**VELAGAPUDI RAMAKRISHNA SIDDHARTHA ENGINEERING COLLEGE**

**(AUTONOMOUS - AFFILIATED TO JNTU-K, KAKINADA)**

**Approved by AICTE & Accredited by NBA**

**KANURU, VIJAYAWADA-7**

**ACADEMIC YEAR**

**(2022-2023)**



#### DEVELOPING A BLOGGING WEBSITE USING REACT

#### Project

**Submitted to: Submitted by:** BLN.PHANEENDRAKUMAR Ch.Deepthi(208W1A1209)

IT Department G.Aashritha(208W1A1215)

VRSEC. N.Sushma(208W1A1235)

K.Prabhu Ram(208W1A1288)

**ACKNOWLEDGEMENT**

I would want to thank V.R. Siddhartha Engineering College in the first place for providing me with the opportunity to complete my project. For his encouragement and support during the course of my endeavor, I am grateful to our principle, **Dr. A.V.RATNA PRASAD**.

I would like to express my gratitude to **Dr.M. Suneetha**, Head of the Department of IT, for her unwavering inspiration and support throughout the course of my work as I submit this project report.

In order to successfully complete this project, I would want to convey my sincere gratitude to my project guide, assistant professor **BLN.PHANEENDRA KUMAR**.

Additionally, I would like to express my gratitude to all faculty members, departmental support staff, friends, and family members who provided their invaluable assistance and support at crucial junctures without which it would have been extremely challenging for me to complete the project work.

**CERTIFICSTIONS/BADGES** DONE AS PART OF THE **COURSES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ROLL NUMBER** | **COURSE NAME** | **PLATFORM** | **DATE OF COMPLETION** | **SHARABLE LINK** |
| 208W1A1209 | Full-stack web development with react | Coursera | nov 2, 2022 | [208W1A1209 coursera](https://www.coursera.org/account/accomplishments/specialization/5ZU5AMNXJBJY) |
| 208W1A1215 | Full-stack web development with react | Coursera | nov 2, 2022 | [208W1A1215 coursera](https://www.coursera.org/account/accomplishments/specialization/BNRET33DZHPF) |
| 208W1A1235 | Full-stack web development with react | Coursera | nov 2, 2022 | [208W1A1235 coursera](https://www.coursera.org/account/accomplishments/specialization/QSHSF3BD29W9) |
| 208W1A1288 | Full-stack web development with react | Coursera | nov 2, 2022 | [208W1A1288 coursera](https://www.coursera.org/account/accomplishments/specialization/BRMQFPGXJEU9?utm_source=link&utm_medium=certificate&utm_content=cert_image&utm_campaign=sharing_cta&utm_product=s12n) |

**INTRODUCTION**

1. **React introduction with features, versions, applications**

**INTRODUCTION**

For creating reusable UI components, ReactJS is a declarative, effective, and adaptable JavaScript toolkit. It is an open-source front end library that is only in charge of the application's view layer. Jordan Walke, a software engineer at Facebook, developed it. Facebook created and maintained it at first, and then used it in products like WhatsApp and Instagram. ReactJS was created by Facebook in its newsfeed area in 2011, but it wasn't made available to the general public until May 2013.

Most websites are now created utilising the MVC (model view controller) architecture. React is the "V" in the MVC architecture, which stands for "view," whereas Redux or Flux offer the architecture.

**APPLICATION**

A ReactJS application is composed of various components, each of which is in charge of producing a discrete chunk of reusable HTML code. The core of every React application is its component. The ability to nest these components with other components enables the construction of complex applications from basic building blocks. To fill the HTML DOM with data, ReactJS uses a virtual DOM-based technique. The virtual DOM operates quickly because it simply modifies certain DOM elements as opposed to reloading the entire DOM each time.

We write React components that correlate to different elements in order to create React apps. These components are organized within higher level components that specify the application structure. Take a form, for instance, which has numerous components including input fields, labels, and buttons. Each form component may be expressed as a React component, which we then combine to create the form component itself. The form's structure and the pieces that go inside of it would be specified by the form's components.

