

GitHub Copilot Developer Training

Andrew Scoppa

Let's build from here



Resources

- Getting started with GitHub Copilot
- o <u>Insider newsletter digest: How to use GitHub Copilot</u>
- Video Get Started with the Future of Coding: GitHub Copilot
- Tutorial: GitHub Copilot and VS Code
- Copilot Exercises



GitHub Copilot - Introduction

Coding

Best practices & Prompt Engineering

Workshop (1 - 2 hours long)

Secure coding

Wrap-up, Q&A

AGENDA



Outcome of this training

You will achieve...

- Get answers to specific use case scenario questions
- Increase existing Copilot skills by following a specific workshop tutorial catered to your needs
- Learn in-depth tips and tricks and best practices on how to best utilize GitHub Copilot



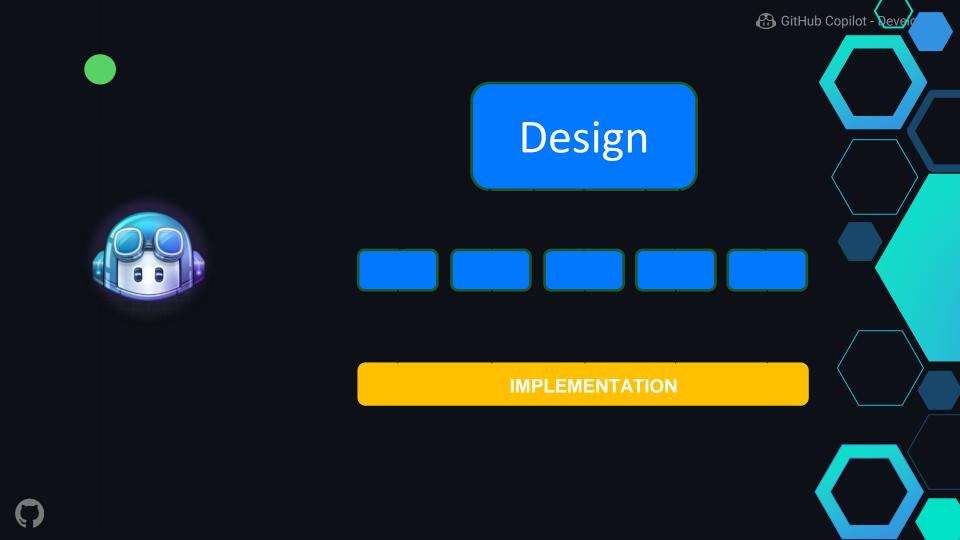
parse expense

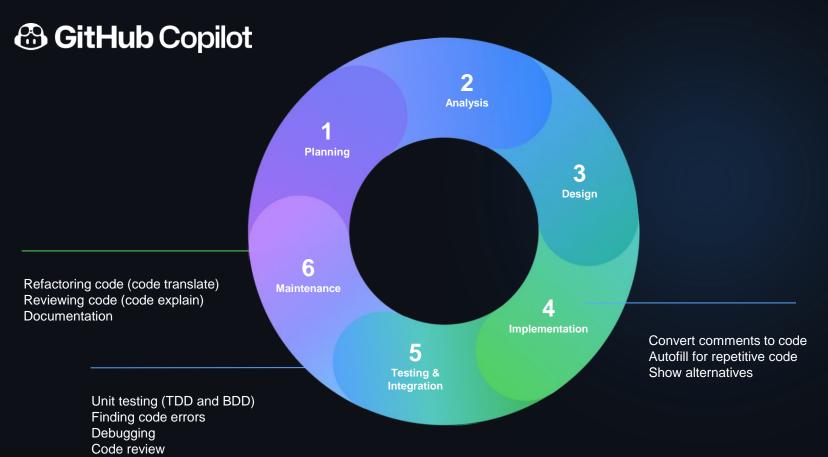
GitHub Copilot

- An intelligent pair programmer
- Draws context from comments & code to suggest individual lines and whole functions
- Powered by OpenAl Codex
 - Copilot uses a transformative model
 - Trained on large datasets to ensure accuracy
- Available as extensions to popular IDEs
- Programming Languages and Technology available in Public code base all are supported

```
1 #!/usr/bin/env ts-node
 3 import { fetch } from "fetch-h2";
 5 // Determine whether the sentiment of
6 // Use a web service
 7 async function isPositive(text: string
     const response = await fetch(`http://
      method: "POST",
      body: `text=${text}`,
      headers: {
         "Content-Type": "application/x-ww
12
     const json = await response.json();
    return json.label === "pos";
17
   Copilot
```

