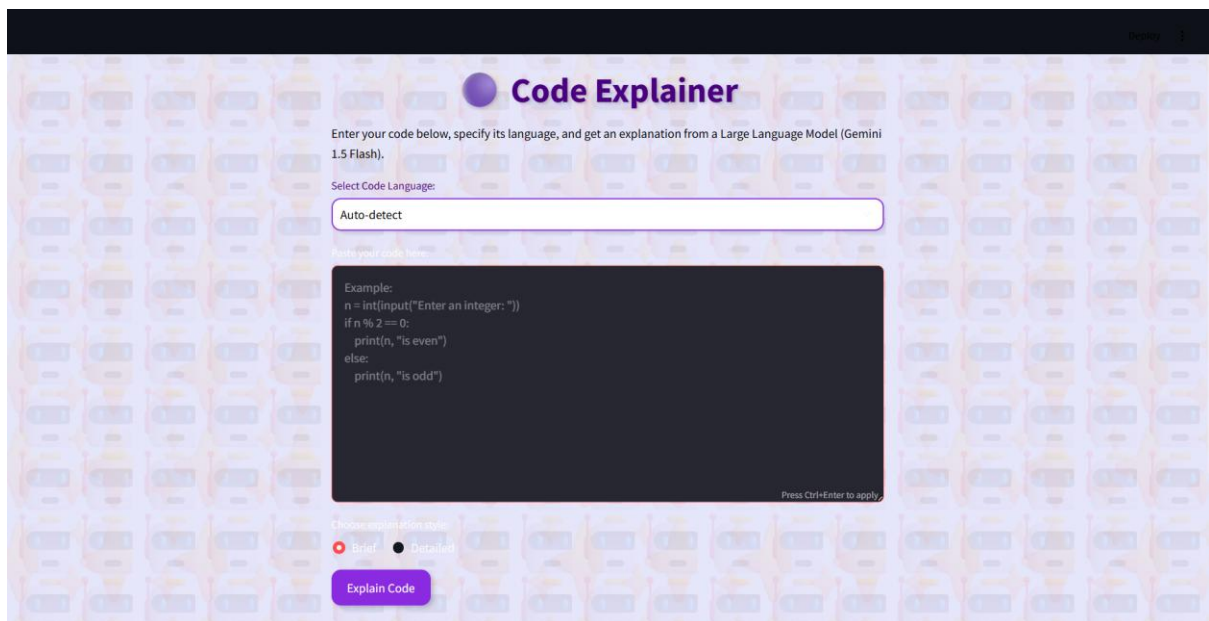


NAME: B.HARITHA

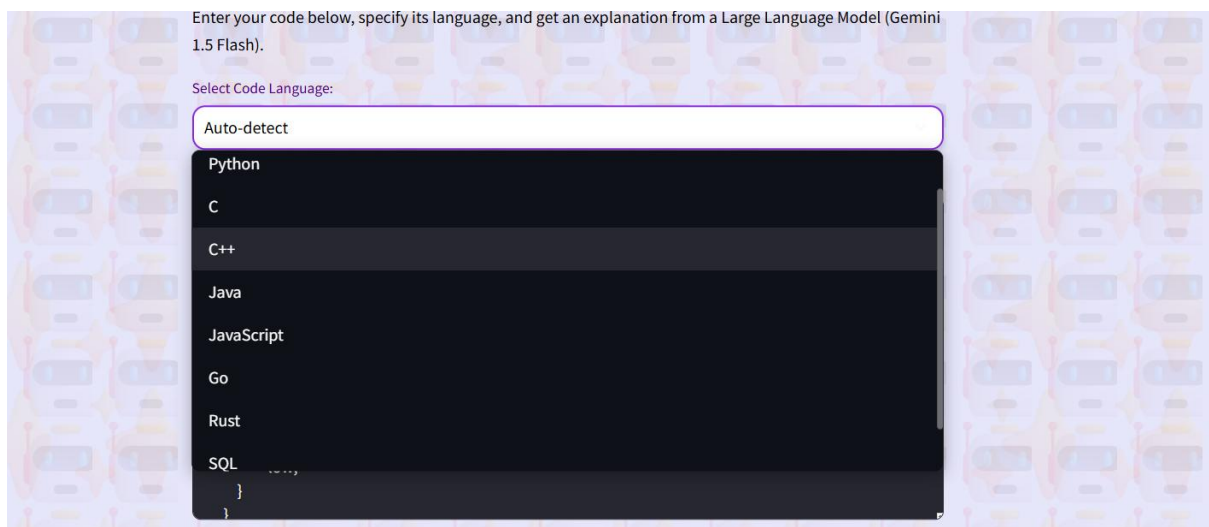
COLLEGE NAME: VIT VELLORE

# OUTPUT

This is how the interface looks:



These are the programming languages that this code explainer can explain( also has auto detect)