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | 0.3.0 | 29/05/2013 | Initial Public Release |
| 2. | 0.4.0 | 20/07/2013 | Support for comment nodes <div>{/\* \*/}</div>, Improved server-side rendering APIs, Removed React.autoBind, Support for the key prop, Improvements to forms, Fixed bugs. |
| 3. | 0.5.0 | 20/10/2013 | Improve Memory usage, Support for Selection and Composition events, Support for getInitialState and getDefaultProps in mixins, Added React.version and React.isValidClass, Improved compatibility for Windows. |
| 4. | 0.8.0 | 20/12/2013 | Added support for rows & cols, defer & async, loop for <audio> & <video>, autoCorrect attributes.Added onContextMenu events, Upgraded jstransform and esprima-fb tools, Upgraded browserify. |
| 5. | 0.9.0 | 20/02/2014 | Added support for crossOrigin, download and hrefLang, mediaGroup and muted, sandbox, seamless, and srcDoc, scope attributes, Added any, arrayOf,component, oneOfType, renderable, shape to React.PropTypes, Added support for onMouseOver and onMouseOut event, Added support for onLoad and onError on <img> elements. |

**VERSIONS**

1. **DESCRIPTION OF THE PROJECT UNDERTAKEN**

This is a straightforward React JS app for a blogging platform. The project has the essential attributes of a website for creating blogs.

The react website is bootstrapped with [Create React App](https://github.com/facebook/create-react-app). To set up the project, install all the packages by using yarn or npm install. Then, run the development server with npm run start.

In this project, there are several components to fulfill different functions of the blogging site. Also, **react router** helps in navigation of different pages. Likewise, **context** api assists to pass data through the component tree conveniently.

The UI of the blogging is design by using **Bulma** which is a free, open source CSS framework based on Flexbox. There is navigation bar at top of the page, a title just below it and main body without any footer.

In this blogging site, login system is elementary with an admin and a regular user. The admin can add new products but a normal user can only add products from product list to their cart. Also, users can clear or checkout the added products from the cart. However, the new added product is stored on the local storage of the browser because there is no database in the project. The product has few details like name, description, price and so on.

**SOFTWARE REQUIREMENTS TO THE PROJECT**

**NODEJS**

As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scale-able network applications. This is in contrast to today’s more common concurrency model, in which OS threads are employed. Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node.js are free from worries of dead-locking the process, since there are no locks. Almost no function in Node.js directly performs I/O, so the process, so the process never blocks except when the I/O is performed using synchronous methods of Node.js standards library. Because nothing blocks, scale-able systems are very reasonable to develop in Node.js.

**VISUAL STUDIO CODE**

Visual Studio Code is a code editor in layman’s terms. Visual Studio Code is “a free-editor that helps the programmer write code, helps in debugging and corrects the code using the intelli-sense method ”. In normal terms, it facilitates users to write the code in an easy manner. Many people say that it is half of an IDE and an editor, but the decision is up to to the coders. Any program/software that we see or use works on the code that runs in the background. Traditionally coding was used to do in the traditional editors or even in the basic editors like notepad! These editors used to provide basic support to the coders.

Some of them were so basic that it was very difficult in writing basic English level programs in them. As time went by, some programming languages needed a specific framework and support for further coding and development it, which was not possible using these editors. VI Editor, Sublime Text Editor, is one of the many kinds of editors that came into existence. The most prominent and which supports almost every coding language is VISUAL STUDIO CODE. Its features let the user modify the editor as per the usage, which means the user is able to download the libraries from the internet and integrate it with the code as per his requirements.