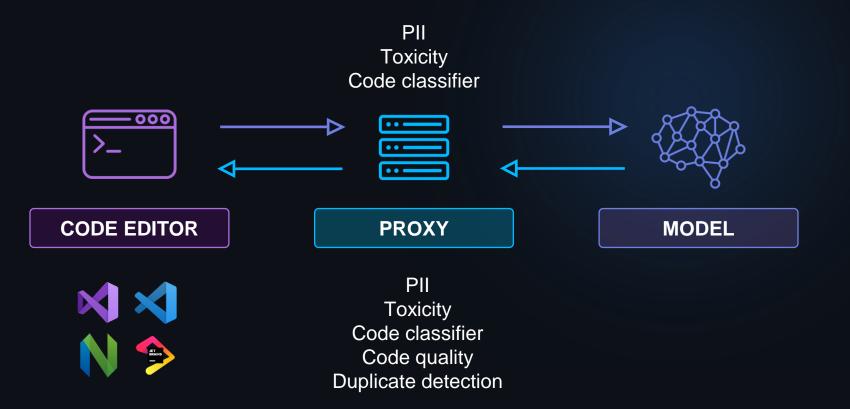
sentiment.ts





Al Pull Requests

Data flow through the Copilot ecosystem





Coding

GH Copilot vs GH Copilot Chat

Slash commands

Data Generation & Testing

Code Refining & Refactoring

Block Files & Debugging

GH Rest APIs

When to use Copilot vs Copilot Chat

Copilot

Copilot Chat

Direct Code Writing

In-Depth Assistance

Seamless IDE Integration

Learning & Teaching

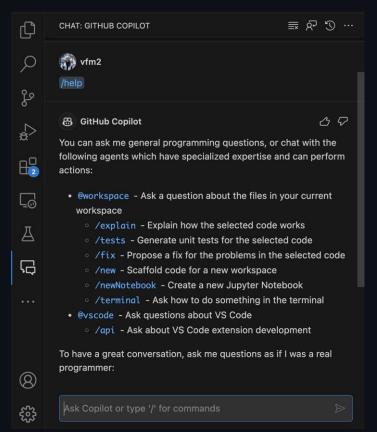
Solo Development

Collaborative Scenarios



Slash Commands

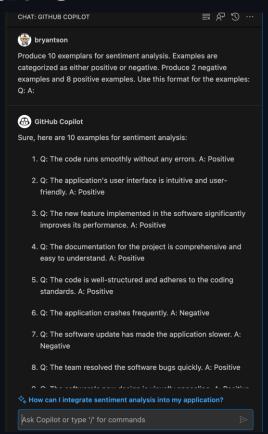
/help to find available commands in your IDE





Data Generation

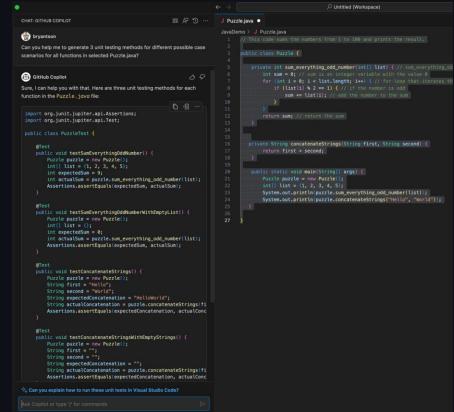
Ask Copilot to generate data





Unit Testing Generation

Works best for general unit testing case generation





Test Driven Development (TDD)

Test Driven
Development
(TDD) practice is
possible

```
EXPLORER

...

JS sample-function, js

JS test-sample-function, js 2

JavascriptTDD

JS sample-function in sample-function, js

// Test functions in sample-function, js

// Import the functions to be tested

const { add, subtract, multiply, divide } = require('./sample-function');

// Test add function

// Test subtract function

12 test 'subtract 2 - 1 to equal 1', () => {

expect(subtract(2, 1)).toBe(1);

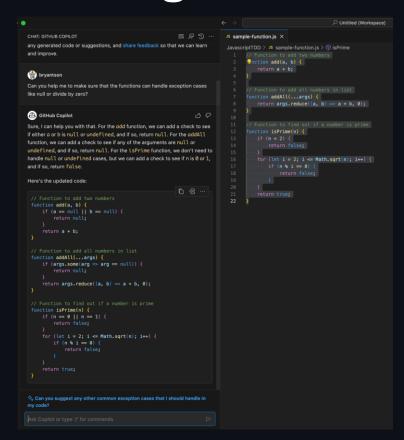
// Test multiply function

// Test multiply function
```



