"CODE EDITOR"

A Project Report Submitted to
Rajiv Gandhi Proudyogiki Vishwavidyalaya

Towards Partial Fulfilment for the Award of
Bachelor Of Technology in Computer Science and Engineering

Submitted By:
Ananya Shrivastava
0827CS201032

Guided By:
Prof. Priyanka Jangde
Prof. Narendra Pal Singh Rathore
Associate Professor
Computer Science and Engineering

ACROPOLIS INSTITUTE OF TECHNOLOGY AND RESEARCH, INDORE

Jan- June 2022

EXAMINER APPROVAL

The Project entitled ***"CODE EDITOR"*** submitted by **Ananya Shrivastava 0827CS201032** has been examined and is hereby approved towards partial fulfilment for the award of Bachelor of Technology degree in Computer Science and Engineering discipline, for which it has been submitted. It understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it has been submitted.

(Internal Examiner)	**(External Examiner)**
Date:	**Date:**

GUIDE RECOMMENDATION

This is to certify that the work embodied in this project entitled **"CODE EDITOR"** submitted by **Ananya Shrivastava** is a satisfactory account of the bonafide work done under the supervision of **Prof.Priyanka Jangde** and **Prof. Narendra Pal Singh** are recommended towards partial fulfilment for the award of the Bachelor of Technology (Computer Science and Engineering) degree by Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal.

(Project Guide) **(Project Coordinator)**

STUDENTS UNDERTAKING

This is to certify that project entitled **"CODE EDITOR"** has developed by us under the supervision of Prof. Ronak Jain and Prof. Narendra Pal Singh Rathore. The whole responsibility of work done in this project is ours. The sole intension of this work is only for practical learning and research.

We further declare that to the best of our knowledge; this report does not contain any part of any work which has been submitted for the award of any degree either in this University or in any other University / Deemed University without proper citation and if the same work found then we are liable for explanation to this.

Ananya Shrivastava (0827CS201032)

Acknowledgement

We thank the almighty Lord for giving me the strength and courage to sail out through the tough and reach on shore safely.

There are number of people without whom this projects work would not have been feasible. Their high academic standards and personal integrity provided me with continuous guidance and support.

We owe a debt of sincere gratitude, deep sense of reverence and respect to our guide and mentors **Prof. Priyanka Jangde** and **Prof. Narendra Pal Singh Rathore**, Associate Professor, AITR, for their motivation, sagacious guidance, constant encouragement, vigilant supervision, and valuable critical appreciation throughout this project work, which helped us to successfully complete the project on time.

We express profound gratitude and heartfelt thanks to **Dr Kamal Kumar Sethi**, HOD CSE, AITR Indore for his support, suggestion, and inspiration for carrying out this project. I am very much thankful to other faculty and staff members of CSE Dept, AITR Indore for providing me all support, help and advice during the project. We would be failing in our duty if do not acknowledge the support and guidance received from **Dr S. C. Sharma**, Director, AITR, Indore whenever needed. We take opportunity to convey my regards to the management of Acropolis Institute, Indore for extending academic and administrative support and providing me all necessary facilities for project to achieve our objectives.

We are grateful to **our parent** and **family members** who have always loved and supported us unconditionally. To all of them, we want to say, "Thank you", for being the best family that one could ever have and without whom none of this would have been possible.

Ananya Shrivastava(0827CS201032)

Executive Summary

CODE EDITOR

This project is submitted to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (MP), India for partial fulfilment of Bachelor of Technology in Computer Science & Engineering branch under the sagacious guidance and vigilant supervision of Prof. Priyanka Jangde and Prof. Narendra Pal Singh Rathore.

A code editor is one of the tools used for software development. This is a text editor designed for writing and editing code. A code editor can be a standalone application or built into an IDE.

Key words: React , Artificial Intelligence, Machine Learning, Dialogflow.

“Difficulties in your life do not come to destroy you… But to help you realise your hidden potential and power, Let difficulties know that you too are difficult!” ~A.P.J. Abdul Kalam

Table Of Contents

Chapter 1: Introduction

Introduction

The world of internet is growing rapidly, many applications that previously created on the desktop start moving to the web. Many applications could be accessed anytime and anywhere easily using internet . Developers need tools to create their applications one of them name code editor. Code Editor is one of the essential tools for programmers, designers and even writers.Integrated Development Environment are often cumbersome for small tasks, such as changing a file or doing a small project.

On the other hand, text editors such as Windows Notepad or Mac TextEdit are Too easy to edit codes. These apps lack needful capabilities and make coding complicated.The solution is to use a code editor. Editing a file or project and managing the contents of a folder are some of the tasks that a code editor is very good at.

1.1 Overview

In the past, the code editors were specific for each platform. But now, most of them are cross-platform, and you can use them on different operating systems in almost the same way. Therefore, programmers can use office computers, personal pcs, or even shared devices and no longer need to be familiar with different environments to get things done.

In addition, you can change the behavior of these editors by changing the configuration settings (e.g., tab length adjustment, line spacing and wrapping, autocompletion, syntax highlighting, etc.). This feature allows the programmer to make the most of the software. On the other hand, the default settings allow the user to have a pleasant experience and use the “ready to use” capabilities

1.2 Background and Motivation

Developers need tools to create their applications one of them name code editor . The purpose of this research is to design and develop a real time code editor application using web socket technology to help us collaborate while working on the project this application provides feature where users can collaborate on a project in real time . The authors using analysis methodology with conducting on a study of the current code editor applications distributing questionnairies and conducting on literature study. It is web application that provides workspace to writing perform, display the results of the code through the terminal and collaborate with the users in real time .The applications main features are providing workspace to make, execute and build the source code. This application supports Html , Css and Javascript.

While working on a specific project I realized that for either it is html, css or javascript I need to open different editors and at that time I was learning react so I decided to make one editor or platform for each of them.

1.3 Problem Statements and Objectives

Using the perfect code editor allows you to simplify and speed up the proces of writing code. This is possible due to the fact that they provide useful features for automatic code generation, auto-correction, quick find and replace, and much more. Source code editors also allow you to quickly find and fix errors.

All code editors highlight the code in different colors, which makes the process of reading the code much more convenient and faster.

Compared to complex IDEs, which, in addition to the code editor, also include a compiler, debugger, and profiler, code editors load faster and also take up less memory. If you are working on a computer with few resources, then it is better to use a code editor.In addition, code editors usually support a large list of programming languages, while IDEs have a limited number.

1.4 Scope of the Project

Using the perfect code editor allows you to simplify and speed up the process of writing code. This is possible due to the fact that they provide useful features for automatic code generation, auto-correction

.

Compared to complex IDEs, which, in addition to the code editor, also include a compiler, debugger, and profiler, code editors load faster and also take up less memory. If you are working on a computer with few resources, then it is better to use a code editor.
In addition, code editors usually support a large list of programming languages, while IDEs have a limited number.

1.5 Group Organization

Along with preliminary investigation and understanding the drawback of the current system, I studied about the topic and its scope. I surveyed various research papers related to chatbots and the technology to be used. I also contributed in documentation phase of the project. Then investigated, found the right technology, and studied it in depth. I also organized and debugged the code of the project. I also tested the overall functionality of the project.

worked on the overall documentation of the project. I also collected the object data and trained the model for it. Moreover, I managed the overall structure of the project, its design and working.

I investigated and found the right technology and studied in deep about it. For the implementation of the project. Implementation logic for the project objective and coding of internal functionalities is also done by me. I worked on the frontend, making the HTML.

1.6 Report Structure

The project ***CODE EDITOR*** is primarily concerned with the **editing all html, css and javascript in one platform** and whole project report is categorized into five chapters.

Chapter 1: Introduction- introduces the background of the problem followed by rationale for the project undertaken. The chapter describes the objectives, scope and applications of the project. Further, the chapter gives the details of team members and their contribution in development of project which is then subsequently ended with report outline.

Chapter 2: Review of Literature- explores the work done in the area of Project undertaken and discusses the limitations of existing system and highlights the issues and challenges of project area. The chapter finally ends up with the requirement identification for present project work based on findings drawn from reviewed literature and end user interactions.

Chapter 3: Proposed System - starts with the project proposal based on requirement identified, followed by benefits of the project. The chapter also illustrate software engineering paradigm used along with different design representation. The chapter also includes block diagram and details of major modules of the project. Chapter also gives insights of different type of feasibility study carried out for the project undertaken. Later it gives details of the different deployment requirements for the developed project.

Chapter 4: Implementation - includes the details of different Technology/ Techniques/ Tools/ Programming Languages used in developing the Project. The chapter also includes the different user interface designed in project along with their functionality. Further it discuss the experiment results along with testing of the project. The chapter ends with evaluation of project on different parameters like accuracy and efficiency.

Chapter 5: Conclusion - Concludes with objective wise analysis of results and limitation of present work which is then followed by suggestions and recommendations for further improvement.

Chapter 2: Review of Literature

Review of Literature

2.1 Preliminary Investigation

Using the perfect code editor allows you to simplify and speed up the process of writing code. This is possible due to the fact that they provide useful features for automatic code generation, auto-correction, quick find and replace, and much more. Source code editors also allow you to quickly find and fix errors.

All code editors highlight the code in different colors, which makes the process of reading the code much more convenient and faster.

Compared to complex IDEs, which, in addition to the code editor, also include a compiler, debugger, and profiler, code editors load faster and also take up less memory. If you are working on a computer with few resources, then it is better to use a code editor.

In addition, code editors usually support a large list of programming languages, while IDEs have a limited number.

