

Online Code Editor

A PROJECT REPORT

Submitted By

**Rashika Garg
(2000290140100)**

**Submitted in partial fulfilment of the
Requirements for the Degree of**

MASTER OF COMPUTER APPLICATIONS

**Under the Supervision of
Dr Amit Kumar Gupta
Associate professor**



Submitted to

**DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206**

(JAN 2022)

CERTIFICATE

Certified that **Rashika Garg (200290140100)** have carried out the project work having “**Title of Report** Online Code Editor” for Master of Computer Applications from Dr A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

Rashika Garg (200290140100)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Dr Amit Kumar Gupta
Associate professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Signature of Internal Examiner

Signature of External Examiner

Dr Ajay Shrivastava
Head, Department of Computer Applications
KIET Group of Institutions, Ghaziabad

ABSTRACT

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 named code editor. The purpose of this research is to design and develop a real-time code editor application using web socket technology to help users collaborate while working on the project. This application provides a feature where users can collaborate on a project in real-time.

The authors using analysis methodology which conducting on a study of the current code editor applications, distributing questionnaires and conducting on literature study. Coder is a web application that provides workspace to writing, perform, display the results of the code through the terminal, and collaborate with other users in real-time.

The application main features are providing workspace to make, execute and build the source code, real-time collaboration, chat, and build the terminal. This application supports C, C++, and Java programming languages.

ACKNOWLEDGEMENTS

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Dr. Amit Kumar Gupta** for his guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Rashika Garg

(200290140100)

List of Chapters

Chapter 1 - Introduction

- 1.1 Project description
- 1.2 Project Scope
- 1.3 Hardware / Software used in Project

Chapter 2 Feasibility Study

- 2.1 Technical feasibility
- 2.2 Operational Feasibility
- 2.3 Behavioral Feasibility
- 2.4 Operational Feasibility

Chapter 3 Database Design

- 3.1 Database Tables
- 3.2 Flow Chart
- 3.3 Use Case Diagram
- 3.4 Sequence Diagram
- 3.5 Collaborative Diagram

.....

.....

Chapter 4 Form Design

- 4.1 Input / Output Form (Screenshot)
- 4.2

Chapter 5 Coding

- Module wise code

Chapter 6 Testing

- 6.1 Unit testing
- 6.2 Integration testing
- 6.3 System testing
- 6.4 Functional testing
- 6.5 Performance testing
- 6.6 Acceptance testing
- 6.7 Alpha testing

CHAPTER 1

INTRODUCTION

1.1 Project Description

A **online-code editor** is a text editor program designed specifically for editing source code of computer programs. It may be a standalone application or it may be built into an integrated development environment (IDE) or web browser. Source-code editors are a fundamental programming tool, as the fundamental job of programmers is to write and edit source code.

Source-code editors have characteristics specifically designed to simplify and speed up typing of source code, such as syntax highlighting, indentation, autocomplete and brace matching functionality. These editors also provide a convenient way to run a compiler, interpreter, debugger, or other program relevant for the software-development process. So, while many text editors like Notepad can be used to edit source code, if they don't enhance, automate or ease the editing of code, they are not *source-code editors*.

Structure editors are a different form of source-code editor, where instead of editing raw text, one manipulates the code's structure, generally the abstract syntax tree. In this case features such as syntax highlighting, validation, and code formatting are easily and efficiently implemented from the concrete syntax tree or abstract syntax tree, but editing is often more rigid than free-form text. Structure editors also require extensive support for each language, and thus are harder to extend to new languages than text editors, where basic support only requires supporting syntax highlighting or indentation. For this reason, strict structure editors are not popular for source code editing, though some IDEs provide similar functionality.

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 pretty printing it with consistent capitalization and spacing. A few source-code editors do both.

1.2 Project Objective

Create an online code editor for HTML, CSS and JS code snippets using HTML, CSS and JavaScript. The code editor's functionality will be similar to that of codepen.io

Online code-editor is a tool that resides on a remote server and is accessible via browsers. Some online code editors have basic features like syntax highlighting or code completion similar to text editors while others are like complete [IDEs](#).

For any developer, be it amateur or professional, often the liberty of using a local code editor may be unavailable. As online code-editors are fast, efficient and greatly popular, it is a familiar tool among developers. If you have used one, ever wondered how it can be made? This module will guide you through the process that can be followed to build your own code-editor for HTML, CSS and JS code snippets. Implementing the project will add an immense value to your profile.

This project is a good start for beginners, a new idea for intermediates and a refreshing hobby project for professionals. It involves the basic use of all of HTML, CSS and JS languages. Reason for following this tech-stack is that these languages are easy to use and are also fast in terms of execution time.

You are free to use JavaScript instead of JavaScript but it is recommended to use JavaScript as it is lighter to [implement](#).

1.3 Project Scope

We can segregate the product architecture based on tools used:

- HTML, CSS - Site user interface (site's skeleton layout)
- JavaScript - Site's core functionality
- GitHub - To host the web application

High-Level Approach

- Creating the structural aspect of code-editor using HTML, CSS (i.e. without any animations and features)
- Implementation of core functionalities and other small features using a JavaScript library
- Publish to GitHub and finally host your deployed code-editor

Applications

Some of the applications of online code editors are:

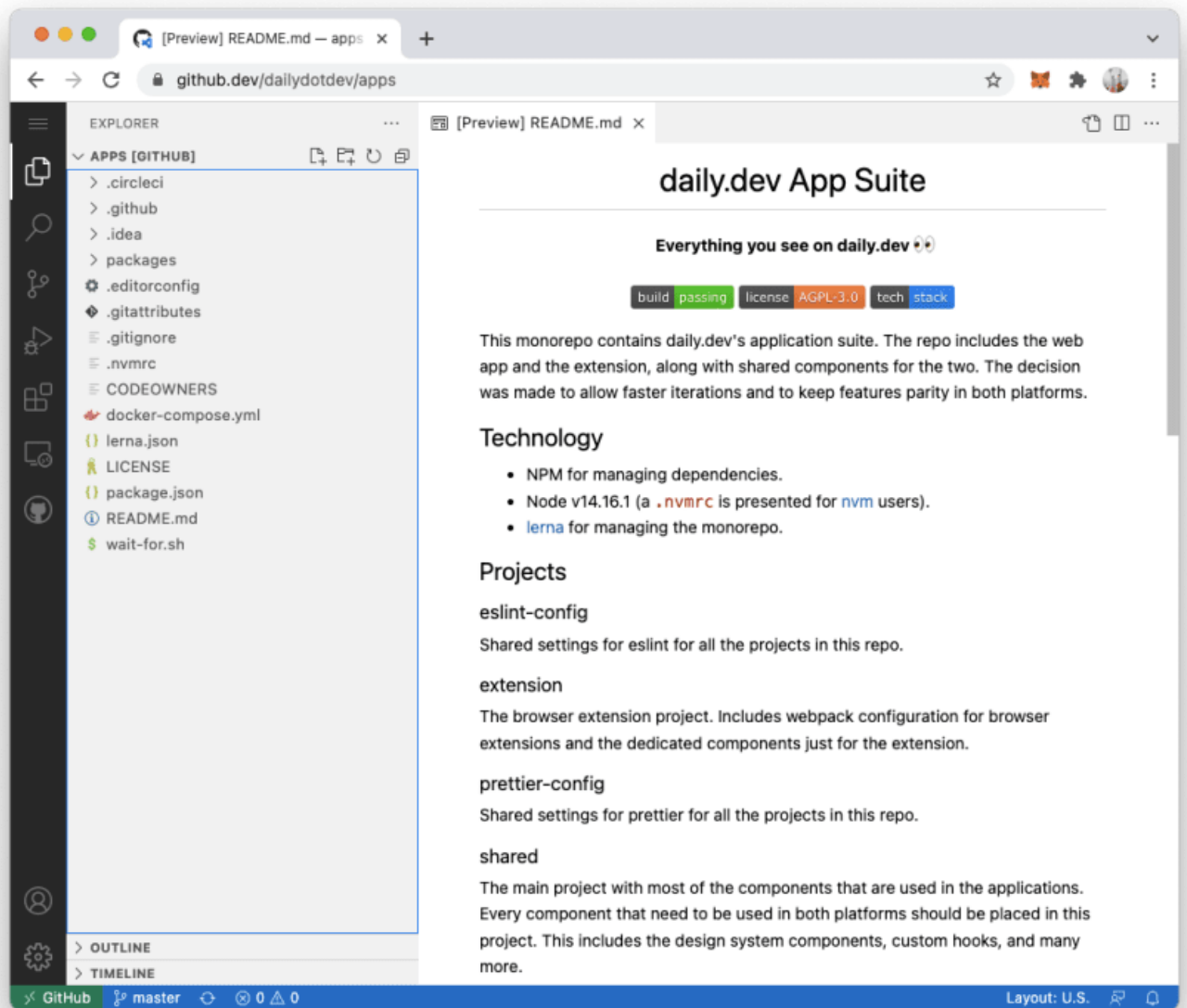
- For online interviews/hiring: With online code editors, you can do interviews with ease. It helps to see candidates' approach towards solutions and work with code.
- Prototyping: You can validate your ideas and get instant feedback from users as sharing and collaboration options are available and saves you from the pain of local setup.

1.4 Advantages of Online Code Editor

- The main components to be created are HTML, CSS, JS buttons (render upon toggling) and a 'run' button. This can be done using simple [lists](#) in HTML.
- Wrap all the components and containers in appropriate [classes](#) and [divisions](#) which will then be used for references in styling. This step must be done in the future too, when the need arises.
- Create (three) containers in which the HTML, CSS, JS code snippets will be inputted by the user. For this use the text area HTML tag.
- Finally wrap the 'run' button with the [iframe](#) [element](#) (can be done later). [iframe](#) element is used to render the desired output as a webpage within a webpage.

It gives a user-friendly framework for different types of programming languages, such as Visual Basic, Java, and PowerBuilder. It contains a complete package including a source code editor, build automation tools, debugger, compiler, interpreter and other features such as syntax highlighting, support for the version control system, auto-completion of keywords (where you start typing the name of a function or variable, it fills out the rest of the name), etc. IDE integrates project files, which you work on and includes version control of source files such as git repository.

And we are welcomed by the full repo as we would get in VS Code.



Pretty cool, right?

So what can we do with this?

- Review pull requests online with ease. You can simply press the `.` in a PR to open it in the editor for a more detailed overview.
- Super easy to search throughout the codebase. Yes, click-through works!
- It's super fast.
- Make changes and PRs directly online
- Extensions are available

All and all, it's a super powerful way to edit code.

The main downside is, of course, not being able to run it.