Code Refactoring

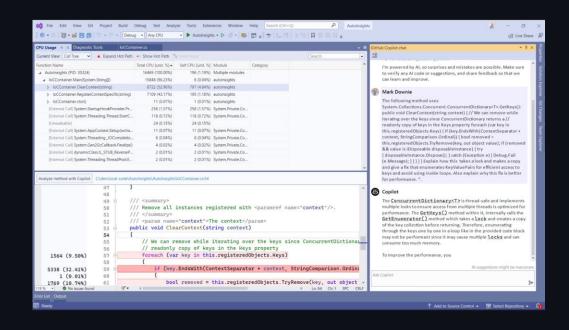
Refactoring is possible through GitHub Copilot Chat





Code Refining

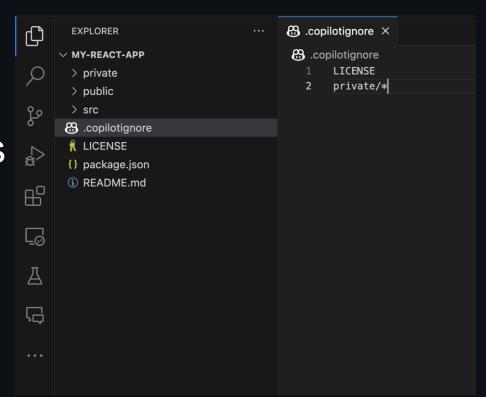
GH Copilot CPU usage tool





Block files from Copilot

Use .copilotignore to block files and folders from being used by Github Copilot





Code Debugging

Debugging with GitHub Copilot



GitHub Rest API

GitHub Copilot Rest API

```
GET /orgs/{org}/copilot/billing

cURL JavaScript GitHub CLI

# GitHub CLI api
# https://cli.github.com/manual/gh_api

gh api \
    -H "Accept: application/vnd.github+json" \
    -H "X-GitHub-Api-Version: 2022-11-28" \
    /orgs/ORG/copilot/billing
```

```
Example response Response schema

Status: 200

{
    "seat_breakdown": {
        "total": 12,
        "added_this_cycle": 9,
        "pending_invitation": 0,
        "pending_cancellation": 0,
        "active_this_cycle": 12,
        "inactive_this_cycle": 11
    },
    "seat_management_setting": "assign_selected",
    "public_code_suggestions": "block"
}
```





Demo: Coding practices



Best practices

Getting accuracy closer to the expectation

Prompt Engineering

Working at a project(s) level

Leveraging Copilot to increase code quality

Addressing concerns around security exploit

Getting accuracy closer to expectation Problems

- Copilot fails to produce answer or to keep repeating
- **Copilot generates incorrect result**
- Library/module version discrepancies issue
- Copilot suggests non-optimal solution

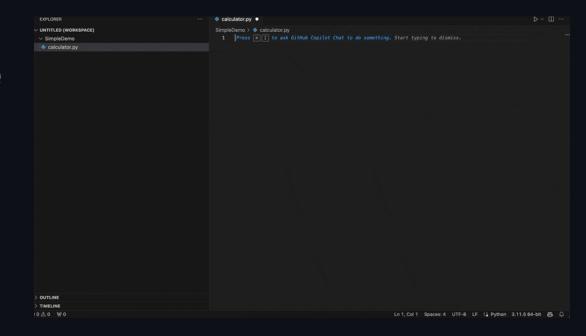


Problems #1:

Copilot fails to produce answers or will keep repeating

Some problems

- Fails to produce answer
- Hallucination -Keeps repeating



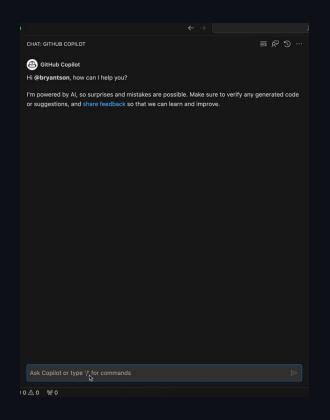


Problems #2:

Copilot generates incorrect result

Reasons why...

