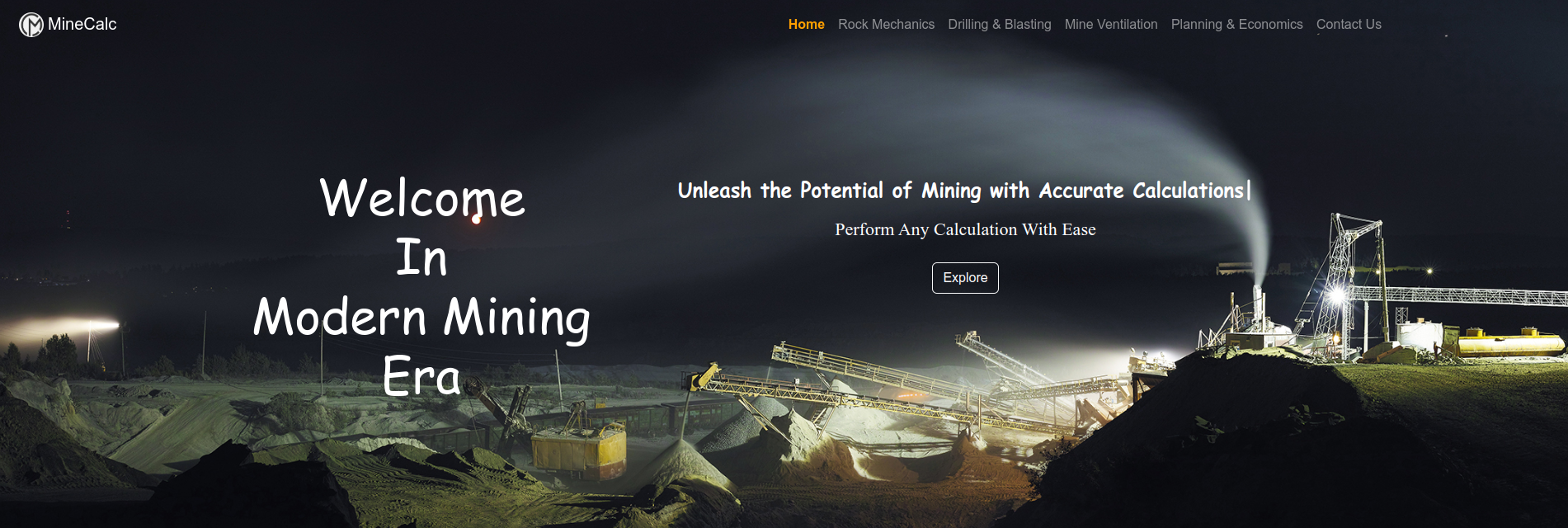
**MineCalc Website**



# 

**Git Repository Link-** <https://github.com/Alok4k45h/MineCalc_Web.git>

**Tech Stack Used-** React Js, Node Js, Django, WebPlotdigitizer, Python. Javascript, Bootstrap, Css, Postman

horizontal line

# 

# Problem Statement

Build an .EXE file or any executable file using any coding language of your choice to estimate the probable penetration rate for a

1) Rotary Percussive drilling machine

2) Rotary drilling machine

The number of algorithm options should be showcased to the user via instruction display. The algorithms to be considered are as follow:

a.) Rotary Percussive:

• USBM, 1969

• Norway, 1979

• Benaola, 1985

b.) Rotary:

• Bauer and Calder, 1967

• Baurer, 1971

• Calder and Workman, 1994

## Solution:-

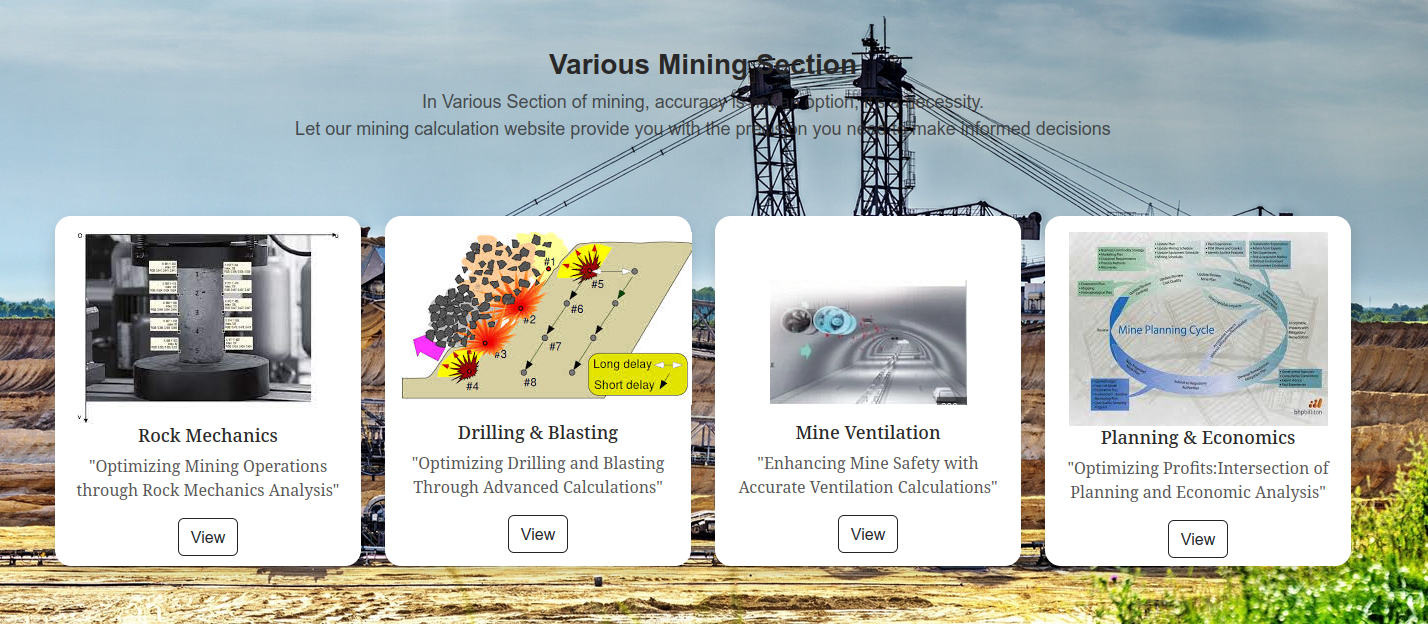
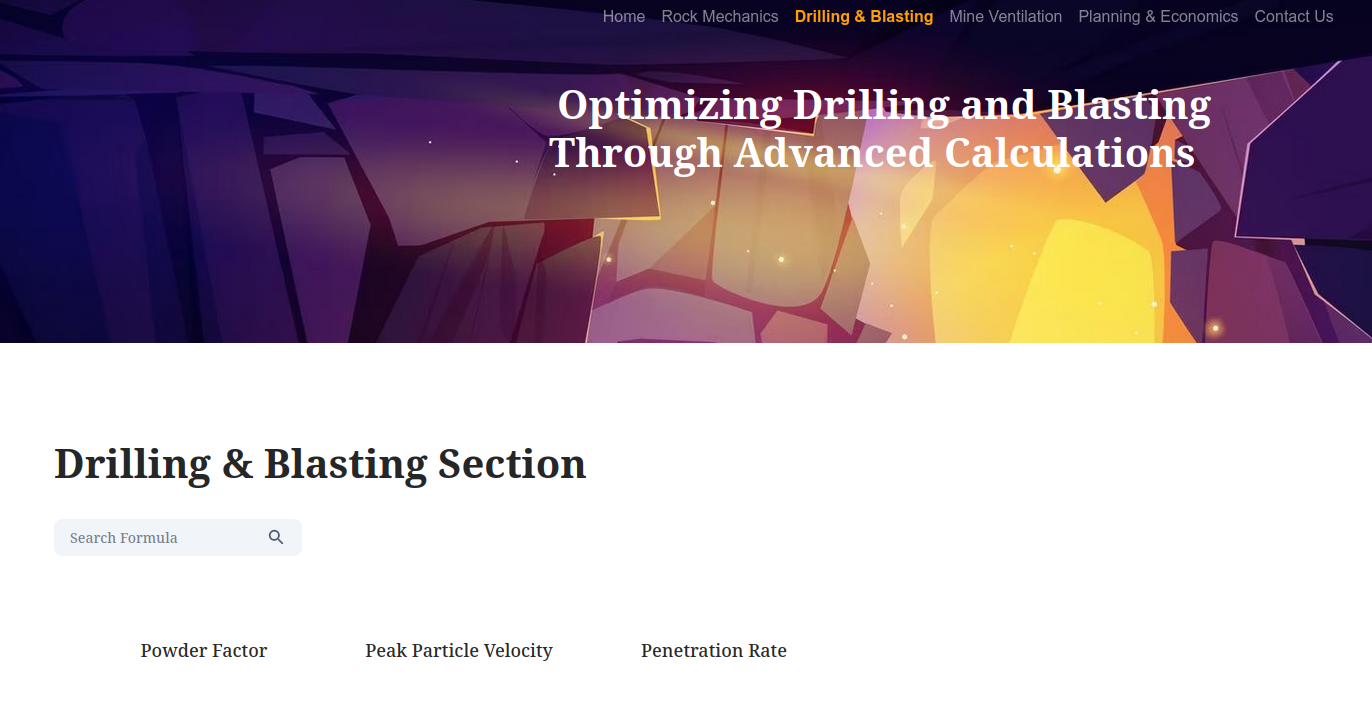
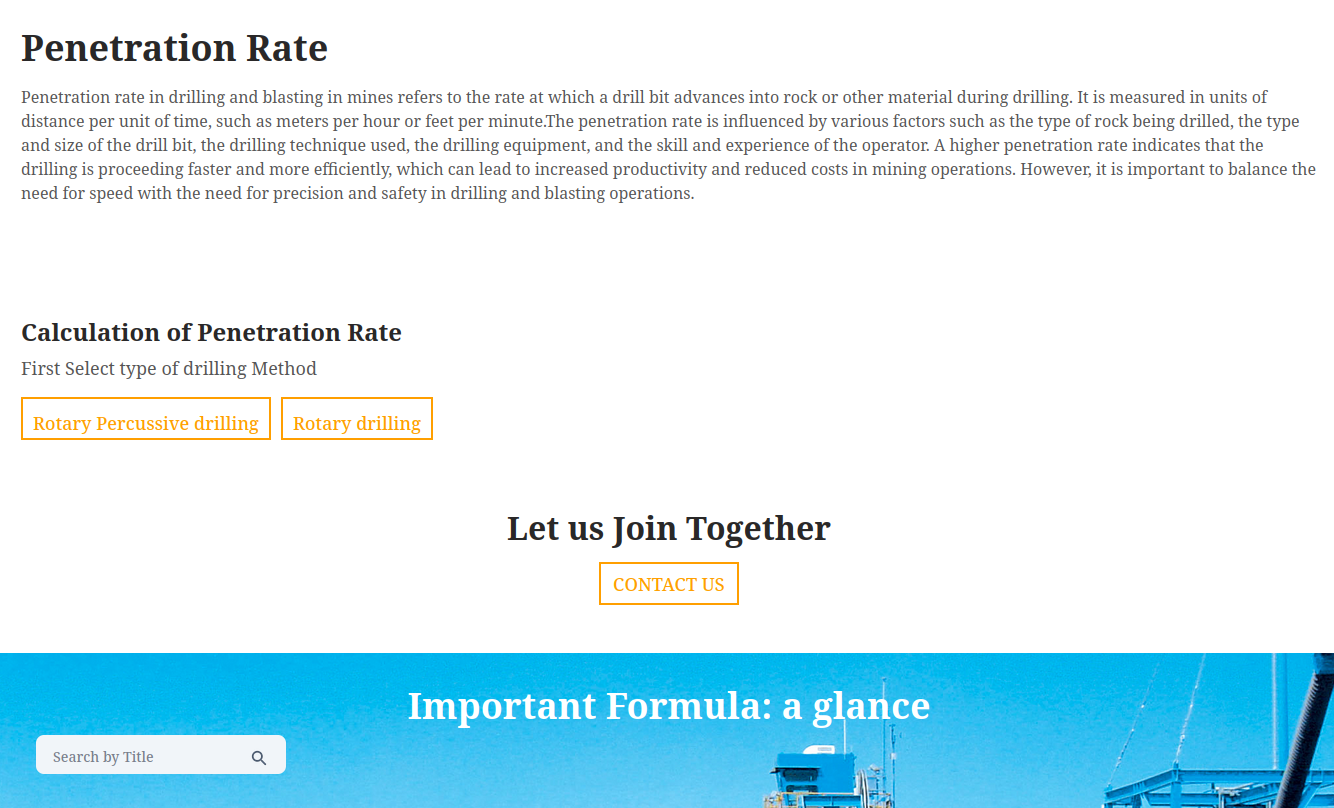
### Introduction