**COMMAND PROMPT**

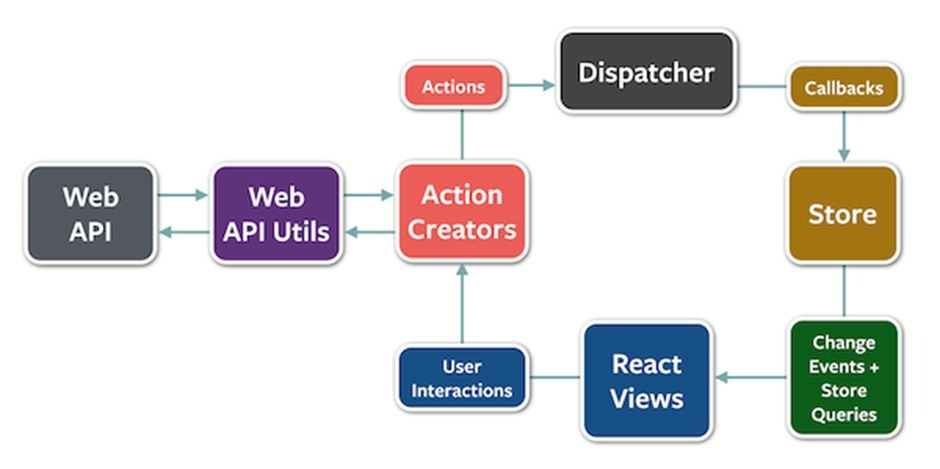
A command prompt is the input field in a text-based [user interface](https://www.techtarget.com/searchapparchitecture/definition/user-interface-UI) screen for an operating system (OS) or program. The prompt is designed to elicit an action. The command prompt consists of a brief text string followed by a blinking [cursor](https://www.techtarget.com/whatis/definition/cursor), which is where the user types command prompt commands.

Command-line interfaces ([CLI](https://www.techtarget.com/searchwindowsserver/definition/command-line-interface-CLI)) and prompts were the standard interface for computers from the early days of computing into the 1980s. Microsoft [**MS-DOS**](https://www.techtarget.com/searchenterprisedesktop/definition/MS-DOS) systems and other early consumer-based computers used CLIs. Current [Windows](https://www.techtarget.com/searchwindowsserver/definition/Windows) systems offer the CLI for administrative tasks. The CLI is still an essential part of the [Linux](https://www.techtarget.com/searchdatacenter/definition/Linux-operating-system) OS.

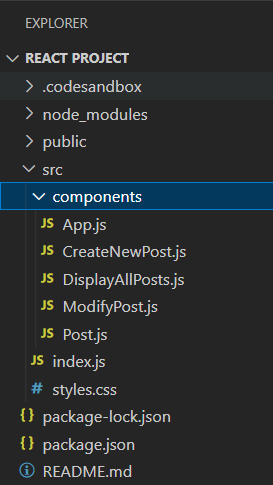
The command prompt is an [executable](https://www.techtarget.com/searchsecurity/definition/executable) CLI program, cmd.exe. At the command prompt, the user types a statement including a base [batch file](https://www.techtarget.com/searchwindowsserver/definition/batch-file) or a command name and any arguments to specify running conditions, logging and so on for the program. In Windows systems, such as Windows 10 and many previous versions of Windows, the command interpreter and executioner are referred to as the [Windows Command Processor](https://answers.microsoft.com/en-us/windows/forum/all/what-is-windows-command-processor/8a2c7d1c-158c-4502-9f8b-fe906ebff30c).

Command prompt interfaces can be powerful and succinct. Some tools that aren't available through the graphical user interface ([GUI](https://www.techtarget.com/whatis/definition/GUI-graphical-user-interface)) can be accessed through the command prompt. It also offers superior automation through [scripting](https://www.techtarget.com/whatis/definition/script), but mastering the commands can be challenging.

**DESIGN AND FLOW DIAGRAM OF THE PROJECT**



**CODING and OUTPUT screenshots:**



**CreateNewPost.js**

import React from "react";

const CreateNewPost = (props) => {

return (

<>

<form onSubmit={props.savePost}>

<h2>Create New Blog</h2>

<label className="col-sm-12 col-form-label">

<b>Title</b>

<input

className="form-control form-control-sm"

autoFocus={true}

type="text"

placeholder="Blog title"

onChange={props.savePostTitleToState}

required

ref={props.getTitle}

/>

</label>

<br />

<label className="col-sm-12 col-form-label">

<b>Content</b>

<textarea

className="form-control form-control-sm"

placeholder="description"

onChange={props.savePostContentToState}

rows="18"

cols="41"

required

ref={props.getContent}

/>

</label>

<br />

<button title="save Blog" className="btn btn-success ml-3">

save

</button>

</form>

</>

);

};

export default CreateNewPost;

**DisplayAllPosts.js**

import React, { useState, useRef } from "react";

import CreateNewPost from "./CreateNewPost";

import ModifyPost from "./ModifyPost";

import Post from "./Post";

const DisplayAllPosts = () => {

// managing states below

const [title, setTitle] = useState("");

const [content, setContent] = useState("");

const [allPosts, setAllPosts] = useState([

]);

