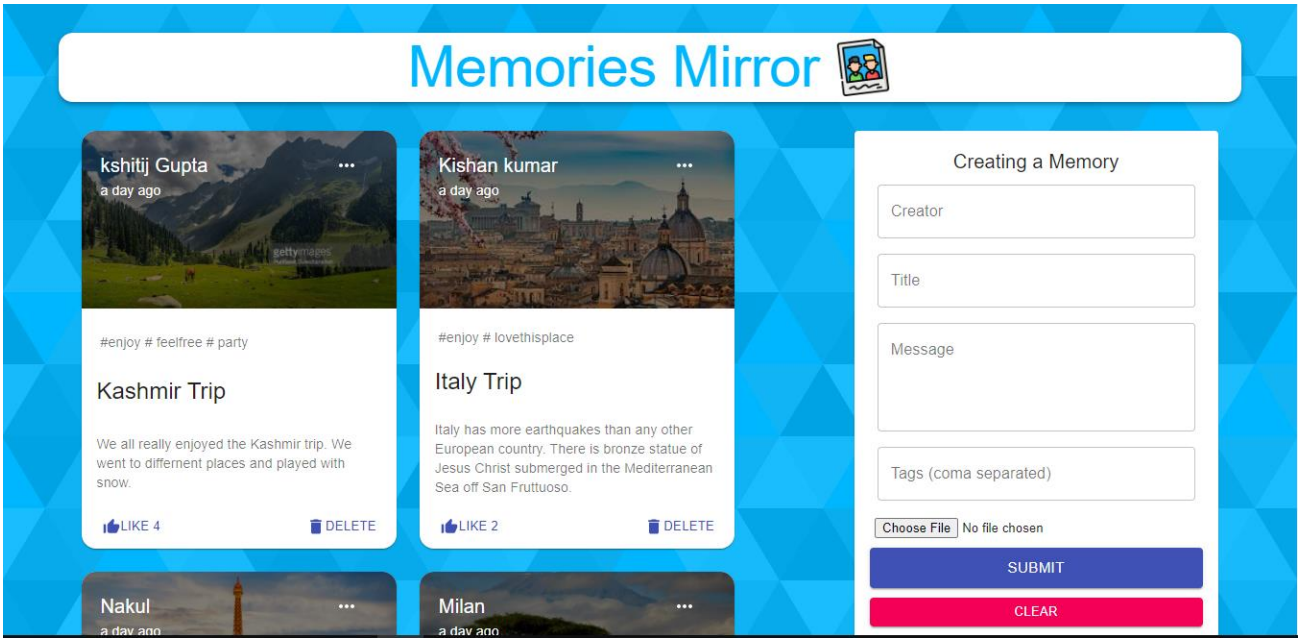
- Windows 8.1 and above
- Visual Studio Code
- Web Browser
- GitHUB Desktop

3. Language and Framework Requirements:

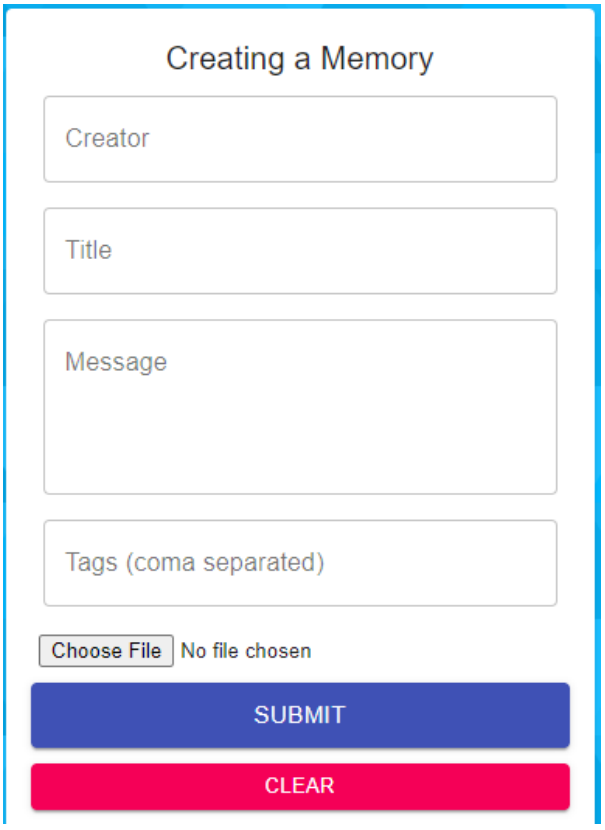
- HTML
- CSS
- Bootstrap
- JavaScript
- ReactJs
- MongoDB
- ExpressJs
- NodeJS

