Create a calculator app using Angular which is capable of performing following operations:

1. Addition of two numbers
2. Subtraction of two numbers
3. Multiplication of two numbers
4. Division of two numbers
5. Factorial of a number
6. Checking if a given number is Prime or not

CODE :

App.component.ts

app.component.ts × app.component.html app.component.css

calculator > src > app > app.component.ts > AppComponent > getAnswer > isPrime

```ts
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'calculator';
  subDisplayText = '';
  mainDisplayText = '';
  operand1!: number;
  operand2!: number;
  operator = '';
  calculationString = '';
  answered = false;

  operatorSet = false;

  pressKey(key: string) {
    if (key === '/' || key === 'x' || key === '-' || key === '+'|| key === 'Factorial'|| key === 'Prime') {
      const lastKey = this.mainDisplayText[this.mainDisplayText.length - 1];
      if (lastKey === '/' || lastKey === 'x' || lastKey === '-' || lastKey === '+'|| key === 'Factorial'|| key === 'Prime') {
        this.operatorSet = true;
      }
      if ((this.operatorSet) || (this.mainDisplayText === '')) {
        return;
      }
      this.operand1 = parseFloat(this.mainDisplayText);
      this.operator = key;
      this.operatorSet = true;
    }
    if (this.mainDisplayText.length === 10) {
      return;
    }
    this.mainDisplayText += key;
  }
  allClear() {
    this.mainDisplayText = '';
    this.subDisplayText = '';
    this.operatorSet = false;
  }
```

```
getAnswer() {
  this.calculationString = this.mainDisplayText;
  this.operand2 = parseFloat(this.mainDisplayText.split(this.operator)[1]);
  if (this.operator === '/') {
    this.subDisplayText = this.mainDisplayText;
    this.mainDisplayText = (this.operand1 / this.operand2).toString();
    this.subDisplayText = this.calculationString;
    if (this.mainDisplayText.length > 9) {
      this.mainDisplayText = this.mainDisplayText.substr(0, 9);
    }
  } else if (this.operator === 'x') {
    this.subDisplayText = this.mainDisplayText;
    this.mainDisplayText = (this.operand1 * this.operand2).toString();
    this.subDisplayText = this.calculationString;
    if (this.mainDisplayText.length > 9) {
      this.mainDisplayText = 'ERROR';
      this.subDisplayText = 'Range Exceeded';
    }
  } else if (this.operator === '-') {
    this.subDisplayText = this.mainDisplayText;
    this.mainDisplayText = (this.operand1 - this.operand2).toString();
    this.subDisplayText = this.calculationString;
  } else if (this.operator === '+') {
    this.subDisplayText = this.mainDisplayText;
    this.mainDisplayText = (this.operand1 + this.operand2).toString();
    this.subDisplayText = this.calculationString;
    if (this.mainDisplayText.length > 9) {
      this.mainDisplayText = 'ERROR';
      this.subDisplayText = 'Range Exceeded';
    }
  }
   else {
    this.subDisplayText = 'ERROR: Invalid Operation';
  }
```

```
      this.subDisplayText = this.mainDisplayText;
      this.mainDisplayText = (factorial).toString();
      this.subDisplayText = this.calculationString;
      if (this.mainDisplayText.length > 9) {
        this.mainDisplayText = 'ERROR';
        this.subDisplayText = 'Range Exceeded';
      }
    }
    else if (this.operator === 'Prime') {

      this.subDisplayText = this.mainDisplayText;
      this.mainDisplayText = (isPrime(this.operand1)).toString();
      this.subDisplayText = this.calculationString;
      if (this.mainDisplayText.length > 9) {
        this.mainDisplayText = 'ERROR';
        this.subDisplayText = 'Range Exceeded';
      }
    }
    else {
      this.subDisplayText = 'ERROR: Invalid Operation';
    }
    this.answered = true;
  }
}
function isPrime(num: number) {
  for(var i = 2; i < num; i++)
    if(num % i === 0) return 'Not Prime';
  return 'Prime';
}
function calcFact( num: number )
      {
      var i;
      var fact = 1;
      for( i = 1; i <= num; i++ )
      {
      fact = fact * i;
      }
      return fact;
      }
```

App.component.html

app.component.ts | app.component.html | app.component.css

calculator > src > app > app.component.html > body > div.container > div.row > div.col-md-4 > div.base > div.keypad > table > tr > td.keys.opkey