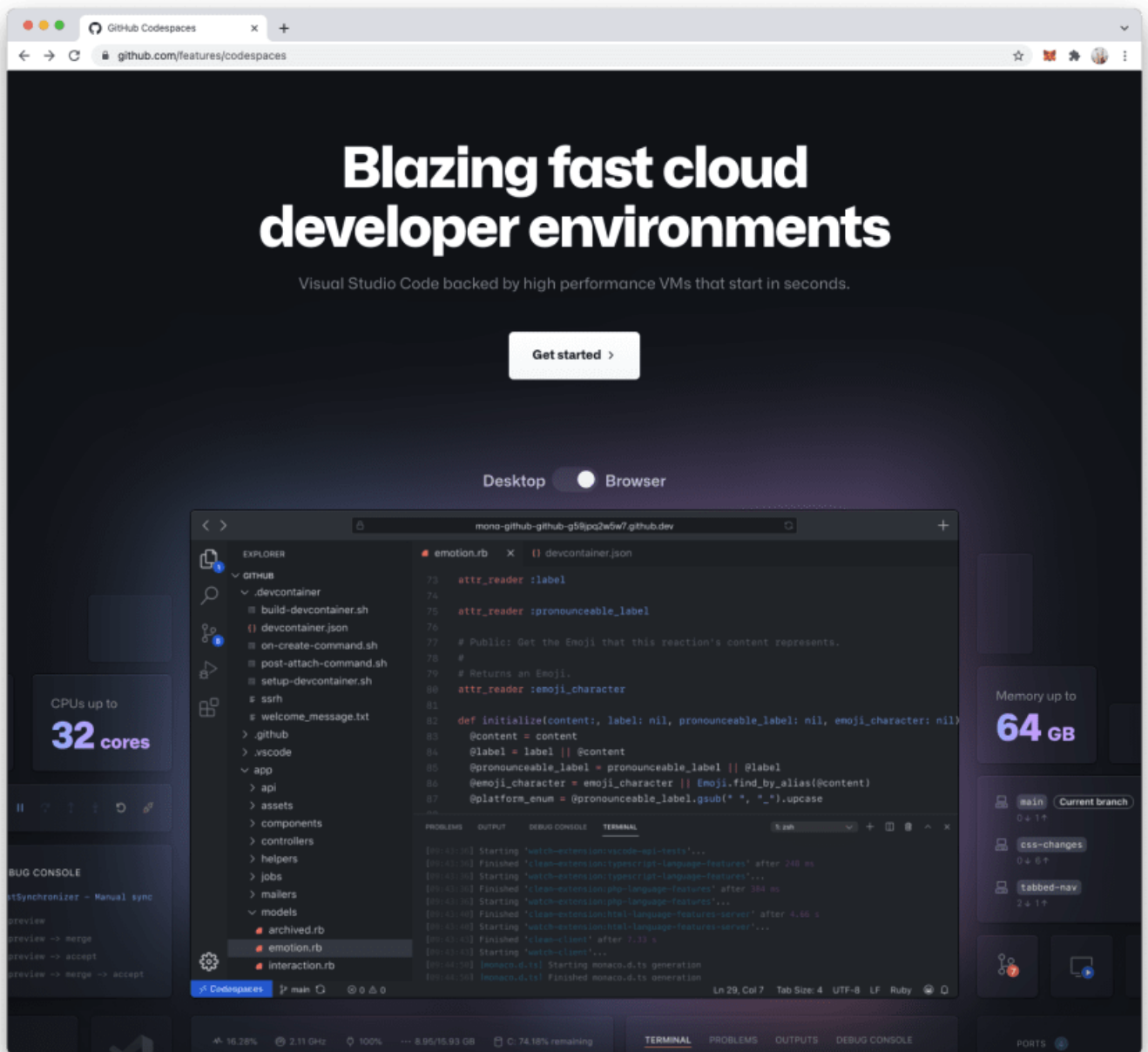
But for PR reviews, this is a good win!

GitHub Code spaces

GitHub recently also launched GitHub Code spaces, which is a blazing fast cloud environment.

The look and feel are the same, but you are running on a VM.

The cool part about Code spaces is that you can run your code there.

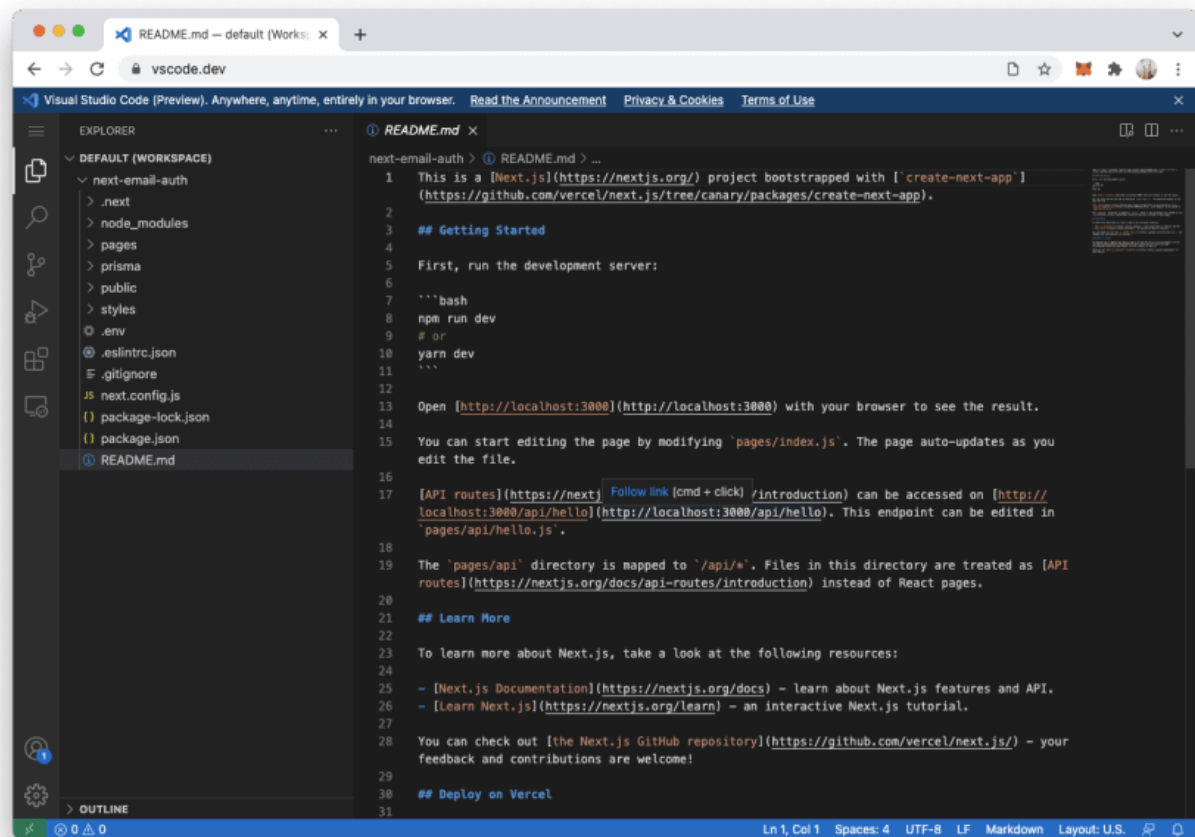


Visual Studio Code in the Browser

More recently, we got a new addition to the online editor family, being VS code themselves.

To open a new editor, you can type `vscode.dev` in your browser, and a new editor will spool up.

The cool part is that this has wider use, as it can even load local files!



It can also open GitHub links and even azure links!

You can simply take any link to those repos and prefix them

Powerful if you ask me!

Some pro's of using the vs code one would be:

- Local file opening
- New projects from scratch, quickly share some project structure
- Azure dev-ops environment
- Again, many amazing extensions are available

The downsides are, for now, no way to run and debug.

What about the future?

We started this article by asking if these editors are the future?
And to be honest with you, I do think it's very viable.

Our hardware becomes less and less critical (meaning whatever we have on it locally).

We can sign in to some online tools and take off from where we left.

This makes total sense in the current day and age.

There are some kinks to iron out for it to be entirely a solution on its own.

For me, those are:

Option to run docker images on the cloud (Codespaces/Vercel/etc)

Peer programming online 👁👁

Debug systems integrated

I'm sure these are even possible with some extensions and very keen to see what this will bring over time!

1.5 Hardware/Software Requirements

Processor	I5
RAM	4 GB
Internet Speed	30 Mbps
Connection Type	Wired or Wireless
Hard drive	1 TB

Software Used

Operating System	Windows, Linux, Mac Os
Browser	Google Chrome(Recommended), Mozilla Firefox, Safari

Chapter 2 Feasibility Study

Feasibility Study

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 named code editor. The purpose of this research is to design and develop a real-time code editor application using web socket technology to help users collaborate while working on the project. This application provides a feature where users can collaborate on a project in real-time. The authors using analysis methodology which conducting on a study of the current code editor applications, distributing questionnaires and conducting on literature study. CodeR is a web application that provides workspace to writing, perform, display the results of the code through the terminal, and collaborate with other users in real-time. The application main features are providing workspace to make, execute and build the source code, real-time collaboration, chat, and build the terminal. This application supports C, C++, and Java programming languages.

2.1 Technical Feasibility

The technological development trend in software engineering has been improving, where the design of software began move from the desktop to the web. Nowadays, many IDE (Integrated Development Environment) applications has been made, such as Eclipse, Visual Studio, etc.¹ , but IDEs which based on desktop still have significant disadvantages such as long time for configuration and installing the plug-in needed for IDE to run the project. This problem could be a huge waste of time when there are many devices that have to be configured² . Many software applications have been run in the cloud, and use a web browser as a user interface that allows ubiquitous access, instant collaboration, and avoid installation and configuration on desktop computers² . One of the technologies used for instant collaboration is single IDE (like pair programming). Pair programming is the practice of having two programmers access and work on the same code in a single development environment³ . In pair programming, programmers have the abilities to create, edit and delete source code in real-time. Pair programming could solve the synchronization problem of program code in order to remain valid, and whenever the code changes any programmer who is working on the same project could see the one who changed the code. Collaborative technologies could help programmers work together while fixing bugs or discuss the program in the same single environment but in different geographical area. Therefore, it needs to make an application that can improve performance while creating program such as real-time collaboration, create, execute and display the result of the program using terminal.

Real-time Communication: Real-time Communication (RTC) is merging of communication and collaboration systems, which combined of communication technologies, like Voice-over-IP (VoIP) telephony and instant messaging, and various collaborative application^{4,5}. RTC technologies consist of four interconnected building blocks; consist of unified communications, presence awareness, contextualization and E-Collaboration portfolio RTC systems usually enable two-person communication and support multi-person conference. By providing and integrating a range of synchronous communication media in one integrated environment, RTC systems allow users to collaborate in real-time, for example editing a document, voice call, multi-person video conference. RTC technology could help to solve the synchronization problem, especially when working in a team. RTC systems offer flexibility and interacting remotely with other users, also it has been facilitated and improved in terms of interaction and communication that could help construct the project more efficiently

- 1.1. State of the Art State of the art of this research is to design a code editor application which has the ability to do a collaboration while working on specific file, syntax checking, run and build those source code through terminal, and users could communicate with the others using chat as media. This application provides workspace where users can create, edit, run and build source code which has been written before and some useful feature like auto complete in C, C++, and Java. For users who want to export the source code files to desktop or import project files from desktop could use upload and download features. Hopefully, this application could be able to help programmer to do some project collaborate in real-time with others, and increase project development performance.

2.2 Operational Feasibility

Real-time Collaborative Programming: While working on development projects, any programmers working on the project by team. Any programmer who has the access to the project can create, change, and add code inside the same project file. So synchronization process is required between programmers to avoid code duplication, and to solve this synchronization problem integrated real-time collaboration is needed in a single environment. The Integrated Development Environment (IDE) is focused to provide collaborative setting for programming teams which has the ability to do real-time text editing, run and build code, chat, and various other features. The ability for editing text in real-time allows multiple users to work together while editing a document and display the changes directly to other users who has the ability to access the same document. There are a number of free applications that support real-time text editing feature, such as Google Docs. This feature not only makes excellent collaboration for common users, but can also very effective in programming. There are also a wide variety of web-based systems that provide collaboration. For example is EtherPad that allows real-time text editing. Ace, CodeMirror are web-based text editing component which designed to be embedded into the IDE or application^{3,7}.

Project or software development requires the coordination and collaboration between programmers, so that the collaboration systems are very useful to improve the efficiency in making project. The effectiveness of collaboration in programming can improve the productivity and quality of project or software⁸. Collaborative programming in real-time support programmers to work on the same programming file. Real-time system will automatically combine the code typed by a programmer without manual command from the programmer (such as update, commit)⁸. Multiple programmers enable to access and edit the same source code directory, even at the same time. In real-time sessions programmer can collaborate with other programmers by joining and leaving a session of real-time during collaborative programming. There are steps to join and leave the session using the join protocol with two-way client to receive a request from a new client who wants to join. To accommodate the new client request, a distributed join-protocol was designed with the following message: • JOIN: new client send a request to the session manager for request to join an existing real-time session. • START: session manager send to all existing clients to inform the start join-protocol procedure to receives a request from a new client. • READY: all existing clients send information to session manager about readiness for entering the state. • FINISH: session manager send information to all existing clients about the completion of all procedures.