// const [allPosts, setAllPosts] = useState([]) // can also be used

const [isCreateNewPost, setIsCreateNewPost] = useState(false);

const [isModifyPost, setIsModifyPost] = useState(false);

const [editPostId, setEditPostId] = useState("");

// Initialize useRef (to empty title and content once saved)

const getTitle = useRef();

const getContent = useRef();

// function 1 (accepting title in state by user input)

const savePostTitleToState = (event) => {

setTitle(event.target.value);

};

// function 2 (accepting content/description in state by user input)

const savePostContentToState = (event) => {

setContent(event.target.value);

};

// function 3 (to save title and content in allPosts state)

const savePost = (event) => {

event.preventDefault();

const id = Date.now();

setAllPosts([...allPosts, { title, content, id }]);

getTitle.current.value = "";

getContent.current.value = "";

toggleCreateNewPost();

};

// function 4 (toggle create new post visibility)

const toggleCreateNewPost = () => {

setIsCreateNewPost(!isCreateNewPost);

};

// function 5 (toggle post editing)

const toggleModifyPostComponent = () => {

setIsModifyPost(!isModifyPost);

};

// function 6 (to edit posts)

const editPost = (id) => {

setEditPostId(id);

toggleModifyPostComponent();

};

/ function 7 (to update the posts)

const updatePost = (event) => {

event.preventDefault();

const updatedPost = allPosts.map((eachPost) => {

if (eachPost.id === editPostId) {

return {

...eachPost,

title: title || eachPost.title,

content: content || eachPost.content

};

}

return eachPost;

});

setAllPosts(updatedPost);

toggleModifyPostComponent();

};

// function 8 (to delete posts)

const deletePost = (id) => {

const modifiedPost = allPosts.filter((eachPost) => {

return eachPost.id !== id;

});

setAllPosts(modifiedPost);

};

if (isCreateNewPost) {

return (

<>

<CreateNewPost

savePostTitleToState={savePostTitleToState}

savePostContentToState={savePostContentToState}

getTitle={getTitle}

getContent={getContent}

savePost={savePost}

/>

{/\* Cancel Button \*/}

<button

className="btn btn-danger cancel-button"

onClick={toggleCreateNewPost}

>

Cancel

</button>

</>

);

} else if (isModifyPost) {

const post = allPosts.find((post) => {

return post.id === editPostId;

});

return (

<>

<ModifyPost

title={post.title}

content={post.content}

updatePost={updatePost}

savePostTitleToState={savePostTitleToState}

savePostContentToState={savePostContentToState}

toggleCreateNewPost={toggleCreateNewPost}

/>

<button

className="btn btn-danger cancel-update-button"

onClick={toggleModifyPostComponent}

>

Cancel

</button>

</>

);

}

return (

<>

<h2>All Blogs</h2>

{!allPosts.length ? (

<div>

<li>There are no posts yet.</li>

</div>

) : (

allPosts.map((eachPost) => (

<Post

id={eachPost.id}

key={eachPost.id}

title={eachPost.title}

content={eachPost.content}

editPost={editPost}

deletePost={deletePost}

/>

))

)}

<button

className="btn btn-outline-info button-edits create-post"

onClick={toggleCreateNewPost}

>

Create New

</button>

</>

);

};

export default DisplayAllPosts;

**ModifyPost.js**

import React from "react";

const ModifyPost = (props) => {

return (

<>

<form>

<h2>Modify Blog</h2>

<label className="col-sm-12 col-form-label">

<b>Title</b>

<input

className="form-control form-control-sm"

defaultValue={props.title}

autoFocus={true}

onChange={props.savePostTitleToState}

placeholder="title"

size="39"

/>

</label>

<br />

<label className="col-sm-12 col-form-label">

<b>Content</b>

<textarea

className="form-control form-control-sm"

defaultValue={props.content}

onChange={props.savePostContentToState}

placeholder="contents"

rows="18"

cols="41"

/>

</label>

<button

title="update changes"

className="btn btn-success ml-3"

onClick={props.updatePost}

>

Update Blog

</button>

</form>

</>

);