We have created a comprehensive and user-friendly web application specifically tailored for the mining industry. Our platform features advanced algorithms designed to assist in solving the mining problem involving mining formula, ranging from drilling and blasting to rock mechanics, mine ventilation, and mine planning and design.

So we are integrating the above all Algorithm of Penetration rate into Our MineCalc Website where the penetration page of this website provide an interface to user to first select the type of drilling method of their choice and after selecting the methods, user have an option to select the available algorithm to calculate the Penetration Rate.

Now, the Algorithm page allows the user to input the data in the specified field to get the answer of penetration rate on clicking the calculate button. Also this page provides a demo data button as well as reset button to fill the typical values to all fields and to clear the input value respectively.

### Navigation To Penetration Rate Page

1. On **Home page** of Website, Navigate to Various mining sections. In this section, Go to the **Drilling and Blasting Section** and **click the View Button**. 
2. On this page of Drilling and Blasting Section, **Click the Penetration Rate text** to open the penetration rate page
3. **So This is the Initial interface of Penetration Rate webPage**

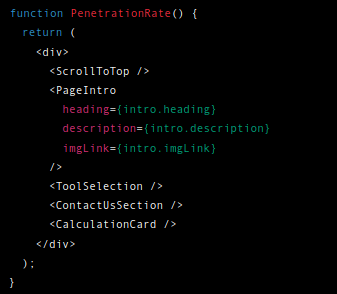
### PseudoCode of Penetration Rate Page

**Steps:-**

1. First imports the necessary components and resources that will be used in the PenetrationRate component.
2. Then, PenetrationRate function is defined. This function is a React component that returns a JSX element. Within the return statement, the components are rendered on the page in a specific order:

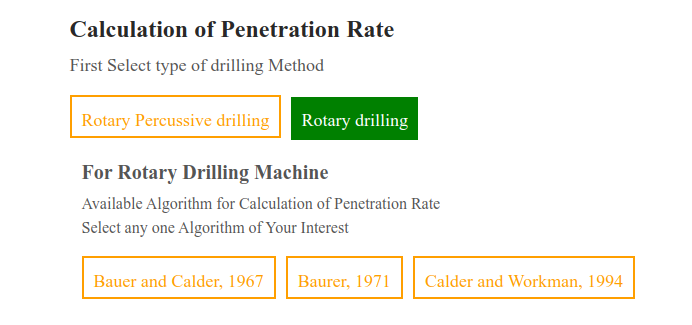
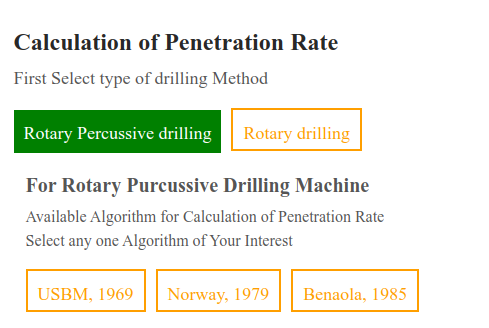
* ScrollToTop component is rendered first, which allows the user to scroll to the top of the page.
* PageIntro component is rendered next with the heading, description, and imgLink props from the intro object.
* ToolSelection component is rendered next which provides the selection of Various methods and Algorithms used to calculate Penetration Rate.
* ContactUsSection component is rendered next which provides the contact section on the homepage.
* CalculationCard component is rendered last which displays the Important Formula calculation card.

### Code



1. Finally, PenetrationRate function is exported as a default export, so it can be used in App.js to render as a web page on Website.