Go to component

```
<body>
  <div class="container">
    <div class="row">
      <div class="col-md-4"></div>
      <div class="col-md-4">
        <div class="base">
          <div class="maindisplay">
            <div class="subdisplay">{{ subDisplayText }}</div>
            {{ mainDisplayText }}
          </div>
          <div class="keypad">
            <table style="width: 100%;">
              <tr>
                <td class="keys ackey" colspan="3" (click)="allClear()">AC</td>
                <td class="keys opkey" colspan="1" (click)="pressKey('/')">/</td>
              </tr>
              <tr>
                <td class="keys numkey" (click)="pressKey('7')">7</td>
                <td class="keys numkey" (click)="pressKey('8')">8</td>
                <td class="keys numkey" (click)="pressKey('9')">9</td>
                <td class="keys opkey" (click)="pressKey('x')">x</td>
              </tr>
              <tr>
                <td class="keys numkey" (click)="pressKey('4')">4</td>
                <td class="keys numkey" (click)="pressKey('5')">5</td>
                <td class="keys numkey" (click)="pressKey('6')">6</td>
                <td class="keys opkey" (click)="pressKey('-')">-</td>
              </tr>
              <tr>
                <td class="keys numkey" (click)="pressKey('3')">3</td>
                <td class="keys numkey" (click)="pressKey('2')">2</td>
                <td class="keys numkey" (click)="pressKey('1')">1</td>
                <td class="keys opkey" (click)="pressKey('+')">+</td>
              </tr>
              <tr>
                <td colspan="2" class="keys numkey" (click)="pressKey('0')">0</td>
                <td class="keys numkey" (click)="pressKey('.')">.</td>
                <td class="keys equalkey" (click)="getAnswer()">=</td>
              </tr>
              <tr>
                <td colspan="2" class="keys opkey" (click)="pressKey('!')">Factorial</td>
                <td colspan="2" class="keys opkey" (click)="pressKey('P')">Prime</td>
              </tr>
            </table>
          </div>
        </div>
      </div>
      <div class="col-md-4"></div>
    </div>
  </div>

</body>
```

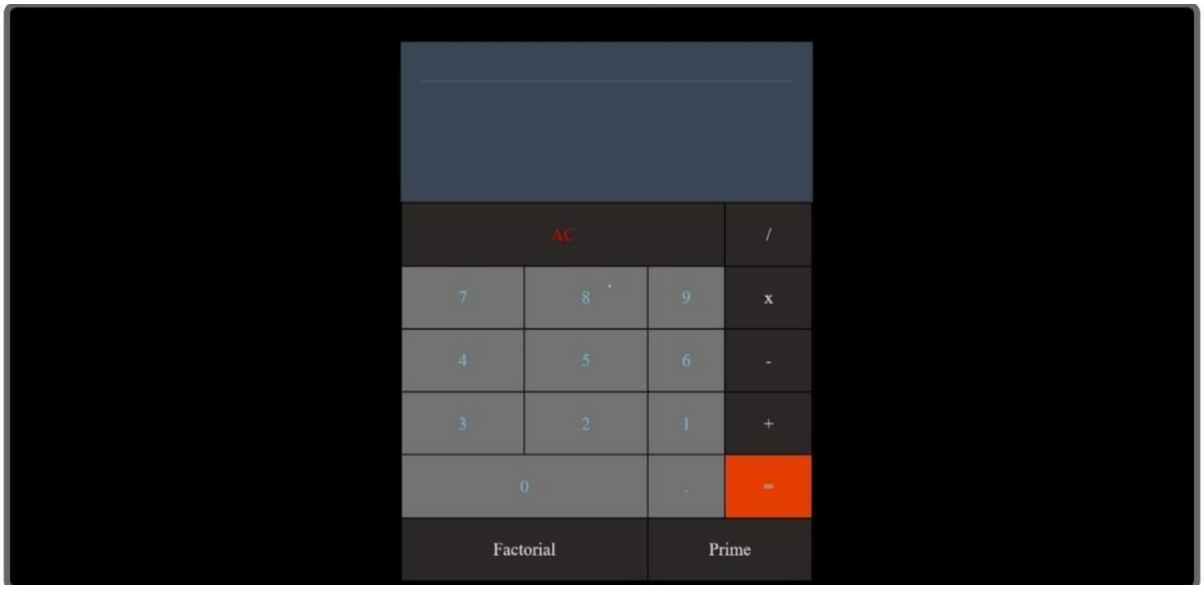
App.component.css

app.component.ts | app.component.html | app.component.css

calculator > src > app > app.component.css > .base

```css
body {
  background-color: #000000;
  box-shadow: 0px 0px 0px 10px #666;
  border: 5px solid black;
  border-radius: 10px;
}
.base {
  background: black;
  margin-top: 5vh;
  margin-left: 65vh;
  border: 3px solid black;
  width: 35%;

}

.maindisplay {
  background: #3A4655;
  height: 20vh;
  padding: 5% !important;
  font-size: 4rem;
  text-align: right;
  font-family: Courier, monospace;
  overflow: auto;
}

.subdisplay {
  border-bottom: 1px solid #727B86;
  height: 15%;
  font-size: 2rem;
  overflow: auto;
}

.keypad {
  height: calc(50%);
}

.keys {
  margin: 0;
  height: 5%;
  background: whitesmoke;
  color: #425062;
  padding: 5%;
  font-size: 2rem;
  text-align: center;
  cursor: pointer;
  opacity: 0.9;
}

.keys:hover {
  opacity: 1;
}

.ackey {
  color: red;
  background: rgb(48, 45, 45);
}

.equalkey {
  color: white;
  background-color: orangered;
}

.numkey {
  color: skyblue;
  background-color: grey;
}

.opkey {
  color: white;
  background-color: rgb(48, 45, 45);
}
```

SCREENSHOT :


AC
/
7
8
9
x
4
5
6
-
3
2
1
+
0
.
=
Factorial
Prime