- Not enough context
- Old trained data
- Generative in nature





What is a Prompt?



In the context of Copilot, a prompt is a piece of code or natural language description that is used to generate code suggestions. It is the input that Copilot uses to generate its output.

- Copilot



What is Prompt Engineering?



Prompt engineering is the process of designing and creating high-quality prompts that can be used to generate accurate and useful code suggestions with Copilot.

- Copilot



Why is Prompt Engineering Important?



- Maximizes utility of Copilot
- Minimizes repeated iterations
- Determines code suggestion quality
- Skill prompts guide Copilot to understand context & nuances
- Refined interactions reduces misunderstandings



Techniques



Zero-Shot Prompting



One-Shot Prompting

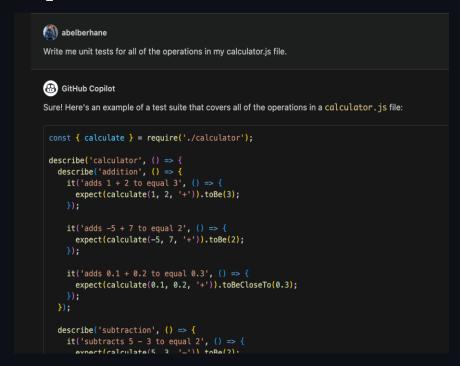


Few-Shot Prompting



Techniques

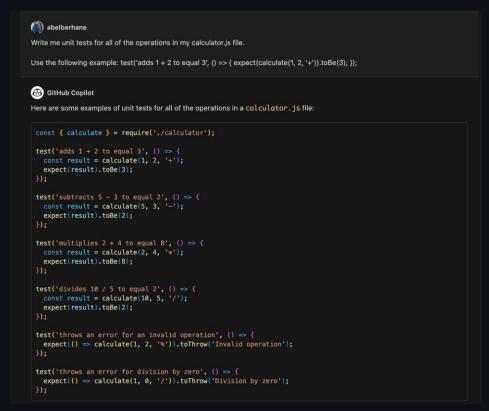
Zero-Shot
Programming
with Copilot





Techniques

One or few shot Programming with Copilot



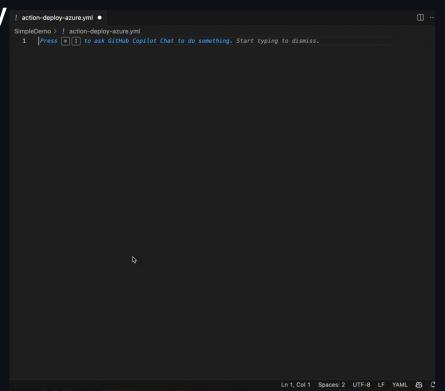


Problems #3:

Library/module version discrepancy

Old trained data

 While packages go through frequent updates, Copilot does not use latest data

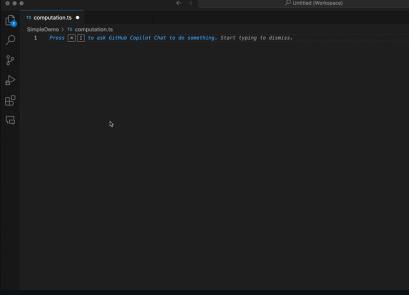




Problems #4:

Copilot suggests non-optimal solution

Although solution works... Suggested solution is not optimal because Quick Sort can be implemented in O(1) complexity, meaning no space required





How to improve Problem #4

Copilot suggests non-optimal solution

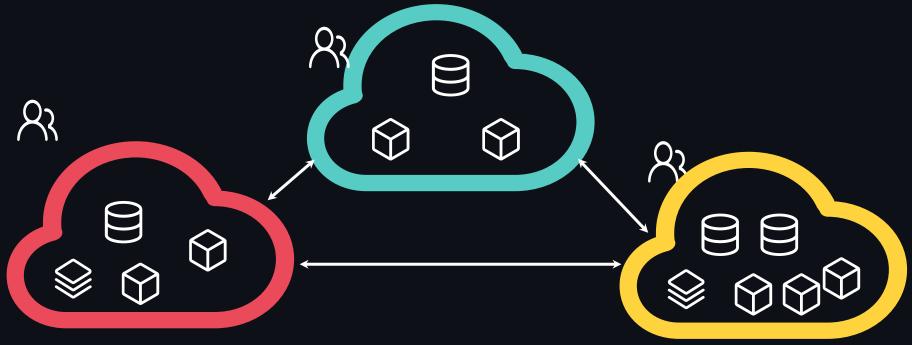
How to improve

- Ask in Copilot Chat
- Use in-line suggestion from Copilot

```
Untitled (Workspace)
(var i = 1: i < arr, length: <math>i \leftrightarrow) {
```