### 3.1 Tool Selection Component



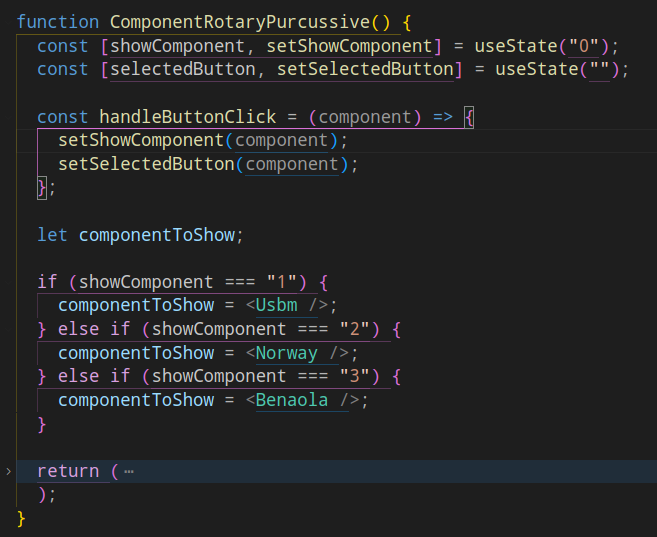
**Pseudocode for Selection of Drilling Methods and Available Algorithms**

**Steps:-**

1. First imports the necessary components and resources that will be used in the ToolSelection component.
2. The code is split into three main sections: ComponentRotaryPercussive, ComponentRotary, and ToolSelection.

## ComponentRotaryPercussive

* This component is used to display a set of algorithms for rotary percussive drilling machines. The component has two states: showComponent and selectedButton. showComponent is used to determine which algorithm component to render, while selectedButton is used to determine the currently selected algorithm button.
* The component has a function handleButtonClick which takes a component argument representing the algorithm to display. This function sets showComponent and selectedButton to component.
* The component then renders a set of algorithm buttons and displays the selected algorithm component based on the showComponent state.



## ComponentRotary

* This component is used to display a set of algorithms for rotary drilling machines. It has the same structure as ComponentRotaryPercussive, with the only difference being the set of algorithms displayed.



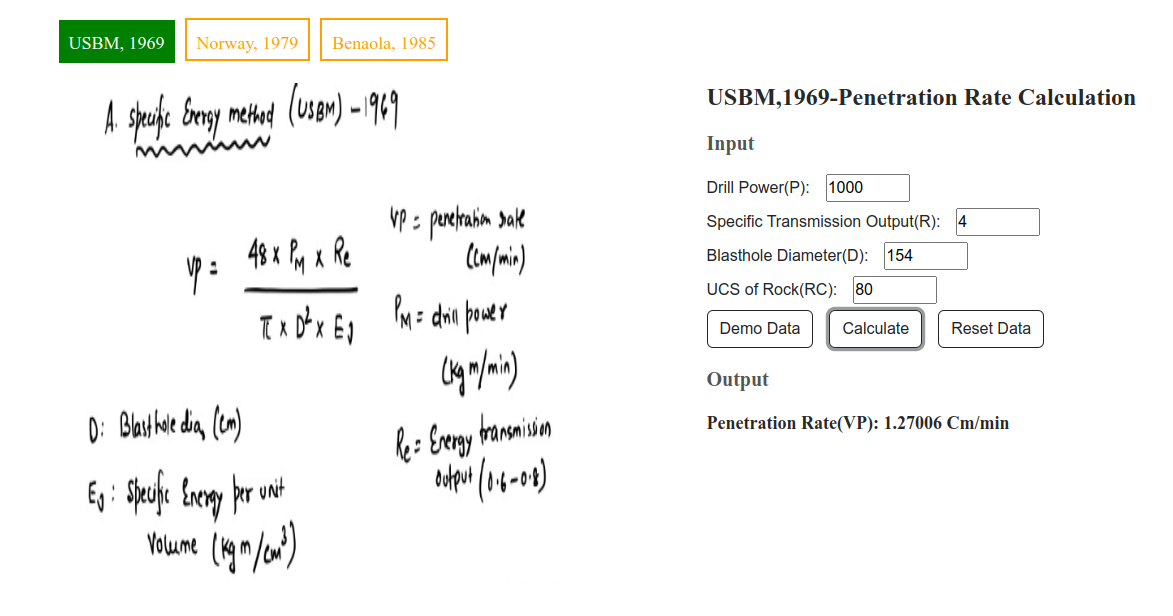
## ToolSelection

* This is the main component which renders the tool selection interface. It has the same structure as the other two components, with two states and a function handleButtonClick. It renders two buttons for selecting the drilling method and displays the appropriate component based on the showComponent state.