Integrating Collaborative Program Development and Debugging within a Virtual Environment: IDEs (Integrated Development Environments) are most used tools in the activity of the programmer. To support software development and software engineering, IDE was developed in order to support collaboration in real-time, where programmers can work together to design, discuss and share in the same software development. A collaborative integrated environment allows programmers to share programming-related tasks. One of the tools that implement collaborative IDE is ICI (Idaho Collaborative IDE)⁹. ICI allows programmers who are in different locations can collaborate in various software development activities in real-time. ICI combines technology between synchronous collaboration and a real-time debugger. The implementation of collaborative technologies is not only used in the world of programming, but also in education. Education of computer science distance requires a more collaborative IDE integration in terms of 10: 1) Supporting real-time collaboration in compiling, linking, running and debugging sessions 2) Providing a technology where developers can more easily communicate either by text or voice Collaborative environment enables programmers to communicate and view the activities of other programmers in the same environment. So that programmers can chat via text or VoIP with other programmers or other project teams in real-time. To work within the same development environment, programmers can invite each other to join in a collaborative IDE session, where they collaborate together in program design, coding or debugging⁹.

ICI supports collaborative software development tasks. The implementation of ICI could be used in a virtual laboratory to help teaching assistants and tutors in teaching a group of students on a remote computer. Below is the flow of information in a distributed team environment ICI that shows two-way communication such as compilers or debuggers.

Real-Time Collaborative Coding in a Web IDE: Now IDE is not only a tool that helps programmers in making projects or Software, but the IDE now developed into tools that could help programmers communicate and collaborate with other programmers to make project more efficiently. One of the IDE that has been created to support real-time collaboration is Coll abode. Coll abode is a collaborative web-based IDE that supports Java programming language. In Coll abode when multiple users make changes, the result of the changes will be distributed to other users immediately without control requires programmers manually⁷ . In addition, more than one user in the same project module can access this application simultaneously. Coll abode made in order to improve the quality of collaboration and project produced. To be able to collaborate on projects, user only need to visit the same URL where the other users are in. Coll abode use Ether Pad to support collaboration among multiple editors. To manage the projects, Coll abode using Eclipse that provides syntax highlighting, compile errors and warnings, continuous compilation, code formatting and refactoring, and the execution of the program code.

There are three interesting models used for close synchronous collaboration⁷ : Test Driven Programming, Micro Outsourcing, and Mobile Instructor. The three models above give much different collaboration between one programmer to another programmer. Each of them has different ways such as using pair programming, side-by-side programming, and other collaborative models.

2.3 Behavioral Feasibility

Coder Features Algorithm: Coder use facebook plugin for login to access the application more easily and integrate with social network to collaborate each other. Coder supports C, C++ and Java as programming languages that users can choose as a main programming language. Users can run and build program to find out the results of the code program. Users can download the source code program along with their parent folders that has been created. Users can do collaboration to create a project together with other user in real-time. Users can communicate with other users who exist in the same project using chat as media. Users can send email to ask something or inform about bugs present in the website. When a user successfully logs into the application, the first page displayed is the workspace page. In Figure 2 explains our algorithm that after the user logged in it will display all the projects, folders, and files that have been made (if any). Project will be displayed in the file directory on the left workspace. If a new user, then the file directory will appear empty. This application provides real-time connections so that any changes that have been made will always be updated simultaneously.

2.4 Operational Feasibility

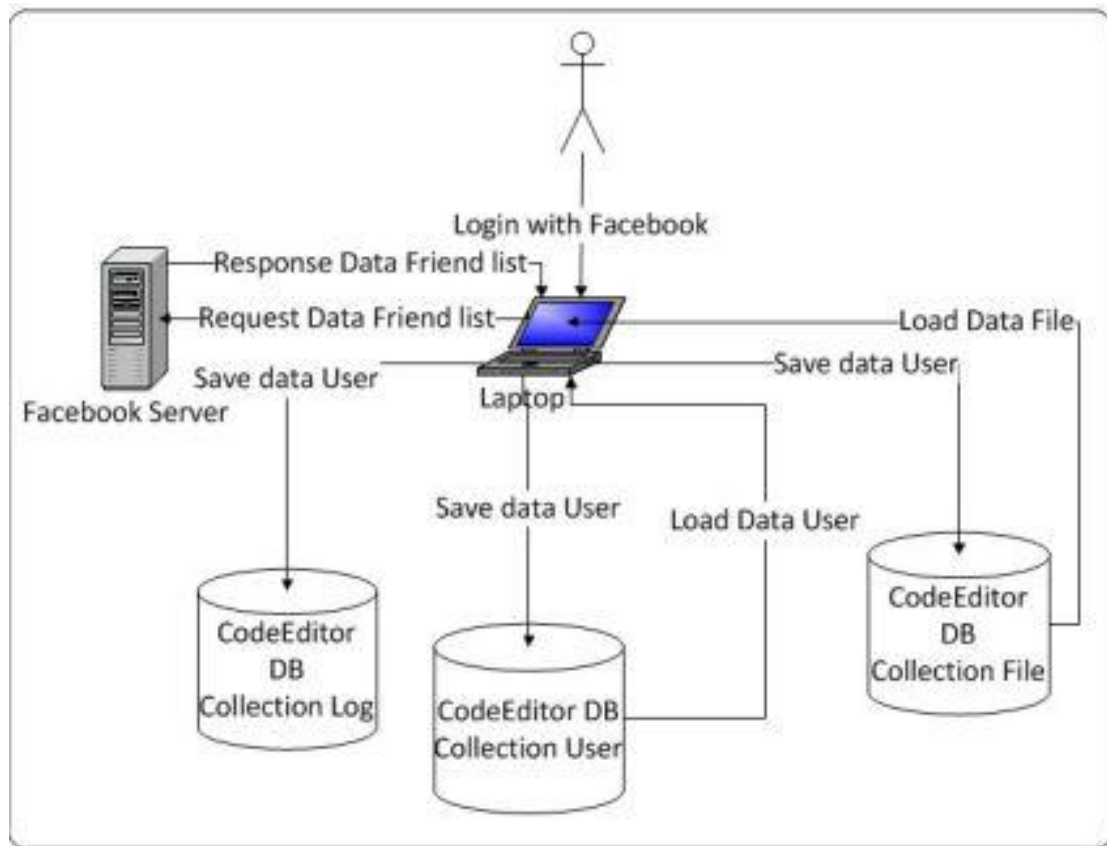
The collaboration in real-time allows a group of users to access and edit the same document at the same time over a connected network. In order to connect in real-time it is needed consistency of data to be accessed at the same time, because each user operation will be executed directly without delay. To ensure consistency of copies of the shared data in collaborative real-time group-ware used operational transformation algorithm.

Operational transformation (OT) become a methods used to maintain consistency of copies of shared data on group editors^{13, 14}. Operational transformation approach explains the two main components¹⁵: the integration algorithm and transformation functions. Integration algorithm used to receive, transmit and execute the operation. While transformation function used to combine two concurrent operations are defined on the same state. Received operations would transform according to the local concurrent operations and then executed in operational transformation¹⁵. In Figure 9 illustrates the effect of the transformation function, which when op2 received on site1 then op2 must be transformed according to the op1. Position the insertion op2 is incremented because op1 has been insert f before s in the state "effect". Furthermore, op'2 executed on site1 and generates state "effects". One of the implementation of operational transformation algorithm is the word processor application. Word processor application has a complex structure and comprehensive editing operations¹⁴. In addition to supporting the operation object creation and deletion, word processor application also supports updating the attributes of existing objects.

As for collaborative word processor application it is important to have the ability to update collaboratively, thus speeding up the creation of documents. To support the concurrent execution of update operation is used extension operational transformation. For example the result of this method is a Co word project that aims to convert MS Word into a real-time collaborative word processor without changing the source code.

Operational transformation has a component that is divided into two layers: the high-level transformation control algorithms, and the low-level transformation functions¹⁴. The strategy used in the extension of OT is to maintain a high-level control algorithm unchanged, but it adds a new update-related transformation functions. In this way makes it possible to reduce the complexity and localize the extension. Update operation is different with Insert and Delete operation, where the update has no effect on the linear address space, while the insert and delete change the linear address space of document. When Insert transformed to concurrent Update, no change on the Insert. However, when Update is transformed to Insert, the Update's position will be increased by one if the Update's position greater than or equal to the Insert position.

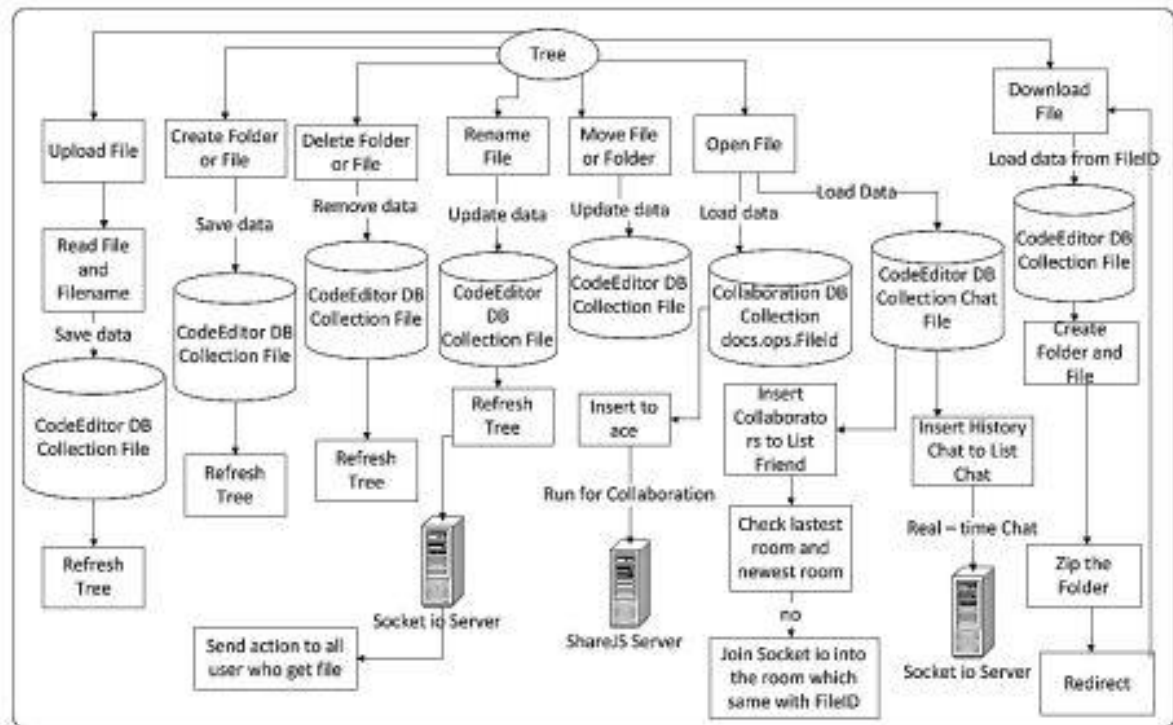
Application Architecture: User have to log in first to be able accessing the workspace. It is shown in the Figure 7. User data will be checked first whether it has been stored previously in the database. If the data exist then new log data will be saved into Collection logs, if not then new data will be created in user, file, and log Collection. Furthermore, the user will be redirected to the workspace page; project file data will be loaded from the user Collection and file database.



Facebook friends list will be loaded after the user redirected into workspace. This friend's list will be used to add new collaborators, and also new collaborators who have been invited will have the ability to communicate with another users via chat. Fig. 7. Login Architecture Figure 8 illustrates the architecture when a user sends a chat to other users and adds collaborator to join the same file project. When the user typing a message and press the enter key, the JSON data will be sent and then the chat data is saved into the chat Collection. After that the data will be sent to socket.io server and the data is returned to all users who have been invited previously. To collaborate with other users, the user could search for friends name who are registered in Facebook using Facebook auto complete engine to retrieve user data.