2.1.1 Current System

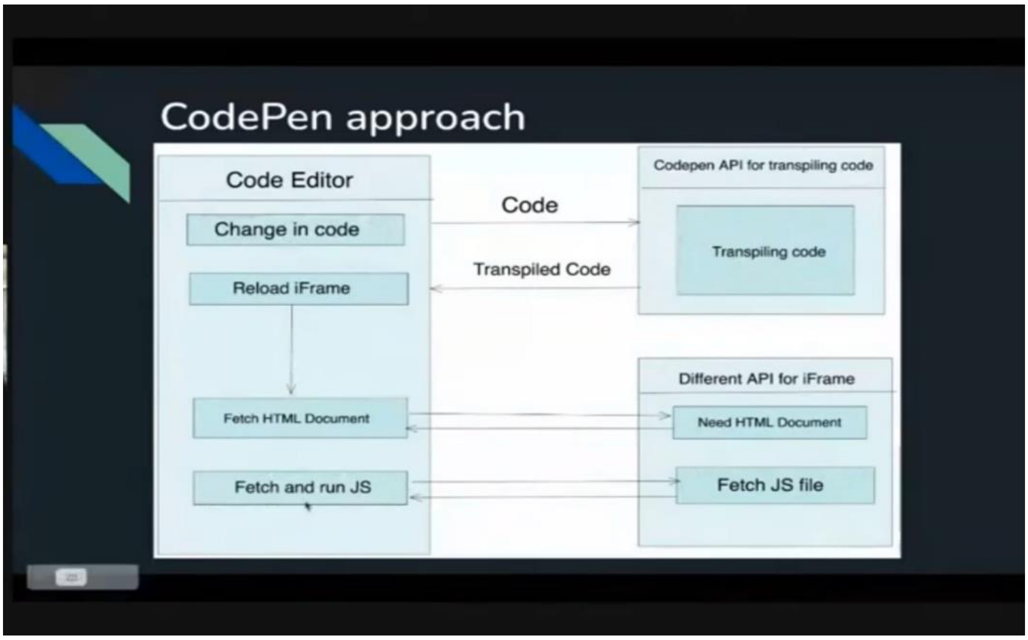
Code editors or source code editors are software that is designed specifically to help developers with coding. These are text editors with additional functionalities to manage and edit code. It can be standalone or it can be a part of an IDE.

When developers write code using these editors, it takes care of syntax. Code editors immediately warn of any syntax errors. Developers don't have to worry about syntax. Auto indentation & auto-completion saves a lot of time. Some editors, like sublime text and visual studio code, have an integrated terminal.

Core Features:

Enlisted below are the various Features of these Editors:

- Syntax highlighting
- Auto indentation
- Auto-completion
- Brace matching



CodePen approach

Code Editor

Change in code

Reload iFrame

Fetch HTML Document

Fetch and run JS

Code

Transpiled Code

Codepen API for transpiling code

Transpiling code

Different API for iFrame

Need HTML Document

Fetch JS file

2.2 Limitations of Current System

The code editors are helpful if you are writing the code from scratch. But if you have to edit the existing code which is written by someone else then IDE is the best option. IDE is helpful in understanding the code written by others as code editors cannot compile or debug the code.

Some features of these editors are better than IDE like theme selection and searches, which are important while writing the code. Meanwhile, instead of editing a few lines and constantly debugging with code editors, you can concentrate more on coding.

Another reason for using these editors instead of IDE is that IDE uses more resources like CPU, memory, and disk space. The coding editors don’t use many resources, hence they are fast.

2.3 Requirement Identification and Analysis for Project

All the project requires research and analysis investing some time on research and analysis will save much time in future to gather other information

An online web code editor is most useful when you do not have the opportunity to use a code editor application, or when you want to quickly try out something on the web with your computer or even your mobile phone. This is also an interesting project to work on because having the knowledge of how to build a code editor will give you ideas on how to approach other projects that require you to integrate a code editor to show some functionality

2.3.1 Conclusion

Technology is not the only requirement to build a successful Code editor. This chapter reviews the literature surveys that have been done during the research work. The related work that has been proposed by many researchers has been discussed. All the necessary requirements to make a successful code editor has been discussed in this chapter.

Chapter 3: Proposed System

Proposed System

3.1 The Proposal

The proposal is to provide one platform to edit all three languages in one so it it easy and is beneficial for the people to access it with ease.

Code editors have syntax highlighting and allow you to automatically format your code. In addition, they allow you to quickly find and replace text, rename objects in code, and much more.

3.2 Benefits of the Proposed system

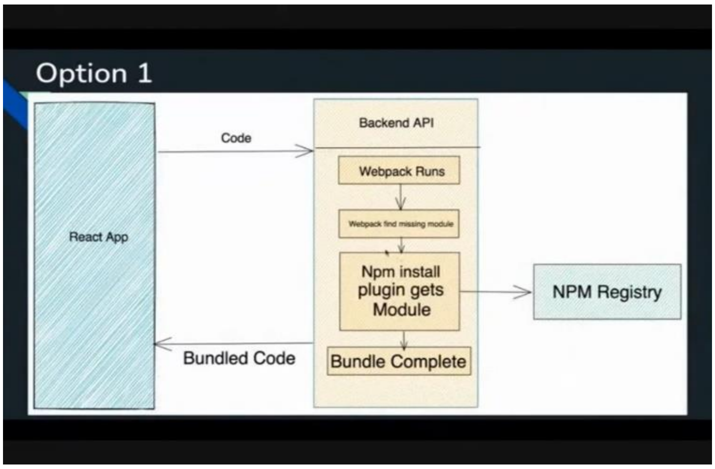
The main advantage of code editors is that they are **enhanced programs** that can help you debug, stage, version, and pip code. You can download a code editor with any space on your laptop or even use one embedded within a web browser and be on your way to writing code in no time. The benefits of the proposed system:

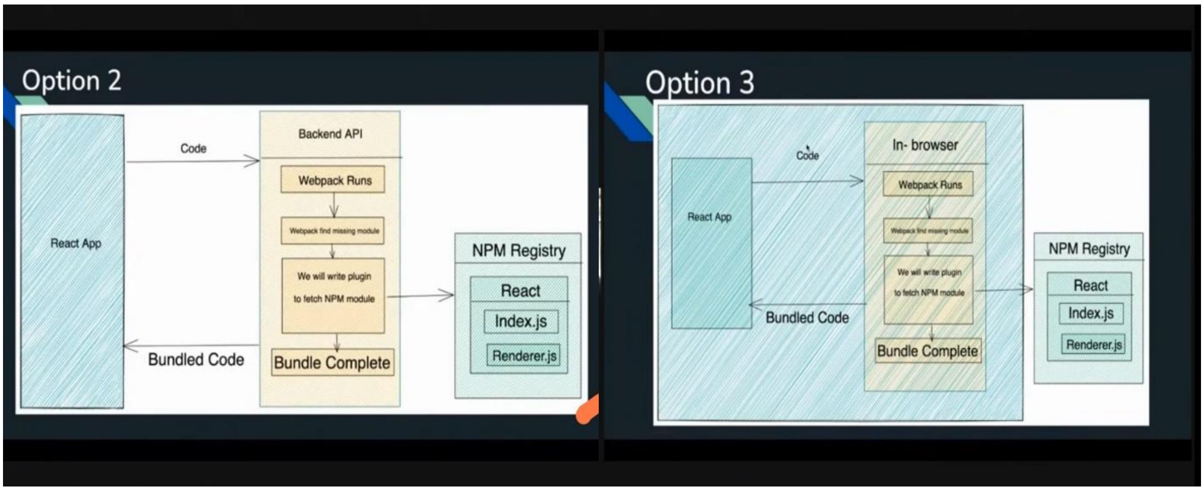
1.You can write all the codes of html, css, and javascript in one screen.

2.Consumes less time.

3. Includes a flexible autocompletion system.

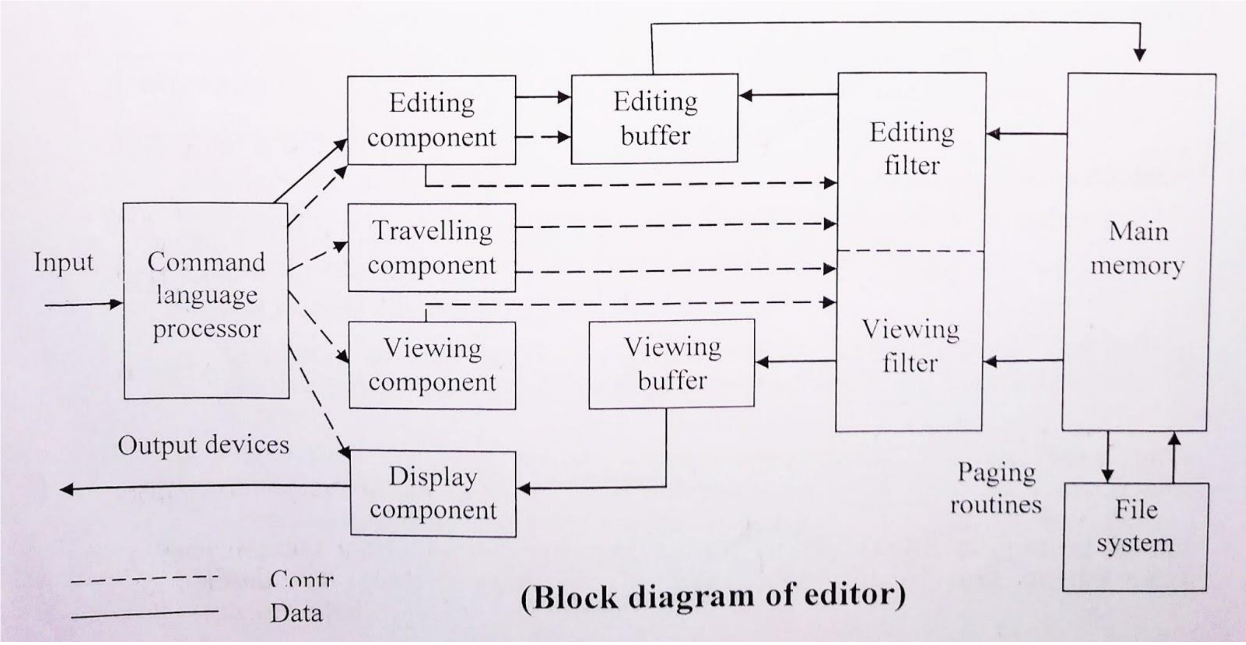
4.Customizes how each language appears on your editor





3.3 Block Diagram

A block diagram is like a flowchart because it provides a visual representation of how different parts of a system work together to produce an output. Block diagrams track the various inputs of a system by creating blocks and using connections to show how those blocks produce a result. They provide a high-level overview of the system by categorizing each element as a general idea rather than an in-depth, interactive part of the system.