Some Screenshots

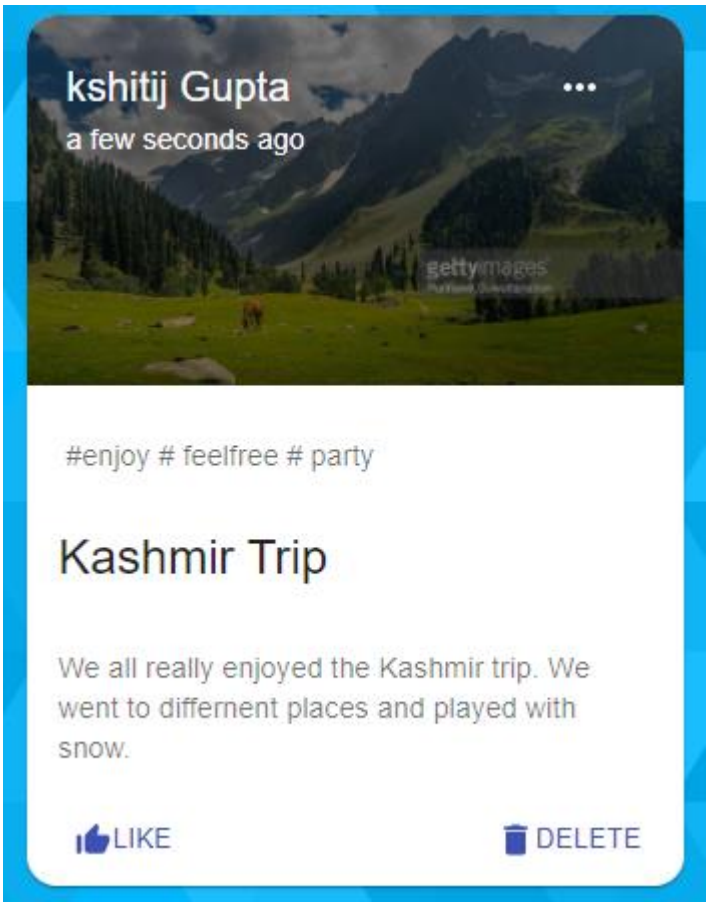
Homepage:



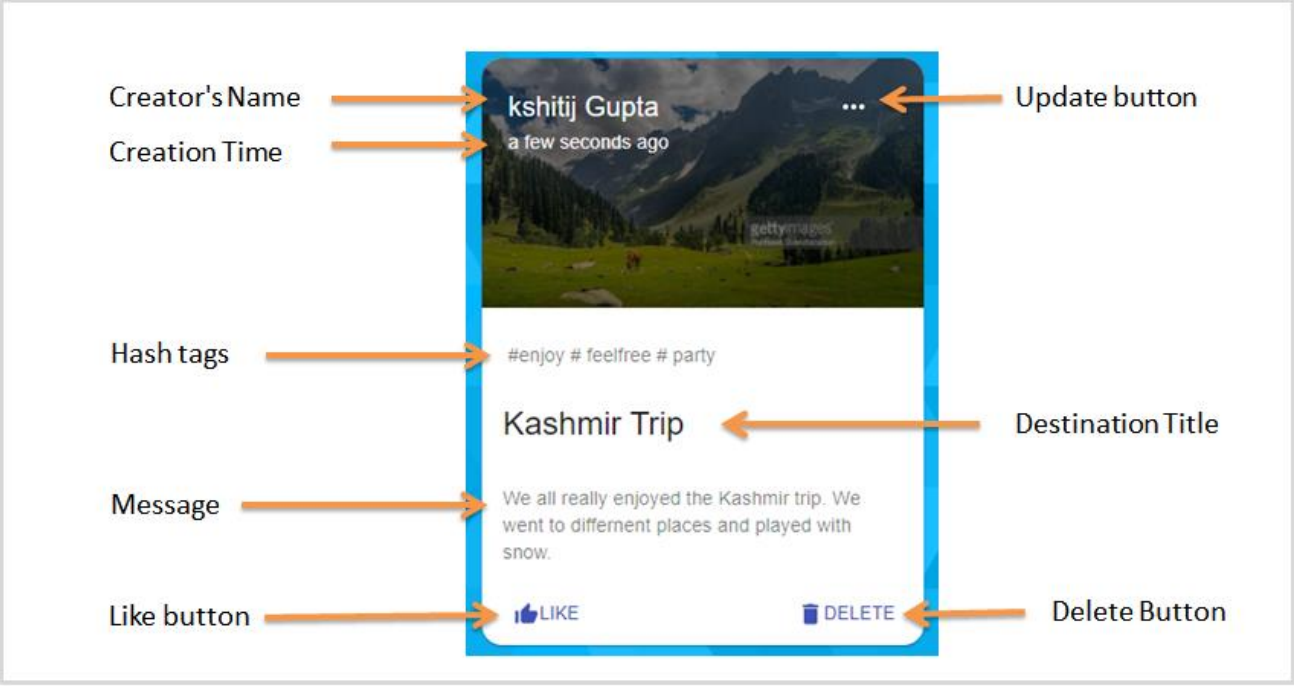
Form to create a memory:



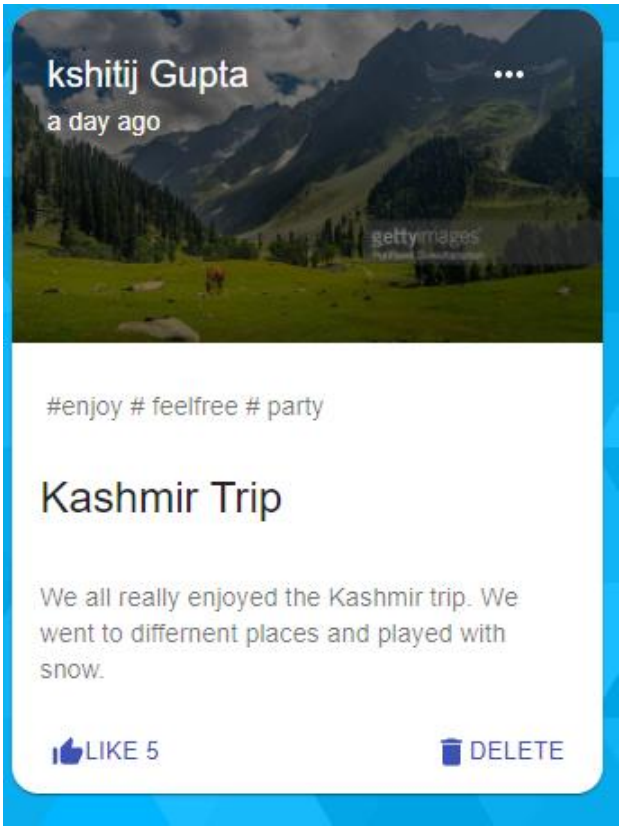
Memory Card Design:



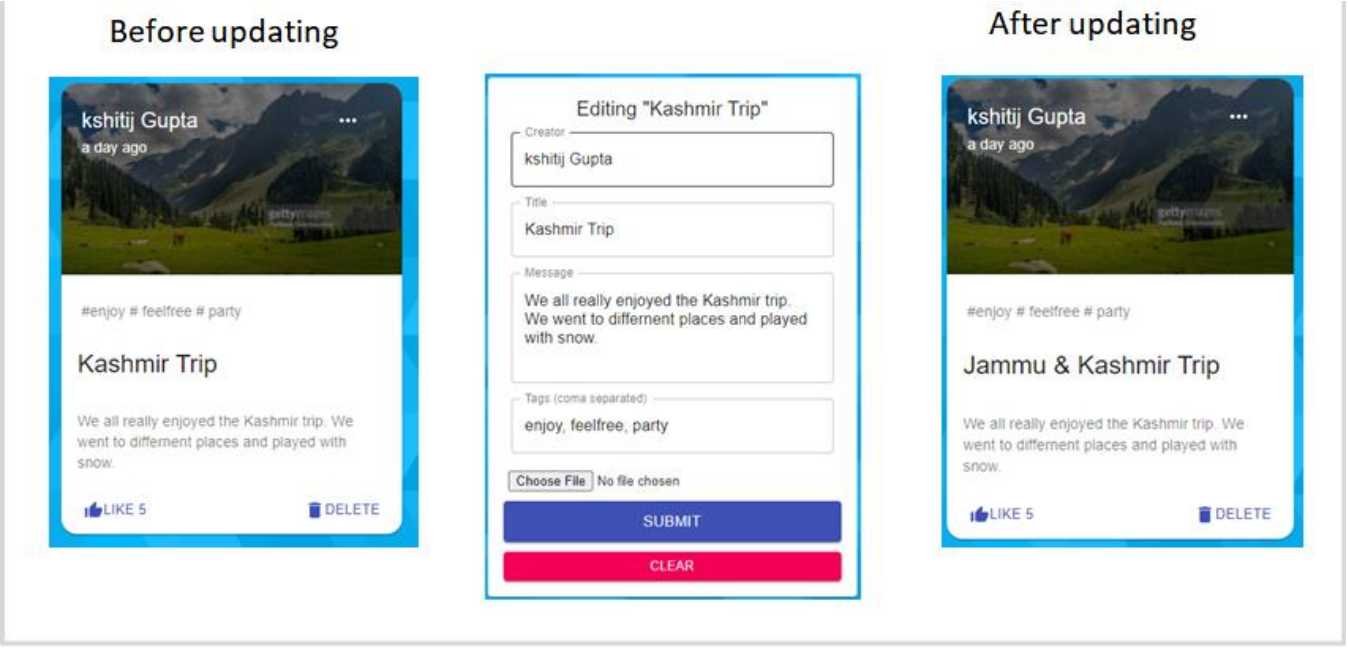
Features of a memory card:



Memory card when user likes it:



When user clicks on update button:



Data is saving in MongoDB:

```
db.getCollection('postmessages').find({})
```

postmessages 0.012 sec. 0 50

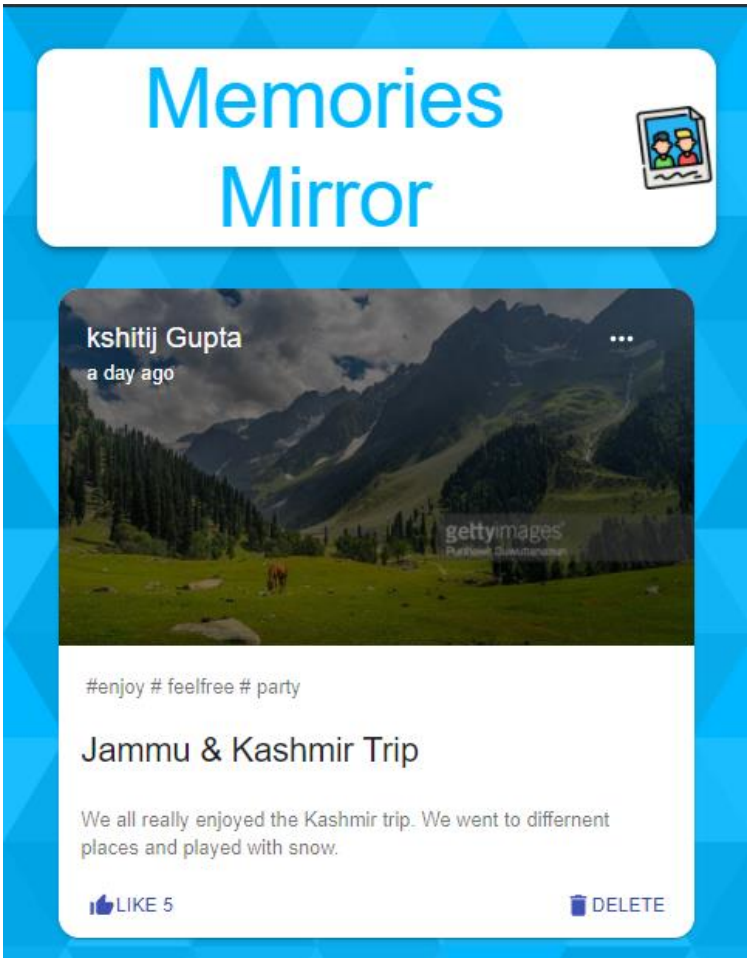
Key	Value	Type
(1) ObjectId("6096a32842798635201216a1")	{ 9 fields }	Object
_id	ObjectId("6096a32842798635201216a1")	ObjectId
tags	[3 elements]	Array
likeCount	5	Int32
createdAt	2021-05-08 12:37:17.617Z	Date
title	Kashmir Trip	String
message	We all really enjoyed the Kashmir trip. We went to differne...	String
selectedFile	...	String
creator	kshitij Gupta	String
__v	0	Int32
(2) ObjectId("6096a40642798635201216a2")	{ 9 fields }	Object
(3) ObjectId("6096a48542798635201216a3")	{ 9 fields }	Object
(4) ObjectId("6096a4fa42798635201216a4")	{ 9 fields }	Object

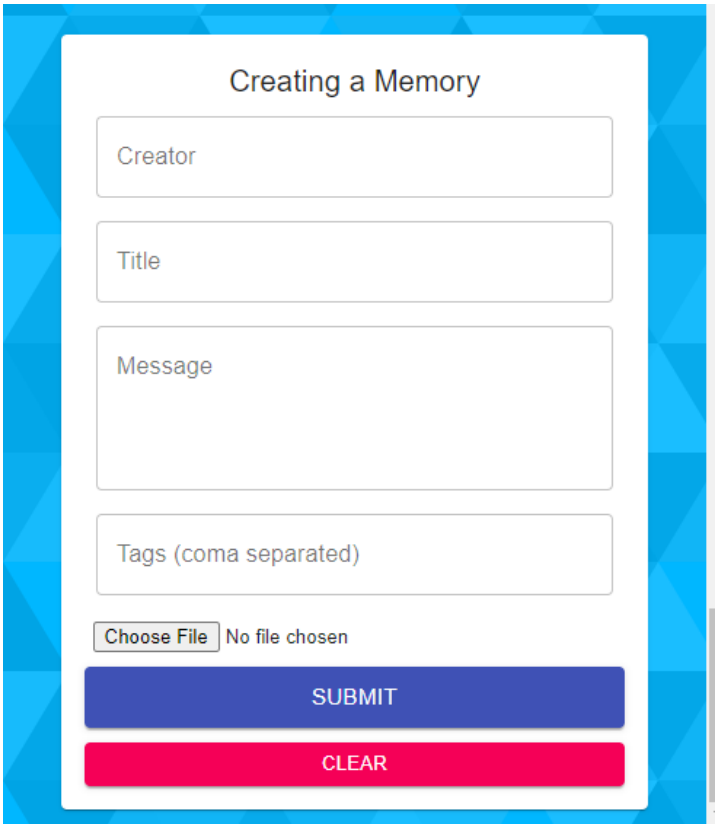
```
db.getCollection('postmessages').find({})
```