After getting the user id, data will be inserted into file Collection and will be forwarded into socket.io server. Finally the collaborator will be put in the same file project User can click the button placed at the left corner of the workspace page to access the folder sand the files in the workspace, and then the folders and the file will be displayed in the form of a tree. This is shown in Figure 9. Action is provided such as create, delete, rename, move the folder or file, open, download, and upload files.

Figure 8 illustrates the architecture when a user sends a chat to other users and adds collaborator to join the same file project. When the user typing a message and press the enter key, the JSON data will be sent and then the chat data is saved into the chat Collection. After that the data will be sent to socket.io server and the data is returned to all users who have been invited previously. To collaborate with other users, the user could search for friends name who are registered in Facebook using Facebook auto complete engine to retrieve user data. After getting the user id, data will be inserted into file Collection and will be forwarded into socket.io server. Finally the collaborator will be put in the same file project User can click the button placed at the left corner of the workspace page to access the folder sand the files in the workspace, and then the folders and the file will be displayed in the form of a tree. This is shown in Figure 9. Action is provided such as create, delete, rename, move the folder or file, open, download, and upload files.



Collaboration Features This application provides a collaboration with the user in real-time as shown in Figure 10. The user can invite other users to collaborate working on the same project through a friend list in the box on the right. After receiving an invitation to join another user in the same project, the user can communicate via chat. If a user is logged on via Facebook account, user can invite friends who are in their friend list. While login by anonymous, users will automatically connects with other users who used anonymous account. In Figure 10 shown the chat box that displays the user name, date, and there is a search feature to search friends name who would like to be invited to collaborate or communicate via chat.

5. Discussion To evaluate the application the authors use ab (Apache HTTP Server Benchmarking Tool) or more commonly called Apache Benchmark. Apache Benchmark is a tool to measure the Apache Hypertext Transfer

requests per second that is capable of Apache. The type of hardware that will greatly affect the performance of the webserver is RAM. The most basic

thing, Apache Benchmark could perform load successive and/or concurrent

load tests on a web page. It also could test the output such as time per

request, concurrency level, transfer rate, failed request, total transferred,

the time taken for the test, complete request, writing errors, HTML

transferred, requests per second. Apache Benchmark provides many

options such as -n (requests) for the number of requests for benchmarking

sessions and -c (concurrency) for the number of multiple requests to

perform at a time. Fig. 10. Collaborate between web and mobile user This

test hits the URL 1000 times (-n 1000) with maximum concurrency of 1000

simultaneous requests (-c 1000). Tests conducted using two kinds

specification of computer and the following two specifications show in

Table 2. Table 2. Specification of the computer used TYPE PROCESSOR RAM

SPEC 1 CPU Pentium Dual Core E2180 4 GB SPEC 2 Core i7 2600 8 GB

Results of experiments with computer spec 1: • Total data transferred is

3575000 bytes for 1000 requests • Test completed in 28.842 seconds. No

failed requests • Requests per seconds: 34.67 • Time per request:

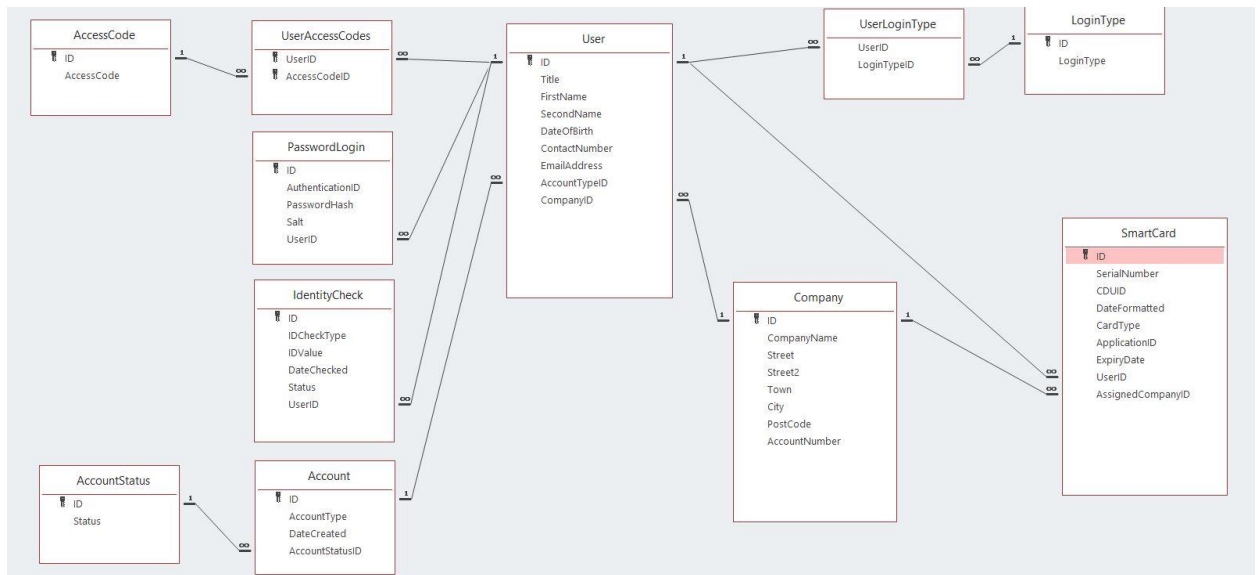
28841.650 milliseconds (for 1000 concurrent requests). So across all

requests it is $28841.650 \text{ ms}/100 = 28842 \text{ ms}$

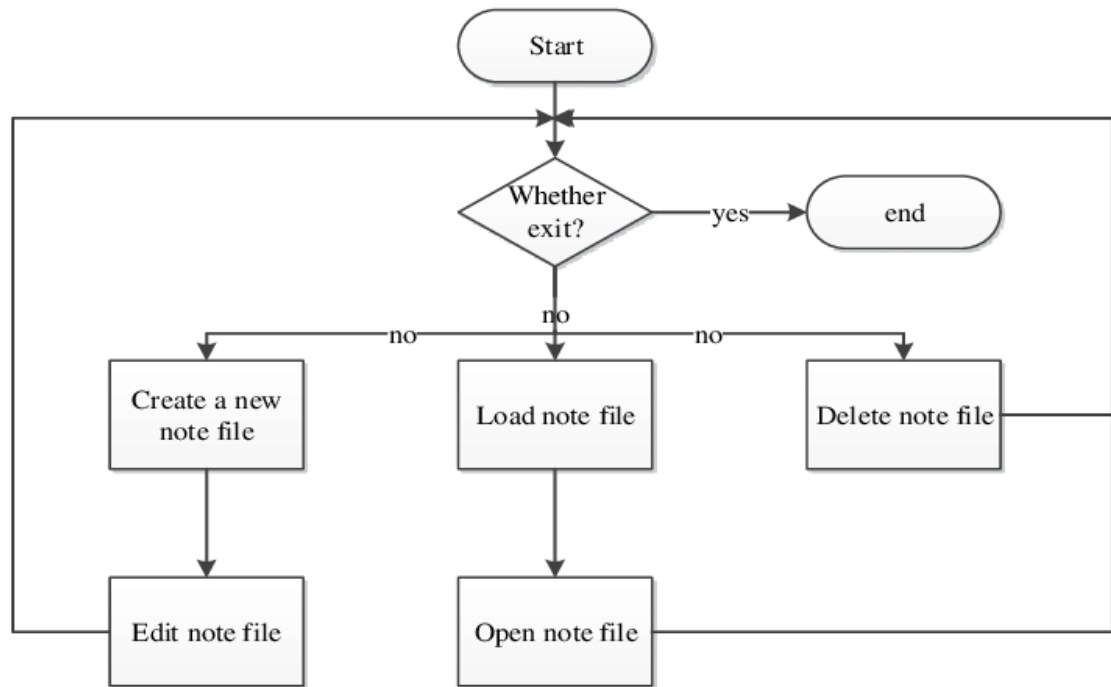
Coder is a web application that helps programmers to create and see the result of the executed source code by terminal, collaborate in real-time with other programmers by chat or invite to join the same project and manage the project such as import, export, shared projects. Coder supports C, C ++, and Java programming languages. Coder has the main features: provide workspace to make, execute and build the source code, real-time collaboration, chat, and build the terminal.

Chapter 3 Database Design

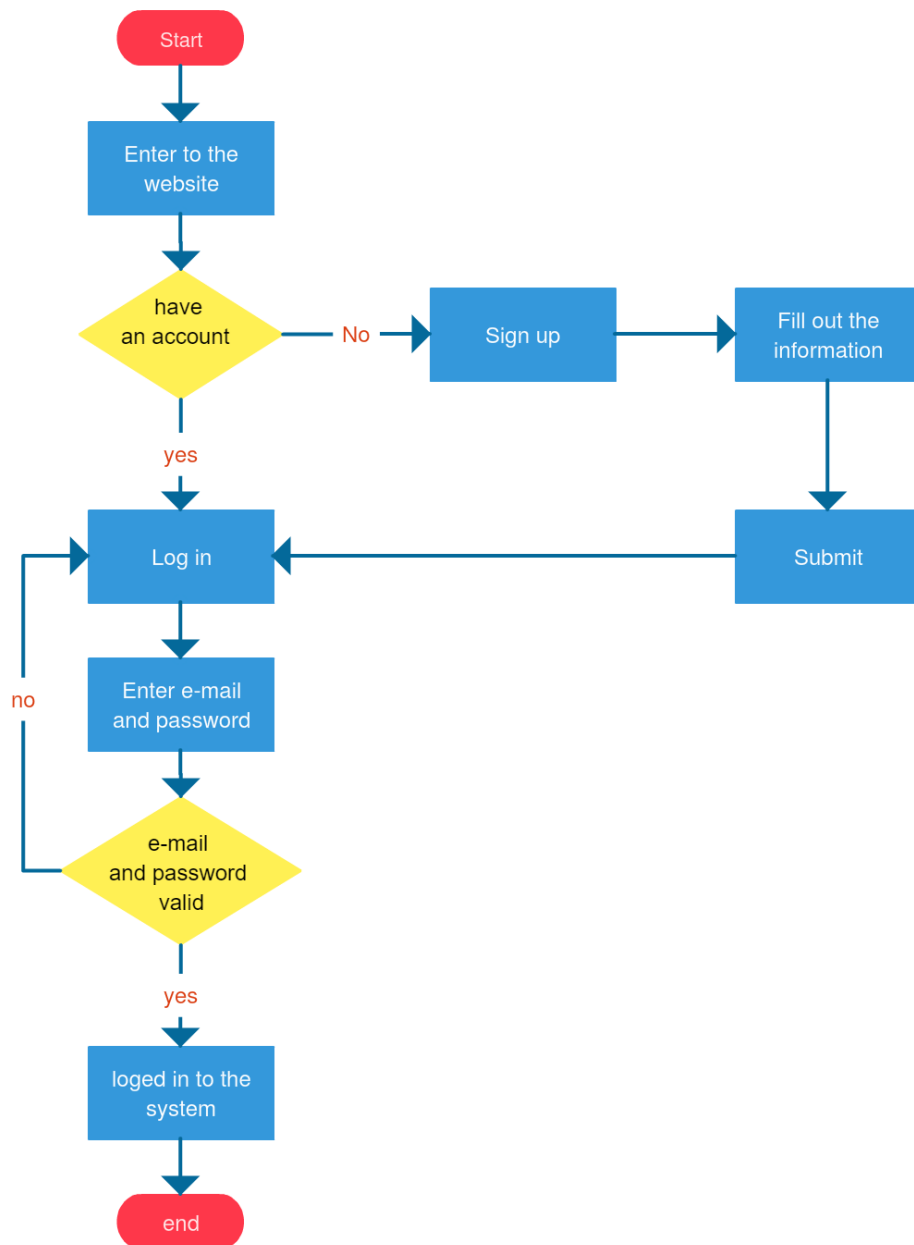
3.1 Database Tables



3.2 Flow Chart



Log in Process Flowchart



Customized editor preferences

Developers can be picky about their workspaces and workflows. Most local IDEs can't be beaten for their ability to be customized. Some cloud IDEs offer little or no ability to customize your workspace, others, however, allow a great deal of customization. In this case, it really depends on which specific IDE you are using.