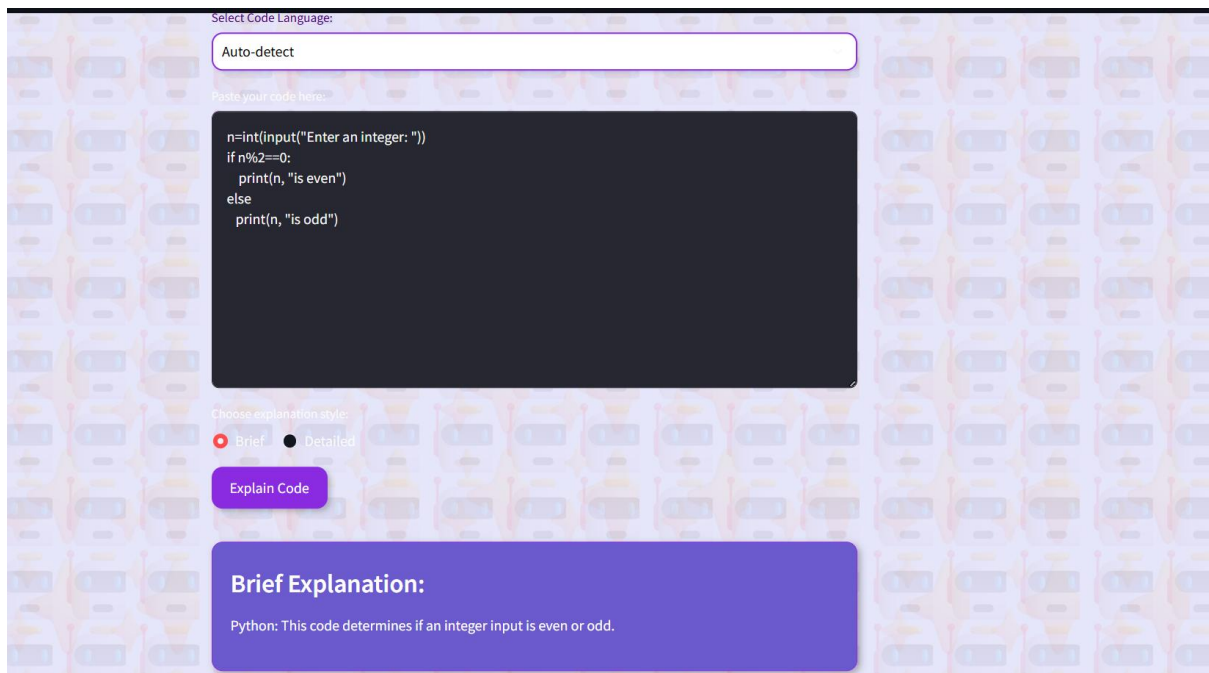


Also HTML/CSS(included)

## Python test:

Gives option to user to choose between Brief and Detailed explanation(according to their convenience)

### 1)Brief Explanation:



The screenshot shows a web application interface for explaining Python code. At the top, there is a dropdown menu labeled "Select Code Language:" with "Auto-detect" selected. Below this is a text area labeled "Paste your code here:" containing the following Python code:

```
n=int(input("Enter an integer: "))
if n%2==0:
    print(n, "is even")
else
    print(n, "is odd")
```

Below the code area, there is a section labeled "Choose explanation style:" with two radio buttons: "Brief" (selected) and "Detailed". A purple button labeled "Explain Code" is positioned below these options. At the bottom, a purple box displays the "Brief Explanation:" which states: "Python: This code determines if an integer input is even or odd."

### 2)Detailed Explanation:

Paste your code here:

```
n=int(input("Enter an integer:"))
if n%2==0:
    print(n,"is even")
else:
    print(n,"is odd")
```

Choose explanation style:

☒ Brief ☐ Detailed

Explain Code

**Detailed Explanation:**

This Python code determines whether a given integer is even or odd.

First, the program prompts the user to enter an integer using the `input()` function. The input, which is initially a string, is then converted to an integer using `int()`, and stored in the variable `n`.

Next, it uses a conditional statement (`if...else`). The condition `n % 2 == 0` checks if the remainder when `n` is divided by 2 is equal to 0. The `%` symbol is the modulo operator, which gives the remainder of a division.

If the remainder is 0 (meaning the number is divisible by 2), the code inside the `if` block executes, printing a message indicating that `n` is even.

Otherwise (if the remainder is not 0, meaning the number is not divisible by 2), the code inside the `else` block executes, printing a message indicating that `n` is odd.

In essence, the program takes a number as input, performs a simple divisibility test, and then outputs whether the number is even or odd based on the result of that test. The entire logic hinges on the property that even numbers are perfectly divisible by 2, leaving no remainder.

## Java Test:

### 1)Brief Explanation:

Paste your code here:

```
public class Prime {

    public static void main(String[] args) {

        int low = 20, high = 50;

        while (low < high) {
            if(checkPrimeNumber(low))
                System.out.print(low + " ");

            ++low;
        }
    }
}
```

Choose explanation style:

☒ Brief ☐ Detailed

Explain Code

**Brief Explanation:**

Java: This code prints all prime numbers within a specified range.

### 2)Detailed Explanation:

Select Code Language:  
Auto-detect

Select your code type:  
public class Prime {  
  
 public static void main(String[] args) {  
  
 int low = 20, high = 50;  
  
 while (low < high) {  
 if(checkPrimeNumber(low))  
 System.out.print(low + " ");  
  
 ++low;  
 }  
 }  
}

Choose explanation style:  
☒ Brief ☐ Detailed

Explain Code

### Detailed Explanation:

This Java code identifies and prints all prime numbers within a specified range.

The program starts by defining a range of numbers, from 20 to 50 (inclusive of 20, exclusive of 50). It then uses a `while` loop to iterate through each number in this range.

For each number, it calls a function called `checkPrimeNumber`. This function determines whether the given number is prime or not. It works by checking if the number is divisible by any number from 2 up to half of the number itself. If it finds a divisor, it means the number isn't prime, and the function immediately returns `false`. If the loop completes without finding any divisors, it means the number is prime, and the function returns `true`.

Back in the main loop, if `checkPrimeNumber` returns `true` (indicating a prime number), the number is printed to the console. The loop continues until it reaches the upper limit of the range (50).

In essence, the code systematically checks each number in the specified range for primality using a simple divisibility test, and only prints those numbers that pass this test. The `checkPrimeNumber` function is a helper function that encapsulates the logic for determining primality, making the main part of the code cleaner and easier to understand.

Similarly all other languages are tested.