Industrial Internship Project Documentation

COURSE NAME: Generative AI Applications using Vertex AI in collaboration with Google.

TEAM ID: SWTID1720075266

Team Members:

1. SAI DHANUSH KORLAM
2. MALLIDI SASI KIRAN REDDY
3. HIMAJA V
4. SHREEJA R

5. # CodeXchange: An AI-Powered Code Translator Tool Using Palm’s Chat-Baison-001

Project Description

CodeXchange is an innovative web application designed to streamline code translation and facilitate seamless collaboration among developers working with different programming languages. Whether you're transitioning applications between platforms, collaborating in multilingual teams, or reusing code across projects, CodeXchange empowers developers to effortlessly translate code snippets between various programming languages. Leveraging advanced translation algorithms and syntax analysis, CodeXchange ensures accurate and reliable code conversion while preserving the original functionality and logic. With its intuitive interface and comprehensive language support, CodeXchange revolutionizes the development workflow, enabling teams to work together efficiently, enhance code reusability, and accelerate project delivery.

Scenario 1: Platform Transition

CodeXchange assists developers in transitioning applications from one platform to another. For instance, a team working on an application written in Python needs to migrate it to Java to leverage Java's robustness and scalability in an enterprise

environment. By inputting the Python code snippets and selecting Java as the target language, developers receive accurately translated code that maintains the original functionality, streamlining the migration process and minimizing the risk of introducing errors.

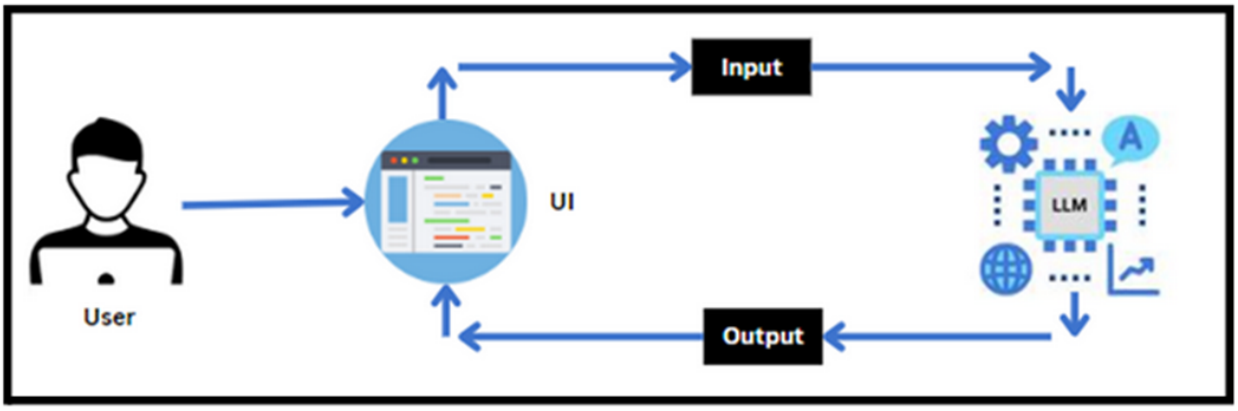
Scenario 2: Multilingual Collaboration

In a collaborative project where team members use different programming languages, CodeXchange facilitates seamless integration by translating code snippets as needed. Suppose one part of the team is proficient in C++ while another prefers Python. Developers can write code in their preferred language and use CodeXchange to translate it, ensuring all team members can work together efficiently without being constrained by language barriers. This enhances productivity and reduces the learning curve associated with adopting new languages.

Scenario 3: Code Reusability Across Projects

CodeXchange promotes code reusability by enabling developers to translate existing code into different languages for new projects. For example, a developer has written a set of utility functions in Java that would be beneficial for a new project being developed in C++. By translating these Java functions into C++ using CodeXchange, the developer can quickly integrate proven code into the new project, saving time and ensuring consistency across different projects.

Technical Architecture:



Project Flow

- User interacts with the UI developed by Streamlit framework.
- The input text(i.e. input code) is then passed to the LLM model specified for the project, chat-bison in this case.
- The input text is then analysed and processed by the model.
- Model autonomously generates the output code in preferred target language.
- The dynamically generated output is passed on back to the UI where user can interact to get deeper insights.

To accomplish this, following activities should be performed:

- Requirements Specification
    - Create a requirements.txt file to list the required libraries
    - Install the required libraries

- Initialization of the Model
    - Generate PALM API
    - Initialize the pre-trained model
- Interfacing with pre-trained Model
    - Write function to change programming language of the code
- Model Deployment
    - Give project title, description and scenarios
    - Translate the code given by user
    - Run the web application

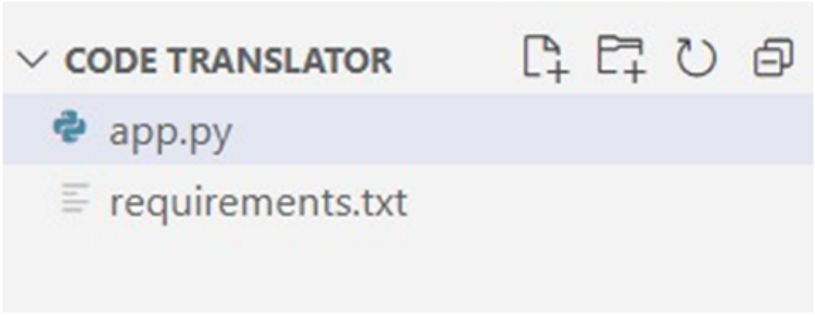
Prior Knowledge

One should have prior knowledge on following topics to complete this project:

- LLM & PALM: https://cloud.google.com/vertex-ai/docs/generative-ai/learn-resources
- Streamlit: https://www.datacamp.com/tutorial/streamlit

Project Structure

- Create the project folder consisting of files as shown below.

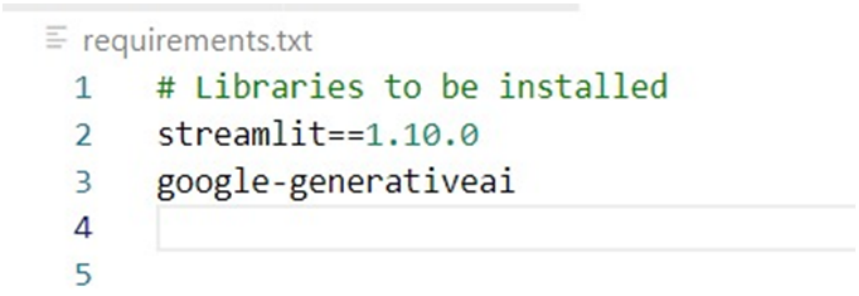


- app.py: An application file consisting of the logic for operations and the Streamlit code for the UI.
- requirements.txt: Text file consisting of the required libraries for the project implementation.

Milestone 1: Requirements Specification

Specifying the required libraries in the requirements.txt file ensures seamless setup and reproducibility of the project environment, making it easier for others to replicate the development environment.

- **Activity 1: Create requirements.txt file**



- Streamlit: Streamlit is a popularPythonframework for developing interactive web applications.
- google-generative:It is a Pythonclient library for accessing Google generative AI API, facilitating interactions with several pre-trained models

- **Actvity 2: Install the required libraries**

```
C:\Windows\System32>cd C:\Users\kiran\Desktop\Code Translator

C:\Users\kiran\Desktop\Code Translator>pip install -r requirements.txt
```

- Open the command prompt or terminal in administrator mode since it may involve modifications to specific secure locations.
- Run the command **“pip install -r requirements.txt”** to install all the requirements specified in the file.

Milestone 2: Initialization of the Model

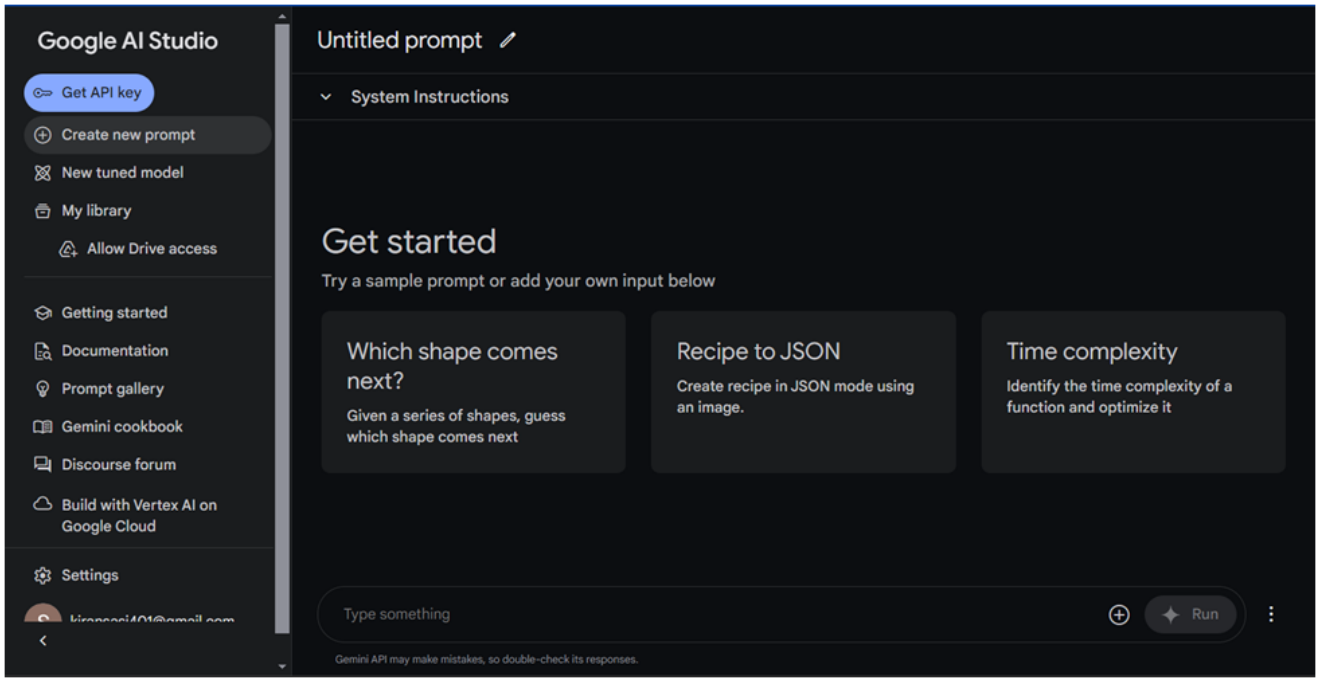
The Google API key is a secure access token provided by Google, enabling developers to authenticate and interact with various Google APIs. It acts as a form of identification, allowing users to access specific Google services and resources.