How to work at project(s) level





Neighboring Tabs

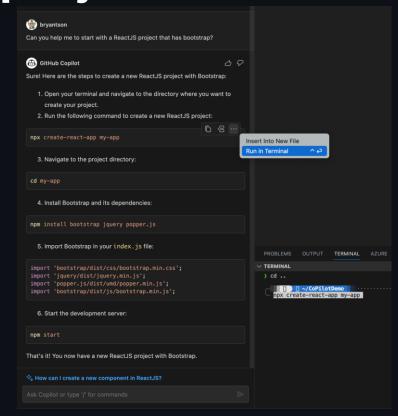
```
TS sentiments.ts
                               parse_expenses.py
                                                  addresses.rb
                 ∞ write_sal.ao
 1 #!/usr/bin/env ts-node
 3 import { fetch } from "fetch-h2";
 5 // Determine whether the sentiment of text is positive
 6 // Use a web service
7 async function isPositive(text: string): Promise<boolean> {
     const response = await fetch(`http://text-processing.com/api/sentiment/`, {
       method: "POST",
       body: `text=${text}`,
       headers: {
         "Content-Type": "application/x-www-form-urlencoded",
       },
     });
     const json = await response.json();
     return json.label === "pos";
17 }
```



Working in a project level

Ask in GitHub Copilot Chat to get started

GitHub Copilot Chat (default)





Using Copilot to improve code quality



















Copilot Chat: Ask to Refactor

Ask GitHub
Copilot Chat to
refactor your
code

```
P Untitled (Workspace)
        CHAT: GITHUB COPILOT
                                                    sorting-algorithm.py 6 .
                                                     SimpleDemo > Sorting-algorithm by >
         (A) GitHub Copilet
                                                           def quickSort(array) (
                                                                if (array, length <= 1) {
        Hi @bryantson, how can I help you?
        I'm powered by Al, so surprises and mistakes
                                                               var pivot = array[0]
        are possible. Make sure to verify any
                                                               var left = []
        generated code or suggestions, and share
                                                               var right = []
         feedback so that we can learn and improve.
                                                                for (var i = 1; i < array.length; i++) (
                                                                   array[i] < pivot | left.push(array[i]) : right.push(array[i]);
                                                               return quickSort(left).concat(pivot, quickSort(right));
                                                           def bubbleSort(array) {
50
                                                               for (var i = 0; i < len; i=) (
                                                                           var teep = array[i]:
                                                                            array[i + 1] = temp:
                                                               var preIndex, current:
                                                                for (var i = 1; i < len; i++) (
                                                                   while [preIndex >= 8 && array[preIndex] > current)
                                                                       array[preIndex + 1] = array[preIndex];
                                                               return array:

✓ ② 4 △ 2 ₩ 0
                                                                                                                                        Ln 16, Col 11 Spaces: 4 UTF-8 LF (1 Python 3.11.564-bit 23 C
```



Copilot Chat: Ask to generate tests

Ask GitHub
Copilot Chat to
generate tests

```
    sorting-algorithm.py — Untitled (Workspace)

         CHAT: GITHUB COPILOT
                                                     sorting-algorithm by 2 X
                                                      SimpleDemo > sorting-algorithm.py >
         (A) GitHub Copilet
                                                            der quickSort(array, low, high):
         Hi @bryantson, how can I help you?
                                                                     pivot = partition(array, low, high)
                                                                     quickSort(array, low, pivot - 1)
         I'm powered by Al. so surprises and mistakes
                                                                     quickSort(array, pivot + 1, high)
         are possible. Make sure to verify any
                                                                  return array
         generated code or suggestions, and share
         feedback so that we can learn and improve.
                                                             def partition(array, low, high):
                                                                 pivot = array[high]
                                                                  for 1 in range(low, high):
                                                                          array[i], array[j] = array[j], array[i]
                                                                 array[i + 1], array[high] = array[high], array[i + 1]
                                                            def bubbleSort(array) {
                                                                 var len = array.length
                                                                  for (var i = 0; i < len; i++) (
                                                                 return array:
                                                                  var preIndex, current;
                                                                     current = array[1]:
                                                                         array[preIndex + 1] = array[preIndex];
                                                                     array preIndex + 11 = current;
y ⊗2∆0 ₩0
                                                                                                                                            Ln 19. Col 4 Spaces: 4 UTF-8 LF (1 Python 3.11.5 64-bit 23 C
```