(Block diagram of editor)

3.4 Feasibility Study

A feasibility study is an analysis of how successfully a system can be implemented, accounting for factors that affect it such as economic, technical and operational factors to determine its potential positive and negative outcomes before investing a considerable amount of time and money into it

3.4.1 Technical

To create any project it is necessary to choose the appropriate technologies which will help to make the website responsive and dynamic choosing right database will save lots of time to store the information so here we chose the most appropriate and trending technologies which is

1.React

2.Code mirror library

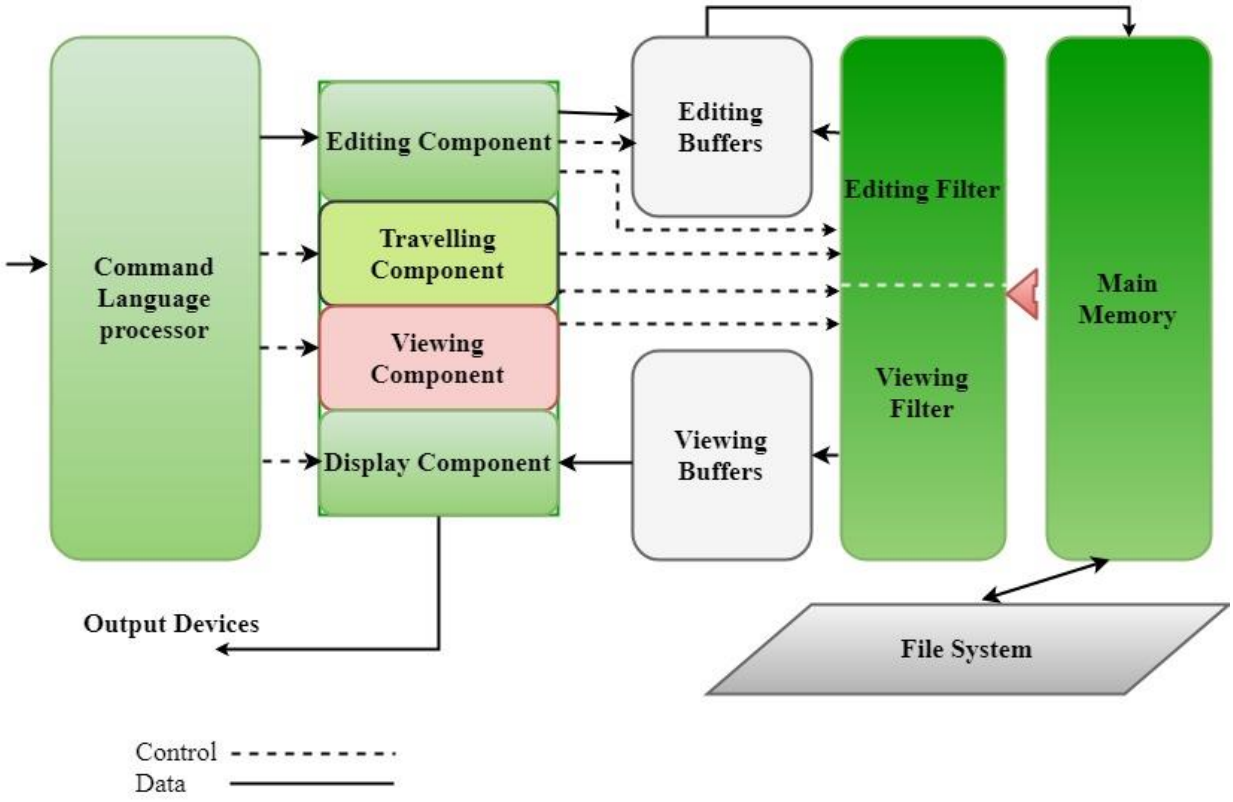
3.4.2 Economical

This project do not require lot of economical support as its maintainance is very low and it only need a domain to work perfectly.

3.4.3 Operational

The main motto of our system is to reduce efforts of developing highly complex code editors. Also, to spend more time on perfecting conversational experience with users.

3.5 Design Representation



3.6 Deployment Requirements

There are various requirements (hardware, software and services) to successfully deploy the system. These are mentioned below :

3.6.1 Hardware

32-bit, x86 Processing system
Windows 7 or later operating system
High processing computer system without GPU or with
GPU(high performance)

3.6.2 Software

My sql

Hosting domain

Chapter 4: Implementation

Implementation

We had created this web app for the purpose to solve the problem of using different platforms for editing your code. It is an online code editor which supports HTML, CSS, and Javascript Development

4.1 Technologies Used

4.1.1 Frontend

HTML

CSS

JAVASCRIPT

4.1.1 Backend

REACT

CODE MIRROR LIBRARY

4.1 Technique Used

4.1.1 REACT

JSX – JSX is JavaScript syntax extension. It isn't necessary to use JSX in React development, but it is recommended.

Components – React is all about components. You need to think of everything as a component. This will help you maintain the code when working on larger scale projects.

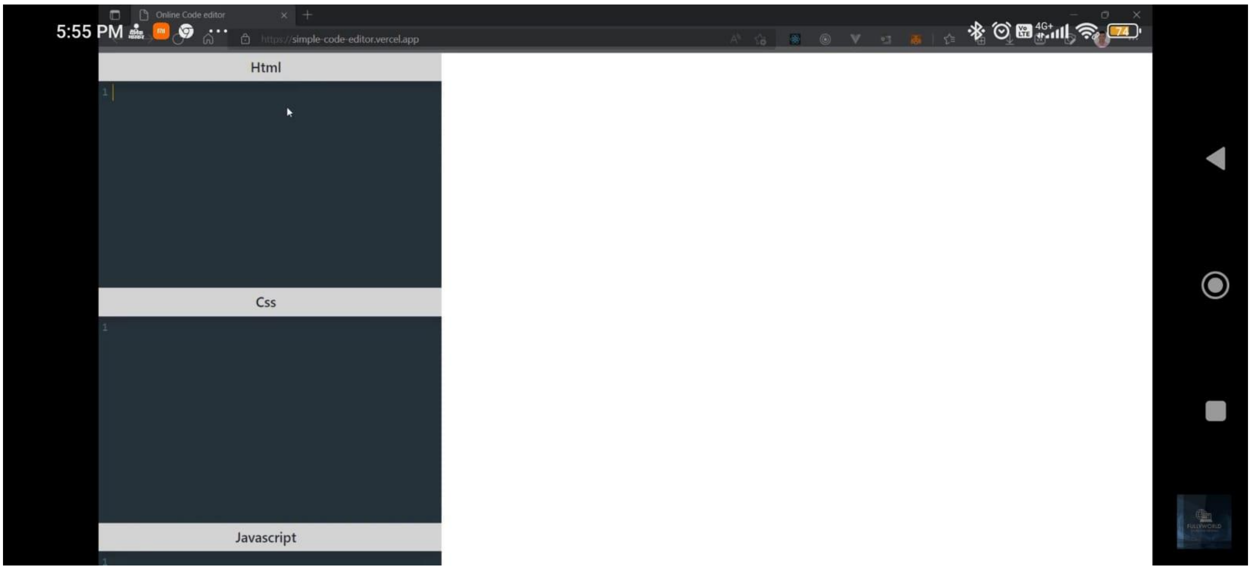
Unidirectional data flow and Flux – React implements one-way data flow which makes it easy to reason about your app. Flux is a pattern that helps keeping your data unidirectional. **License –** React is licensed under the Facebook Inc. Documentation is licensed under CC BY 4.0.

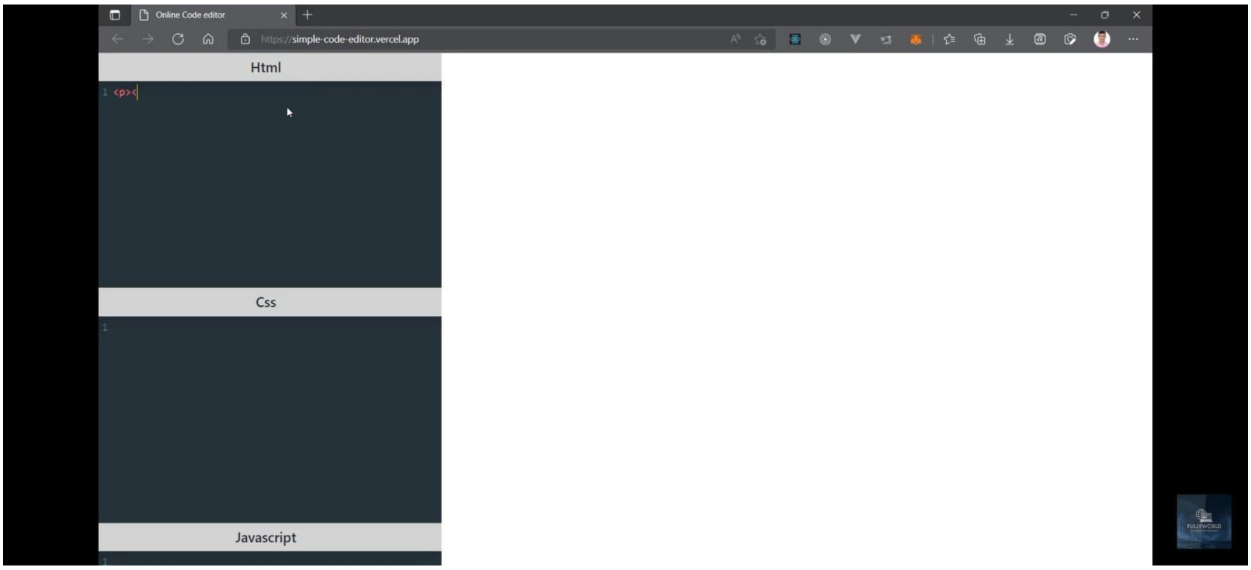
4.2 Tools Used

4.2.1 Node.js Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js serves as a medium between Dialogflow agent and our web page. It provides a connectivity between the stored questions database and front-end of website.