This key plays a crucial role in authorizing and securing API requests, ensuring that only authorized users can access and utilize Google's services.For initializing the model we need to generate PALM API.

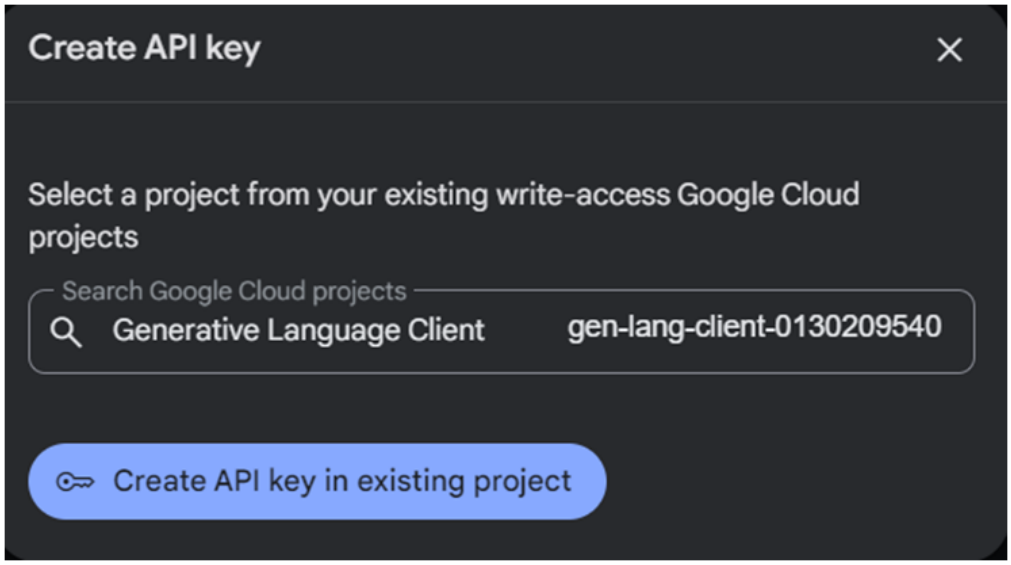
- **Activity 1: Generating PALM API key**

  - Refer to the below link for generating a new API key provided by Google.https://aistudio.google.com/app/prompts/new_chat

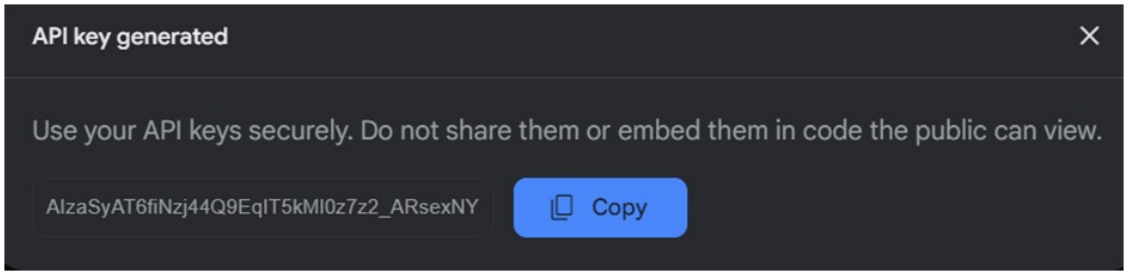
Sign in to the google account and get the below page.



Click on Get API key, and then click on “Create API key” . The select “Generative Language Client” for Google cloud projects and create.



Now, copy the generated API key or loading the pre-trained models.



- **Activity 2: Initialize the pre-trained model**

- Import the necessary libraries installed through the requirements.txt file previously.

```
import streamlit as st
import google.generativeai as palm
```

Streamlit: Import streamlit framework as **st**for creating UI directly with the python script.

Palm: Import palm module from google.generativeai package for accessing google pre-trained models.

- Now, configure the PALM API with the API key generated.

```
# Configuring Palm API with API key
palm.configure(api_key="AIzaSyAT6fiNzj44Q9EqIT5kMl0z7z2_ARsexNY")
```

Here configure function is used for configuring the Google API with the API key generated **“AIzaSyAT6fiNzj44Q9EqIT5kMl0z7z2_ARsexNY”.**

Now, specify the model to be used.

```
5   # Define model to be used
6   model_name="models/chat-bison-001"
```

This line of code will initialize the model_name variable with the pre-trained "chat-bison-001" for further API calls.

Milestone 3: Interfacing with Pre-trained Model

In this milestone, we will build a prompt template to generate code based on user description and language.

- **Activity 1: Function to change the programming language of the code**

```
8   # Define function to translate code from one language to other
9   def translate_code(code_snippet,target_language):
10      prompt=f"Translate the following code to {target_language}:\n\n{code_snippet}"
11      response=palm.chat(model=model_name,messages=[prompt])
12      return response.candidates[0]["content"]
13
```

  - The **"translate_code"** function will take the input of user-written code and

the specification of the target language.

- It builds a prompt for the model to translate the code.
- Then, the prompt along with the inputs is passed to the chat-bison-001 model through **“chat”** function of the PALM API.
- Finally, the function returns the first candidate of the result(i.e. translated code) generated by the model.

Milestone 4: Model deployment

In this milestone, the created codeXchange model is deployed for the users to easily interact with the model through the simple UI in browser built through the Streamlit, which is an open source python framework for creating interactive web applications with low-code.

- **Activity 1: Describe Project Title, Description and Scenarios**

```python
# Streamlit application
def main():
    st.title("CodeXchange: AI-Powered Code Translation Tool")

    st.markdown("""
    <h2>Project Description</h2>
    <p>CodeXchange is an innovative web application designed to streamline code translation and facilitate seamless collaboration among
    developers</p>""", unsafe_allow_html=True)

    st.markdown("""
    <h2>Scenario 1: Platform Transition</h2>
    <p>CodeXchange assists developers in transitioning applications from one platform to another. For instance, a team working on an
    application written in Python needs to migrate it to Java to leverage Java's robustness and scalability in an enterprise
    environment. By inputting the Python code snippets and selecting Java as the target language, developers receive accurately
    translated code that maintains the original functionality, streamlining the migration process and minimizing the risk of
    introducing errors</p>""", unsafe_allow_html=True)

    st.markdown("""
    <h2>Scenario 2: Multilingual Collaboration</h2>
    <p>In a collaborative project where team members use different programming languages, CodeXchange facilitates seamless integration
    by translating code snippets as needed. Suppose one part of the team is proficient in C++ while another prefers Python. Developers
    can write code in their preferred language and use CodeXchange to translate it, ensuring all team members can work together
    efficiently without being constrained by language barriers. This enhances productivity and reduces the learning curve associated
    with adopting new languages.</P>""", unsafe_allow_html=True)

    st.markdown("""
    <h2>Scenario 3: Code Reusability Across Projects</h2>
    <p>CodeXchange promotes code reusability by enabling developers to translate existing code into different languages for new
    projects. For example, a developer has written a set of utility functions in Java that would be beneficial for a new project being
    developed in C++. By translating these Java functions into C++ using CodeXchange, the developer can quickly integrate proven code
    into the new project, saving time and ensuring consistency across different projects.</p>""", unsafe_allow_html=True)
```

- This main() function serves as the entry point for the application.

- It involves the streamlit code for creating a simple UI for users to interact with the model.

- It involves a title, a comprehensive description of the model, and it’s ability to streamline the code translation for the developers to collaborate among themselves improving the efficiency and production.