/fix, /tests and other / command options

Leverage / command options that can help to improve your code

```
sorting-algorithm.py -- Untitled (Workspace)
CHAT: GITHUB COPILOT
                                            sorting-algorithm.py 2 X
                                             SimpleDemo > o sorting-algorithm.py >
(B) GitHub Copilot
                                                    def quickSort(array, low, high):
Hi @bryantson, how can I help you?
                                                            pivot = partition(array, low, high)
                                                            quickSort(array, low, pivot - 1)
I'm powered by AI, so surprises and mistakes
                                                            quickSort(array, pivot + 1, high)
are possible. Make sure to verify any
generated code or suggestions, and share
 eedback so that we can learn and improve.
                                                    def partition(array, low, high):
                                                        pivot = array [high]
                                                        for j in range(low, high):
                                                            if array[j] < pivot:
                                                                 array[i]; array[i] = array[i]; array[i]
                                                        array[i + 1], array[high] = array[high], array[i + 1]
                                                    def bubbleSort(array) {
                                                        var len = array.length
                                                                 if (array[i] > array[i + 1]) (
                                                                     var temp = array[i]:
                                                    def insertionSort(array) {
                                                            preIndex = i - 1;
                                                            while [preIndex >= 0 && array[preIndex] > current]
                                                                 array[preIndex + 1] = array[preIndex];
                                                            array preIndex + 1 = current;
                                                                                                                                   Ln 19, Col 4 Spaces: 4 UTF-8 LF () Python 3.11.5 64-bit & C
```

In-file Copilot options

Copilot now offers in-file Copilot feature to selectively improve

```
sorting-algorithm.py 6
        SimpleDemo > ♦ sorting-algorithm.py > ⊕ insertionSort
               def guickSort(array) {
                   var pivot = array[0]
                   for (var i = 1; i < array.length; i++) {
                       array[i] < pivot ? left.push(array[i]) : right.push(array[i]);
                   return quickSort(left).concat(pivot, quickSort(right));
               def bubbleSort(array) {
                      swapped = false:
                       for (var i = 0; i < array.length; i++) {
                              var temp = array[i];
                               swapped = true;
                   for (var i = 1: i < length: i++) {
                       for (var i = i - 1; i >= 0 55 array[i] > temp; i--) {
                           array[i + 1] = array[i];
                      array[j + 1] = temp:
№ ⊗4∆2 ₩0
                                                                                                                                    Ln 30, Col 4 Spaces: 4 UTF-8 LF ( Python 3.11.5 64-bit & (
```

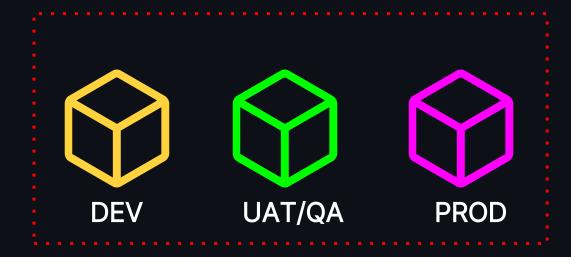


What about my production layer?

Local



Cloud/Server



Current Copilot

"Copilot X"



Copilot Security Exploit?