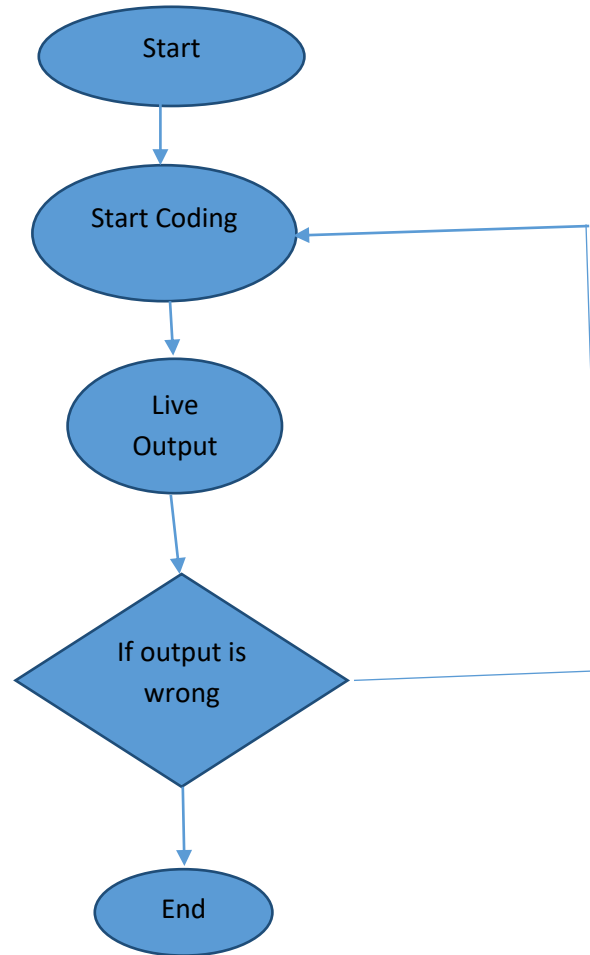
Use case best served by: either, depending on which product you are using

Conclusion

Both local and cloud IDEs have distinctive advantages and disadvantages. What works best for you depends upon your situation and needs.

If you would like to explore the potential for cloud-based IDEs in your workflow, try out our open-source project, code-server, to host VS Code on a server and access it anywhere remotely from the browser. We are also currently offering **a two-month free trial of Coder**, our enterprise product that builds upon code-server and adds features designed for teams including support for additional 10+ IDEs and advanced security features.

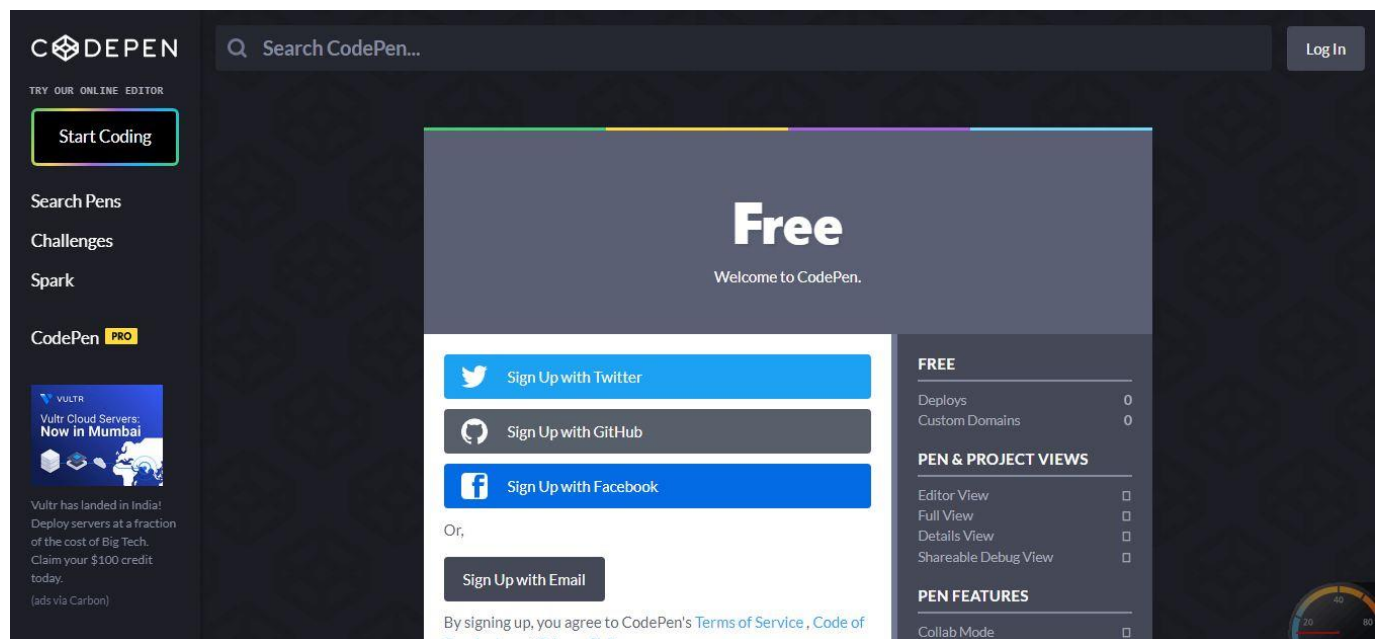
3.4 Sequence Diagram



Chapter 4

Form Design

SIGNUP FORM:



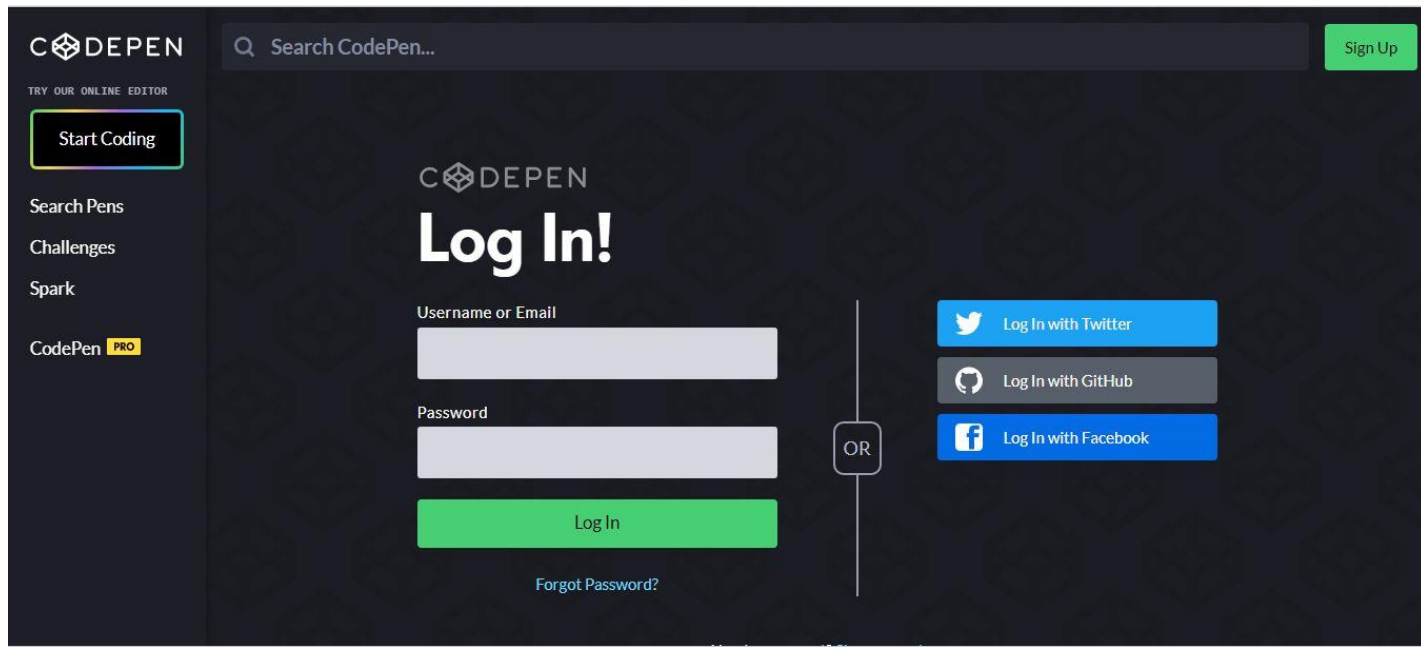
The image shows the CodePen website's signup form. The form is titled "Free" and "Welcome to CodePen." It features a dark theme with a sidebar on the left containing navigation links like "Start Coding", "Search Pens", "Challenges", "Spark", and "CodePen PRO". The main content area has a search bar and a "Log In" button. The signup form itself is centered and includes social media login options (Twitter, GitHub, Facebook) and a "Sign Up with Email" button. To the right of the form, there is a table showing the "FREE" plan's features and a list of "PEN & PROJECT VIEWS" and "PEN FEATURES".

FREE	
Deploys	0
Custom Domains	0

PEN & PROJECT VIEWS	
Editor View	<input type="checkbox"/>
Full View	<input type="checkbox"/>
Details View	<input type="checkbox"/>
Shareable Debug View	<input type="checkbox"/>

PEN FEATURES	
Collab Mode	<input type="checkbox"/>

LOGIN FORM:



The image shows the CodePen login interface. On the left is a dark sidebar with the CodePen logo, a search bar, and links to 'Start Coding', 'Search Pens', 'Challenges', 'Spark', and 'CodePen PRO'. The main area has a dark background with a subtle pattern. It features the CodePen logo, a large 'Log In!' heading, and two input fields for 'Username or Email' and 'Password'. A green 'Log In' button is below the password field. To the right of the inputs is a vertical line with a box containing 'OR'. Further right are three social login buttons: 'Log In with Twitter' (blue), 'Log In with GitHub' (grey), and 'Log In with Facebook' (blue). A 'Sign Up' button is in the top right corner. A 'Forgot Password?' link is centered below the 'Log In' button.

CODEPEN

TRY OUR ONLINE EDITOR

Start Coding

Search Pens

Challenges

Spark

CodePen PRO

Search CodePen...

Sign Up

CODEPEN

Log In!

Username or Email

Password

Log In

Forgot Password?

OR

Log In with Twitter

Log In with GitHub

Log In with Facebook

REGISTRATION FORM:

Your Name

Choose a username

codepen.io/username

Email

Choose Password

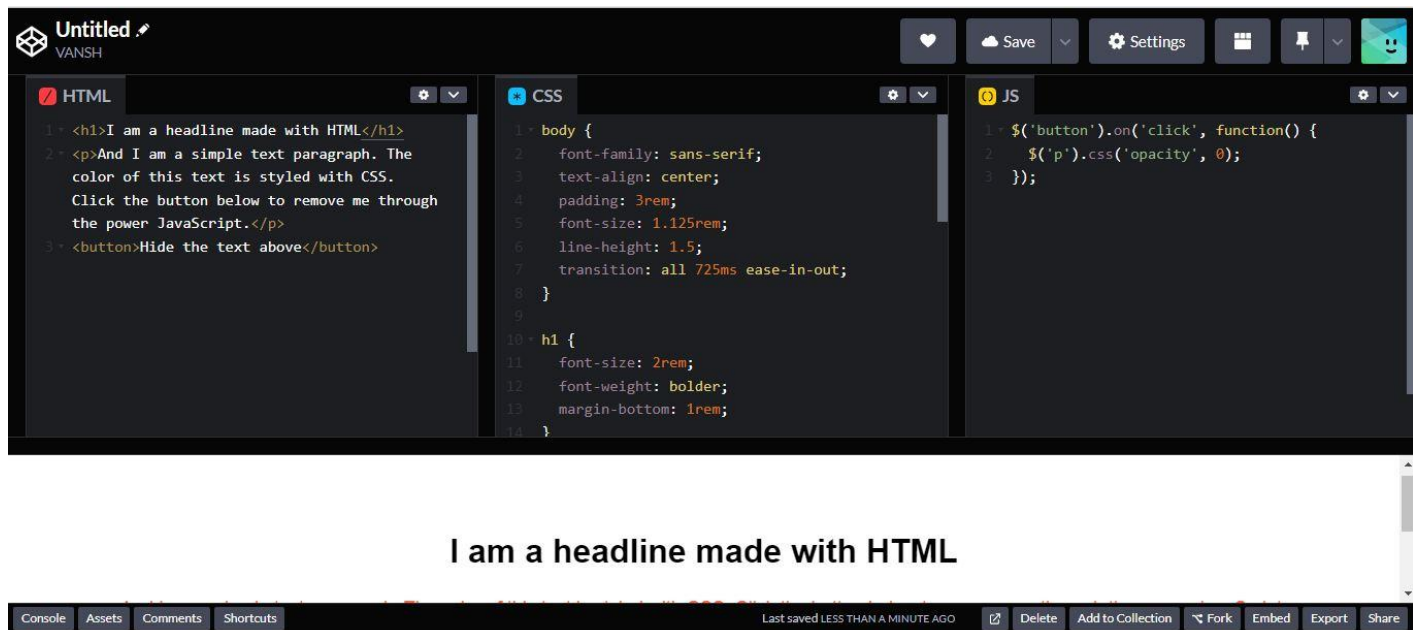
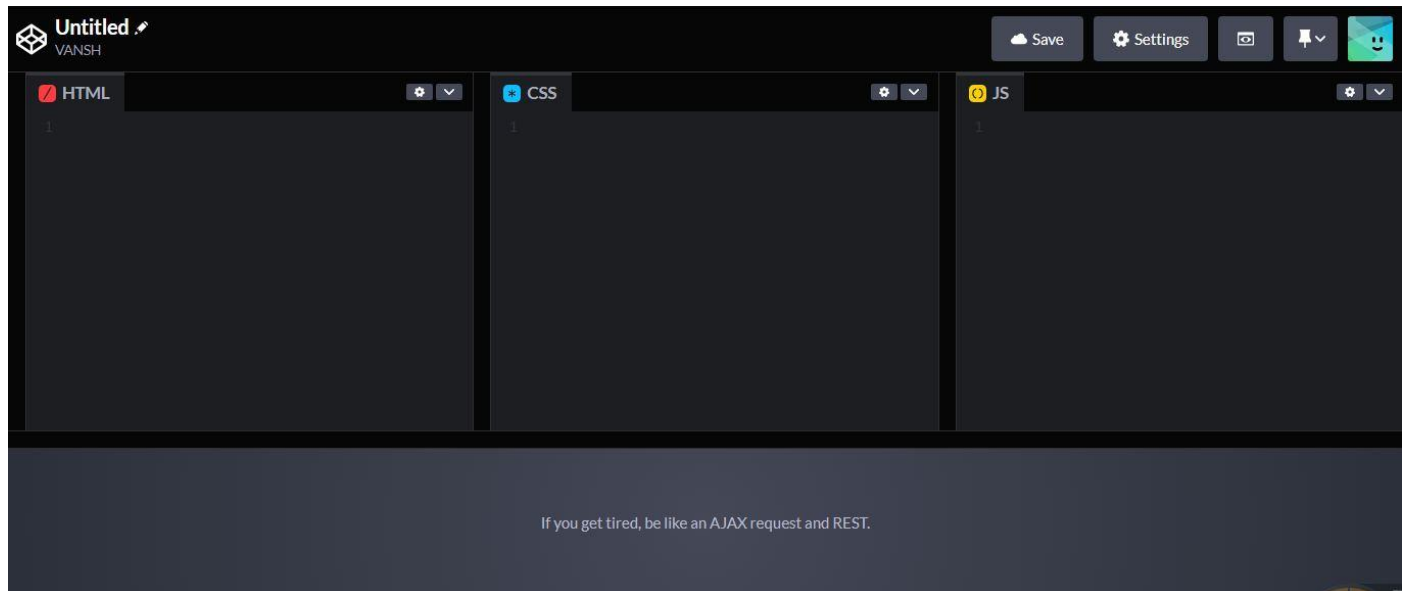
Your password must:

- Include an UPPER and lowercase letter
- Include a number
- Include one or more of these special characters: .@!%*#?&><)(^~_
- Be between 8 and 100 characters

Submit

By signing up, you agree to CodePen's [Terms of Service](#) , [Code of Conduct](#) , and [Privacy Policy](#) .

CODE EDITOR:



Chapter 5 Coding

LOGIN PAGE:

<p>We offer two popular choices: <a href="<https://github.com/postcss/autoprefixer>" target="blank" rel="noopener">Autoprefixer (which processes your CSS server-side) and <a href="<https://leaverou.github.io/prefixfree/>" target="_blank" rel="noopener">-prefix-free (which applies prefixes via a script, client-side).</p>

</div>

</div>

<ul class="radio-list">

<input type="radio" id="prefix-autoprefixer" name="prefix" value="autoprefixer">

<label for="prefix-autoprefixer" class="small-inline">Autoprefixer</label>

<input type="radio" id="prefix-prefixfree" name="prefix" value="prefixfree">

<label for="prefix-prefixfree" class="small-inline">Prefixfree</label>

<input type="radio" id="prefix-neither" name="prefix" value="neither">

<label for="prefix-neither" class="small-inline">Neither</label>

</div>

<div id="external-css-resources" class="settings-row">

<h4>

Add External Stylesheets/Pens

</h4>

<p>Any URL's added here will be added as <code><link></code>s in order, and before the CSS in the editor. You can use the CSS from another Pen by using it's URL and the proper URL extension.</p>

<div class="resource-search-bar">

<svg class="icon icon-mag" viewBox="0 0 56.7 56.7">

<input id="external-css-search" type="text" value="" placeholder="Search for resources (Bootstrap, Foundation, Animate.css...)">

</div>

<div class="algolia-shoutout">

</div>

<div class="help-flyout-link">

<svg class="icon-help" viewBox="0 0 100 100">

</svg>

<p>You can apply CSS to your Pen from any stylesheet on the web. Just put a URL to it here and we'll apply it, in the order you have them, before the CSS in the Pen itself.</p>

<p>You can also link to another Pen here (use the <code>.css</code> URL Extension) and we'll pull the CSS from that Pen and include it. If it's using a matching preprocessor, use the appropriate URL Extension and we'll combine the code before preprocessing, so you can use the linked Pen as a true dependency.</p>

</div>

<div class="algolia-shoutout">

Powered by

</svg>

</div>

</div>

</script>

<div class="external-resource-actions group">

+ add another resource

</div>

</div>

<h3 aria-label="Packages">Packages</h3>

<div class="item-settings-packages settings-row" id="item-settings-packages">

<h4>Add Packages</h4>

</svg>

<p>Using packages here is powered by Skypack, which makes packages from npm not only available on a CDN, but prepares them for native JavaScript ES6 <code>import</code> usage.</p>

<p>All packages are different, so refer to their docs for how they work.</p>

<p>If you're using React / ReactDOM, make sure to turn on Babel for the JSX processing.</p>

</div>

</div>

</div>

</div>

</div>

<div class="settings tab-page" id="settings-details">

<div id="pen-details" class="pen-details">

<div id="pen-details-form" class="pen-details-form">

<h3 aria-label="Pen Details">

Details

</h3>

<div class="settings-row top-label-form normal-labels">

<h4>

<label for="item-details-title">

Pen Title

</label>

</h4>

<input type="text" id="item-details-title" class="item-details-title" name="item-details-title" value="" maxlength="255" placeholder="Untitled">

</div>

<div class="settings-row top-label-form normal-labels">

<h4>

<label for="item-details-description">

Pen Description

</label>

```
</h4>
<textarea class="item-details-description" id="item-details-description"
placeholder="Explain what's going on in your Pen here. This text is
searchable, so it can also help others find your work. Remember to
credit others where credit is due. Markdown supported."></textarea>
</div>
<div class="tags settings-row top-label-form normal-labels">
<h4>
<label for="pen-tags" class="tags-label">
Tags <em>comma separated, <span id="max-tags-label">max of
five</span></em>
</label>
</h4>
<input type="text" id="pen-tags" class="pen-tags">
<div class="active-tags" id="active-tags">
```

```
<div>
<p class="hint">If active, Pens will autosave every 30 seconds after
being saved once.</p>
<div class="ios-toggle-mega-label-wrap">
<span class="ios-toggle ios-toggle-reverse">
<input type="checkbox" id="auto-save" name="auto-save" checked>
<label for="auto-save"></label>
<label for="auto-save" class="ios-toggle-mega-label"></label>
</span>
</div>
</div>
<div>
<div class="settings-row">
<h4>Auto-Updating Preview</h4>
<p class="hint">If enabled, the preview panel updates automatically as
```

you code. If disabled, use the "Run" button to update.</p>

```
<div class="ios-toggle-mega-label-wrap">
<span class="ios-toggle ios-toggle-reverse">
<input type="checkbox" id="auto-run" name="auto-run" checked>
<label for="auto-run"></label>
<label for="auto-run" class="ios-toggle-mega-label"></label>
</span>
</div>
```

```
</div>
```

```
<div class="settings-row">
```

```
<h4>Format on Save</h4>
```

```
<p class="hint">If enabled, your code will be formatted when you
actively save your Pen. <strong><span class="inline-
note">Note:</span> your code becomes un-folded during
formatting.</strong></p>
```

```
<div class="ios-toggle-mega-label-wrap">
<span class="ios-toggle ios-toggle-reverse">
<input type="checkbox" id="format_on_save"
name="format_on_save">
<label for="format_on_save"></label>
<label for="format_on_save" class="ios-toggle-mega-label"></label>
</span>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="settings tab-page" id="settings-editor">
```

```
<h3 aria-label="Editor Settings">Editor Settings</h3>
```

```
<div id="editor-settings-form" class="settings-row top-label-form">
```

```
<h4>
```

Code Indentation

```
</h4>
```

```
<ul class="radio-list">
```

```
<li>
```

```
<input type="radio" id="indent-with-spaces" name="indent-with"
value="spaces" checked>
```

<label for="indent-with-spaces" class="small-inline">Spaces</label>

<input type="radio" id="indent-with-tabs" name="indent-with" value="tabs">

<label for="indent-with-tabs" class="small-inline">Tabs</label>

</div>

<div class="settings-row top-label-form normal-labels">

<h4>

<label for="tab-size">

Code Indent width

</label>

</h4>

<div class="custom-select-wrap">

<select id="tab-size" class="fullwidth" name="tab-size">

<option value="1">1</option>

<option value="2">2</option>

<option value="3">3</option>

<option value="4">4</option>

<option value="5">5</option>

<option value="6">6</option>

</select>

<div class="select-icon">

<svg viewBox="-122.9 121.1 105.9 61.9" class="icon-arrow-down-mini">

<path d="M-63.2,180.3l43.5-43.5c1.7-1.7,2.7-4,2.7-6.5s-1-4.8-2.7-6.5c-1.7-1.7-4-2.7-6.5-2.7s-4.8,1-6.5,2.7l-37.2,37.2l-37.2-37.2

c-1.7-1.7-4-2.7-6.5-2.7s-4.8,1-6.5,2.6c-1.9,1.8-2.8,4.2-

2.8,6.6c0,2.3,0.9,4.6,2.6,6.5l0,0c11.4,11.5,41,41.2,43,43.3l0.2,0.2

C-73.5,183.9-66.8,183.9-63.2,180.3z" />

</svg>

<svg viewBox="-122.9 121.1 105.9 61.9" class="icon-arrow-down-mini">

```

<path d="M-63.2,180.3l43.5-43.5c1.7-1.7,2.7-4,2.7-6.5s-1-4.8-2.7-6.5c-
1.7-1.7-4-2.7-6.5-2.7s-4.8,1-6.5,2.7l-37.2,37.2l-37.2-37.2
c-1.7-1.7-4-2.7-6.5-2.7s-4.8,1-6.5,2.6c-1.9,1.8-2.8,4.2-
2.8,6.6c0,2.3,0.9,4.6,2.6,6.5l0,0c11.4,11.5,41,41.2,43,43.3l0.2,0.2
C-73.5,183.9-66.8,183.9-63.2,180.3z" />
</svg>
</div>
</div>
</div>
<div class="settings-row">
<h4>Want to change your Syntax Highlighting theme, Fonts and
more?</h4>
<p>Visit <a href="/settings/editor" target="_blank">your global Editor
Settings</a>.</p>
</div>
</div>
<div class="settings tab-page" id="settings-template">
<div id="pen-template" class="pen-details">
<div class="top-label-form">
<h3>Template</h3>
<div class="settings-row">
<h4>Make Template?</h4>
<p>Templates are Pens that can be used to start other Pens quickly from
the create menu. The new Pen will copy all the code and settings from
the template and make a new Pen (that is not a fork). You can <a
target="_blank" rel="noopener" href="/you/pens/templates">view all of
your templates</a>, or <a target="_blank" rel="noopener"
href="https://blog.codepen.io/documentation/api/templates/">learn
more in the documentation</a>.</p>
<div class="ios-toggle-mega-label-wrap" id="item-details-template-
wrap">
<span class="ios-toggle ios-toggle-reverse">
<input type="checkbox" name="item-details-template" id="item-
details-template">
<label for="item-details-template"></label>
<label for="item-details-template" class="ios-toggle-mega-

