API INTEGRATION IN SYSTEM DEVELOPMENT

Prepared by:

KRISTIANNE ALEZA MARIE JAVIER-MAGBANUA
Assistant Professor I, College of Computer Studies

E-mail Address: kristianne.javier@minsu.edu.ph

Learning Objectives

By the end of this lesson, students should be able to:

- Explain what an API is and its purpose in system integration.
- Identify the different types of APIs and their uses.
- Demonstrate a simple API integration using C#.
- Perform an activity that retrieves and displays data from a public API.

What is an API?

- **API** stands for **Application Programming Interface**. It is a **set of protocols, definitions, and tools** that allows one software application to interact with another.

What is an API?

An API is like a waiter in a restaurant.

- You (the client) tell the waiter (API) what you want (request).
- The waiter gives your order to the kitchen (server).
- The kitchen prepares it and the waiter brings it back (response).
- This is how **applications communicate without directly accessing each other's code or databases**.

Types of APIs

Type	Description	Example / Use Case
REST API (Representational State Transfer)	Uses standard HTTP methods (GET, POST, PUT, DELETE). Returns data usually in **JSON** format.	OpenWeather API, YouTube Data API
SOAP API (Simple Object Access Protocol)	Uses XML for data exchange. Common in enterprise systems.	Bank or government web services
GraphQL API	Allows clients to request exactly the data they need — nothing more, nothing less.	GitHub API
WebSocket API	Maintains a continuous connection for **real-time data**.	Chat apps or live stock prices

Components of an API Request

Component	Description	Example
Endpoint URL	The web address (like a website) where the API can be accessed.	https://api.openweathermap.org/data/2.5/weather
Method	The action you want to perform: GET, POST, PUT, DELETE.	GET
Parameters	Inputs or filters that customize your request.	?q=Manila&appid=API_KEY
Headers	Extra information like access tokens or format type.	Authorization: Bearer TOKEN
Response	The data returned by the server (often JSON).	{"temp":30,"city":"Manila"}

Example: API Integration Using C#

```
0 references
class Program
{
    0 references
    static async Task Main()
    {
        string apiUrl = "https://api.agify.io?name=ALEZA"; // API endpoint URL

        using (HttpClient client = new HttpClient()) // Create an instance of HttpClient
        {
            HttpResponseMessage response = await client.GetAsync(apiUrl); // Send GET request to the API

            if (response.IsSuccessStatusCode) //    Check if the response is successful
            {
                string jsonData = await response.Content.ReadAsStringAsync(); // Read the response content as a string
                Console.WriteLine("Data from API:");
                Console.WriteLine(jsonData); // Print the JSON data to the console
            }
            else
            {
                Console.WriteLine("Error: Unable to retrieve data");
            }
        }
    }
}
```

Example: API Integration Using C#

Code Line	Explanation
using System.Net.Http;	Imports the library that allows sending HTTP requests.
HttpClient client = new HttpClient();	Creates an object that can send requests to a web server.
GetAsync(apiUrl)	Sends a GET request to the specified API URL.
response.Content.ReadAsStringAsync();	Reads the data returned by the server (JSON format).
Console.WriteLine(jsonData);	Displays the data on the console.

Real-World Examples of API Usage

Category	API Example	Purpose	Sample Use Case
Finance	PayPal, Stripe	Online payments	E-commerce checkout
Data	OpenWeatherMap	Data retrieval	Travel or weather apps
Social Media	Facebook Graph	Authentication & posts	Log in with Facebook
Education	Google Classroom	Sync grades & classes	Student portal integration
Transport	Google Maps	Route visualization	Delivery tracking
Healthcare	IBM Watson	AI-powered analysis	Health chatbot
Enterprise	SAP / ERPNext	Business process integration	POS-to-ERP sync

Summary: API Integration in Systems

- **Application Programming Interfaces (APIs)** play a vital role in allowing different systems, applications, and platforms to communicate and share data seamlessly. Through APIs, developers can extend the functionality of their applications without reinventing the wheel — for example, retrieving weather data, authenticating users, or processing payments.

References

- **Microsoft Learn (2024).** *Call a Web API from a .NET client (C#)*
- **RapidAPI (2024).** API Hub and Tutorials
- **Postman Blog (2024).** Understanding REST APIs
- **OpenWeather API Documentation.** https://openweathermap.org/api
- **Microsoft Learn:** Call a web API with HttpClient
- **CoinGecko API Documentation:** https://www.coingecko.com/en/api
- **GitHub REST API Guide:** https://docs.github.com/en/rest
- **W3Schools REST API Tutorial:** https://www.w3schools.com/js/js_api_intro.asp

ACTIVITY: Connect and Display API Data

- **Instructions:**
- Create a **C# Console Application** or a simple **Windows Form App**.
- Integrate the following API:
  👉 https://api.genderize.io?name=yourname
- Display the **name** and **predicted gender** in the output.
- Example:
  - Name: Aleza
  - Predicted Gender: female

ACTIVITY: Display Weather Data

```
using System;
using System.Net.Http;
using System.Threading.Tasks;

0 references
class Program
{
    0 references
    static async Task Main()
    {
        string city = "Manila";
        string apiKey = "YOUR_API_KEY"; // Register for free at openweathermap.org
        string apiUrl = $"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={apiKey}&units=metric";

        using (HttpClient client = new HttpClient())
        {
            HttpResponseMessage response = await client.GetAsync(apiUrl);
            string result = await response.Content.ReadAsStringAsync();
            Console.WriteLine("Weather Data for " + city + ":");
            Console.WriteLine(result);
        }
    }
}
```