};

export default ModifyPost;

**Post.js**

import React from "react";

const Post = ({ id, title, content, editPost, deletePost }) => {

return (

<>

<div className="card card-width bg-dark">

<section key={id}>

<h3>{title}</h3>

<hr className="new1"></hr>

<p>{content}</p>

<span title="edit post" onClick={() => editPost(id)}>

<i className="edit-button far fa-edit fa-2x button-css" />

</span>

<span title="delete post" onClick={() => deletePost(id)}>

<i className="delete-button fas fa-trash fa-2x ml-2 button-css" />

</span>

</section>

</div>

</>

);

};

export default Post;

**App.js**

import React from "react";

import "../styles.css";

import DisplayAllPosts from "./DisplayAllPosts";

//import Post from "./Post";

const App = () => {

return (

<>

<DisplayAllPosts />

</>

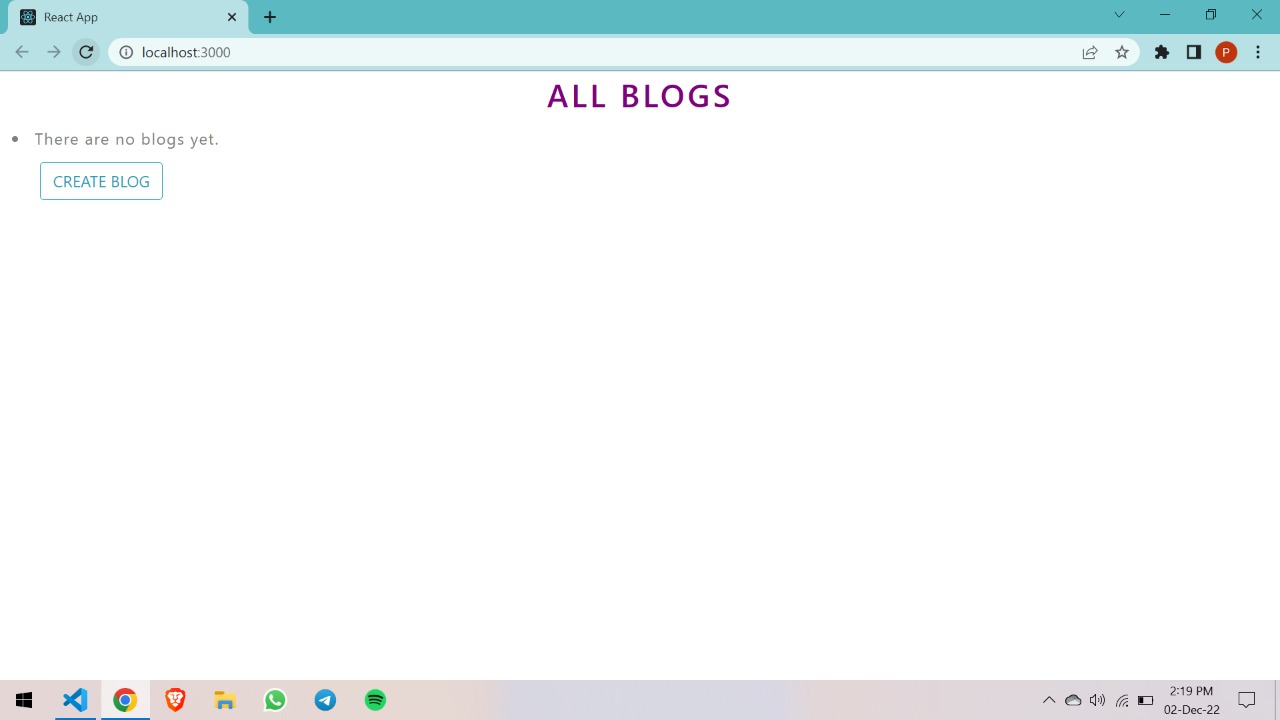
);

};