```

```
label"><span>On</span></label>
```

```
</span>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="settings tab-page" id="settings-screenshot">
```

```
<div id="pen-screenshot" class="pen-details">
```

```
<h3>Screenshot</h3>
```

```
<div id="pen-screenshot-form" class="pen-details-form">
```

```
<div class="thumbnail-upload-section settings-row">
```

```
<h4>
```

Screenshot or Custom Thumbnail

```
</h4>
```

Screenshots of Pens are shown in mobile browsers, RSS feeds, to users who chose images instead of iframes, and in social media sharing.</p>

```
<div id="settings-screenshot-wrap" class="settings-screenshot-wrap">
```

```
<div id="screenshot-type" class="screenshot-description">
```

```
<div class="screenshot-area-screenshot screenshot-image" id="custom-screenshot" style="background-image:
```

```
url(https://shots.codepen.io/anon/pen/-800.jpg?version=0);">
```

```
<a id="delete-screenshot" class="button mini-button button-dark delete-screenshot inline-display-none">× Delete</a>
```

```
</div>
```

```
<div class="screenshot-description-container">
```

This Pen is using the default Screenshot, generated by CodePen. Upgrade to PRO to upload your own thumbnail that will be displayed on previews of this pen throughout the site and when sharing to social media.</p>

```
<a href="/features/pro" class="button yellow">Upgrade to PRO</a>
```

```
</div>
```



```
</div>
</div>
</div>
</div>
</div>
</div>
</div>
<footer>
<div class="save-and-close-wrap">
<input type="button" class="button button-small green button-no-right-
margin close" value="Close" id="close-settings">
</div>
</footer>
</div>
<div class="page-wrap">
<div class="boxes">
<div class="mobile-editor-nav">
<button id="html-toggle"
class="selected"><span>HTML</span></button>
<button id="css-toggle"><span>CSS</span></button>
<button id="js-toggle"><span>JS</span></button>
<button id="result-toggle"
class="selected"><span>Result</span></button>
</div>
<div class="top-boxes editor-parent" data-number-of-editors="3"
elementtiming="pen-editors">
<div class="editor-resizer html-editor-resizer" title="Double-click to
expand."></div>
<div id="box-html" class="box box-html" data-type="html">
<div class="powers">
<div class="powers-drag-handle" title="Double-click to
expand."></div>
<div class="editor-actions-left">
<h2 class="box-title html-editor-title" id="html-editor-title">
</svg>
```

```
<span class="box-title-name">
HTML
</span>
<span class="box-title-preprocessor-name "></span>
</h2>
</div>
<div class="editor-actions-right">
<div class="collaborators-indicators"></div>
<button id="settings-pane-html" class="button button-medium mini-
button settings-nub" data-type="html" title="Open HTML Settings">
```

```
</svg>
</button>
<button class="button mini-button button-medium editor-dropdown-
button editor-dropdown-button-html" data-dropdown="#editor-
dropdown-html" aria-haspopup="true" aria-expanded="false">
<span class="visually-hidden">
```

HTML Options

```
</span>
<svg viewBox="-122.9 121.1 105.9 61.9" class="icon-arrow-down-
mini" width="10" height="10">
```

```
</svg>
</button>
<ul id="editor-dropdown-html" class="link-list is-dropdown editor-
dropdown editor-dropdown-html" data-dropdown-position="css" data-
dropdown-type="html">
<li class="editor-dropdown-list-item">
<button id="tidy-html" class="invisible-button tidy-code-button" data-
editor-type="html">
```

Format HTML

```
</button>
```


<li class="editor-dropdown-list-item">

<button id="view-compiled-html" class="invisible-button view-compiled-button" data-editor-type="html">

View Compiled HTML

</button>

<li class="editor-dropdown-list-item">

<button id="analyze-html" class="invisible-button analyze-button" data-editor-type="html">

Analyze HTML

</button>

<li class="editor-dropdown-list-item">

<button id="maximize-html-editor" class="invisible-button maximize-button" data-editor-type="html">

Maximize HTML Editor

</button>

<li class="editor-dropdown-list-item">

<button id="minimize-html-editor" class="invisible-button minimize-button" data-editor-type="html">

Minimize HTML Editor

</button>

<li class="editor-dropdown-list-item">

<button id="fold-all-html" class="invisible-button fold-all-button" data-editor-type="html">

Fold All

</button>

<li class="editor-dropdown-list-item">

<button id="unfold-all-html" class="invisible-button unfold-all-button" data-editor-type="html">

Unfold All

```
</button>
</li>
</ul>
</div>
</div>
<div class="code-wrap notranslate" translate="no">
<pre id="html" class="code-box" aria-labeledby="html-editor-title">
<code>
</code>
</pre>
<div class="error-bar" id="error-bar-html">
<span class="error-icon" data-type="html">
!
</span>
</div>
<span class="code-editor-status"></span>
</div>
</div>
<div class="editor-resizer css-editor-resizer" title="Double-click to
expand."></div>
<div id="box-css" class="box box-css" data-type="css">
<div class="powers">
<div class="powers-drag-handle" title="Double-click to
expand."></div>
<div class="editor-actions-left">
<h2 class="box-title css-editor-title" id="css-editor-title">
<svg viewBox="0 0 15 15" class="file-type-icon" id="icon-file-css">
</svg>
<span class="box-title-name">
CSS
</span>
<span class="box-title-preprocessor-name "></span>
</h2>
</div>
```

```
<div class="editor-actions-right">
<div class="collaborators-indicators"></div>
<button id="settings-pane-css" class="button button-medium mini-
button settings-nub" data-type="css" title="Open CSS Settings">
<svg viewBox="0 0 100 100" id="icon-gear" width="10" height="10">

</svg>
```

```
<div class="console-entries short-no-scroll"></div>
```

```
<div class="console-command-line">
<span class="console-arrow forwards"></span>
<textarea class="console-command-line-input auto-expand" rows="1"
data-min-rows="1"></textarea>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
<div id="asset-bin-goes-here"></div>
<footer id="react-pen-footer" class="site-footer editor-
footer"></footer>
<div id="keycommands" class="modal modal-neutral">
<div class="keycommands-container">
<section class="editor-commands inline-editor-commands">
<h2>Editor Commands</h2>
<div class="key-group">
<kbd class="keycommand">
<span class="key pc_only">Ctrl</span>
<span class="key mac_only">Ctrl</span>
<span class="key">Space</span>
</kbd>
Autocomplete
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key">F</span>
</kbd>
Find
</div>
<div class="key-group">
```

```
<kbd class="keycommand">  
<span class="key -command"></span>  
<span class="key">G</span>  
</kbd>
```

Find Next

```
</div>  
<div class="key-group">  
<kbd class="keycommand">  
<span class="key -command"></span>  
<span class="key" title="Shift">␣</span>  
<span class="key">G</span>  
</kbd>
```

Find Previous

```
</div>  
<div class="key-group">  
<kbd class="keycommand">  
<span class="key -command"></span>  
<span class="key pc_only" title="Shift">␣</span>  
<span class="key mac_only">Opt</span>  
<span class="key">F</span>  
</kbd>
```

Find & Replace

```
</div>  
<div class="key-group mac_only">  
<kbd class="keycommand">  
<span class="key -command"></span>  
<span class="key" title="Shift">␣</span>  
<span class="key">F</span>  
</kbd>
```

Format Code

```
</div>  
<div class="key-group">  
<kbd class="keycommand">  
<span class="key -command"></span>
```

```
<span class="key">[</span>
</kbd>
```

Indent Code Right

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key">]</span>
</kbd>
```

Indent Code Left

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key" title="Shift">↑</span>
<span class="key">Tab</span>
</kbd>
```

Auto Indent Code

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key">/</span>
</kbd>
```

Line Comment

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key pc_only" title="Shift">↑</span>
<span class="key mac_only">Opt</span>
<span class="key">/</span>
</kbd>
```

Block Comment

```
</div>
```


<p class="inline-tab-triggers">Also see: Tab Triggers</p>

</section>

<section class="editor-commands">

<h2>Editor Focus</h2>

<div class="key-group">

<kbd class="keycommand">

Alt

Opt

1

</kbd>

HTML Editor

</div>

<div class="key-group">

<kbd class="keycommand">

Alt

Opt

2

</kbd>

CSS Editor

</div>

<div class="key-group">

<kbd class="keycommand">

Alt

Opt

3

</kbd>

JS Editor

</div>

<div class="key-group">

<kbd class="keycommand">

Alt
Opt
4
</kbd>

Toggle Console

</div>
<div class="key-group">
<kbd class="keycommand">

Alt
Opt
0
</kbd>

Preview

</div>
<div class="key-group">
<kbd class="keycommand">
Esc
</kbd>

Exit currently focused editor

</div>
</section>
<section class="editor-commands">
<h2>Misc</h2>
<div class="key-group">
<kbd class="keycommand">

⇧
7
</kbd>

Re-run Preview

</div>

```
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key" title="Shift">␣</span>
<span class="key">8</span>
</kbd>
```

Clear All Analyze Errors

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key" title="Shift">␣</span>
<span class="key">9</span>
</kbd>
```

Open This Dialog

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key" title="Shift">␣</span>
<span class="key">0</span>
</kbd>
```

Open Debug View

```
</div>
<h2>HTML Specific</h2>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key" title="Shift">␣</span>
<span class="key">A</span>
</kbd>
```

Wrap With...

```
</div>
<h2>Pen Actions</h2>
```

```
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key">P</span>
</kbd>
```

Create New Pen

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key">S</span>
</kbd>
```

Save

```
</div>
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key" title="Shift">⬆</span>
<span class="key">S</span>
</kbd>
```

```
Save As Private <span style="border-radius: 2px; padding: 1px 6px 2px 5px; color: black !important; background: var(--cp-color-yellow); white-space: nowrap; text-shadow: none; letter-spacing: 0; font-size: 71%; position: relative; top: -1px;">PRO</span>
</div>
```

```
<div class="key-group">
<kbd class="keycommand">
<span class="key -command"></span>
<span class="key">I</span>
</kbd>
```

Info Panel (if owned)

```
</div>
```

Login.js:-

```

import React from "react";
import { Row, Col, Form, Input, Button } from "antd";
import { loginUser } from "../redux/actions/usersActions";
import { useDispatch } from "react-redux";
import { Link } from "react-router-dom";

import AOS from 'aos';
import 'aos/dist/aos.css';

AOS.init();

function Login() {
  const dispatch = useDispatch()
  function login(values){

    dispatch(loginUser(values))

  }
  return (
    <div className="login">
      <Row justify="center" className="flex align-items-center">
        <Col lg={5}><h1 className="heading1" data-aos='slide-
left'>Shey</h1></Col>
        <Col lg={10} sm={24} className="bs p-5 login-form">
          <h3>Login</h3>
          <hr />
          <Form layout="vertical" onFinish={login}>
            <Form.Item
              label="username"
              name="username"
              rules={[{ required: true }]}
            >
              <Input />
            </Form.Item>