- **Activity 2: Translate the code given by the user**

```
    st. subheader("Code Translation")
    source_code_snippet = st.text_area("Enter the code snippet you want to translate:")
    target_language = st.selectbox("Select the target programming language:", ["Python", "Java", "C++"])
    if st.button("Translate Code"):
        if source_code_snippet.strip():
            with st.spinner("Translating code ... "):
                try:
                    translated_code = translate_code(source_code_snippet, target_language)
                    st.success("Code translation complete!")
                    st.code(translated_code, language=target_language. lower())
                except Exception as e:
                    st.error(f"Error translating code: {e}")
        else:
            st.error("Please enter a code snippet.")
if __name__ == "__main__":
    main()
```

- Here, a sub-header “Code Translation” is visible followed by a text area where the user can enter his code.

- Then, a dropdown menu is displayed with several target programming languages like C, C++, Java, Python, and JavaScript among which the user can choose.

- Next, a button named “Translate Code” is created, if it is clicked, subsequent code will be executed.

- If the text area is not empty, a spinner with the indication of “translating

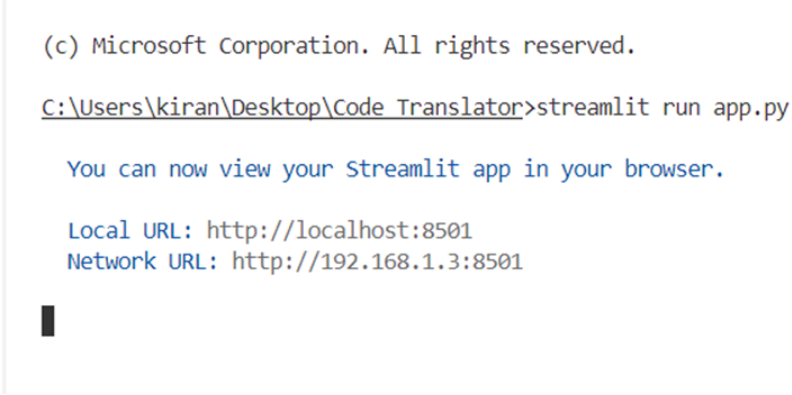
code…” will appear.

- An error is displayed if any exception occurs in the try block.
- Then, **“translate_code’** function is called to translate the code to the specified target language.
- Then, a success message will be displayed along with the translated code in the code block created.
- If the user has not entered any code and clicked the translate button, error will be displayed appropriately.
- Finally, the main() function is executed whenever the app.py is executed directly.

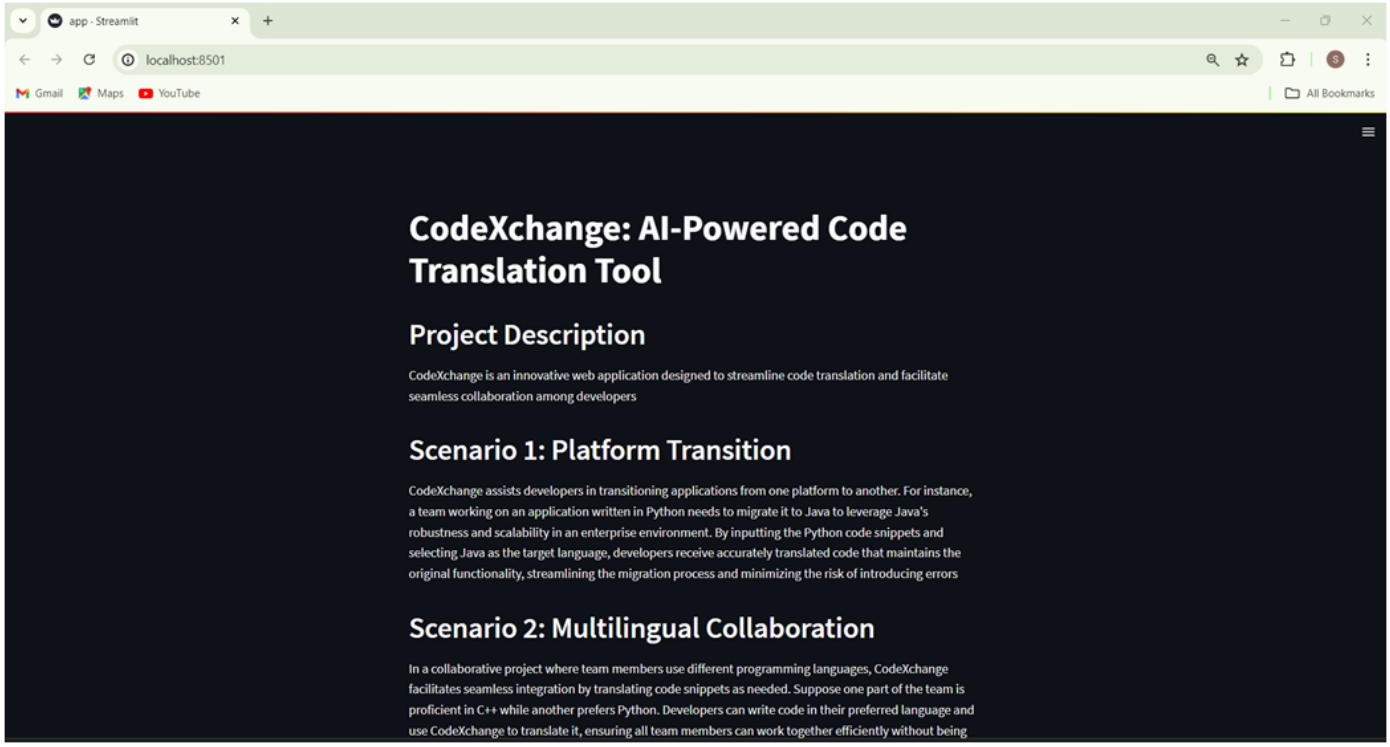
- **Activity 3: Run the Web Application**

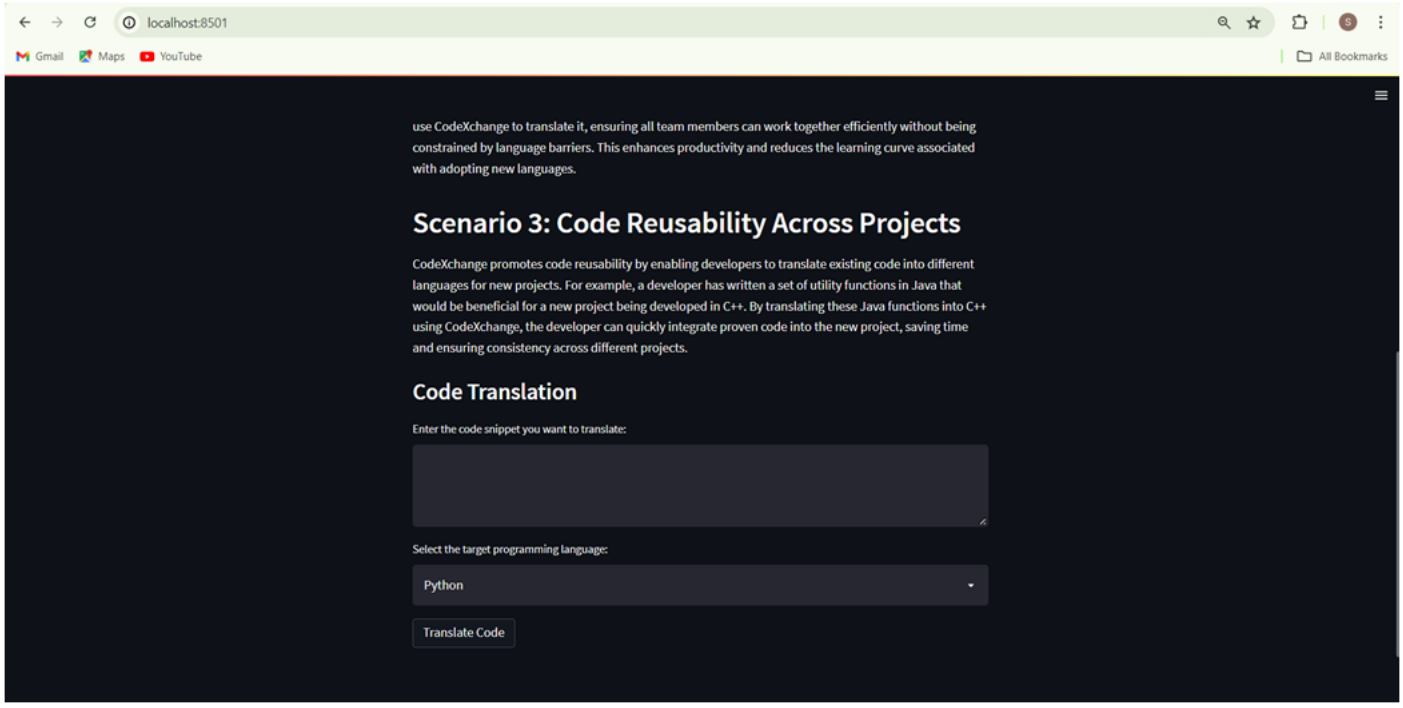
- Now, open the terminal in the IDE(i.e. Visual Studio Code) which is by default navigated to the project folder’s location.

- Type “streamlit run app.py” command to directly run the script.