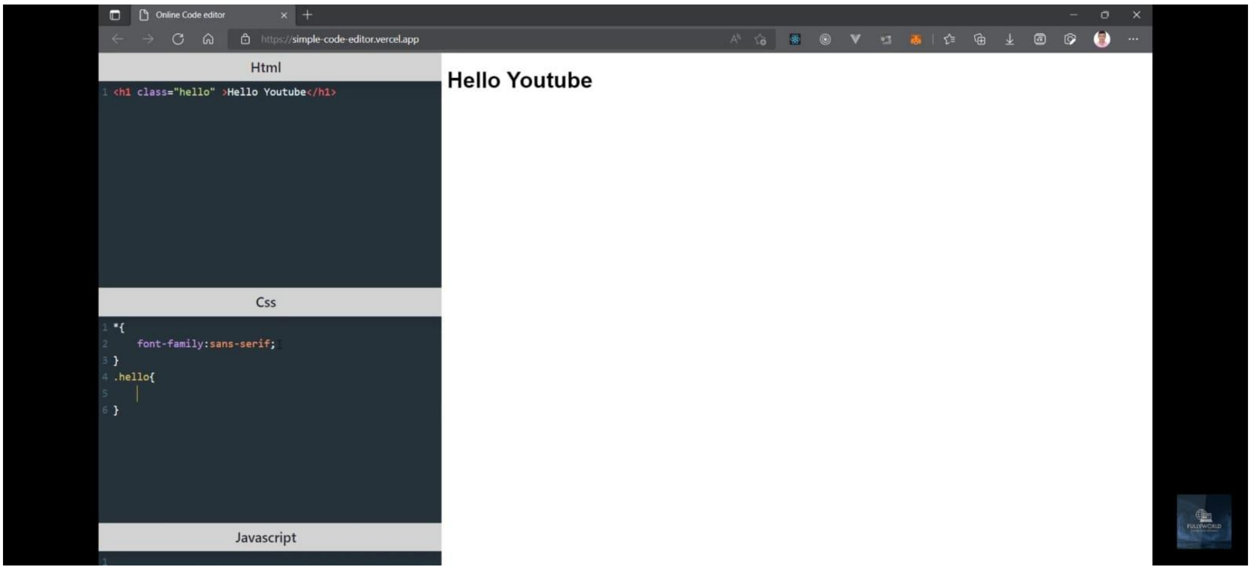
4.3 Language Used : In our project, JavaScript is used. JavaScript is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. JavaScript is used in backend for creating all functionalities and behaviour essential for running Dialogflow in parallel with the website.

4.4 Screenshots





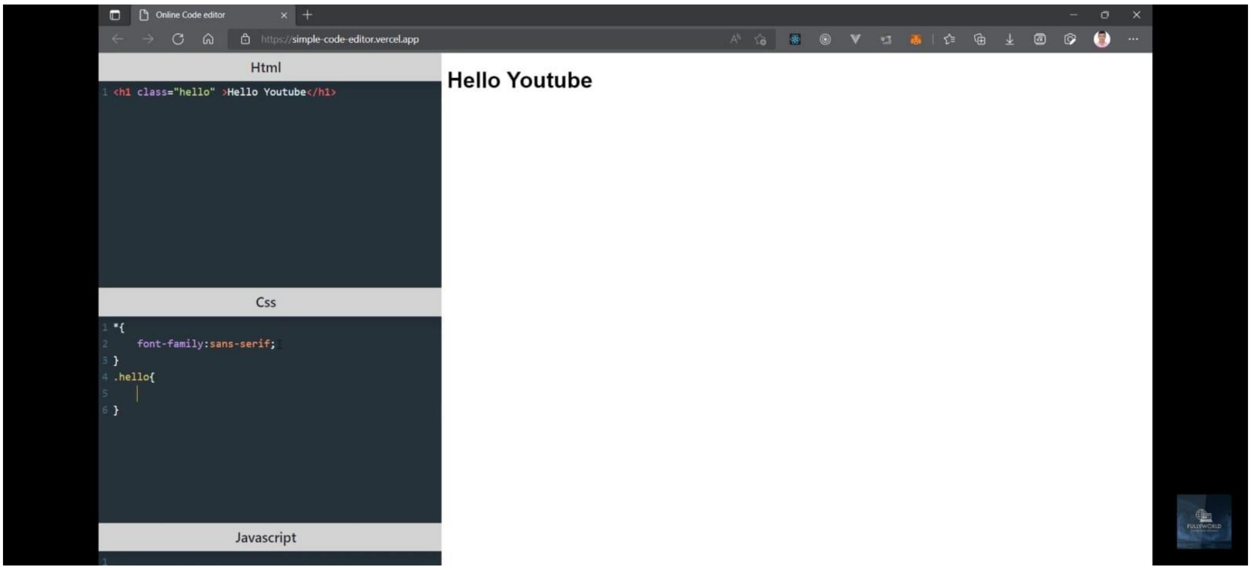
```
<p><
```



```
<h1 class="hello" >Hello Youtube</h1>
```

```
*{
    font-family:sans-serif;
}
.hello{
    
}
```

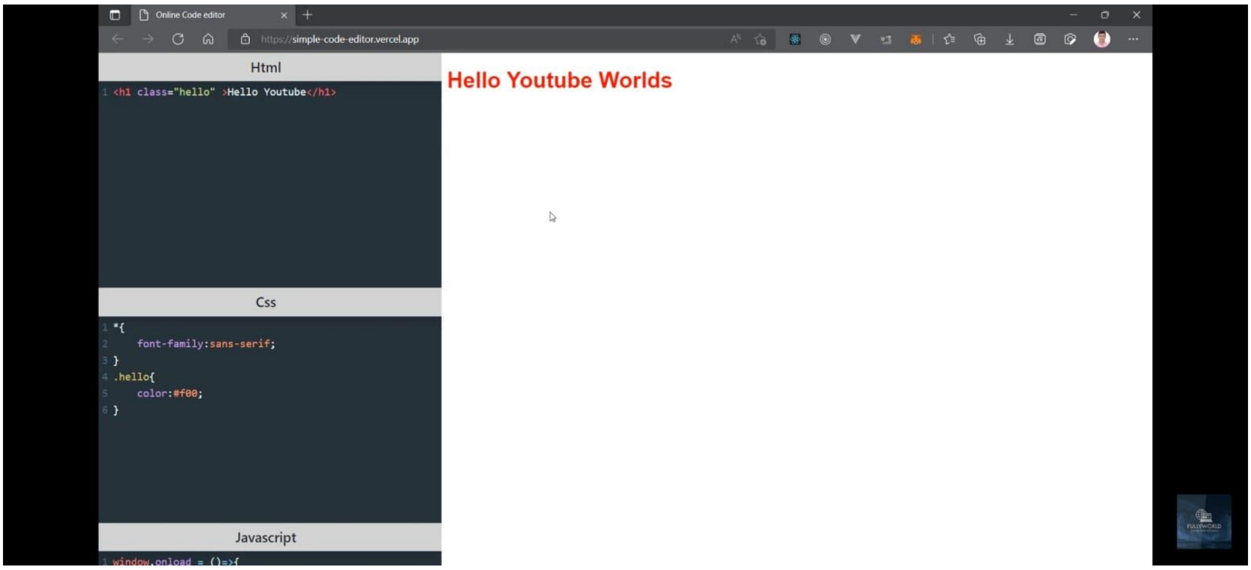
Hello Youtube



```html
<h1 class="hello" >Hello Youtube</h1>
```

```css
*{
    font-family:sans-serif;
}
.hello{
    
}
```

```
Hello Youtube
```