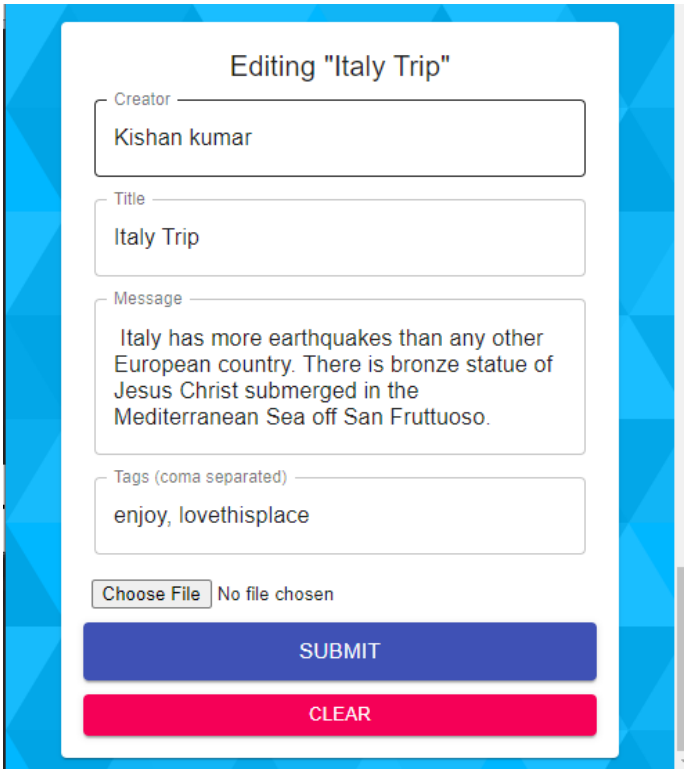
postmessages 0.023 sec. 0 50

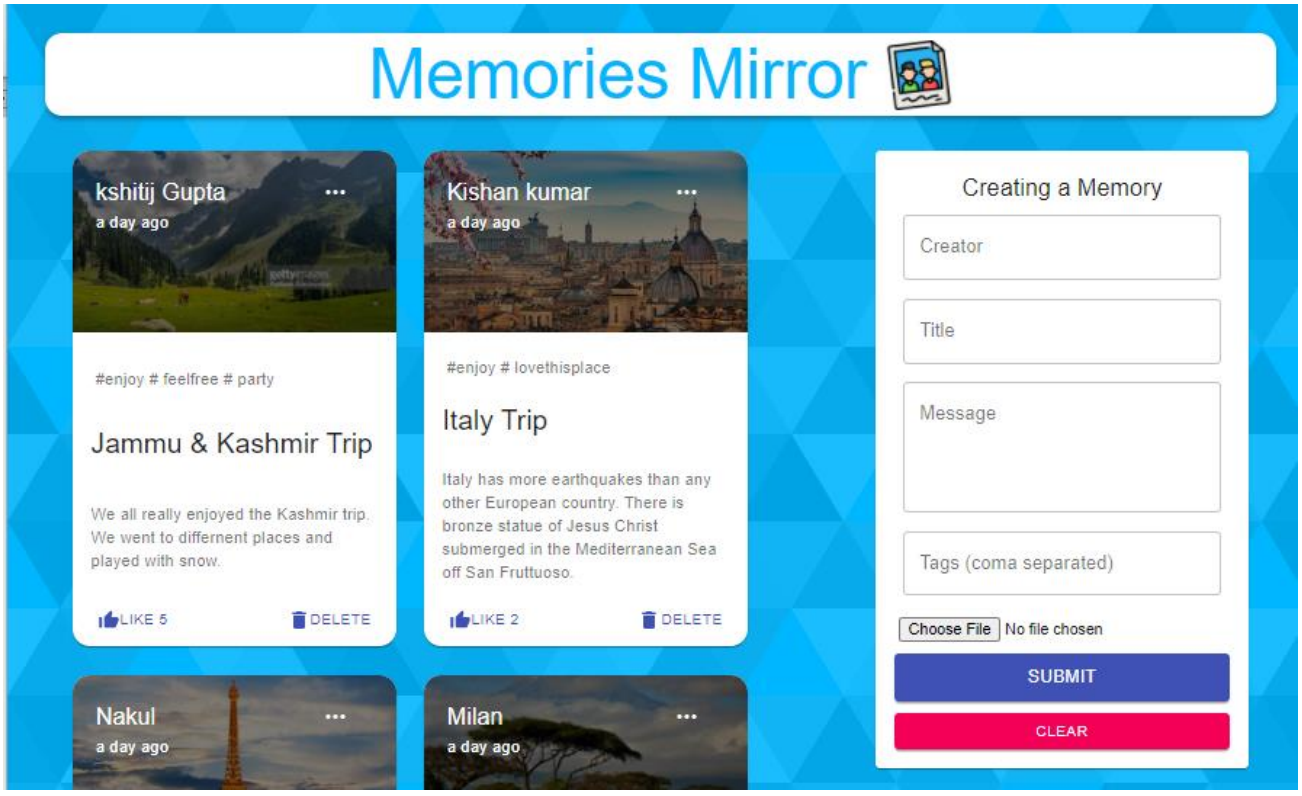
Key	Value	Type
_id	ObjectId("6096a40642798635201216a2")	ObjectId
tags	[2 elements]	Array
likeCount	2	Int32
createdAt	2021-05-08 12:37:17.617Z	Date
title	Italy Trip	String
message	Italy has more earthquakes than any other European cou...	String
selectedFile	...	String
creator	Kishan kumar	String
__v	0	Int32
(3) ObjectId("6096a48542798635201216a3")	{ 9 fields }	Object
_id	ObjectId("6096a48542798635201216a3")	ObjectId
tags	[2 elements]	Array
likeCount	2	Int32
createdAt	2021-05-08 12:37:17.617Z	Date
title	Paris	String
message	Paris (nicknamed the City of light) is the capital city of Fr...	String
selectedFile	...	String
creator	Nakul	String
__v	0	Int32
(4) ObjectId("6096a4fa42798635201216a4")	{ 9 fields }	Object

