

The Agent Awakens

A New Era of AI-Powered Development



Kartik Gupta

Senior Solutions Engineer, GitHub APAC

**You are just a machine, an
imitation of life.**

**Can a robot write a symphony?
create a beautiful
masterpiece?**



Can you?

Last Session

You met Copilot

Autocomplete

Coding suggestions while writing
code

Chat

Chat in natural language to Copilot

Edits

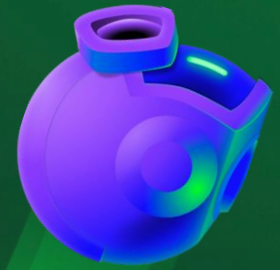
Multi file edits

Prompt files

Adding extra information

This Session

You and Copilot build *real* things together



 Yesterday

**Yesterday we type
prompts
It works when you do.**

 Yesterday

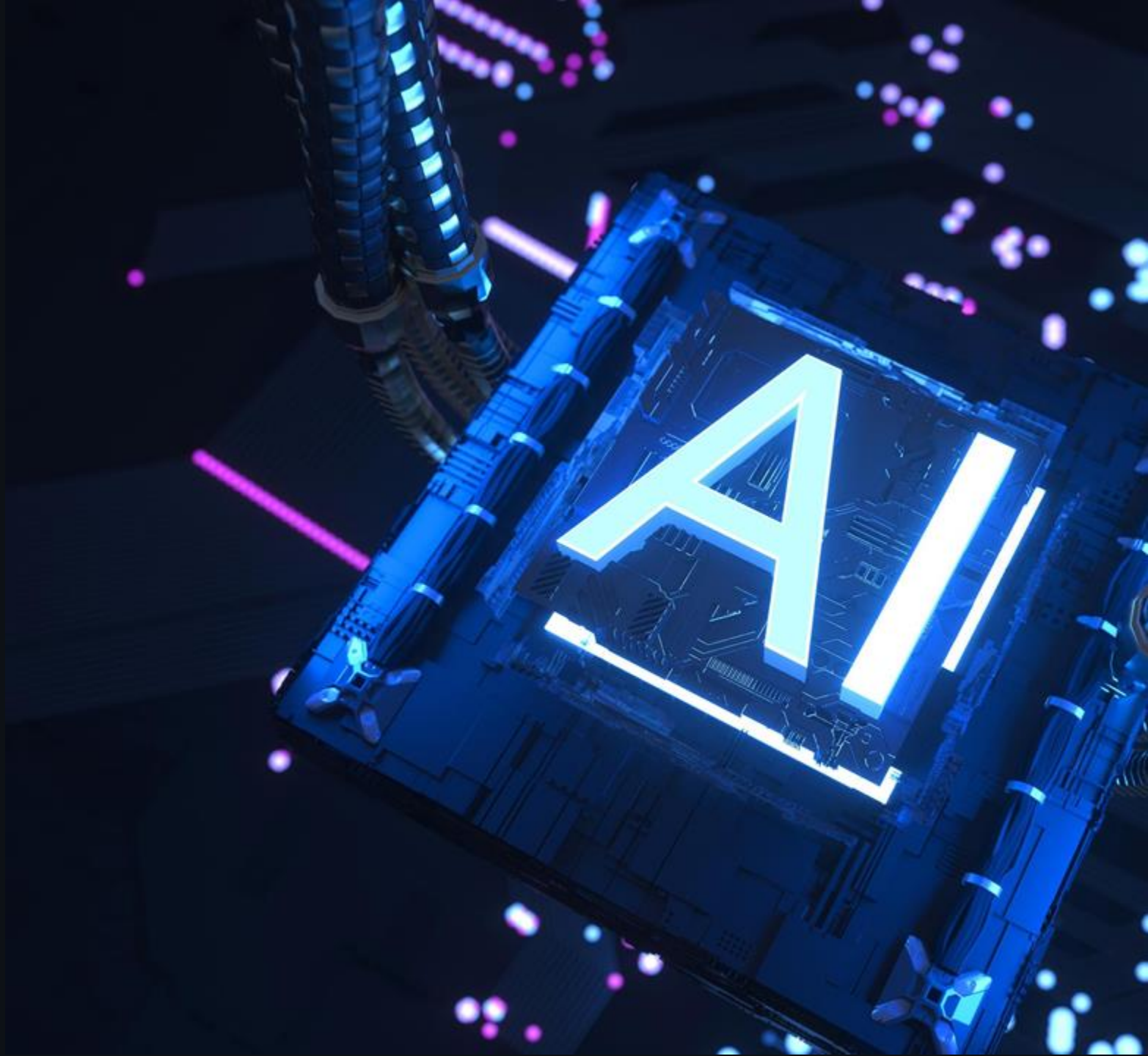
**Yesterday we type
prompts**
It works when you do.