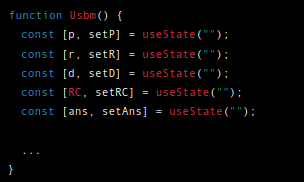
### 3.1.1 Rotary Percussive Drilling

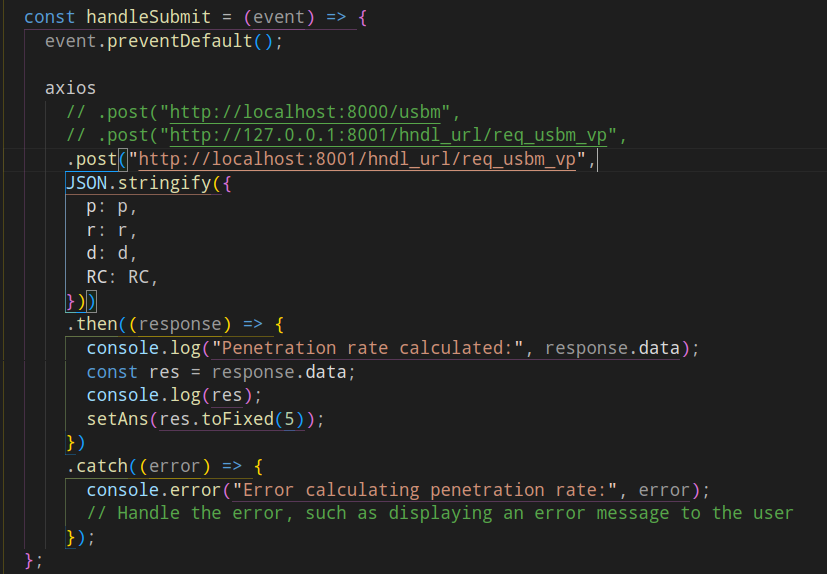
### USBM, 1969

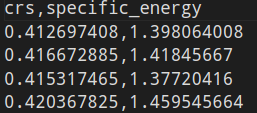


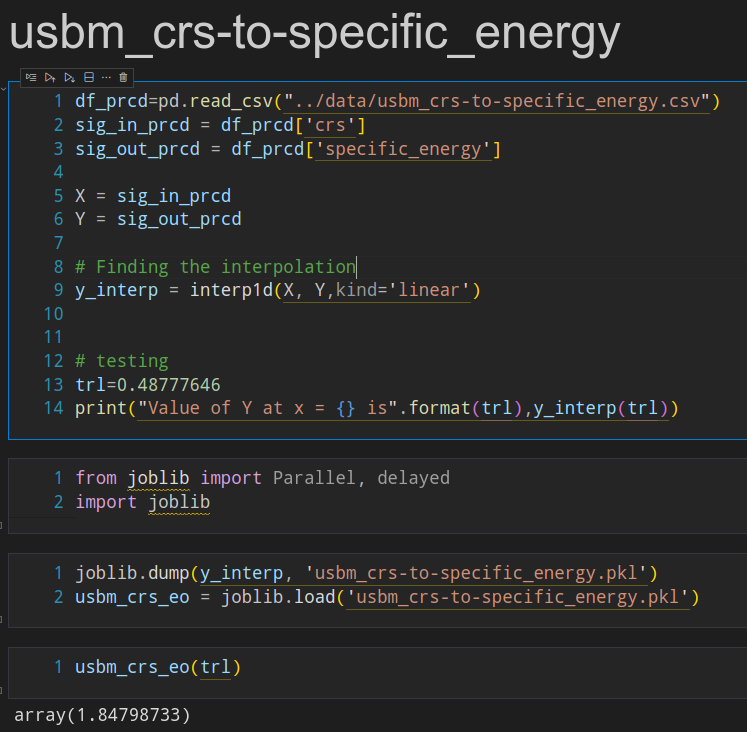
**Pseudocode for USBM, 1969**

**Steps:-**

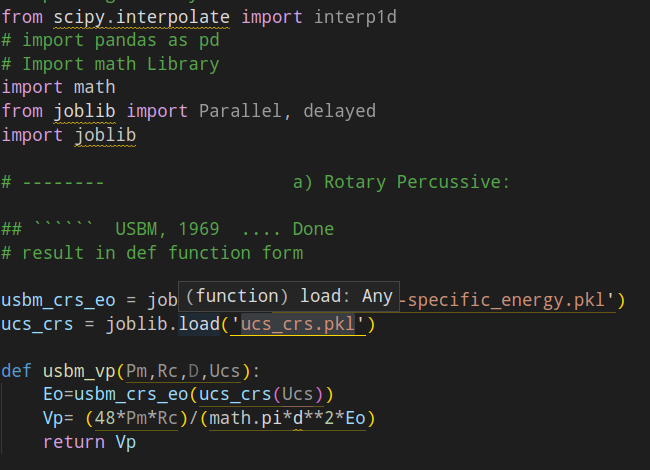
1. First Imports the required modules for the component. React and useState are imported from the 'react' module, axios is imported to make HTTP requests, and "./Usbm.css" is a CSS file for styling the component.
2. Now We have to Define a functional component named Usbm. It uses the useState hook to initialize and manage state variables p, r, d, RC, and ans.
3. After this, the Next code section defines the handleSubmit function, which is called when the user submits the form. It prevents the default behavior of the form submission and sends an HTTP POST request with user input data in .json format using axios. The request is sent to the URL "<http://localhost:8001/hndl_url/req_usbm_vp>" in the backend with the data p, r, d, and RC in the request body. If the request is successful, the response is logged to the console, and the result is stored in the state variable ans.