Responsive Nature






Editing "Italy Trip"
Creator
Kishan kumar
Title
Italy Trip
Message
Italy has more earthquakes than any other European country. There is bronze statue of Jesus Christ submerged in the Mediterranean Sea off San Fruttuoso.
Tags (coma separated)
enjoy, lovethisplace
Choose File
No file chosen
SUBMIT
CLEAR


Memories Mirror
kshitij Gupta
a day ago
#enjoy # feelfree # party
Jammu & Kashmir Trip
We all really enjoyed the Kashmir trip. We went to different places and played with snow.
LIKE 5
DELETE
Kishan kumar
a day ago
#enjoy # lovethisplace
Italy Trip
Italy has more earthquakes than any other European country. There is bronze statue of Jesus Christ submerged in the Mediterranean Sea off San Fruttuoso.
LIKE 2
DELETE
Nakul
a day ago
Milan
a day ago
Creating a Memory
Creator
Title
Message
Tags (coma separated)
Choose File
No file chosen
SUBMIT
CLEAR

Conclusion:

It will be a wonderful and learning experience for me while working on this project. I decided to work on this project because I want to create a app where people from all over the place can share their precious memories with others.

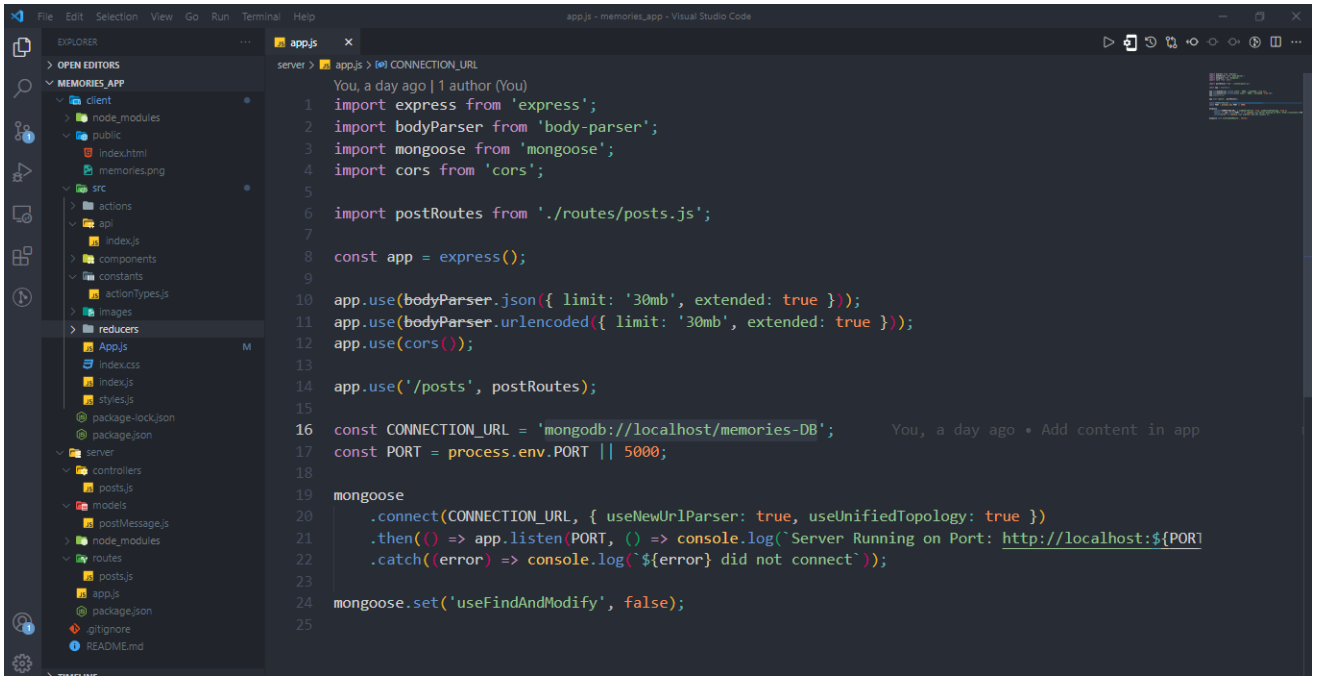
I am trying to make a good interactive User Interface (UI) so that the user will not face any difficulties while using. The joy of work and thrill involved while tackling the various problems and challenges will give me the feel of the developer industry.