 Today

**Today we design
interactions – shaping
solutions with language**

Prompt Royale Special Edition

Prompt Crafting





GitHub Copilot figuring out
what the hell I'm trying to code



**Suppose you're
working with a travel
agent to help plan a
trip to Japan...**

A travel agent wouldn't be able to help you plan accordingly without the following information:

- Budget
- Origin
- Destinations
- Departure and return dates
- Preference of nonstop vs layovers



Prompt crafting: the process of giving clear natural language instructions to computer programs, such as the large language models (LLMs) that drive GitHub Copilot functionalities.



What does Copilot interpret as a **prompt**?

- It can be a question, statement, or a description of what you want Copilot to do
- Accepted in the form of comments in your file or inputs in Copilot Chat



Copilot is **probabilistic**,
not deterministic

Copilot's LLMs are sophisticated probability engines, so you might not get the same results for the exact same prompts every time

Pillars of effective Copilot prompts

Context

Information provided in the prompt that helps Copilot understand the task better.

Intent

The specific goal or purpose you have in mind when creating a prompt.

Clarity

The quality of being unambiguous and easy to understand.

Specificity

The level of detail and precision in a prompt.

Patterns in your code that will
help Copilot

Variable names

Use descriptive variable names to make your intentions clear.

```
total_attendees = 5
```

Method Signatures

Define method signatures with unambiguous parameter names and types.

```
calculateAverage(int  
num)
```

Naming Conventions

Maintain consistent naming conventions for variables and functions

i.e. using **camelCase** for variable names consistently

Input/Output Format

Describe the expected input and output formats.

"Write a function that takes an array of strings as input and returns true if a palindrome is found."

Error Handling

Specify error handling scenarios

"Exit where the integer input is empty and throw an error if the input is not an integer at all"

Control Structures

Describe control flow structures.

"Write a while loop to find the first prime number given a list of integers. If no such number exists, the loop should exit."

Techniques to maximize the
accuracy of suggestions from
Copilot

Include a high level description of your requirements at the top of your file.

This is especially complemented by following up with specific instructions in the form of steps

```
JS weather.js > ...
1  //Create a class that will retrieve weather metrics from the US National
2  //Include the ability to retrieve the current weather, 7 day forecast,
3  //API calls should be handled asynchronously
4  //1.Import axios to make API calls
5  //2.Import the conversion.js file and use the functions to convert the
6  //3.Store API endpoint as a constant
7  //4.Functions should take a zip code as a parameter where applicable
8  //5.Handle exceptions and return appropriate error codes with user friendly
9  //6.Functions should return the data in a format that can be used by the
10
11  const axios = require('axios');
12  const conversion = require('./conversion');
13
14  const BASE_URL = 'https://api.weather.gov/points/';
15
16  // define Weather class
17  class Weather {
18      constructor(zip) {
19          this.zip = zip;
20      }
21  }
```