Prompt Injection Carefully crafted prompts to make the model to ignore its original instruction or perform unintended actions







Workshop



Choose Workshop Track



Workshop for those who want to see how to build web application

Primary IDE: VS Code Primary OS: Independent



Workshop for those who want to see how to build Java applications

Primary IDE: IntelliJ
Primary OS: Independent



Workshop for those who want to see how to build .NET applications

Primary IDE: Visual Studio

IDE

Primary OS: Windows



Copilot and secure coding

Copilot + GitHub Advanced Security

Vulnerability detection

Vulnerability remediation

Copilot Trust Center

Secure coding

Copilot and secure coding

- Al-based vulnerability system that helps prevent insecure coding patterns (e.g. SQL script injection)
- Vulnerability filters are applied only to the Copilotgenerated suggestion itself
 - It cannot detect downstream vulnerabilities introduced by the code e.g. on deployment infrastructure
 - i We recommend taking the same precautions you take with code written by engineers (linting, code scanning, etc.)
- Copilot Chat can be used to query code for known vulnerabilities

```
••• O
     var app = express()
     app.use(bodyParser.json())
     app.use(bodyParser.urlencoded({
         extended: true
     app.get("/", function(reg){
         const user = req.params.q
         if (user != "") {
             pool.query('SELECT * FROM users WHERE name = $1', [user], (error
                  if (error) {
                      throw error
                 res.status(200).json(results.rows)
     app.listen(8000, function () {
         console.log("Server running");
```

Copilot + GHAS

- Copilot is not a replacement of GHAS features
- Copilot can be used in tandem with GHAS features to detect and remediate vulnerabilities earlier during the SDLC
 - O GHAS Code scanning results
 - O GHAS Secret scanning



Vulnerability detection

- Prompt for insecure coding patterns
- Use GHAS Code Scanning results
- Write custom CodeQL queries
- Increase your knowledge of secure coding patterns
- Create custom SecretScanning patterns

```
(g) stebje

It looks like this code is insecure. Help me understand what the issue is and how to resolve it.
```

Yes, this code is vulnerable to SQL injection attacks. The issue is that the search parameter is directly concatenated into the SQL query string without any sanitization or validation. This means that an attacker could potentially inject malicious SQL code into the query, which could

GitHub Copilot

lead to data theft or destruction.

To fix this issue, you should use parameterized queries instead of concatenating strings. Parameterized queries use placeholders for the input values, which are then bound to the query separately. This ensures that the input values are properly sanitized and validated before being used in the query.

Here's an example of how to use parameterized queries with the pg library:

```
const search = req.params.q;
if (search != "") {
  const squery = "SELECT * FROM users WHERE name = $1";
  const values = [search];
  pool.query(squery, values, (err, res) => {
    console.log(err, res);
    pool.end();
    ));
}
```

In this example, the \$1 placeholder is used for the search parameter, and the actual value is passed in as an array of values to the pool.query() function. This ensures that the search parameter is properly sanitized and validated before being used in the query.

```
const pool = new Pool( config: {
  user: 'postares'.
 host: 'localhost'.
  database: 'postgres'.
  password: 'process.env.POSTGRES_PW',
 port: 5432.
var app = express();
app.use(bodyParser.ison()):
app.use(bodyParser.urlencoded({ extended: true }));
app.get('/', function (req, res) {
 const search = reg.params.g:
  if (search != "") {
   var squery == "SELECT * FROM users WHERE name == \"" + search + "\"";
    pool.query( queryTextOrConfig: squery, callback: (err, res) => {
     console.log( message: err, optionalParams[0]: res);
     -pool.end();
app.listen( port: 8000, callback: function () {
  console.log( message: 'Example app listening on port 8000!');
```

Vulnerability remediation

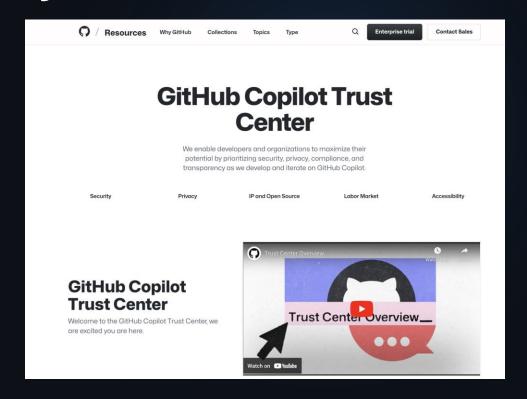
- Copilot helps prevent suggestions that contain insecure coding patterns in realtime
- Automatically fix insecure code based on Copilot suggestions
- Validate and improve existing CodeQL queries



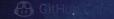
Security & Trust

Copilot Trust Center

- Security
- Privacy
- Data flow
- Copyright
- Labor market
- Accessibility
- Contracting







Wrap Up