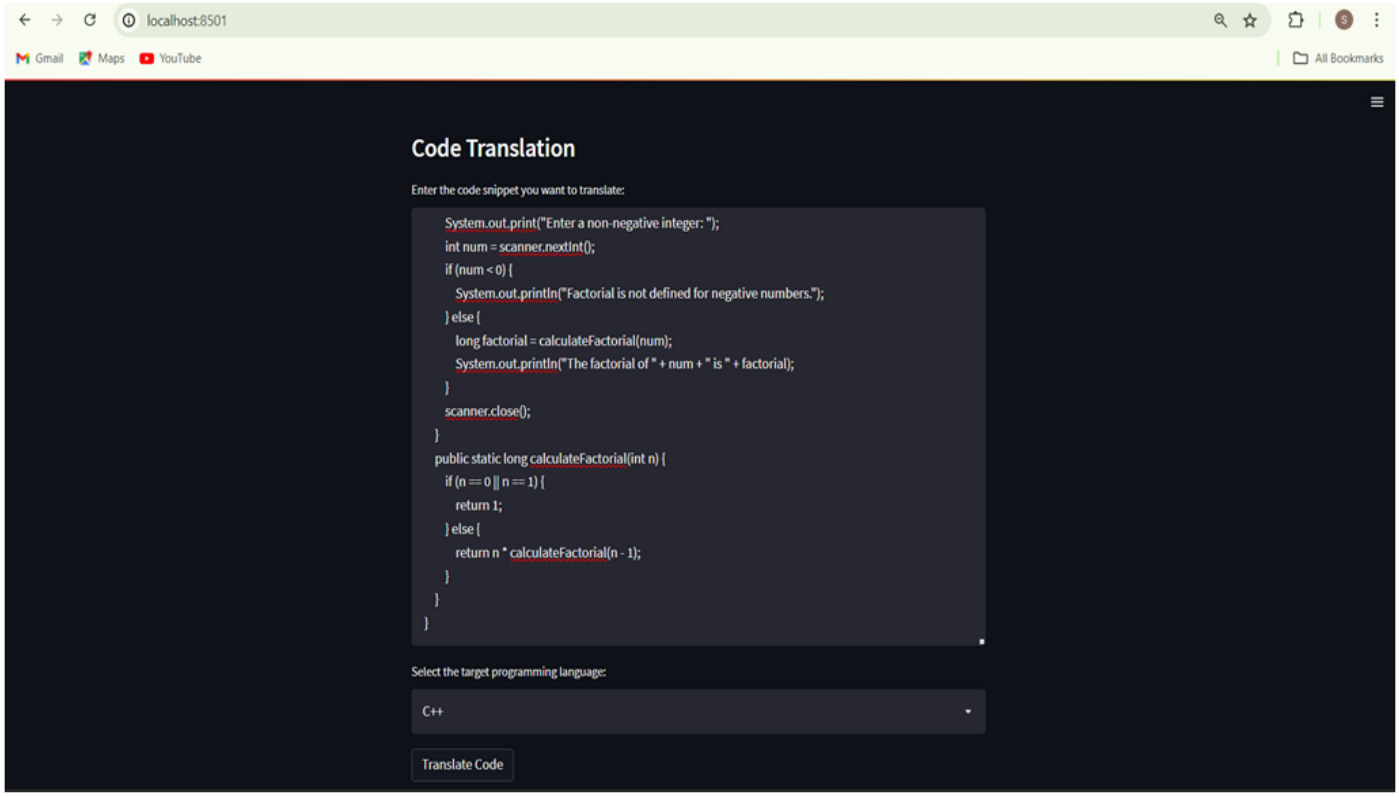
- Now, navigate to the localhost with specified port in the browser to view the web application.

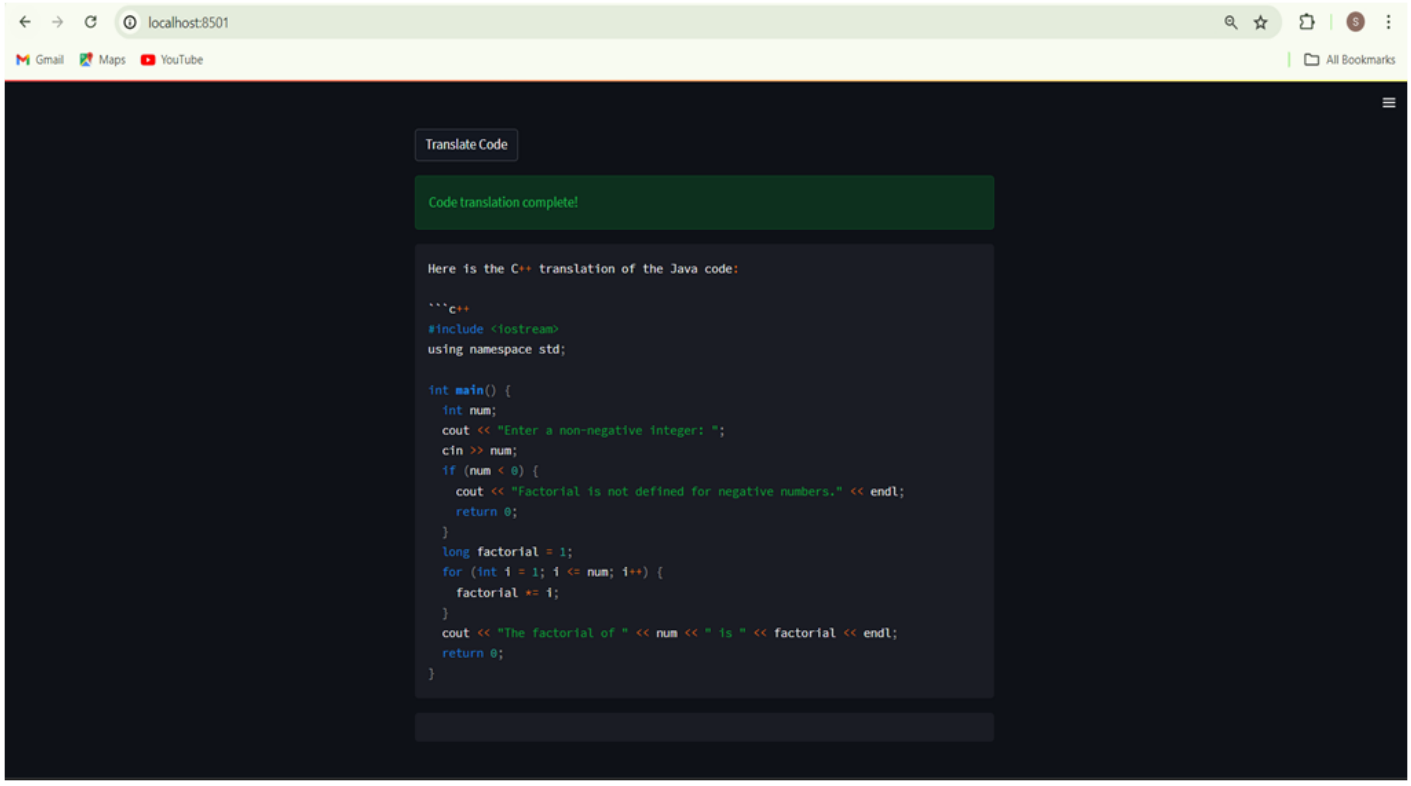

app - Streamlit
localhost:8501
Gmail
Maps
YouTube
All Bookmarks
CodeXchange: AI-Powered Code Translation Tool
Project Description
CodeXchange is an innovative web application designed to streamline code translation and facilitate seamless collaboration among developers
Scenario 1: Platform Transition
CodeXchange assists developers in transitioning applications from one platform to another. For instance, a team working on an application written in Python needs to migrate it to Java to leverage Java's robustness and scalability in an enterprise environment. By inputting the Python code snippets and selecting Java as the target language, developers receive accurately translated code that maintains the original functionality, streamlining the migration process and minimizing the risk of introducing errors
Scenario 2: Multilingual Collaboration
In a collaborative project where team members use different programming languages, CodeXchange facilitates seamless integration by translating code snippets as needed. Suppose one part of the team is proficient in C++ while another prefers Python. Developers can write code in their preferred language and use CodeXchange to translate it, ensuring all team members can work together efficiently without being


localhost:8501
Gmail
Maps
YouTube
All Bookmarks
use CodeXchange to translate it, ensuring all team members can work together efficiently without being constrained by language barriers. This enhances productivity and reduces the learning curve associated with adopting new languages.
Scenario 3: Code Reusability Across Projects
CodeXchange promotes code reusability by enabling developers to translate existing code into different languages for new projects. For example, a developer has written a set of utility functions in Java that would be beneficial for a new project being developed in C++. By translating these Java functions into C++ using CodeXchange, the developer can quickly integrate proven code into the new project, saving time and ensuring consistency across different projects.
Code Translation
Enter the code snippet you want to translate:
Select the target programming language:
Python
Translate Code