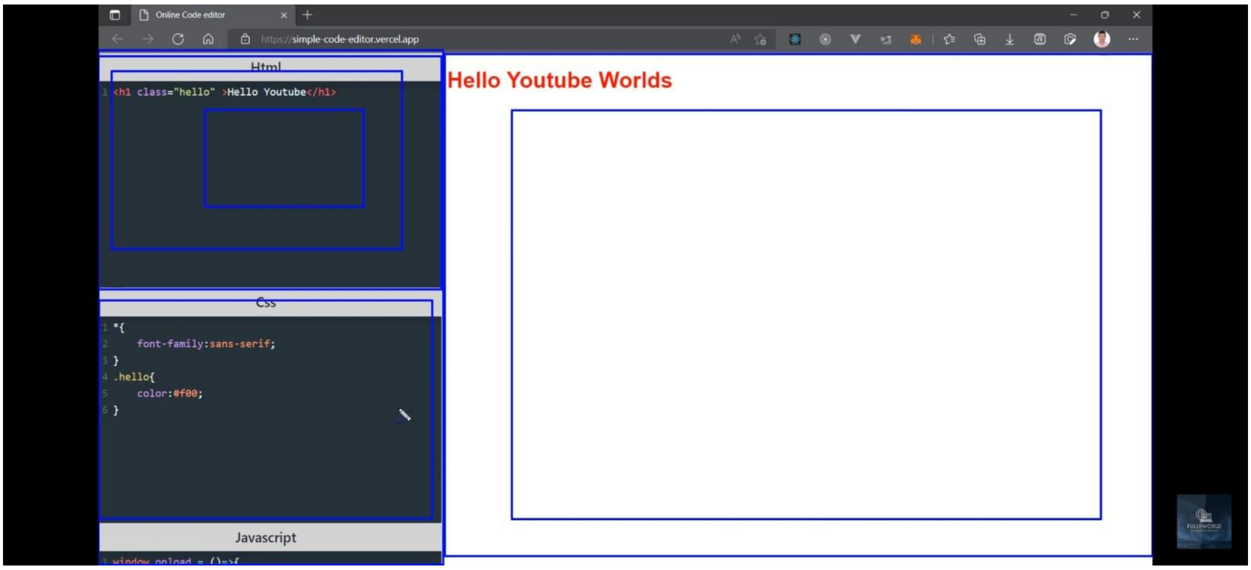


```html
<h1 class="hello" >Hello Youtube</h1>
```

```css
*{
    font-family:sans-serif;
}
.hello{
    color:#f00;
}
```

```javascript
window.onload = ()=>{
```

```
Hello Youtube Worlds
```

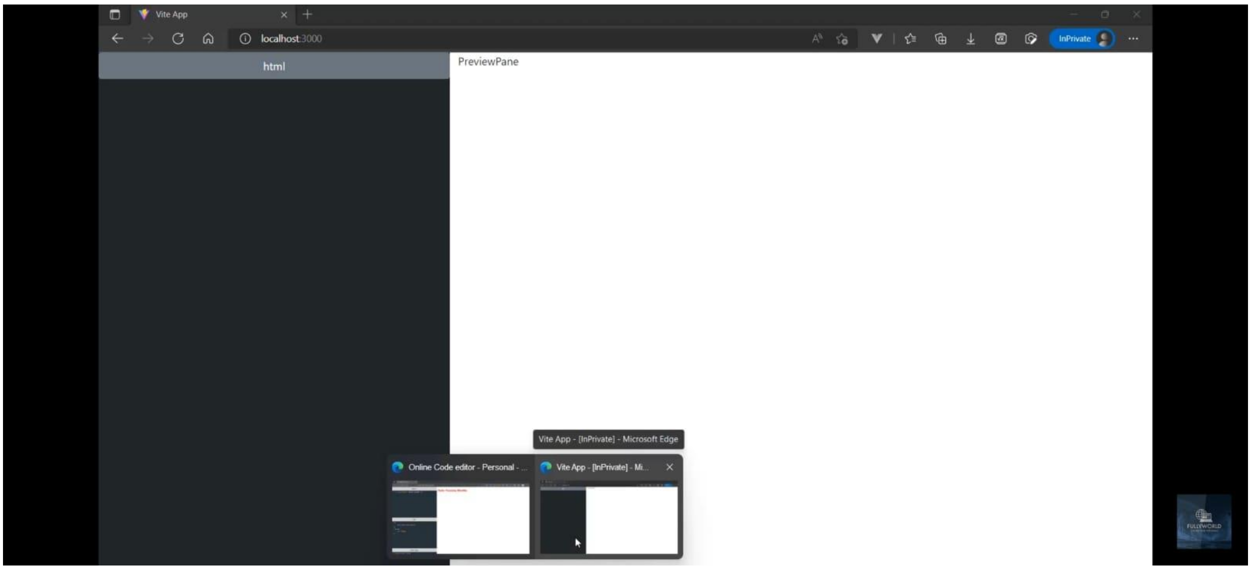


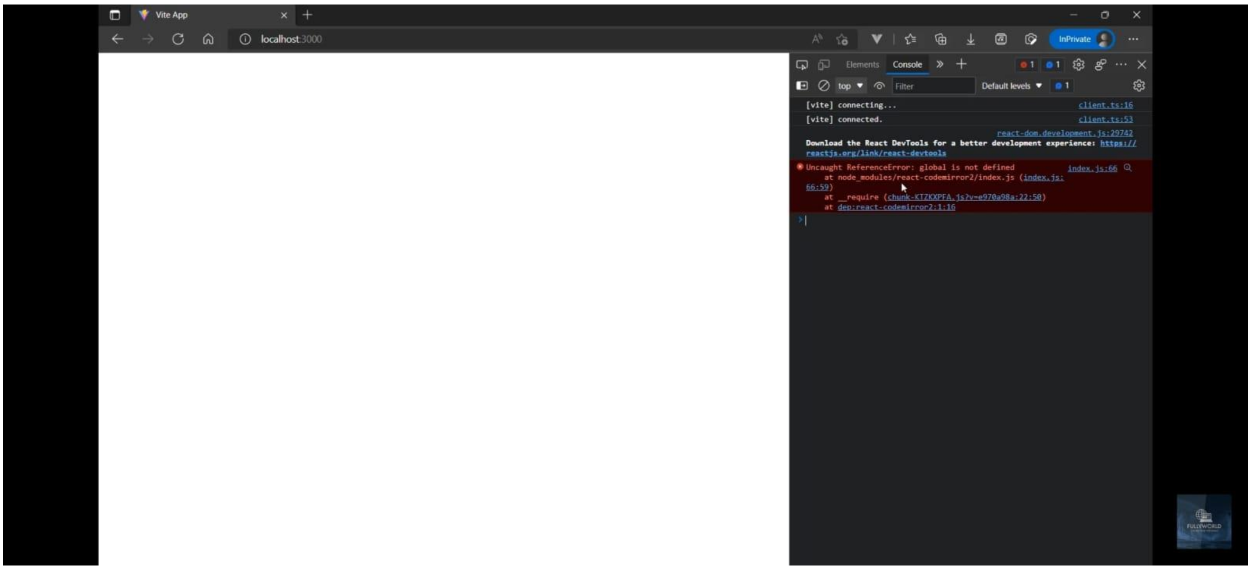
```html
<h1 class="hello" >Hello Youtube</h1>
```

```css
*{
    font-family:sans-serif;
}
.hello{
    color:#f00;
}
```

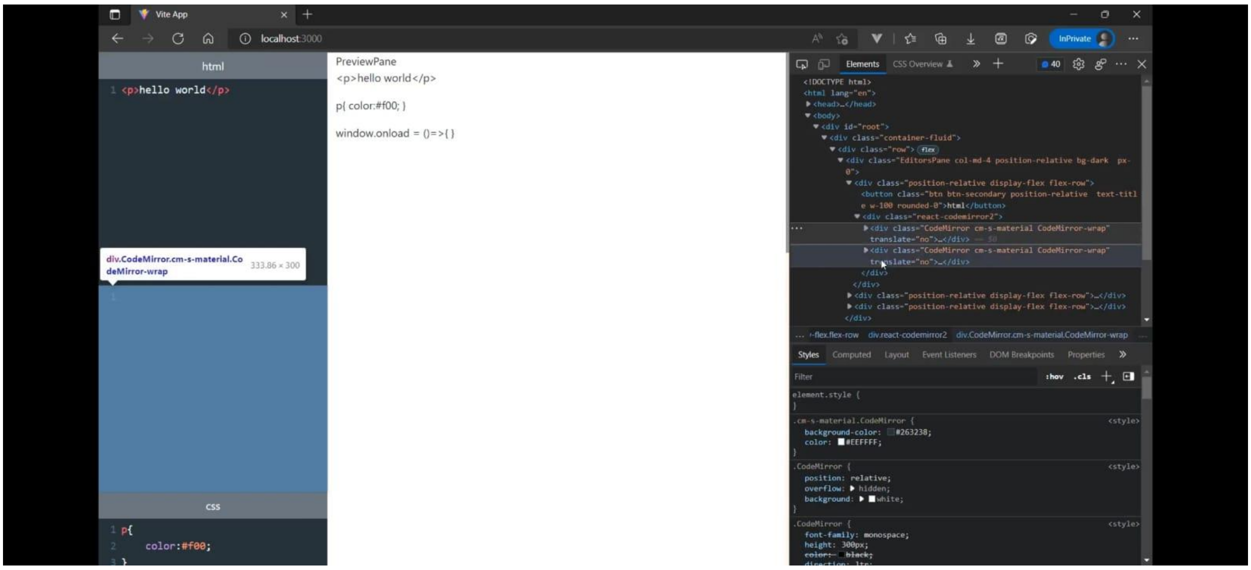
```javascript
window.onload = ()=>{
```

Hello Youtube Worlds


Vite App
localhost:3000
html
PreviewPane
Vite App - [InPrivate] - Microsoft Edge
Online Code editor - Personal - ...
Vite App - [InPrivate] - Mi...
InPrivate



```
[vite] connecting...                                                   client.ts:16
[vite] connected.                                                      client.ts:53
                                                          react-dom.development.js:29742
Download the React DevTools for a better development experience: https://
reactjs.org/link/react-devtools
Uncaught ReferenceError: global is not defined                         index.js:66
    at node_modules/react-codemirror2/index.js (index.js:
66:59)
    at __require (chunk-KTZKXPFA.js?v=e970a98a:22:50)
    at dep:react-codemirror2:1:16
```