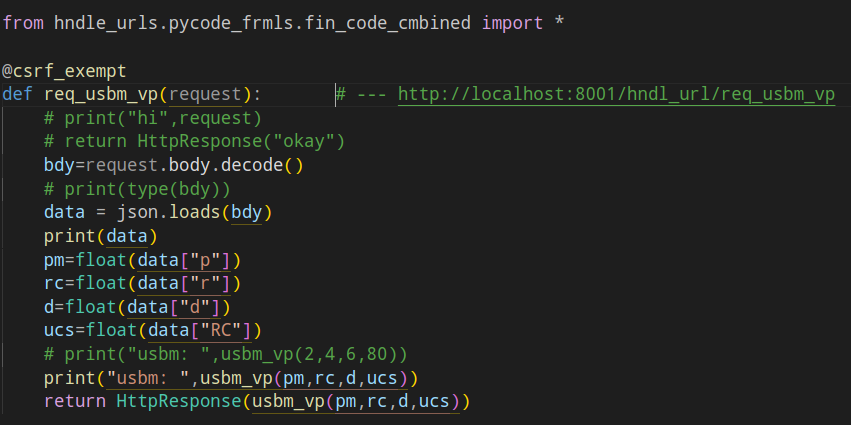
1. **In Backend** on specified URL "<http://localhost:8001/hndl_url/req_usbm_vp>", we have python code as shown below:-
2. First, We have to upload Image to webplotdigitizer( <https://apps.automeris.io/wpd/> - an online application to process images to digitize graphs).
3. Then, USBM crs to specific energy Graph is digitized.
4. Now, Crs to specific energy is interpolated using python scipy module to predict the value at intermediate crs.
5. Then, the Interpolated variable is saved to raw binary format (.pkl).

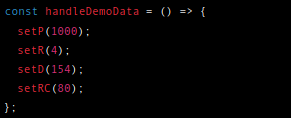


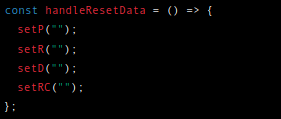
1. Later the same (usbm\_crs-to-specific\_energy.pkl) state variable is used to predict the specific energy coming from the frontend user.
2. The same thing is done with UCS to Crs prediction via (ucs\_crs.pkl).
3. Later everything is encoded into the python function.



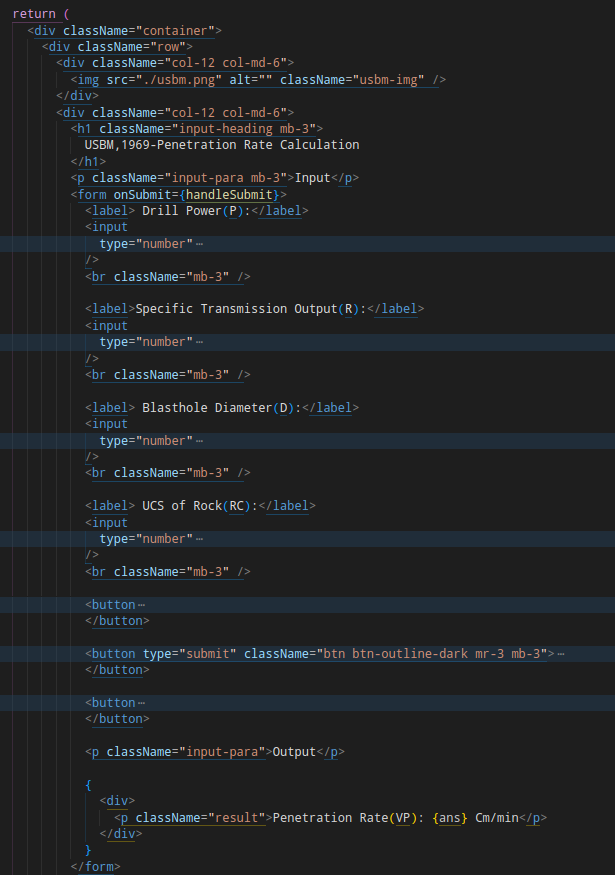
1. At the backend, the above function is processed over incoming user request via request handling and the result is returned as response to frontend user request, As shown below:-



1. **In Frontend**, We have defined the handleDemoData function, which sets the state variables p, r, d, and RC to preset demo data when called.
2. Then we have the handleResetData function, which resets the state variables p, r, d, and RC to their initial empty values.



1. Finally We have a return Statement in which we have code for displaying formula images on the webpage and also taking input from users along with 3 buttons to calculate, set demo data and reset the demo data to empty, And after that we are displaying results on Webpage which is stored in the state variable ans.

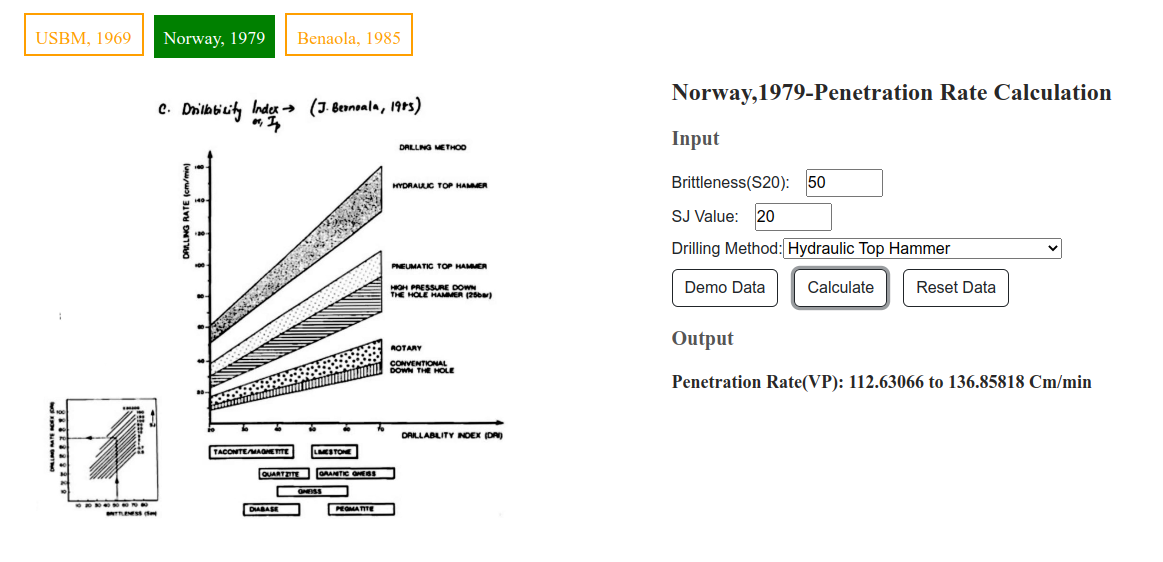


### Note- In Similar Manner, We have written Code for the remaining Rotary Percussive Drilling algorithms as well as for all Rotary Drilling algorithms.

