GitHub Copilot Codathon Event Guide

Introduction to the Codathon

Welcome to the GitHub Copilot Codathon, a hands-on coding marathon where you'll solve programming challenges with GitHub's AI pair programmer. This 4-6 hour event is organized similar to a mini hackathon, where participants address small challenges to demonstrate GitHub Copilot's features and encourage innovative problem-solving.

GitHub Copilot is meant to assist, not replace you. During the Codathon, you will pair with Copilot to write code faster and explore innovative ideas. Try out advanced features like agent mode, where Copilot autonomously handles multi-step tasks, including analysing code, writing and editing files, running commands, and correcting errors. This allows you to focus on creativity and problem-solving.

Objective

The Codathon aims to familiarize participants with GitHub Copilot's AI by engaging in practical challenges. Through this event, participants will learn to use Copilot effectively to enhance productivity, improve coding quality, and solve complex tasks efficiently.

Guidelines

- **Tech Stack Freedom**: Use any programming language or framework. Choose familiar languages (JavaScript, Python, Java, C#, etc.) or try new ones – Copilot supports many.
- **Individual Participation**: This is a solo Codathon. Work individually on the challenges. Engage with Copilot and do all coding yourself, but feel free to discuss ideas and ask questions.
- **Use Copilot's Power**: Leverage **GitHub Copilot**, including autocomplete suggestions, the Copilot Chat interface, and agent mode if available. Experiment with these features to see how Copilot can assist beyond completing code lines.
- **Best Practices with Copilot**: To get the most out of Copilot, keep these best practices in mind:
    - **Provide Context** – Give Copilot as much context as possible about what you're trying to do.
    - **Use Meaningful Names** – Name your variables and functions clearly. Copilot's suggestions improve when it can infer intent from your naming.
    - **Start Small and Iterate** – Break down large problems into manageable pieces. Don't try to get Copilot to solve an entire challenge in one giant prompt.
    - **Validate and Learn** – Always review Copilot's output and test your code. Copilot is a powerful assistant, but it's not infallible – it might occasionally suggest something that doesn't fit or that contains a bug

Challenges

1. Develop a Shopping Cart

Goal

The goal of these requirements is to develop a comprehensive shopping cart system that can effectively manage and reflect various aspects of an e-commerce platform. This includes retrieving and displaying products, facilitating user searches, and handling shopping cart functionalities.

Requirements

- Create a Rest API with methods to:
  - Get the automobile parts list with pagination.
  - Retrieve automobile part details by ID.
  - Search automobile parts by name, description, manufacturer, or price.
- List products on the main page.
- Add a search bar to filter products.
- Navigate to the product description page on click.
- (Optional) Add a price slicer for filtering products.
- Create a shopping cart that can:
  - Add products.
  - Remove products.
  - Calculate the total price

2. **Trip Budget Estimator**

Goal

- Build a simple web or mobile app that takes in a destination, number of days, and travel style (budget/luxury) and returns an estimated trip cost along with change fee before departure and cancellation charges before departure (ADT).

Requirements

- Use static datasets or mock APIs for flights, accommodation, food, and transport and Sightseeing.
- Clean UI, form inputs, and budget breakdown
- In case of destination India, Apply standard GST 18%.
- Currency conversion based on user inputs using mock APIs or below mentioned API.
- Display change Fee (Adult Before departure) and cancellation charges (Adult Before departure) by calling ITQ LLMFR API with static provided request.

A) Currency conversion API Details –

Register in to below API for getting 'YOUR_API_KEY' and consume API for your required conversions.

URL -https://v6.exchangerate-api.com

B) ITQ LLMFR API Details-

Use below credentials and sample request

URL- https://llmmfr.itq.in/MiniService.svc?wsdl use `GetPenalties Method, Credentials must be passed in SOAP header along with the soap action.`

Sample Request –

<soapenv:Envelope xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/ xmlns:tem="http://tempuri.org/"><soapenv:Header/><soapenv:Body><tem:GetPenalties><tem:xmlFile><![CDATA[<GetMiniRules><FareBasis FBC="VL3YXSDY"><Origin>DEL</Origin><Destination>DXB</Destination><DepartureDate>13May25</DepartureDate><Airlines><Airline>AI</Airline></Airlines></FareBasis><PaxType><PTC>ADT</PTC></PaxType><PCC/></GetMiniRules>]]></tem:xmlFile></tem:GetPenalties></soapenv:Body></soapenv:Envelope>

Credentials-

Securitykey	Pqy@325647
branchcode	P1922840
username	Universal API/uAPI7599224455-7aed6507
SOAPAction	http://tempuri.org/IMiniService/GetPenalties

3. Chat Based on Web Sockets

Goal

The goal of this challenge is to develop a chat based on web sockets. The chat should allow users to send and receive messages in real time.

- (Optional) The chat should also allow users to send and receive images.
- (Optional) The chat should keep the session to start again from where it was left off.

Requirements

- Enable real-time messaging.
  - (Optional) Add a login screen for chat.
  - (Optional) Support sending and receiving images.
  - (Optional) Maintain session continuity.

Resources

- Get a list of random users: https://randomuser.me/api/?results=10
- Get random images: https://picsum.photos/v2/list?page=2&limit=100
- Implement chat with socket.io: https://socket.io/docs/v4/index.html

4. Interactive To-Do List App

Goal

Build a simple to-do list application that runs in the browser.

Requirements

- **Task Management**: Users should be able to add, edit, delete, and mark tasks as complete.
- **Categorization**: Enable users to categorize tasks (e.g., work, personal, urgent).
- **Due Dates and Reminders**: Allow users to set due dates and receive reminders for upcoming tasks.
- **Priority Levels**: Users should be able to assign priority levels to tasks (e.g., high, medium, low).
- **Search and Filter**: Provide functionality to search and filter tasks based on keywords, categories, or priority.

5. CSV Data Analyzer (Data Science) –Optional

Goal

Use Copilot to help analyze a dataset and extract some insights.

Requirements

Write a program that reads the CSV and prints out a summary of the data – such as the total and average sales for each category, or the highest, lowest, and average grade for the class.

Use copilot to generate data or pick any publicly available dataset

Add UI to build a dashboard for visualization

Try to use complex or multiple datasets