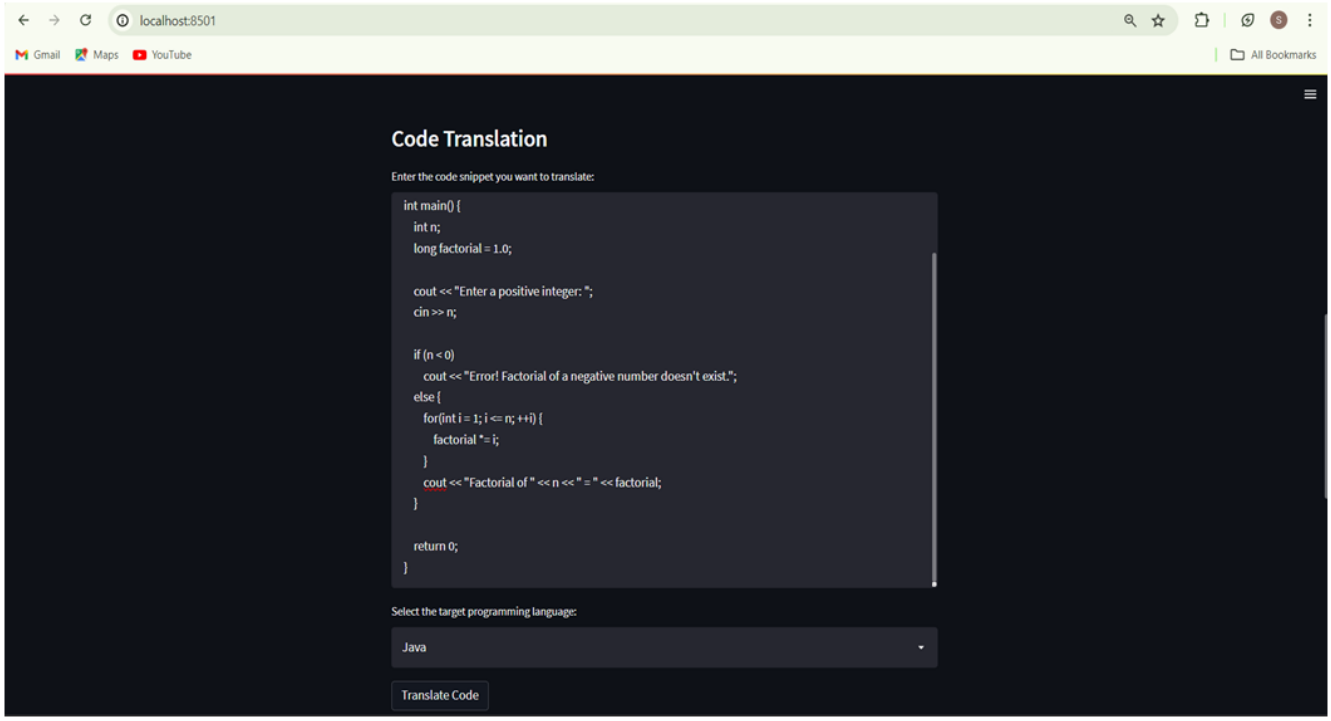
Output/Test Cases:

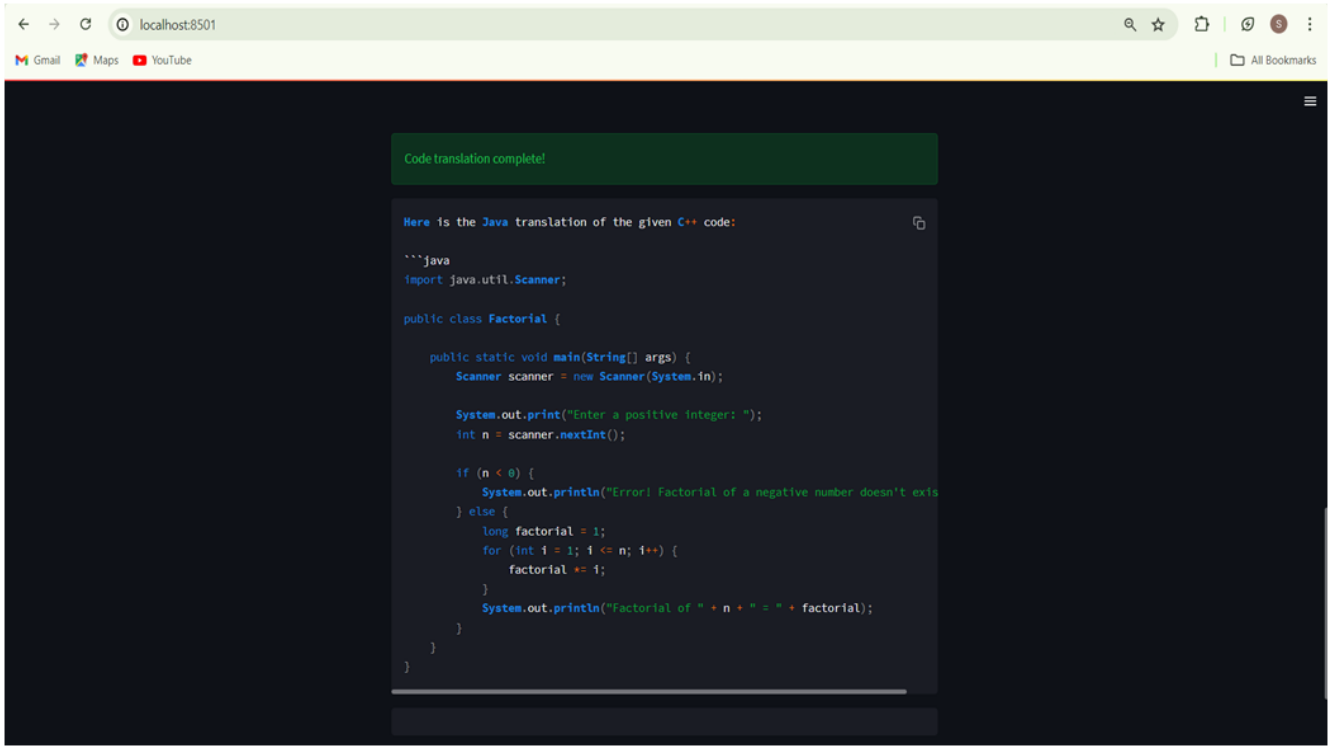
1. **Translating Java code to C++**



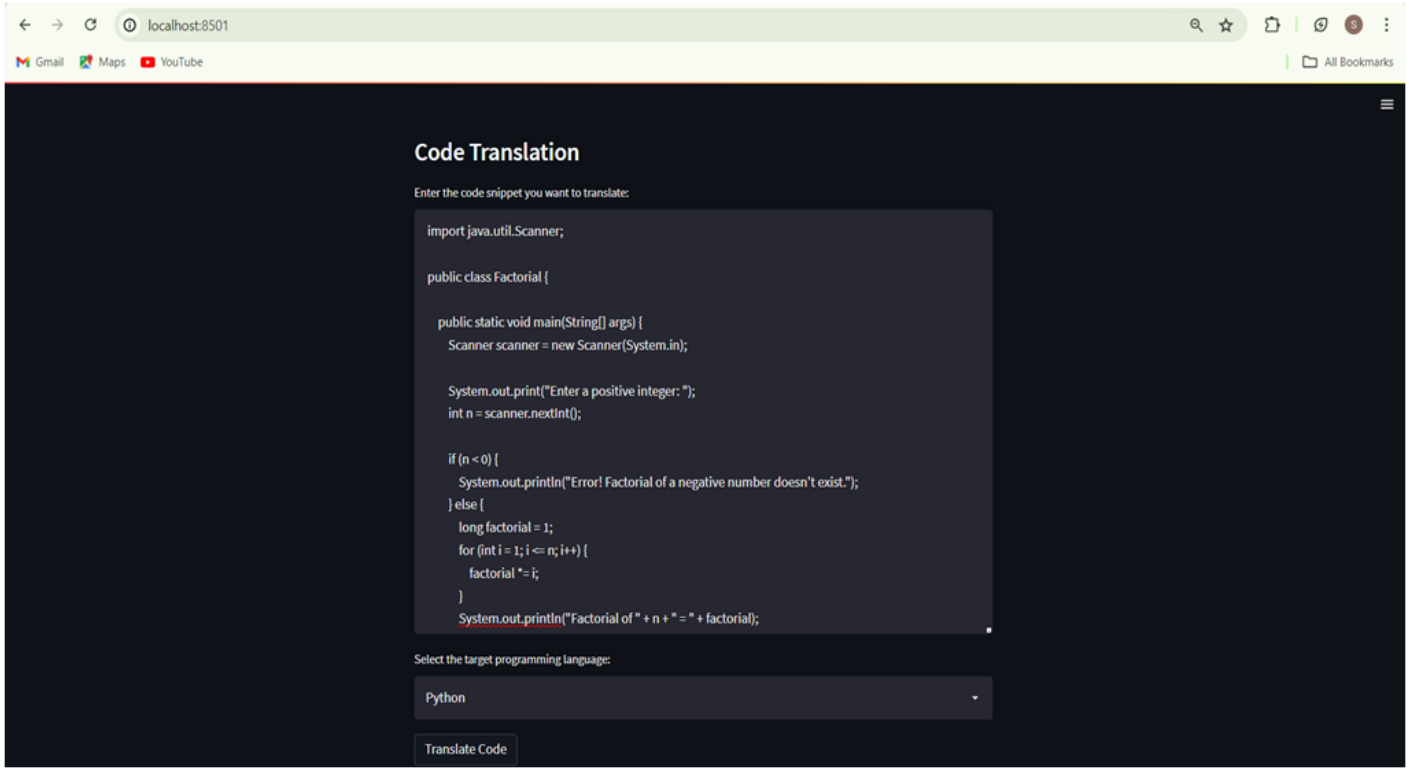


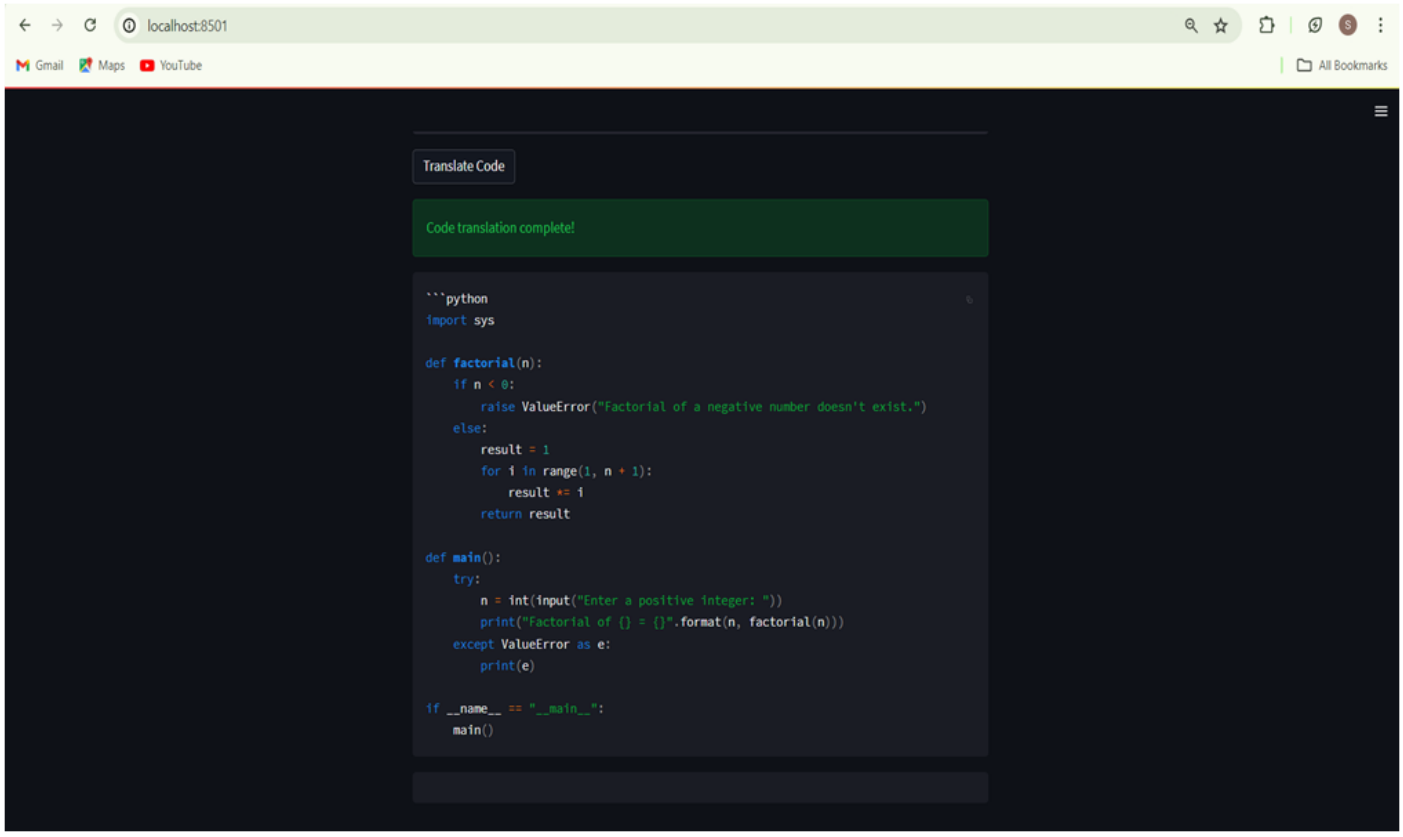
2. **Translating C++ code to Java**


localhost:8501
Gmail
Maps
YouTube
All Bookmarks
Code Translation
Enter the code snippet you want to translate:
int main() {
int n;
long factorial = 1.0;
cout << "Enter a positive integer: ";
cin >> n;
if (n < 0)
cout << "Error! Factorial of a negative number doesn't exist.";
else {
for(int i = 1; i <= n; ++i) {
factorial *= i;
}
cout << "Factorial of " << n << " = " << factorial;
}
return 0;
}
Select the target programming language:
Java
Translate Code

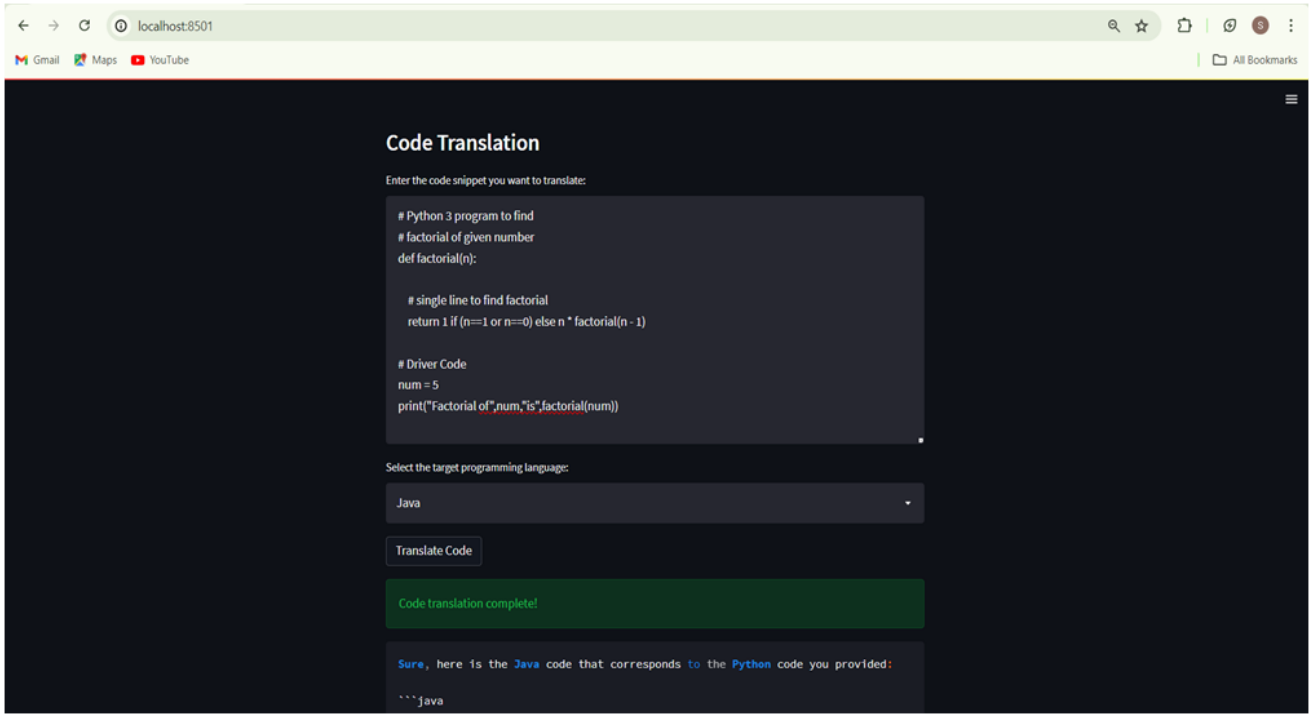

localhost:8501
Gmail
Maps
YouTube
All Bookmarks
Code translation complete!
Here is the Java translation of the given C++ code:
```java
import java.util.Scanner;
public class Factorial {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.print("Enter a positive integer: ");
int n = scanner.nextInt();
if (n < 0) {
System.out.println("Error! Factorial of a negative number doesn't exis
} else {
long factorial = 1;
for (int i = 1; i <= n; i++) {
factorial *= i;
}
System.out.println("Factorial of " + n + " = " + factorial);
}
}
}
```

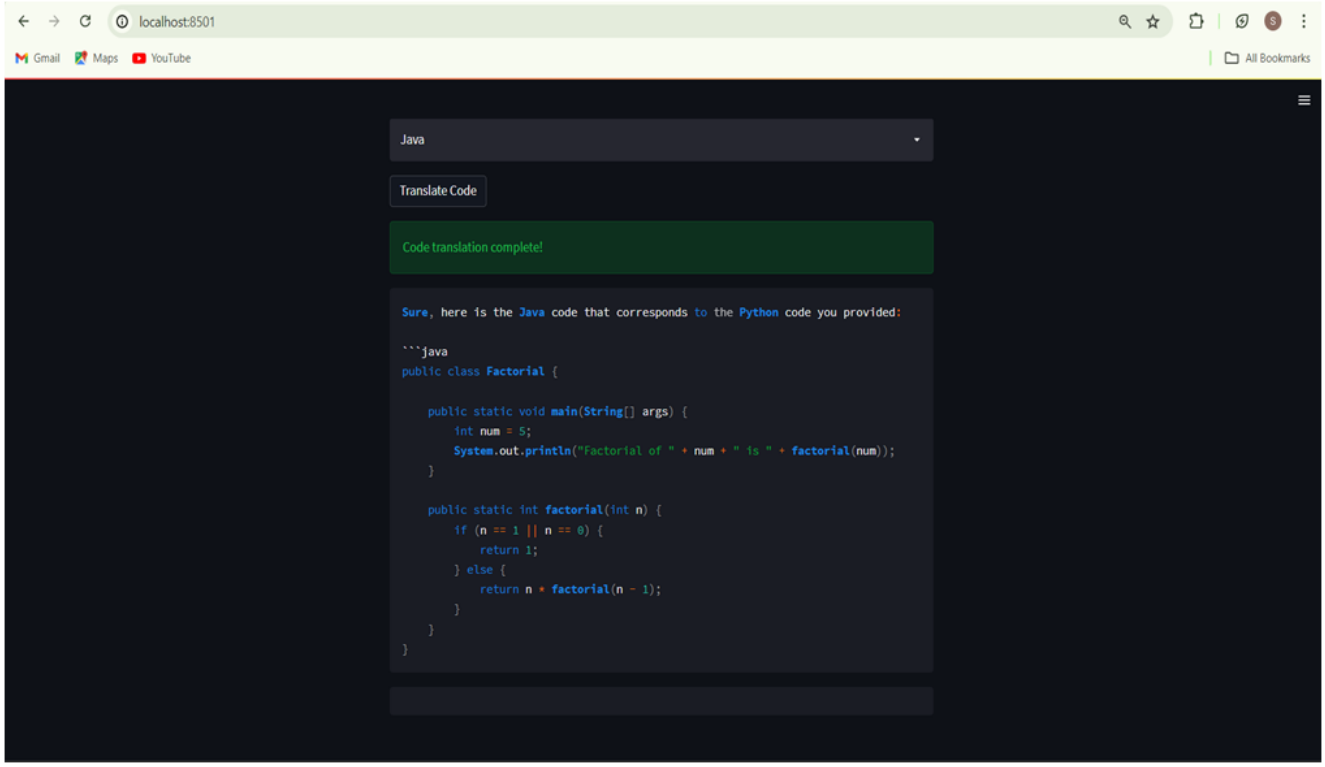
3. **Translating Java code to Python:**



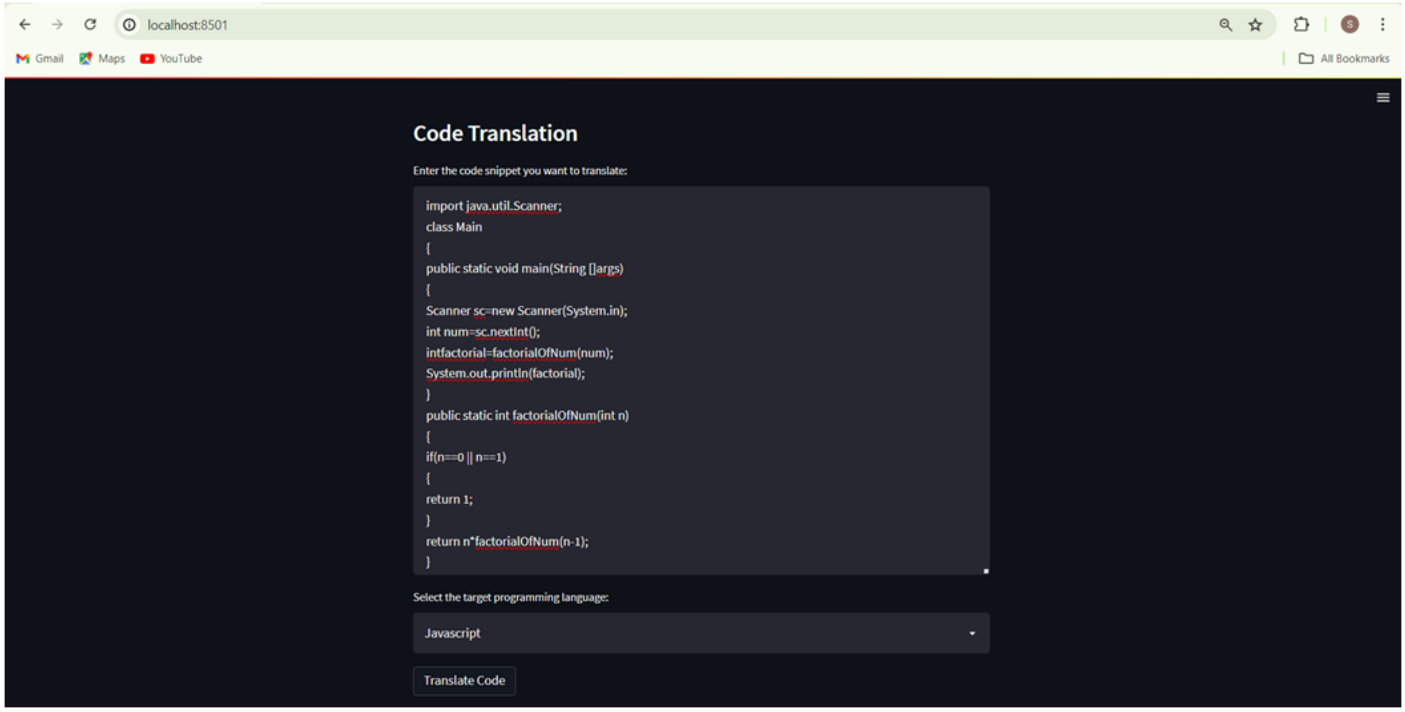


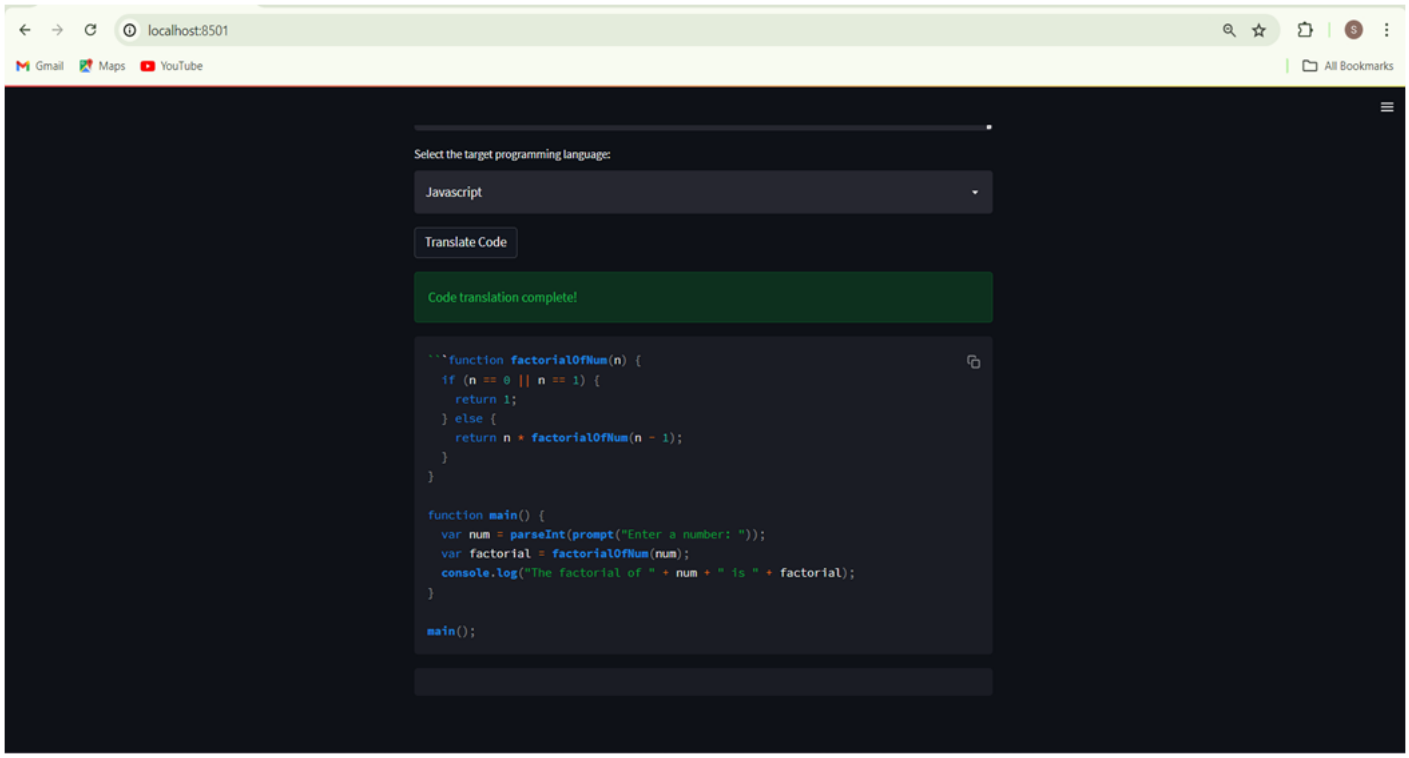
4. **<u>Translating Python code to Java:</u>**


localhost:8501
Gmail
Maps
YouTube
All Bookmarks
Code Translation
Enter the code snippet you want to translate:
# Python 3 program to find
# factorial of given number
def factorial(n):
# single line to find factorial
return 1 if (n==1 or n==0) else n * factorial(n - 1)
# Driver Code
num = 5
print("Factorial of",num,"is",factorial(num))
Select the target programming language:
Java
Translate Code
Code translation complete!
Sure, here is the Java code that corresponds to the Python code you provided:
```java

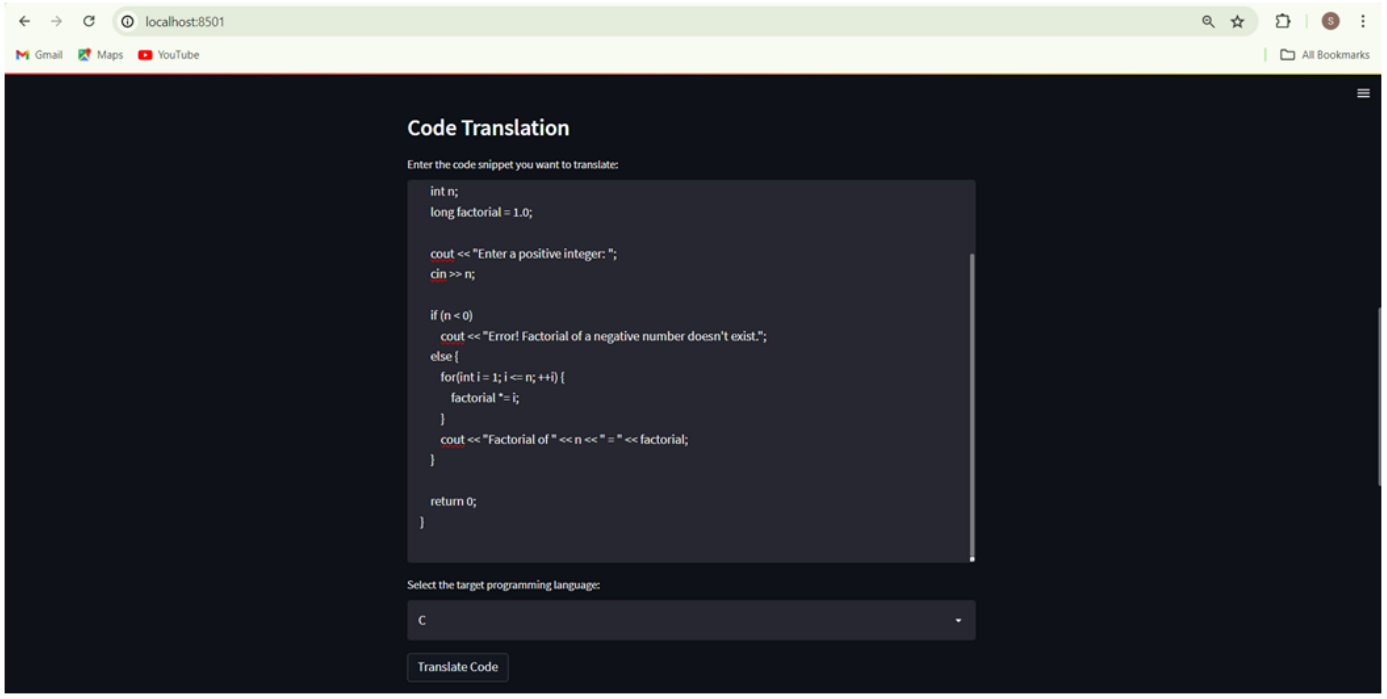

localhost:8501
Gmail
Maps
YouTube
All Bookmarks
Java
Translate Code
Code translation complete!
Sure, here is the Java code that corresponds to the Python code you provided:
```java
public class Factorial {
public static void main(String[] args) {
int num = 5;
System.out.println("Factorial of " + num + " is " + factorial(num));
}
public static int factorial(int n) {
if (n == 1 || n == 0) {
return 1;
} else {
return n * factorial(n - 1);
}
}
}

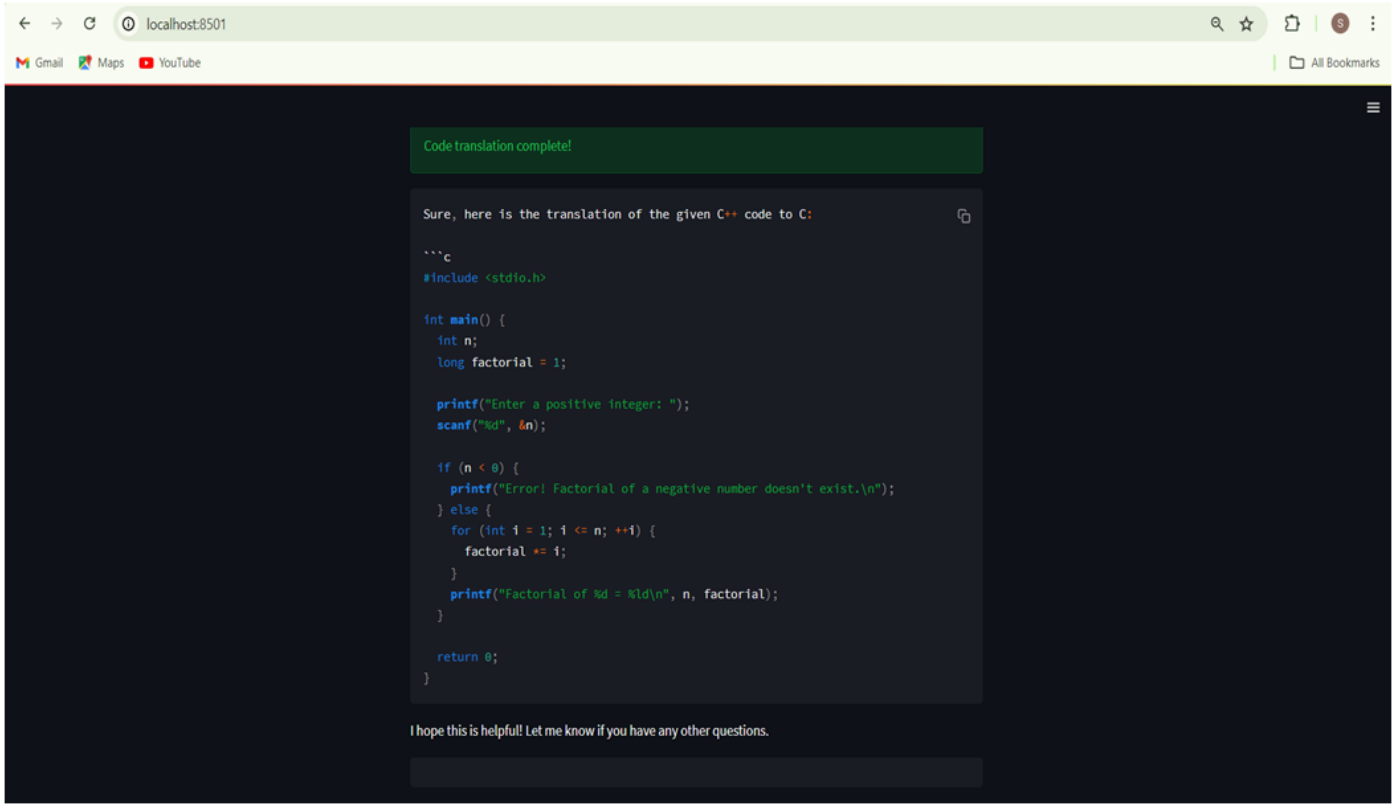
5. **Translating Java code to JavaScript:**






6. **Translating C++ code to C:**


localhost:8501
Gmail
Maps
YouTube
All Bookmarks
Code Translation
Enter the code snippet you want to translate:
int n;
long factorial = 1.0;
cout << "Enter a positive integer: ";
cin >> n;
if (n < 0)
cout << "Error! Factorial of a negative number doesn't exist.";
else {
for(int i = 1; i <= n; ++i) {
factorial *= i;
}
cout << "Factorial of " << n << " = " << factorial;
}
return 0;
}
Select the target programming language:
C
Translate Code


localhost:8501
Gmail
Maps
YouTube
All Bookmarks
Code translation complete!
Sure, here is the translation of the given C++ code to C:
```c
#include <stdio.h>
int main() {
int n;
long factorial = 1;
printf("Enter a positive integer: ");
scanf("%d", &n);
if (n < 0) {
printf("Error! Factorial of a negative number doesn't exist.\n");
} else {
for (int i = 1; i <= n; ++i) {
factorial *= i;
}
printf("Factorial of %d = %ld\n", n, factorial);
}
return 0;
}
I hope this is helpful! Let me know if you have any other questions.
```

- The code used in all the test cases is the basic factorial program.

- Hence, this is the implementation of the codeXchange: an AI powered tool for translating code in one programming language to the other.