**Beschreibung der Funktionalitäten**

Dieser Code implementiert ein Programm, das mit zwei verschiedenen APIs arbeitet:

1. **OpenWeatherMap API**: Abfrage von Wetterdaten für eine Stadt. Es gibt die aktuelle Temperatur in Celsius und eine Wetterbeschreibung zurück.
2. **Google Maps Geocode API**: Abfrage von geographischen Koordinaten oder einer Adresse basierend auf einer Eingabeadresse.

Das Programm ermöglicht es dem Benutzer, eine API auszuwählen, eine Eingabe zu tätigen (Stadt für das Wetter oder Adresse für Google Maps), und zeigt die entsprechenden Daten (Wetterinformationen oder Koordinaten/Adresse) an.

**Funktionen:**

* **Abfrage von Wetterdaten**: Das Programm fragt die OpenWeatherMap API ab, um das Wetter für eine eingegebene Stadt abzurufen.
* **Abfrage von Geodaten**: Das Programm fragt die Google Maps API ab, um Koordinaten oder eine Adresse für eine eingegebene Adresse zu erhalten.
* **Anzeige der Ergebnisse**: Das Programm zeigt die abgerufenen Daten in deutscher Sprache an (z. B. Temperatur in Celsius, Wetterbeschreibung, Koordinaten oder Adresse).

**Anweisungen zur Installation, Kompilierung und Nutzung**

**1. Installation von Abhängigkeiten**

Für das Projekt werden folgende Bibliotheken benötigt:

* **cURL**: Für HTTP-Anfragen an die APIs.
* **nlohmann/json**: Für das Parsen der JSON-Antworten der APIs.

**a. Installation von cURL:**

* Für Windows: Du kannst cURL über den Windows-Paketmanager winget installieren:

Code kopieren

winget install curl

* Alternativ kannst du cURL von curl.se herunterladen und installieren.

**b. Installation von nlohmann/json:**

* Du kannst die JSON-Bibliothek über CMake von GitHub einbinden:

bash

Code kopieren

git clone https://github.com/nlohmann/json.git

Stelle sicher, dass du die json.hpp-Datei in deinem Projekt verwendest.

**2. Projekt einrichten**

* Lade oder erstelle die Quelldateien in einem Verzeichnis.
* Erstelle eine CMake-Konfigurationsdatei (CMakeLists.txt), um das Projekt zu kompilieren.

Beispiel für die CMakeLists.txt:

cmake

Code kopieren

cmake\_minimum\_required(VERSION 3.10)

project(cURLProjekt)

set(CMAKE\_CXX\_STANDARD 14)

# cURL und JSON-Header einbinden

find\_package(CURL REQUIRED)

include\_directories(${CURL\_INCLUDE\_DIRS})

include\_directories(path/to/json)

add\_executable(cURLProjekt main.cpp WeatherAPI.cpp GoogleMapsAPI.cpp CurlUtils.cpp MainController.cpp)

target\_link\_libraries(cURLProjekt ${CURL\_LIBRARIES})

**3. Kompilierung des Projekts**

* Öffne das Projekt in deiner IDE (z. B. CLion) oder benutze die Kommandozeile, um das Projekt zu kompilieren:
  + In CLion: Drücke Shift + F9 (Build).
  + Mit CMake in der Kommandozeile:

bash

Code kopieren

mkdir build

cd build

cmake ..

make

**4. API-Schlüssel**

* Du musst für beide APIs (OpenWeatherMap und Google Maps) API-Schlüssel erhalten:
  + **OpenWeatherMap API**: Registriere dich auf [openweathermap.org](https://openweathermap.org/) und hole dir einen API-Schlüssel.
  + **Google Maps API**: Registriere dich auf console.cloud.google.com und erhalte einen API-Schlüssel.

Ersetze im Code die Platzhalter für die API-Schlüssel:

cpp

Code kopieren

apiKey = "dein\_api\_schluessel\_fuer\_openweathermap"; // OpenWeatherMap

apiKey = "dein\_api\_schluessel\_fuer\_google\_maps"; // Google Maps

**5. Nutzung des Programms**

* Nach der Kompilierung kannst du das Programm ausführen.
* Es fragt den Benutzer, welche API er verwenden möchte (1 für OpenWeatherMap oder 2 für Google Maps).
* Je nach Auswahl wird entweder das Wetter für eine Stadt oder die Koordinaten bzw. die Adresse einer eingegebenen Adresse angezeigt.

Beispiel:

bash

Code kopieren

Wähle eine API:

1. OpenWeatherMap

2. Google Maps

Gib deine Wahl ein (1/2): 1

Gib die Ortschaft ein: Berlin

Wetter in Berlin:

Temperatur: 5.0 °C

Wetterbeschreibung: klar

Oder für Google Maps:

bash

Code kopieren

Wähle eine API:

1. OpenWeatherMap

2. Google Maps

Gib deine Wahl ein (1/2): 2

Gib die Adresse ein: Berlin, Germany

Koordinaten: Breitengrad: 52.5200, Laengengrad: 13.4050

**6. Fehlerbehebung**

* Falls du beim Kompilieren oder Ausführen Fehler bekommst, stelle sicher, dass alle Bibliotheken korrekt eingebunden sind und dass du die richtigen API-Schlüssel verwendest.
* Überprüfe die Ausgabe auf mögliche Fehler bei der API-Abfrage, z. B. ungültige Anfragen oder fehlende API-Schlüssel.

**Fazit**

Das Programm ermöglicht eine einfache und benutzerfreundliche Abfrage von Wetter- und Geodaten über OpenWeatherMap und Google Maps. Mit einer korrekten Konfiguration und API-Schlüsseln kannst du es schnell einrichten und verwenden