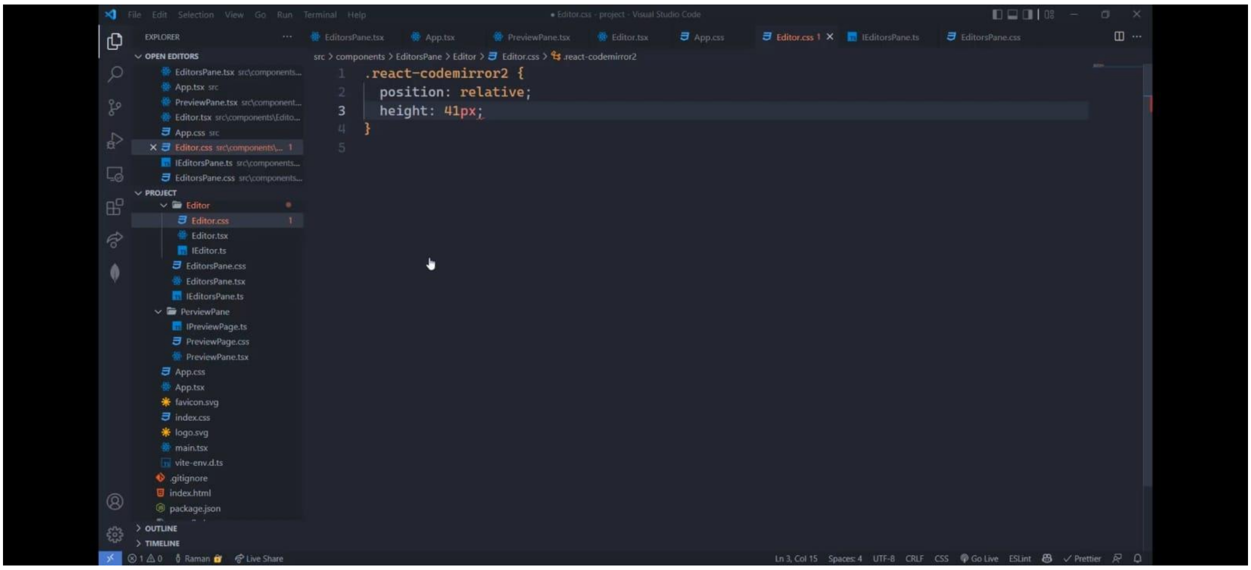
```html
<p>hello world</p>
```

```css
p{
    color:#f00;
}
```

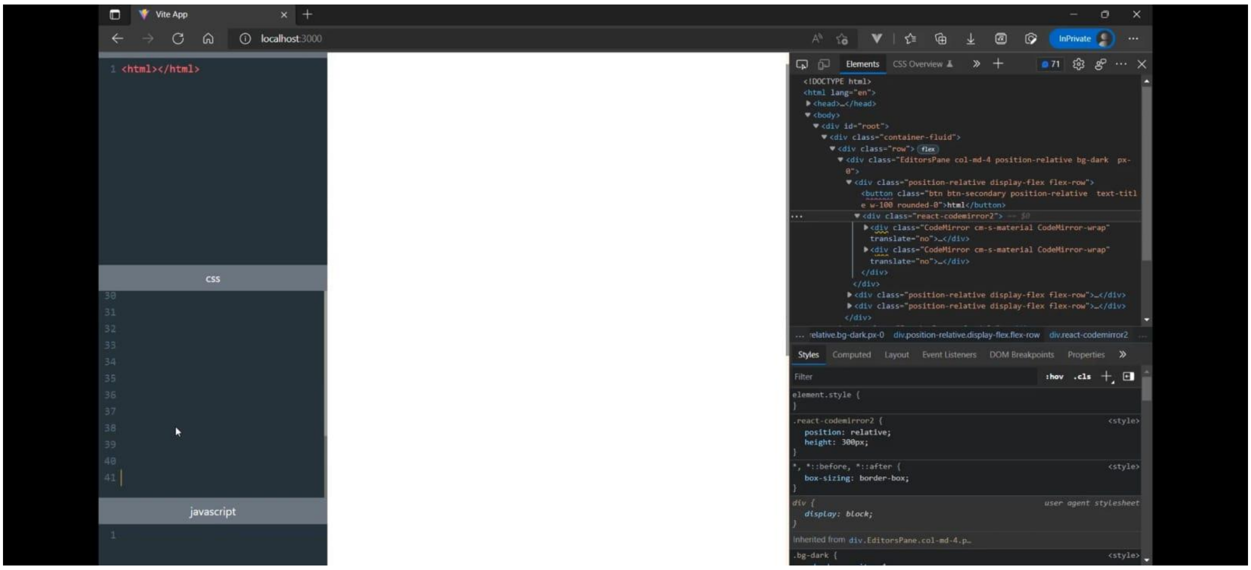
```
<p>hello world</p>

p{ color:#f00; }

window.onload = ()=>{ }
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
 ▼<body>
   ▼<div id="root">
     ▼<div class="container-fluid">
       ▼<div class="row"> flex
         ▼<div class="EditorsPane col-md-4 position-relative bg-dark  px-
           0">
           ▼<div class="position-relative display-flex flex-row">
              <button class="btn btn-secondary position-relative  text-titl
              e w-100 rounded-0">html</button>
             ▼<div class="react-codemirror2">
               ▶<div class="CodeMirror cm-s-material CodeMirror-wrap"
                translate="no">…</div> == $0
               ▶<div class="CodeMirror cm-s-material CodeMirror-wrap"
                translate="no">…</div>
              </div>
            </div>
           ▶<div class="position-relative display-flex flex-row">…</div>
           ▶<div class="position-relative display-flex flex-row">…</div>
          </div>
… -flex.flex-row  div.react-codemirror2  div.CodeMirror.cm-s-material.CodeMirror-wrap
```

```css
element.style {
}
.cm-s-material.CodeMirror {                                            <style>
  background-color: #263238;
  color: #EEFFFF;
}
.CodeMirror {                                                          <style>
  position: relative;
  overflow: ▶ hidden;
  background: ▶ white;
}
.CodeMirror {                                                          <style>
  font-family: monospace;
  height: 300px;
  ~~color: black;~~
  direction: ltr;
```



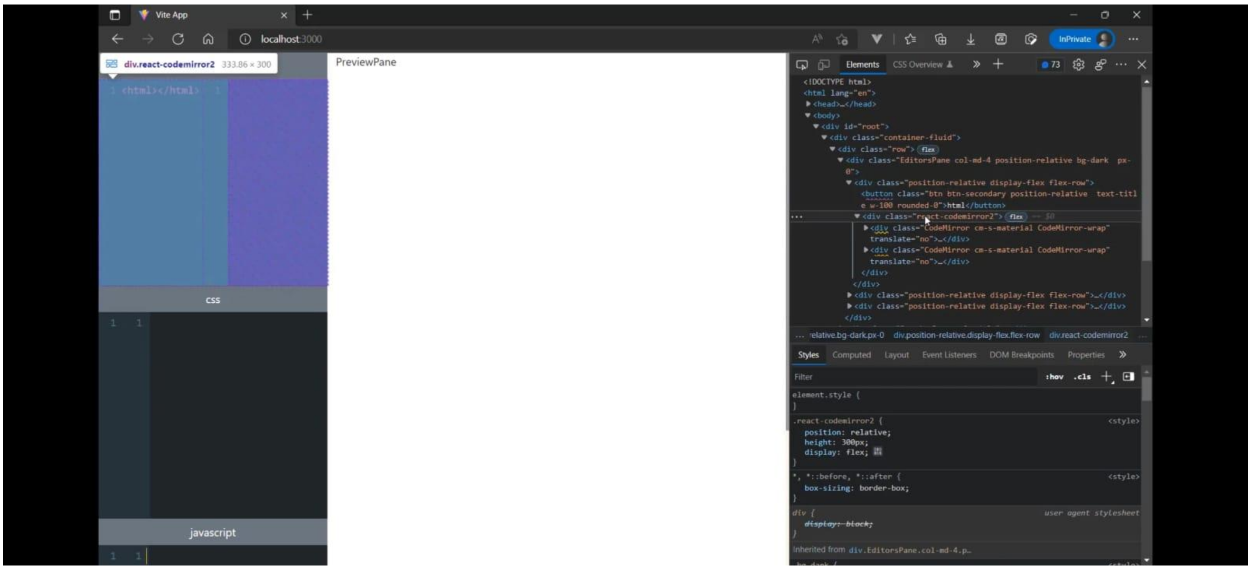
```css
.react-codemirror2 {
  position: relative;
  height: 41px;
}
```



```html
<html></html>
```

```html
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
 ▼<body>
   ▼<div id="root">
     ▼<div class="container-fluid">
       ▼<div class="row"> flex
         ▼<div class="EditorsPane col-md-4 position-relative bg-dark  px-0">
           ▼<div class="position-relative display-flex flex-row">
              <button class="btn btn-secondary position-relative  text-title w-100 rounded-0">html</button>
             ▼<div class="react-codemirror2"> == $0
               ▶<div class="CodeMirror cm-s-material CodeMirror-wrap" translate="no">…</div>
               ▶<div class="CodeMirror cm-s-material CodeMirror-wrap" translate="no">…</div>
              </div>
            </div>
           ▶<div class="position-relative display-flex flex-row">…</div>
           ▶<div class="position-relative display-flex flex-row">…</div>
          </div>
```