Provide examples in your prompt to clarify expectations

```
calculations.py > ...
1  """
2  Create a Python function, `sum_array_values`, that finds
3  and returns the sum of integers in an array.
4
5  1. Verifiy if the array is empty and exit if it is.
6  2. Throw error if the array contains non-integer values.
7  3. Return the sum of the array values if the array is valid.
8
9  Please consider the following example data:
10 Example 1: [5, 10, 15, 20, 8]
11 Example 2: [2, 4, 6, 1, 9, 12]
12 The generated code should work correctly for arrays like above
13 """
14 #define sum_array_values function
15 def sum_array_values(array):
16     # 1. Verifiy if the array is empty and exit if it is.
17     if len(array) == 0:
18         return 0
19     # 2. Throw error if the array contains non-integer values.
20     for i in array:
21         if type(i) != int:
22             raise TypeError("The array contains non-integer values.")
23     # 3. Return the sum of the array values if the array is valid.
24     return sum(array)
```

Keep files open in your project folder that are relevant to the requirements of your current file.

```
J BookDatabaseImpl.java • J BookDatabase.java J BookService.java J BookServiceException.java
src > main > java > com > github > demo > service > J BookDatabaseImpl.java > BookDatabaseImpl > getBook

146
147
148 +
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177

@Override
public List<Book> getBooksByAuthor(String author) throws BookServiceException {
    // 1.create list of type Books
    // 2.create Prepared statement and query to get books by Author
    // 3.iterate through results and add to list
    // 4.close connection if prepared statement is not null
    // 5.return list

    List<Book> books = new ArrayList<Book>();

    if (!isValid()) {
        throw new BookServiceException(message:"Database connection is not valid, check
    }

    PreparedStatement ps = null;

    try {
        ps = connection.prepareStatement(sql:"SELECT * FROM books WHERE author LIKE ?")
        ps.setString(parameterIndex:1, "%" + author + "%");

        ResultSet results = ps.executeQuery();

        while (results.next()) {
            Book book = new Book(
                results.getString(columnLabel:"author"),
                results.getString(columnLabel:"title"),
                results.getString(columnLabel:"image"));
            books.add(book);
        }
    } catch (SQLException e) {
        logger.error("Failed while searching for {}", author);
        throw new BookServiceException(e);
    }
}
```

If at first you
don't succeed,
iterate!

Initial prompt

```
1 //Write a JavaScript function that finds
2 //the maximum value in an array.
3 function max(array) {
4     return Math.max.apply(null, array);
5 }
```