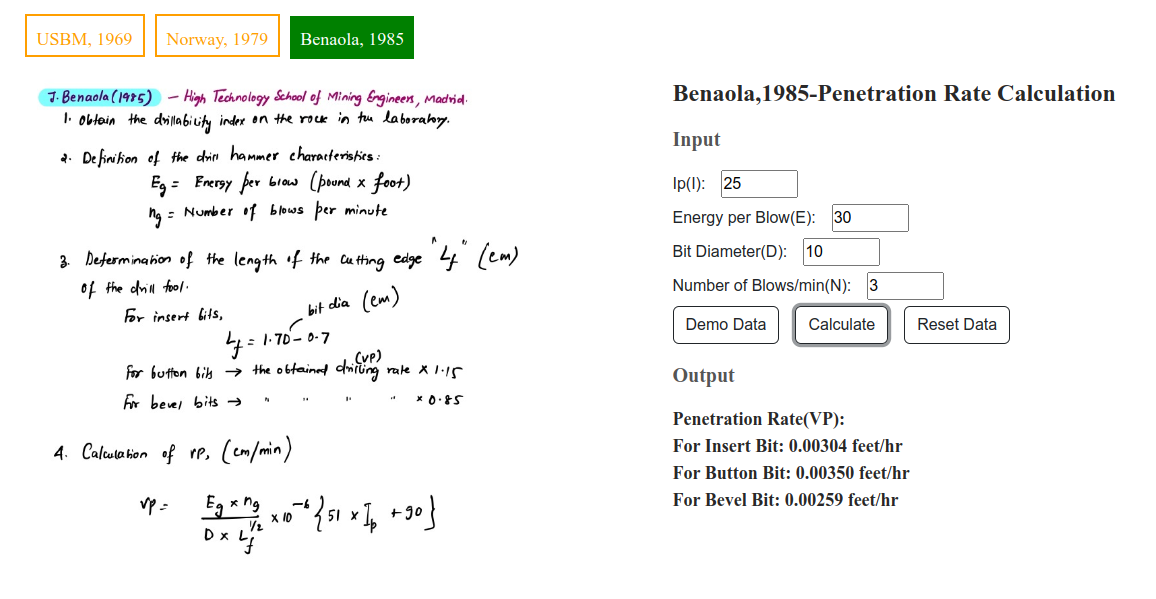
### The Github Repository contains the actual code.