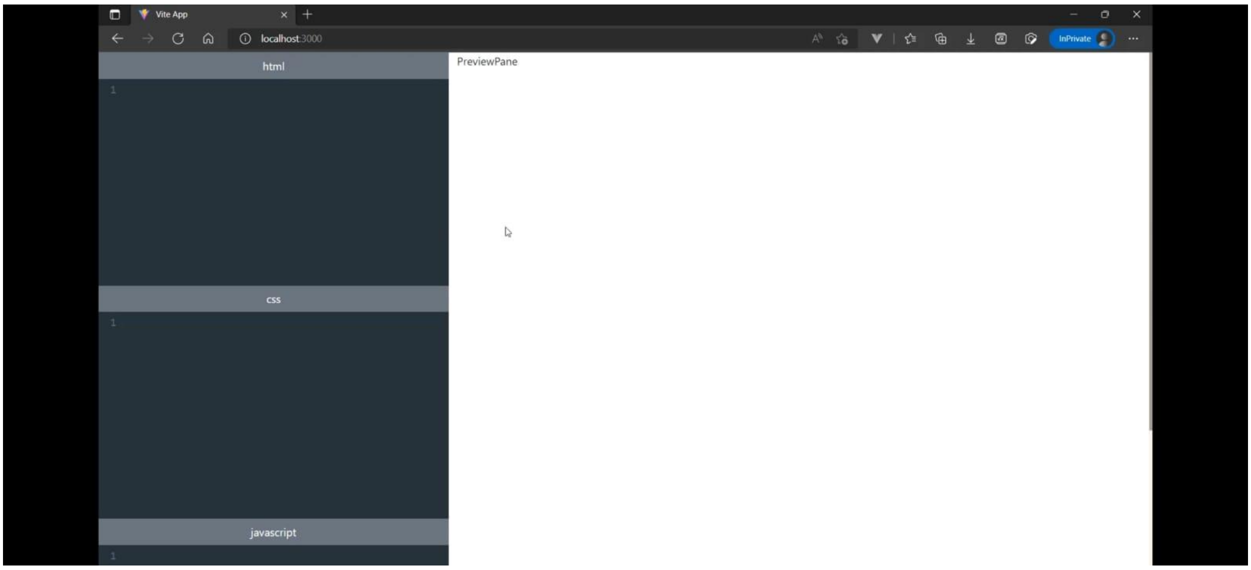
```css
element.style {
}
.react-codemirror2 {                                  <style>
  position: relative;
  height: 300px;
}
*, *::before, *::after {                              <style>
  box-sizing: border-box;
}
div {                                  user agent stylesheet
  display: block;
}
Inherited from div.EditorsPane.col-md-4.p…
.bg-dark {                                            <style>
```

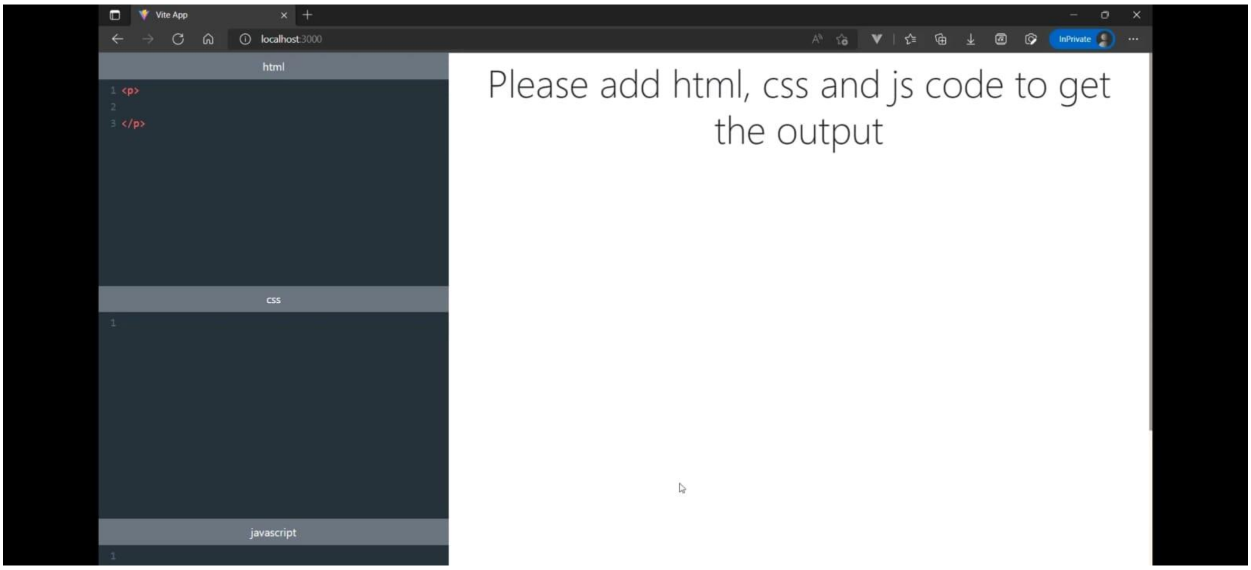


```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
 ▼<body>
   ▼<div id="root">
     ▼<div class="container-fluid">
       ▼<div class="row"> flex
         ▼<div class="EditorsPane col-md-4 position-relative bg-dark  px-
          0">
           ▼<div class="position-relative display-flex flex-row">
              <button class="btn btn-secondary position-relative  text-titl
              e w-100 rounded-0">html</button>
             ▼<div class="react-codemirror2"> flex == $0
               ▶<div class="CodeMirror cm-s-material CodeMirror-wrap"
                 translate="no">…</div>
               ▶<div class="CodeMirror cm-s-material CodeMirror-wrap"
                 translate="no">…</div>
              </div>
            </div>
           ▶<div class="position-relative display-flex flex-row">…</div>
           ▶<div class="position-relative display-flex flex-row">…</div>
          </div>
```

… relative.bg-dark.px-0 div.position-relative.display-flex.flex-row div.react-codemirror2

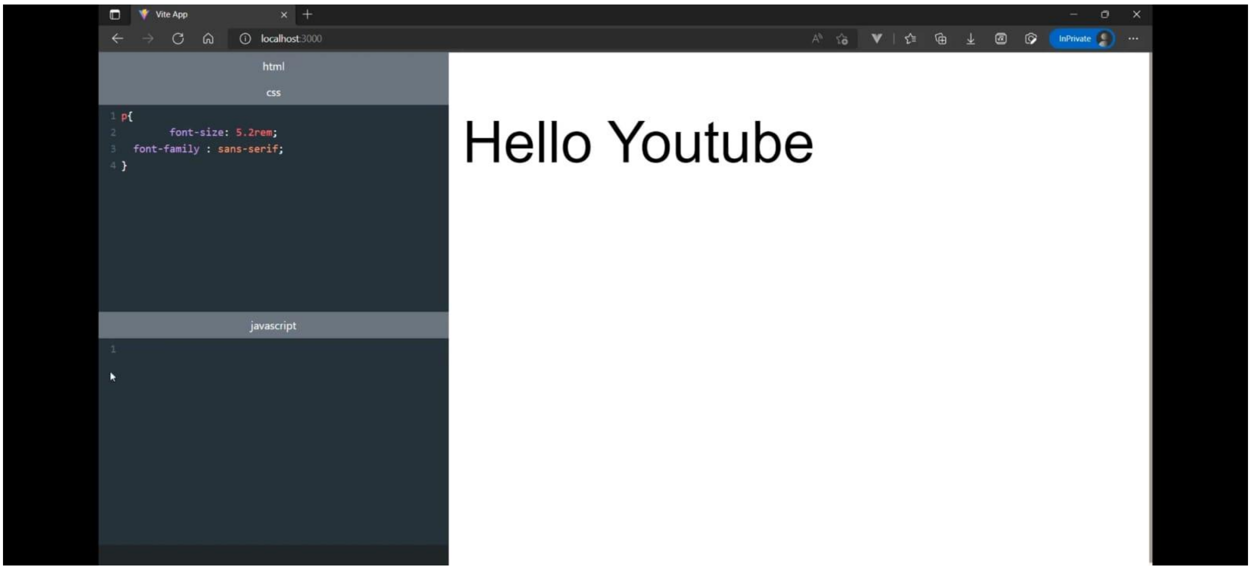
```
element.style {
}
.react-codemirror2 {                                  <style>
  position: relative;
  height: 300px;
  display: flex;
}
*, *::before, *::after {                              <style>
  box-sizing: border-box;
}
div {                                  user agent stylesheet
  ~~display: block;~~
}
Inherited from div.EditorsPane.col-md-4.p…
```





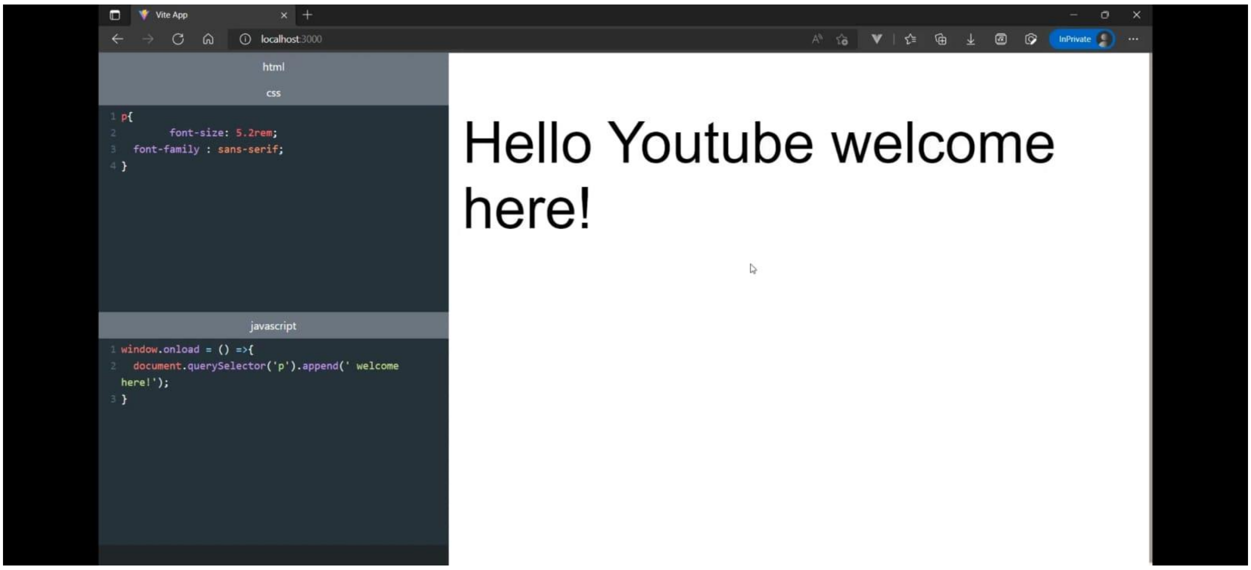
```
<p>

</p>
```

Please add html, css and js code to get the output



```
p{
        font-size: 5.2rem;
  font-family : sans-serif;
}
```

Hello Youtube



```css
p{
        font-size: 5.2rem;
  font-family : sans-serif;
}
```

```javascript
window.onload = () =>{
  document.querySelector('p').append(' welcome
here!');
}
```

Hello Youtube welcome here!

4.5 Testing

Tests can be conducted based on two approaches –

- Functionality testing
- Implementation testing

The texting method used here is Black Box Testing. It is carried out to test functionality of the program. It is also called 'Behavioural' testing.

4.5.1 Strategy Used

We have used the built-in test feature to uncover bugs and prevent regressions. To test our agent, we created test cases using the simulator to define test cases, then executed test cases as needed. A test execution verifies that agent responses have not changed for end-user inputs defined in the test case.

Chapter 5: Conclusion

Conclusion

5.1 Conclusion

In the end, we have:

- A code editor that can compile HTML, CSS AND JAVASCRIPT languages
- Interacting and Hosting APIs on RapidAPI.
- Using keyboard events in React using custom hooks
- Lots and lots of fun! ;)

At last, if you want to take the project to a deeper, here are some of the features that you could consider implementing:

- Login and Registration module – so that you can save your code in your own personal dashboard.
- A way to share code with other people over the internet
- Profile page and customizations.
- Pair programming on a single code snippet using Socket Programming and Operational Transformations.
- Bookmark favourite code snippets
- Custom dashboard of your code snippets (that are saved) - just like CodePen.

5.2 Limitations of the Work

A source-code editor can check syntax while code is being entered and immediately warn of syntax problems. A few source-code editors compress source code, typically converting common keywords into single-byte tokens, removing unnecessary whitespace, and converting numbers to a binary form. Such tokenizing editors later uncompress the source code when viewing it, possibly prettyprinting it with consistent capitalization and spacing .In this project, we've created a single screen for students to get edit and run their programs in single screen without switching the screen for different languages of front end. We have used HTML, CSS, JS and React.

Bibliography

[1] Aliv Faizal Muhammad, Dwi Susanto, Akhmad Alimudin, Farah Adila, Moh. Hasbi Assidiqi, Salim Nabhan, “Developing English Conversation chatbot using Dialogflow.”

[2] Sushil S. Ranavare, R. S. Kamath, “Artificial Intelligence based Chatbot for Placement Activity at College Using DialogFlow.”

[3] Sandeep A. Thorat, Vishakha Jadhav, “A Review on Implementation Issues of Rule-based Chatbot Systems.”

[4] N. M. Madhu Manjunath, S. Ravindra, “A Dialogflow-Based Chatbot for Karnataka Tourism.”

[5] Roberto Reyes, David Garza, Leonardo Garrido, Víctor De la Cueva, Jorge Ramirez, “Methodology for the Implementation of Virtual Assistants for Education Using Google Dialogflow.”

[6] Tubagus Prasetio Anwarulloh, Richi Dwi Agustia, “Development of the code by Einstein Application as a virtual teacher of physical learning in the house using Android based Google Dialogflow API.”

SOURCE CODE

```
1.<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8" />
  <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <meta name="theme-color" content="#000000" />
  <meta
    name="description"
    content="Web site created using create-react-app"
  />
  <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
  <!--
    manifest.json provides metadata used when your web app is installed on a
    user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
  -->
  <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
  <!--
    Notice the use of %PUBLIC_URL% in the tags above.
    It will be replaced with the URL of the `public` folder during the build.
    Only files inside the `public` folder can be referenced from the HTML.

    Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
    work correctly both with client-side routing and a non-root public URL.
    Learn how to configure a non-root public URL by running `npm run build`.
  -->
```

```
    <title>Code Editor</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>
```

```
2. {
  "short_name": "React App",
  "name": "Create React App Sample",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

```
3. # https://www.robotstxt.org/robotstxt.html
User-agent: *
Disallow:
```

```
4. import React, { useState, useEffect } from 'react';
import Editor from './Editor'
import useLocalStorage from '../hooks/useLocalStorage'

function App() {
  const [html, setHtml] = useLocalStorage('html', '')
  const [css, setCss] = useLocalStorage('css', '')
  const [js, setJs] = useLocalStorage('js', '')
const [srcDoc, setSrcDoc] = useState('')

useEffect(() => {
 const timeout = setTimeout(() => {
  setSrcDoc(`
   <html>
    <body>${html}</body>
    <style>${css}</style>
    <script>${js}</script>
   </html>
  `)
 }, 250)

 return () => clearTimeout(timeout)
}, [html, css, js])

return (
 <>
  <div className="pane top-pane">
   <Editor
    language="xml"
    displayName="HTML"
    value={html}
    onChange={setHtml}
   />
   <Editor
    language="css"
```

```
     displayName="CSS"
     value={css}
     onChange={setCss}
    />
    <Editor
     language="javascript"
     displayName="JS"
     value={js}
     onChange={setJs}
    />
   </div>
   <div className="pane">
    <iframe
     srcDoc={srcDoc}
     title="output"
     sandbox="allow-scripts"
     frameBorder="0"
     width="100%"
     height="100%"
    />
   </div>
  </>
 )
}

export default App;
```

5. import React, { useState } from 'react'

```
import 'codemirror/lib/codemirror.css'
import 'codemirror/theme/material.css'
import 'codemirror/mode/xml/xml'
import 'codemirror/mode/javascript/javascript'
import 'codemirror/mode/css/css'
import { Controlled as ControlledEditor } from 'react-codemirror2'
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
import { faCompressAlt, faExpandAlt } from '@fortawesome/free-solid-svg-icons'

export default function Editor(props) {
 const {
  language,
  displayName,
  value,
  onChange
 } = props
 const [open, setOpen] = useState(true)

 function handleChange(editor, data, value) {
  onChange(value)
 }

 return (
  <div className={`editor-container ${open ? '' : 'collapsed'}`}>
   <div className="editor-title">
    {displayName}
    <button
```

```
     type="button"
     className="expand-collapse-btn"
     onClick={() => setOpen(prevOpen => !prevOpen)}
    >
     <FontAwesomeIcon icon={open ? faCompressAlt : faExpandAlt} />
    </button>
   </div>
   <ControlledEditor
    onBeforeChange={handleChange}
    value={value}
    className="code-mirror-wrapper"
    options={{
     lineWrapping: true,
     lint: true,
     mode: language,
     theme: 'material',
     lineNumbers: true
    }}
   />
  </div>
 )
}
6. body {
 margin: 0;
}

.top-pane {
 background-color: hsl(225, 6%, 25%);
}

.pane {
 height: 50vh;
 display: flex;
}

.editor-container {
 flex-grow: 1;
 flex-basis: 0;
 display: flex;
 flex-direction: column;
 padding: .5rem;
 background-color: hsl(225, 6%, 25%);
}

.editor-container.collapsed {
 flex-grow: 0;
}

.editor-container.collapsed .CodeMirror-scroll {
 position: absolute;
 overflow: hidden !important;
}

.expand-collapse-btn {
 margin-left: .5rem;
 background: none;
 border: none;
 color: white;
 cursor: pointer;
}

.editor-title {
 display: flex;
 justify-content: space-between;
 background-color: hsl(225, 6%, 13%);
 color: white;
 padding: .5rem .5rem .5rem 1rem;
 border-top-right-radius: .5rem;
 border-top-left-radius: .5rem;
}

.CodeMirror {
 height: 100% !important;
}

.code-mirror-wrapper {
 flex-grow: 1;
 border-bottom-right-radius: .5rem;
 border-bottom-left-radius: .5rem;
 overflow: hidden;
}
```

```
7. import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './components/App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

```
8. # See https://help.github.com/articles/ignoring-files/ for more about ignoring files.

# dependencies
/node_modules
/.pnp
.pnp.js

# testing
/coverage

# production
/build

# misc
.DS_Store
.env.local
```

```
.env.development.local
.env.test.local
.env.production.local

npm-debug.log*
yarn-debug.log*
yarn-error.log*
```

```
9. {
  "name": "codepen-clone",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@fortawesome/fontawesome-svg-core": "^1.2.30",
    "@fortawesome/free-solid-svg-icons": "^5.14.0",
    "@fortawesome/react-fontawesome": "^0.1.11",
    "@testing-library/jest-dom": "^4.2.4",
    "@testing-library/react": "^9.5.0",
    "@testing-library/user-event": "^7.2.1",
    "codemirror": "^5.57.0",
    "react": "^16.13.1",
    "react-codemirror2": "^7.2.1",
    "react-dom": "^16.13.1",
    "react-scripts": "3.4.3"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

10. # See https://help.github.com/articles/ignoring-files/ for more about ignoring files.

```
# dependencies
/node_modules
/.pnp
.pnp.js

# testing
```

```
/coverage

# production
/build

# misc
.DS_Store
.env.local
.env.development.local
.env.test.local
.env.production.local

npm-debug.log*
yarn-debug.log*
yarn-error.log*
```