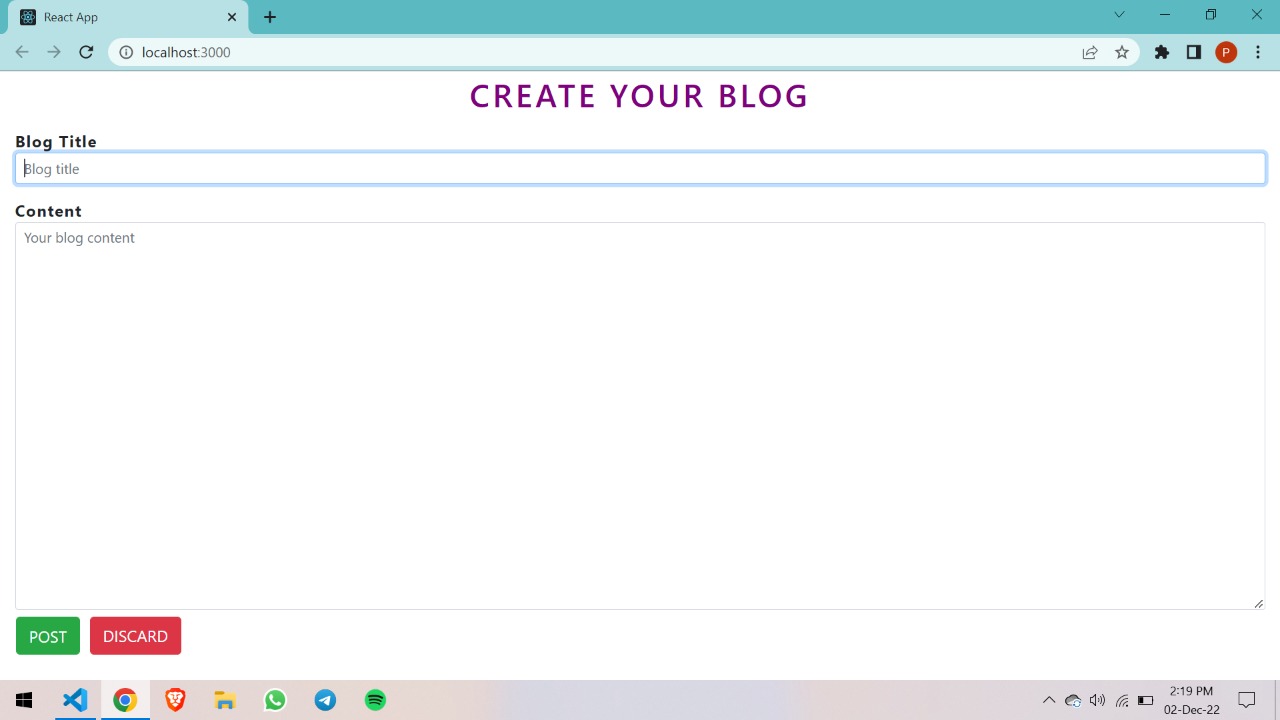
export default App;