### GitHub Repository Link- <https://github.com/Alok4k45h/MineCalc_Web.git>

### Norway, 1979 (Sample Output Web Page)

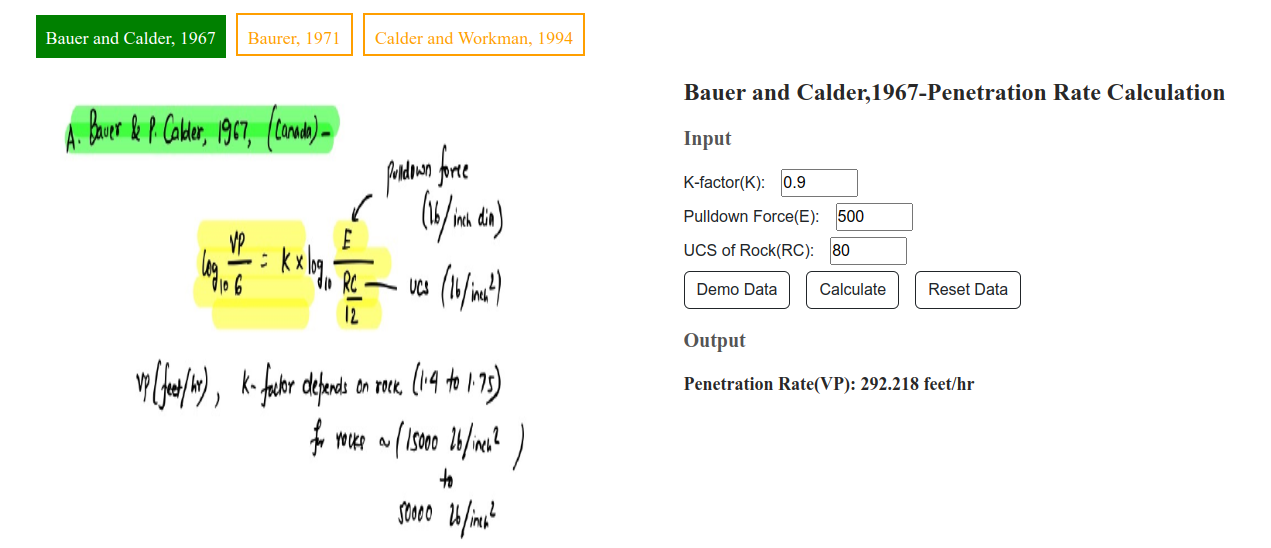


### Benaola, 1985 (Sample Output Web Page)

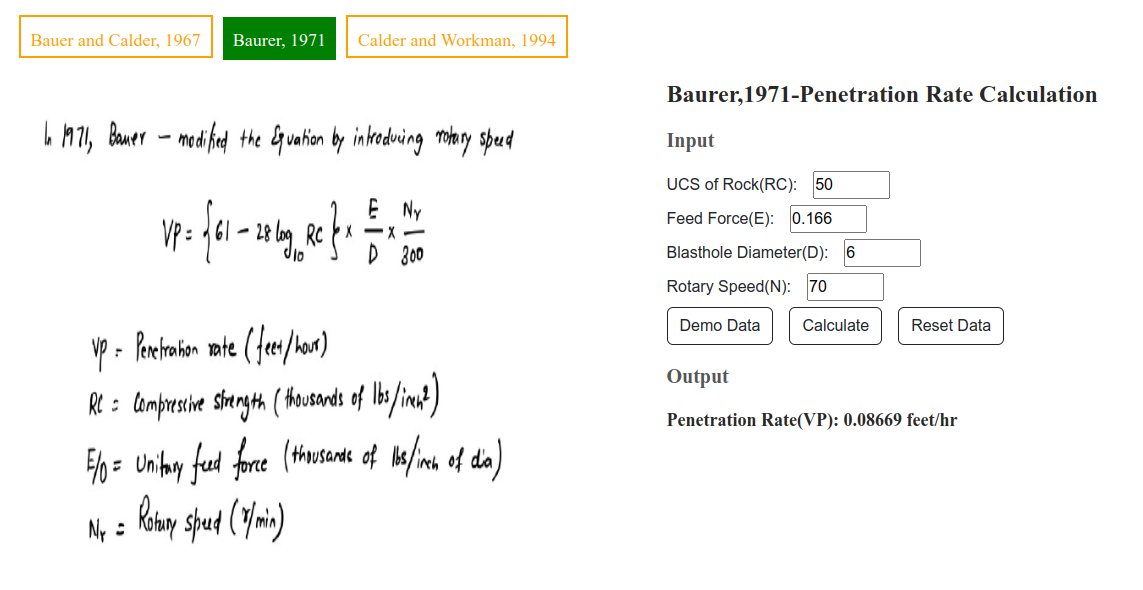


### 3.1.2 Rotary Drilling

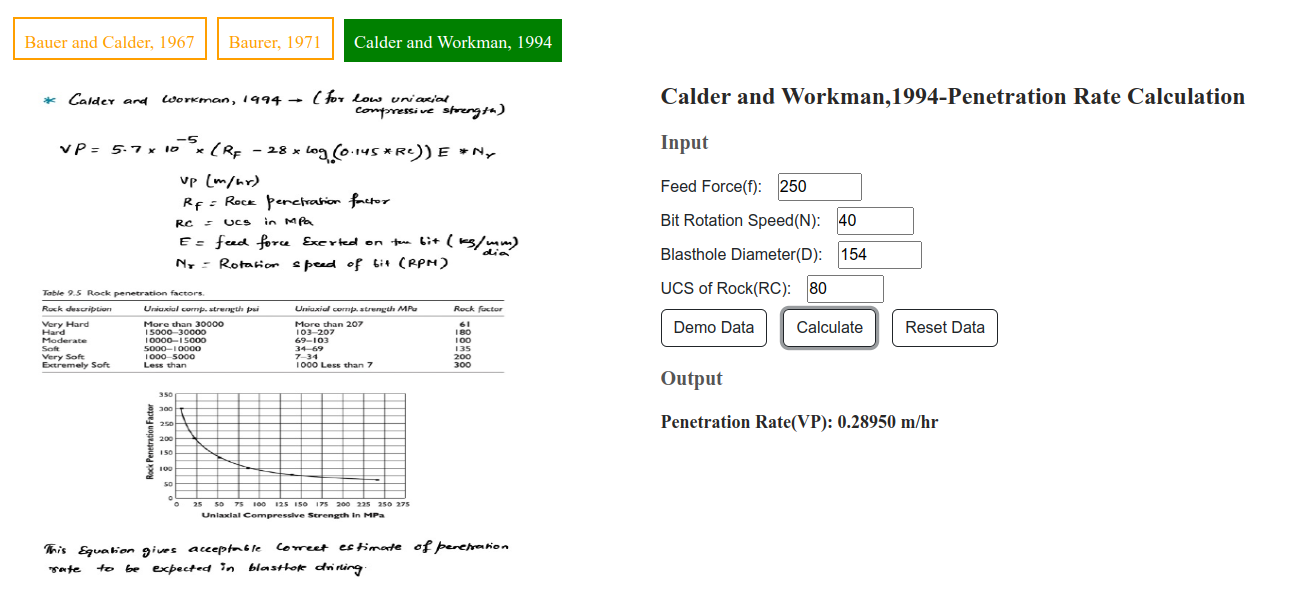
### Bauer & Calder, 1967 (Sample Output Web Page)



### Baurer, 1971 (Sample Output Web Page)



### Calder & Workman, 1994 (Sample Output Web Page)



### Conclusion:

Finally we have incorporated **Rotary Percussive( USBM-1969, Norway-1979, Benaola-985) and Rotary ( Bauer and Calder-1967, Baurer-1971, Calder and Workman-1994) algorithms** in our **MineCalc Website**. The user-friendly interface allows for easy input of data, and the remote accessibility of the platform ensures that users can access the website from anywhere. We are excited to provide this powerful resource to mining professionals and enthusiasts alike, and look forward to seeing the impact it will have on the industry.

**\*\*\*\*** **The End\*\*\*\***