Second prompt

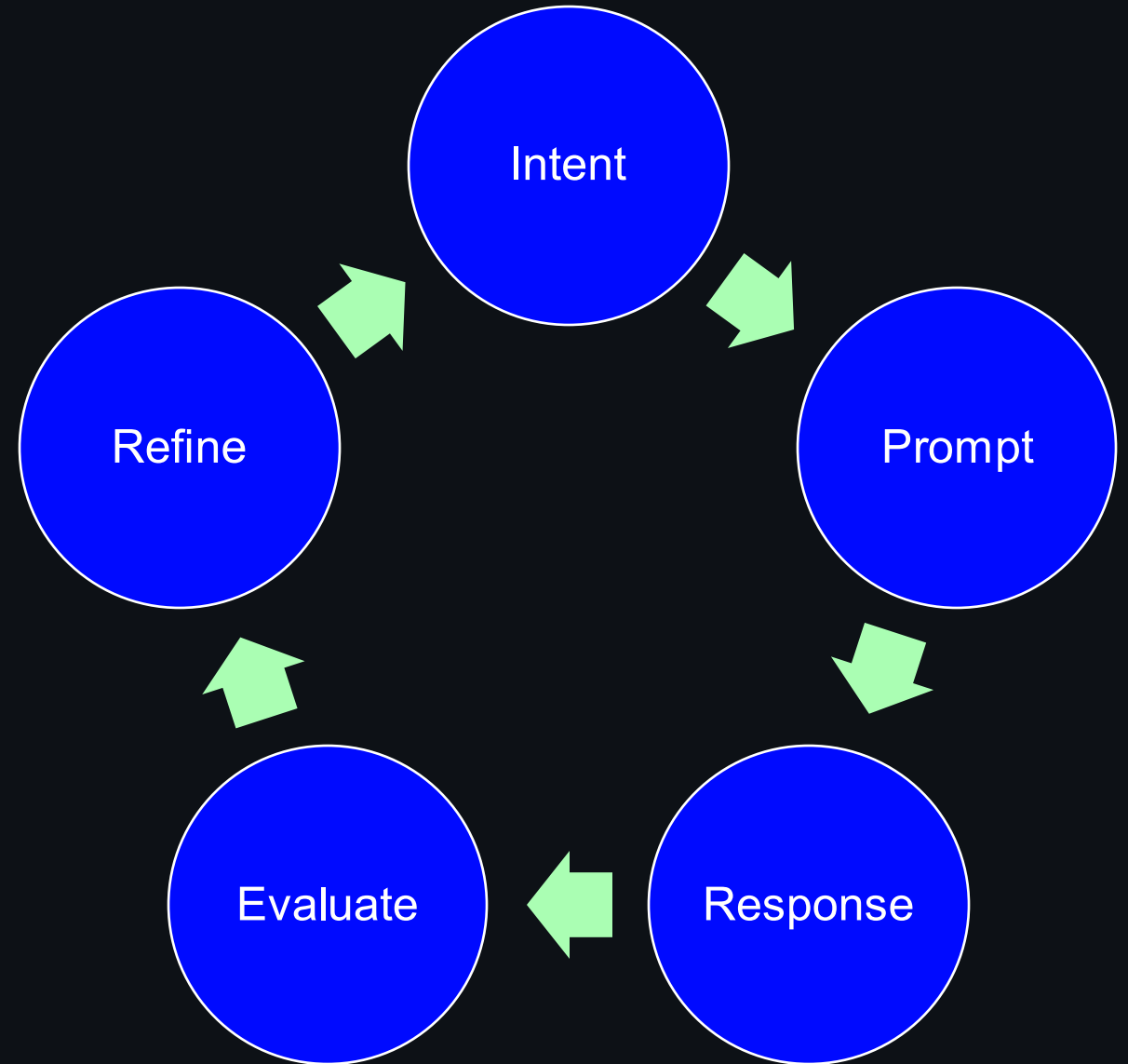
```
1  ✓ //Create a JavaScript function, `findMax`, that takes an array of
2    //and returns the string with the maximum length.
3    //Ensure that the function works correctly for arrays that are em
4    //as well as arrays that contain both strings and numbers.
5  ✓ function findMax(array) {
6      var max = array[0].length;
7      array.map(v => max = Math.max(max, v.length));
8      result = array.filter(v => v.length == max);
9      return result;
10 }
```

How experts use Copilot

**Layering intent using
natural language**

Chaining small steps

**Shaping output using
constraints**



Prompting Patterns that Work

- **Write a test for...**
- **Migrate this function to use `async/await`**
- **Rewrite class using composition instead of inheritance**
- **Suggest performance improvements for this loop**
- **Refactor to enhance <insert nfr>**

What Copilot can actually do?