            <Form.Item
              label="password"
              name="password"
              rules={[{ required: true }]}
            >
              <Input />
            </Form.Item>

            <Button htmlType="submit" className='mb-3'>Login</Button><br />

            <Link to='/register' className='mt-3'>Not registered ? , Click here to
register</Link>
          </Form>
        </Col>
        <Col lg={5}><h1 className='heading2' data-aos='slide-
right'>Jobs</h1></Col>
      </Row>
    </div>
  )
}

```

```

    </div>
  );
}

export default Login;

```

SignUp.js:-

```

import React, { useEffect } from "react";
import DefaultLayout from "../components/DefaultLayout";
import { useSelector, useDispatch } from "react-redux";
import { getAllJobs } from "../redux/actions/jobActions.";
import { Row, Col, Button } from "antd";
import { Link } from "react-router-dom";
import moment from "moment";
function Home() {
  const { jobs } = useSelector((state) => state.jobsReducer);
  const dispatch = useDispatch();
  useEffect(() => {
    dispatch(getAllJobs());
  }, []);

  return (
    <div>
      <DefaultLayout>
        <Row gutter={16}>
          {jobs.map((job) => {
            return <Col lg={12} sm={24}>
              <div className="job-div bs m-2 p-2">
                <h4>{job.title}</h4>
                <p>{job.company}</p>
                <hr />
                <p>{job.smallDescription}</p>
                <div className="flex">
                  <p>Salary : <b>{job.salaryFrom} - {job.salaryTo}</b> , </p>
                  <p style={{marginLeft:20}}>Experience : <b>{job.experience}</b>
                    Years</p>
                </div>
                <hr />
                <div className="flex justify-content-between">
                  <Link to={` /jobs/${job._id}`}><Button>View</Button></Link>
                  <p>Posted on : {moment(job.createdAt).format('MMM DD
yyyy')}</p>
                </div>
              </div>
            </Col>;
          })}
        </Row>
      </DefaultLayout>
    </div>
  );
}

```

```

        </Row>
      </DefaultLayout>
    </div>
  );
}

export default Signup;

```

App.js:-

```

import React, { useState, useEffect } from 'react'
import Editor from "../components/Editor"
import useLocalStorage from "../hooks/useLocalStorage"

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

  useEffect(() => {
    const timeout = setTimeout(() => {
      setSrcDoc(`
        <html>
          <body>${html}</body>
          <style>${css}</style>
          <script>${javascript}</script>
        </html>
      `)
    }, 250)
    // console.log(srcDoc)
    return () => clearTimeout(timeout)
  }, [html, css, javascript])

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

```

```

        onChange={setCss}
      />
      <Editor
        language="javascript"
        label="JavaScript"
        value={javascript}
        onChange={setJavascript}
      />
    </div>
    <div className="bottom-pane">
      <iframe
        srcDoc={srcDoc}
        title="output"
        sandbox="allow-scripts"
        frameBorder="0"
        width="100%"
        height="100%"
      ></iframe>
    </div>
  </div>
);
}

export default App;

```

Index.CSS:-

```

body {
  margin: 0;
}

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

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

.bottom-pane {
  height: 50vh;
}

.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;
}

```

Index.js:-

```

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

```

```
import reportWebVitals from './reportWebVitals';

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

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

ReportWebVital:-

```
const reportWebVitals = onPerfEntry => {
  if (onPerfEntry && onPerfEntry instanceof Function) {
    import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
      getCLS(onPerfEntry);
      getFID(onPerfEntry);
      getFCP(onPerfEntry);
      getLCP(onPerfEntry);
      getTTFB(onPerfEntry);
    });
  }
};

export default reportWebVitals;
```

UseLocalStorage.js:-

```
import { useEffect, useState } from "react";

const PREFIX = "online-editor-"

const useLocalStorage = (key, initialValue) => {
  const prefixedKey = PREFIX + key;

  const [value, setValue] = useState(() => {
    const jsonValue = localStorage.getItem(prefixedKey)
    if (jsonValue) {
      return JSON.parse(jsonValue)
    }
  })

  useEffect(() => {
    setValue(initialValue)
  }, [initialValue])

  return [value, setValue]
}
```

```

        if (typeof initialValue === "function") {
            return initialValue()
        } else {
            return initialValue
        }
    })

    useEffect(() => {
        localStorage.setItem(prefixedKey, JSON.stringify(value))
    }, [prefixedKey, value])

    return [value, setValue];
}

export default useLocalStorage;

```

Editor.js:-

```

import React, { useState } from 'react'
import { Controlled as CodeMirror } from 'react-codemirror2'
import "codemirror/lib/codemirror.css";
import "codemirror/theme/material.css";
import "codemirror/mode/xml/xml"
import "codemirror/mode/css/css"
import "codemirror/mode/javascript/javascript"
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
import { faExpandAlt, faCompressAlt } from '@fortawesome/free-solid-svg-icons'
import "../index.css"

const Editor = (props) => {
    const [open, setOpen] = useState(true)
    const { language, label, value, onChange } = props

    const handleChange = (editor, data, value) => {
        onChange(value)
    }

    return (
        <div className={`editor-container ${open ? '' : 'collapsed'}`}>
            <div className="editor-title">
                {label}
                <button
                    type="button"
                    onClick={() => setOpen(!open)}
                    className="expand-collapse-btn"
                >
                    <FontAwesomeIcon icon={open ? faCompressAlt : faExpandAlt} />
                </button>
            </div>
            <CodeMirror
                value={value}

```

```
        className="code-mirror-wrapper"
        onBeforeChange={handleChange}
        options={{
          lineWrapping: true,
          lint: true,
          mode: language,
          lineNumbers: true,
          theme: "material"
        }}
      />
    </div>
  )
}

export default Editor
```

Chapter 6

Testing

Test Scenario for Online Editor

Online text editors are designed to replace the desktop dependency for text editing. Here you get all the features necessary for editing the text. You can use different fonts. You can format the text with different options such as italicize, justify the alignment and do many other text processing. All of these settings can be preserved depending on the exported format from the text editor. So we are going to test the online text editor with following test cases in mind. If you have any specific text editors test cases then feel free to let me know in the comments. Let's take a look at the text editor offered by writeurl's online editor.

Unit Testing

While coding, the programmer performs some tests on that unit of program to know if it is error free. Testing is performed under white-box testing approach. Unit testing helps developers decide that individual units of the program are working as per requirement and are error free.

Integration Testing

Even if the units of software are working fine individually, there is a need to find out if the units if integrated together would also work without errors. For example, argument passes and data updating etc.

System Testing

The software is compiled as product and then it is tested. This can be accomplished using one or more of the following tests:

Functionality testing

Tests all functionalities of the software against the requirement.

[OBJ]

Now that you know the text editor in question. Let's discuss test cases for online editor.

Test Cases for Online Text Editor

Format Options

Check if the text can be bolded with the short key CTRL+B.

Check if the text can be bolded with the bold icon press action.

Check if the text can be italicize with the short key CTRL+I.

Check if the text can be bolded with the Italic icon press action.

Check if the text can be underlined with shortcutkey CTRL+U.

Check if the text can be underlined with underline icon press.

Check if the text can be strikethrough with the strike icon press.

Check if the text can be unbolded through unbold icon press.

Check if the text can be superscripted with icon press.

Check if the text can be subscripted with icon press.

Check if the text can be increased indentation with icon press.

Check if the text can be decreased indentation with icon press.

Performance testing

This test proves how efficient the software is. It tests the effectiveness and average time taken by the software to do desired task. Performance testing is done by means of load testing and stress testing where the software is put under high user and data load under various environment conditions.

Acceptance Testing

When the software is ready to hand over to the customer it must go through last phase of testing where it is tested for user-interaction and response. This is important because even if the software matches all user requirements and if user does not like the way it appears or works, it may be rejected.

Alpha testing

The team of developer themselves perform alpha testing by using the system as if it is being used in work environment. They try to find out how user would react to some action in software and how the system should respond to inputs.

Beta testing

After the software is tested internally, it is handed over to the users to use it under their production environment only for testing purpose. This is not yet the delivered product. Developers expect that users at this stage will bring minute problems, which were skipped to attend.

Check if the text can be left aligned.

Check if the text can be right aligned.

Check if the text can be center aligned.

Check if the text can be justify aligned.

Check if the line spacing of the text can be adjusted from the drop down menu.

Check if the line spacing values are correct.

Check if the text format can be undo.

Check if the text format can be redo.

Check if the undo button works.

Check if the redo button works.

Font Options

Check if the text headlines can be changed with h1, h2, h3 tags.

Check if the text headlines are as per the tag specifications.

Check if the text fonts can be changed.

Check if the font size can be changed.

Check if the fonts are preserved after the file export.

Check if the special character can be added into the content.

Check if the list element can be added.

Check if the list element are properly exported.

File Options

Check if the text is autosaved by the editor.

Check if the text can be exported to the format of your choice on desktop.

Check if the file can be shared on social media.

Check if the file can be uploaded to cloud accounts.

Check if the closure of tab prompts the file save option.

These are some of the generic level of the test cases that you can check for the online editor. This can be used with both online and offline text editors and IDE. And you can manage to add and remove more test scenarios depending on the available set of feature.

Research Paper

1. <https://www.researchgate.net/scientific-contributions/Asst-Prof-2196623378>
2. https://www.researchgate.net/publication/340379240_Introducing_Reactjs
3. https://www.researchgate.net/publication/318131748_An_Overview_of_Blockchain_Technology_Architecture_Consensus_and_Future_Trends
4. https://www.researchgate.net/publication/262946235_Researching_SME_entrepreneurial_research_A_study_of_Journal_of_Research_in_Marketing_and_Entrepreneurship_JRME_2000-2011
5. https://www.researchgate.net/publication/318131748_An_Overview_of_Blockchain_Technology_Architecture_Consensus_and_Future_Trends
6. https://www.researchgate.net/publication/288331382_Mobile_computing_issues_and_challenges
7. https://www.researchgate.net/publication/317101504_Computer_Networking_A_Survey
8. https://www.researchgate.net/publication/224238658_Data_structures_and_algorithms_in_pen-based_computing_environments
9. https://www.researchgate.net/publication/49616224_Data_mining_techniques_and_applications
10. https://www.researchgate.net/publication/318310544_Nodejs_Challenges_in_Implementation
11. https://www.researchgate.net/publication/318310544_Nodejs_Challenges_in_Implementation
12. https://www.researchgate.net/publication/327120267_MongoDB_-_a_comparison_with_NoSQL_databases
13. https://www.researchgate.net/publication/323015324_A_Survey_on_Java_Programming_Language_and_Methods_of_Improvisation
14. https://www.researchgate.net/publication/321636664_The_C_Programming_Language

15.https://www.researchgate.net/publication/2334185_An_Overview_of_the_C_Programming_Language

16.

https://www.academia.edu/Documents/in/DATABASE_MANAGEMENT_SYSTEM

17.https://www.academia.edu/Documents/in/Computer_Graphics

18.https://www.researchgate.net/publication/273693976_A_Review_on_Internet_of_Things_IoT

19.https://www.researchgate.net/publication/317101504_Computer_Networking_A_Survey

20.https://www.researchgate.net/publication/311488672_SQL_From_Traditional_Databases_to_Big_Data

21.https://www.researchgate.net/publication/49616224_Data_mining_techniques_and_application

22.https://www.researchgate.net/publication/335015436_A_REVIEW_ON_CLIENT-SERVER_BASED_APPLICATIONS_AND_RESEARCH_OPPORTUNITY

23.https://www.researchgate.net/publication/273693976_A_Review_on_Internet_of_Things_IoT

24.https://www.researchgate.net/publication/332456776_Research_and_Analysis_of_the_Front-end_Frameworks_and_Libraries_in_E-Business_Development

25.https://www.researchgate.net/publication/258328266_Database_Management_Systems_A_NoSQL_Analysis