Online Git Repository

https://github.com/kshitijgupta468035/memories_app

Appendices

Backend



postMessage.js - memories_app - Visual Studio Code

server > models > postMessage.js > ...

```python
import mongoose from 'mongoose';

const postSchema = mongoose.Schema({
    title: String,
    message: String,
    creator: String,
    tags: [String],
    selectedFile: String,
    likeCount: {
        type: Number,
        default: 0,
    },
    createdAt: {
        type: Date,
        default: new Date(),
    },
});

var PostMessage = mongoose.model('PostMessage', postSchema);

export default PostMessage;
```

posts.js - memories_app - Visual Studio Code

server > routes > posts.js > ...

```python
import express from 'express';

import { getPosts, getPost, createPost, updatePost, likePost, deletePost } from '../controllers/pos

const router = express.Router();

router.get('/', getPosts);
router.post('/', createPost);
router.get('/:id', getPost);
router.patch('/:id', updatePost);
router.delete('/:id', deletePost);
router.patch('/:id/likePost', likePost);

export default router;
```

```
    const { id } = req.params;

    if (!mongoose.Types.ObjectId.isValid(id)) return res.status(404).send(`No post with id: ${id}`);

    const post = await PostMessage.findById(id);

    const updatedPost = await PostMessage.findByIdAndUpdate(id, { likeCount: post.likeCount + 1 },

    res.json(updatedPost);
};
      You, a day ago • Add likePost Method in posts.js
export default router;
```

Frontend

```
You, seconds ago | 1 author (You)
import React, { useState, useEffect } from 'react';
import { Container, AppBar, Typography, Grow, Grid } from '@material-ui/core';
import { useDispatch } from 'react-redux';

import Posts from './components/Posts/Posts';
import Form from './components/Form/Form';
import { getPosts } from './actions/posts';
import useStyles from './styles';
import memories from './images/memories.png';

Complexity is 14 You must be kidding
const App = () => {
    const [currentId, setCurrentId] = useState(0);
    const dispatch = useDispatch();
    const classes = useStyles();

    useEffect(() => {
        dispatch(getPosts());
    }, [currentId, dispatch]);
      You, a day ago • Add code in component App.js
    return (
        <Container maxWidth="lg">
            <AppBar className={classes.appBar} position="static" color="inherit">
                <Typography className={classes.heading} variant="h2" align="center">
                    Memories Mirror
                </Typography>
                <img className={classes.image} src={memories} alt="icon" height="60" />
```

```
import React, { useState, useEffect } from 'react';
import { TextField, Button, Typography, Paper } from '@material-ui/core';
import { useDispatch, useSelector } from 'react-redux';
import FileBase from 'react-file-base64';

import useStyles from './styles';
import { createPost, updatePost } from '../../actions/posts';



const Form = ({ currentId, setCurrentId }) => {
    const [postData, setPostData] = useState({ creator: '', title: '', message: '', tags: '', selec

    const post = useSelector((state) => (currentId ? state.posts.find((message) => message._id ===
    const dispatch = useDispatch();
    const classes = useStyles();

    useEffect(() => {
        if (post) setPostData(post);
    }, [post]);

    const clear = () => {
        setCurrentId(0);
        setPostData({ creator: '', title: '', message: '', tags: '', selectedFile: '' });
    };
```

```
const Post = ({ post, setCurrentId }) => {
  const dispatch = useDispatch();
  const classes = useStyles();

  return (
    <Card className={classes.card}>
      <CardMedia className={classes.media} image={post.selectedFile || 'https://user-images.github
      <div className={classes.overlay}>
        <Typography variant="h6">{post.creator}</Typography>
        <Typography variant="body2">{moment(post.createdAt).fromNow()}</Typography>
      </div>
      <div className={classes.overlay2}>
        <Button style={{ color: 'white' }} size="small" onClick={() => setCurrentId(post._id)}><Mor
      </div>
      <div className={classes.details}>
        <Typography variant="body2" color="textSecondary" component="h2">{post.tags.map((tag) => `#
      </div>
      <Typography className={classes.title} gutterBottom variant="h5" component="h2">{post.title}</
      <CardContent>
        <Typography variant="body2" color="textSecondary" component="p">{post.message}</Typography>
      </CardContent>
      <CardActions className={classes.cardActions}>
        <Button size="small" color="primary" onClick={() => dispatch(likePost(post._id))}><ThumbUp/
        <Button size="small" color="primary" onClick={() => dispatch(deletePost(post._id))}><Delete
      </CardActions>
    </Card>
  );
};
```

```
import axios from 'axios';

const url = 'http://localhost:5000/posts';

export const fetchPosts = () => axios.get(url);
export const createPost = (newPost) => axios.post(url, newPost);
export const likePost = (id) => axios.patch(`${url}/${id}/likePost`);
export const updatePost = (id, updatedPost) => axios.patch(`${url}/${id}`, updatedPost);
export const deletePost = (id) => axios.delete(`${url}/${id}`);
```

```
import { makeStyles } from '@material-ui/core/styles';

export default makeStyles(() => ({
    appBar: {
        borderRadius: 15,
        margin: '30px 0',
        display: 'flex',
        flexDirection: 'row',
        justifyContent: 'center',
        alignItems: 'center',
    },
    heading: {
        color: 'rgba(0,183,255, 1)',
    },
    image: {
        marginLeft: '15px',
    },
}));
```