Mobile UI Scaffolding



End to end enterprise applications



Game dev loop



Refactor old code



Build reusable libraries



Add tests to legacy code

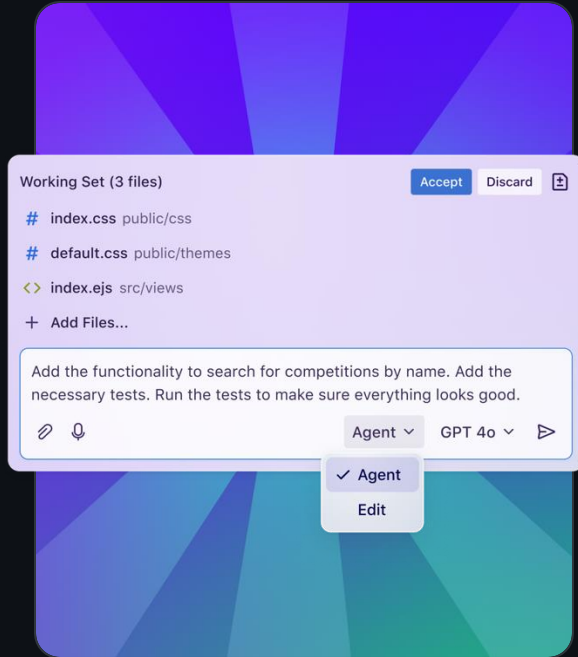
Demo

**Let's pick one and build
together**

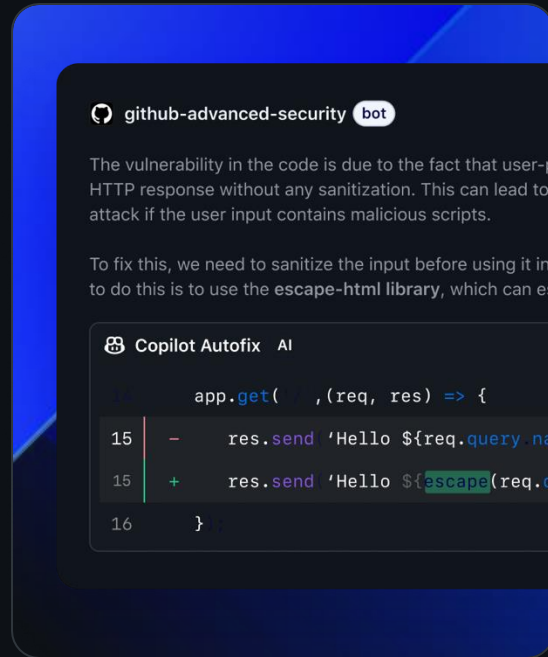
Winning the Prompt Royale

- 1. Don't just ask for code. Ask for clarity.**
- 2. Treat Copilot like a collaborator**
- 3. Break big ideas into smaller promptable units**
- 4. Comment your intent—Copilot pays attention**
- 5. Use files + context deliberately**

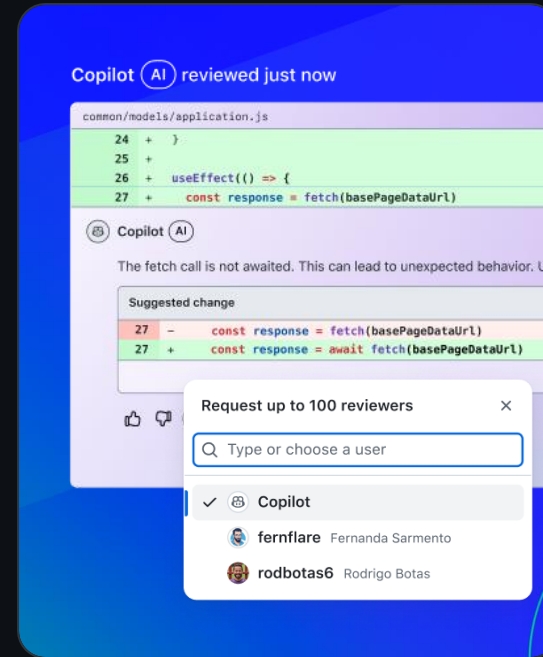
Intent Driven Future



Agent mode



Copilot Autofix



Code Review



Coding Agent

**YOU GET A GITHUB COPILOT
LICENSE**

**YOU ALL GET A GITHUB COPILOT
LICENSE**

makeameme.org