**Output Screens:**

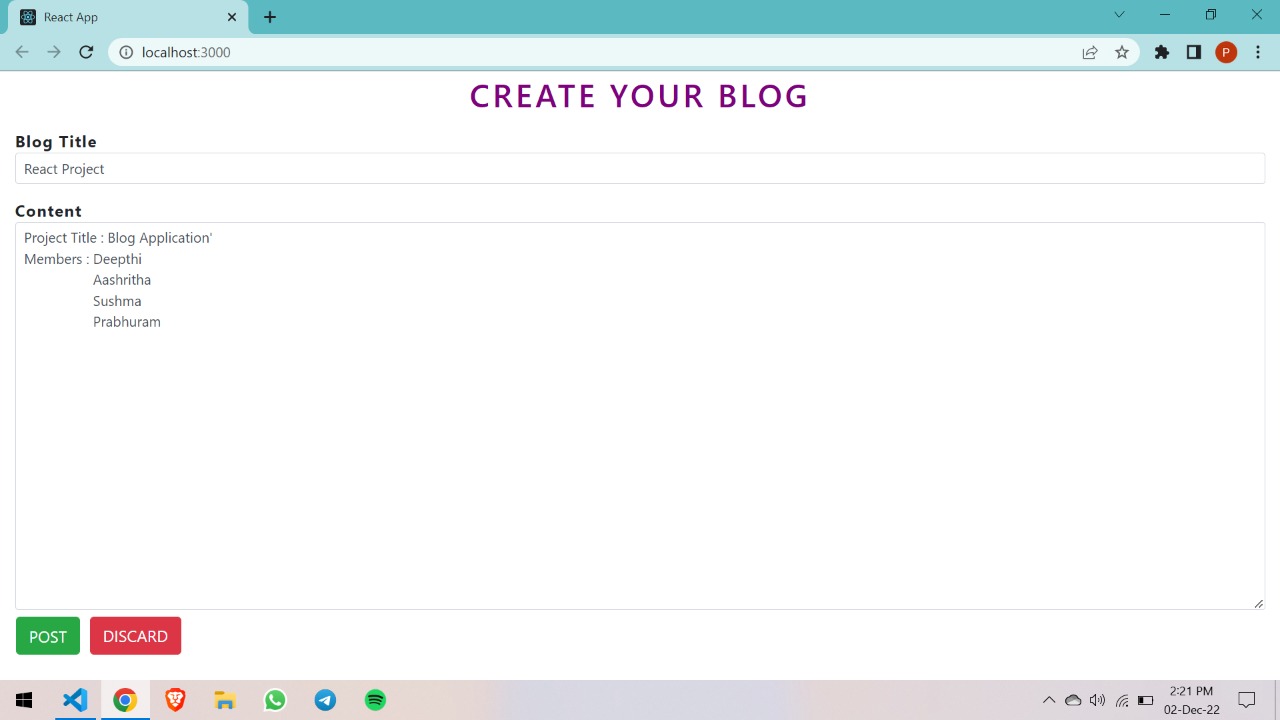
**Home/All blogs page(empty):**

****

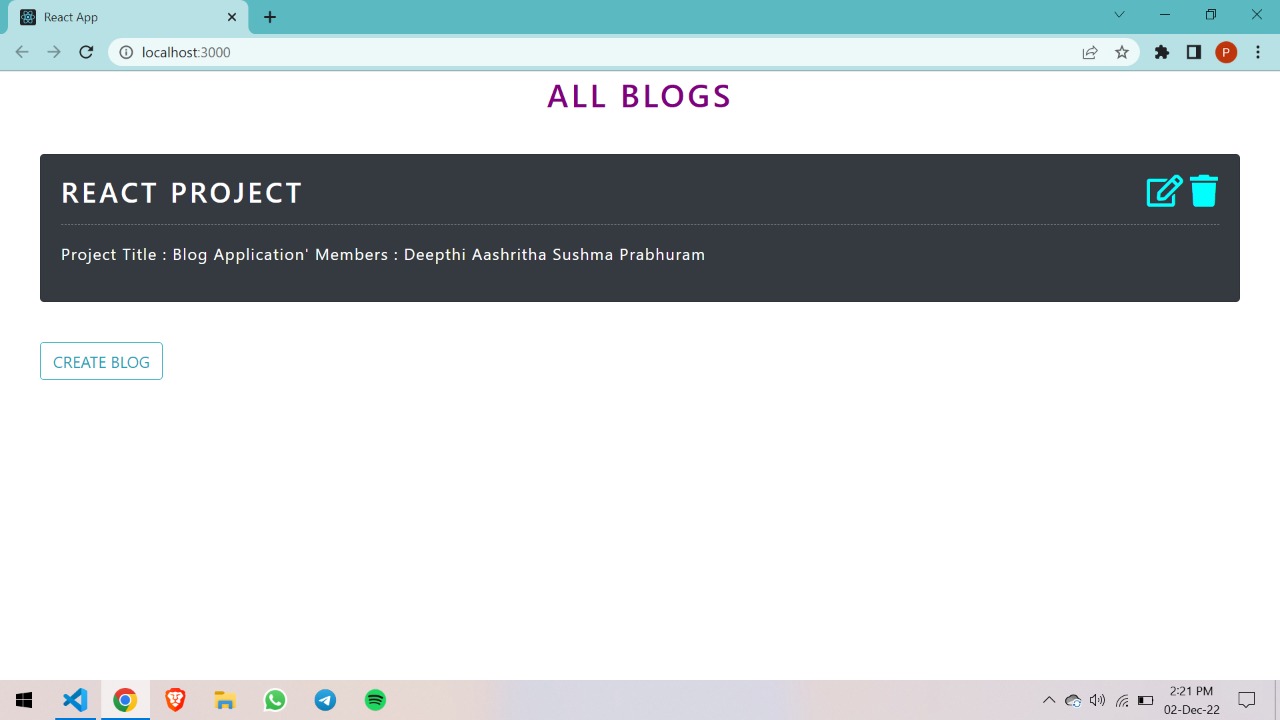
**Blog Creation page:**

****

**Creating process page:**

****

**Added Blog page:**

****

**CONCLUSION:**

We have created a blogging platform where users can post their own ideas and habits, as well as impress readers by expressing their emotions.

**FUTURE SCOPE OF THE PROJECT**

1. We can build an app for this as an extension to this website.
2. Each blogger can have their own login and account storage system, and they can decide whether to make their posts public or private.

**REFERENCES**

**1.React-Bootstrap**

[**https://react-bootstrap.github.io/getting-started/introduction**](https://react-bootstrap.github.io/getting-started/introduction)

**2.React-icons**

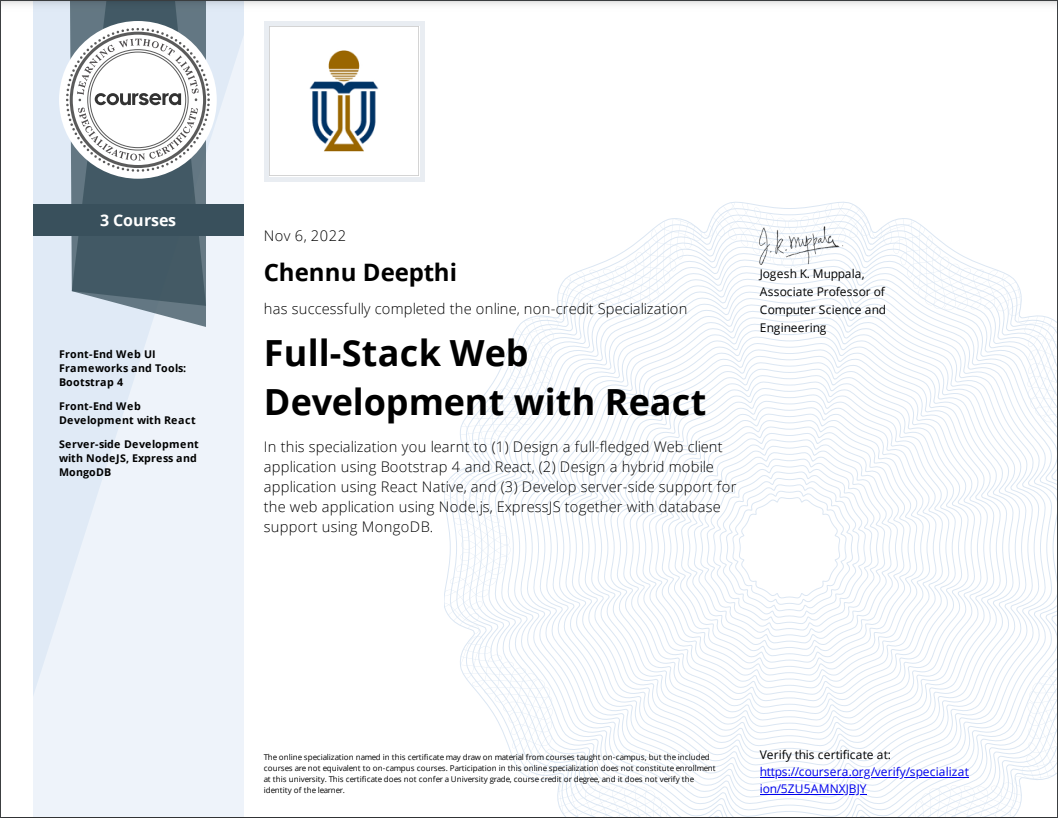
[**https://react-icons.github.io/react-icons/**](https://react-icons.github.io/react-icons/)

**3.React JavaScript**

[**https://reactjs.org/docs/getting-started.html**](https://reactjs.org/docs/getting-started.html)

**PROOF OF CERTIFICATES**

**208W1A1209**



**208W1A1215**



**208W1A1235**



**208W1A1288**