styles.js - memories_app - Visual Studio Code

client > src > components > Form > styles.js > ...

```javascript
import { makeStyles } from '@material-ui/core/styles';

export default makeStyles((theme) => ({
    root: {
        '& .MuiTextField-root': {
            margin: theme.spacing(1),
        },
    },
    paper: {
        padding: theme.spacing(2),
    },
    form: {
        display: 'flex',
        flexWrap: 'wrap',
        justifyContent: 'center',
    },
    fileInput: {
        width: '97%',
        margin: '10px 0',
    },
    buttonSubmit: {
        marginBottom: 10,
    },
}));
```

index.js - memories_app - Visual Studio Code

client > src > reducers > index.js > ...

You, a day ago | 1 author (You)

```javascript
import { combineReducers } from 'redux';

import posts from './posts';

export const reducers = combineReducers({ posts });
```

posts.js - memories_app - Visual Studio Code

client > src > reducers > posts.js > ...

You, a day ago | 1 author (You)

```
import { FETCH_ALL, CREATE, UPDATE, DELETE, LIKE } from '../constants/actionTypes';


export default (posts = [], action) => {
    switch (action.type) {
        case FETCH_ALL:
            return action.payload;
        case LIKE:
            return posts.map((post) => (post._id === action.payload._id ? action.payload : post));
        case CREATE:
            return [...posts, action.payload];
        case UPDATE:
            return posts.map((post) => (post._id === action.payload._id ? action.payload : post));
        case DELETE:
            return posts.filter((post) => post._id !== action.payload);
        default:
            return posts;
    }
};
```

Post.js - memories_app - Visual Studio Code

client > src > components > Posts > Post > Post.js > ...

You, a day ago | 1 author (You)

```
import React from 'react';
import { Card, CardActions, CardContent, CardMedia, Button, Typography } from '@material-ui/core/';
import ThumbUpAltIcon from '@material-ui/icons/ThumbUpAlt';
import DeleteIcon from '@material-ui/icons/Delete';
import MoreHorizIcon from '@material-ui/icons/MoreHoriz';
import moment from 'moment';
import { useDispatch } from 'react-redux';

import { likePost, deletePost } from '../../../actions/posts';
import useStyles from './styles';

Complexity is 25 You must be kidding
const Post = ({ post, setCurrentId }) => {
  const dispatch = useDispatch();
  const classes = useStyles();

  return (
    <Card className={classes.card}>
      <CardMedia className={classes.media} image={post.selectedFile || 'https://user-images.githubu
      <div className={classes.overlay}>
        <Typography variant="h6">{post.creator}</Typography>
        <Typography variant="body2">{moment(post.createdAt).fromNow()}</Typography>
      </div>
      <div className={classes.overlay2}>
        <Button style={{ color: 'white' }} size="small" onClick={() => setCurrentId(post._id)}><Mor
      </div>
      <div className={classes.details}>
```

References

Beta Labs

https://www.youtube.com/

DevDocs API Documentation

W3Schools Online Web Tutorials

Stack Overflow - Where Developers Learn, Share, & Build Careers

We have used these resources as a reference to build our project. These all resources are good e-learning platforms and give us a lot of information about the applications we are going to make on our website.

Signature of Project Guide: