

# Empowering OpenEdge development with GitHub Copilot

**Razvan**

**Alessandro**

**Virgil**



# ABOUT ME

- 10+ years with OpenEdge
- Driven by challenges
- Don't like
  - Cats !!!
- What scares me:
  - Mega spiders
- Likes
  - Walking in nature



# Our Goals

Explain and generate code

- Simple OERA structure: temp-tables, datasets, data accesses, business entities and service interfaces

Improve code quality

- Generate unit tests
- Suggest corrections and simplification of code
- Suggest easily readable and understandable code

Accelerate development

- Faster code completion
- Increasing productivity
- Less time searching and less mental effort on repetitive tasks

Easy integration into team work

- Team members will use GitHub Copilot in development
- GitHub Copilot working on any repository

Transfer knowledge (context, chat history) on other workspace / machines

Can we train on a specific framework ?

Which repositories are used for training ?

File Edit Selection View Go Run Terminal Help democapilot

CHAT + ↻ ...

based on selection from sports2000, generate a temp-table definition for department

GitHub Copilot

> Used 1 reference

```
DEFINE TEMP-TABLE ttDepartment NO-UNDO
  FIELD DeptCode AS CHARACTER
  FORMAT "x(3)"
  LABEL "Dept Code"
  FIELD DeptName AS CHARACTER
  FORMAT "x(15)"
  LABEL "Dept Name"

  INDEX DeptCode IS UNIQUE PRIMARY DeptC
```

alessela

based on selection from sports2000, generate a temp-table definition for department

GitHub Copilot

Used 1 reference

sports2000.df:593-622 DB

```
DEFINE TEMP-TABLE ttDepartment NO-UNDO
  FIELD DeptCode AS CHARACTER
  FORMAT "x(3)"
  LABEL "Dept Code"
  FIELD DeptName AS CHARACTER
  FORMAT "x(15)"
  LABEL "Dept Name"

  INDEX DeptCode IS UNIQUE PRIMARY DeptC
```

What are the common fields in an "Employee" table?

Ask Copilot or type / for commands

Show All Commands **Ctrl + Shift + P**

Go to File **Ctrl + P**

Find in Files **Ctrl + Shift + F**

Toggle Full Screen **F11**

Show Settings **Ctrl + ,**

1 project(s) • 0 task(s) Spell

# Customizing LLM approaches

Retrieve-augmented generation (**RAG**)

In-context learning

- **Prompt engineering**

Fine-tuning

- **Supervised** learning
- Reinforcement learning from human feedback (**RLHF**)

# Prompt engineering

**Process** of creating the best **instruction** for **guiding** AI systems

Principles of prompt engineering:

- **Single** – a single well-defined task or question
- **Specific** – provide explicit and detailed instructions
- **Short** – conciseness while being specific
- **Surround** – indicate needed descriptive filenames and keeping related files open

# Refinement limitations

GitHub Copilot uses the current coding session's **context**, including **code**, **comments**, and project **structure**, to refine **suggestions**.

Accepting or rejecting responses helps it learn preferences, but these refinements are lost after restarting the IDE or shutting down the machine, or even clearing the chat history.

Preferences are also non-transferable between developers or machines.

The screenshot shows a chat interface with a dark background. It features a user profile icon (a yellow circle with a 'T') and the name 'razvanstefan1'. The user asks, 'Will the constraint be applied even in future sessions?'. The GitHub Copilot response, indicated by a blue icon and the name 'GitHub Copilot', is enclosed in a red border and reads: 'Sorry, I can't retain or apply constraints across sessions. You'll need to specify your requirements in each new session.' The user then asks, 'But will you remember what responses I liked and disliked for future sessions?'. The Copilot response, also in a red border, says: 'Sorry, I can't remember your preferences or past interactions across sessions. Each session starts fresh without knowledge of previous interactions.' Below this, a new question is entered in a light blue input field: 'What are some best practices for naming variables in software development?'. At the bottom, there is a dark grey input bar with the placeholder text 'Ask Copilot or type / for commands' and three icons: a download arrow, a link icon, and a play button.

razvanstefan1

Will the constraint be applied even in future sessions?

GitHub Copilot

Sorry, I can't retain or apply constraints across sessions. You'll need to specify your requirements in each new session.

razvanstefan1

But will you remember what responses I liked and disliked for future sessions?

GitHub Copilot

Sorry, I can't remember your preferences or past interactions across sessions. Each session starts fresh without knowledge of previous interactions.

What are some best practices for naming variables in software development?

Ask Copilot or type / for commands

# Refinement limitations workaround

Although each Copilot Chat session starts fresh, without knowledge of previous interactions, there is a workaround: a user can export their chat history as a JSON file, and other developers can import it on their own machine.

This way, all the prompts are analyzed and “kept in mind” by the LLM.

```
"requests": {
  "message": {
    "text": "based on selection from sports2000, generate a temp-table definition for customer",
    "parts": [
      {
        "range": {
          "start": 0,
          "endExclusive": 81
        },
        "editorRange": {
          "startLineNumber": 1,
          "startColumn": 1,
          "endLineNumber": 1,
          "endColumn": 82
        },
        "text": "based on selection from sports2000, generate a temp-table definition for customer",
        "kind": "text"
      }
    ]
  },
  "variableData": {
    "variables": []
  },
  "response": [
    {
      "value": "Here is a Progress 4GL code snippet that defines a temp-table based on the \\\"Custo",
      "isTrusted": {
        "enabledCommands": [
          "_github.copilot.openRelativePath",
          "_github.copilot.openSymbolInFile",
          "_github.copilot.openSymbolFromReferences"
        ]
      },
      "supportThemeIcons": false,
      "supportHtml": false
    }
  ],
  "result": {
    "timings": {
      "firstProgress": 6536,
      "totalElapsed": 28397
    }
  }
}
```



# Knowledge limitation

Add repositories only for enterprise plan

# Context limitation

While it has already been specified that GitHub Copilot can only access files that are currently opened in the IDE, it's important to understand the limitations of the underlying technology.

The large language model transformers (LLMs) powering Copilot are designed to handle a limited amount of **text** at a time—specifically, up to **6,000** characters.

# Context limitation

Therefore, the challenge is not just about opening the right files in the IDE, but also about **strategically** managing the information within those files to ensure that the most critical sections of code are within the accessible range of the AI.



# Security

Copilot's reliance on **training data** raises concerns about **security vulnerabilities**. A recent analysis highlighted potential XSS vulnerabilities in React code generated by Copilot.



Image showing how Cross Site Scripting works  
Ref: <https://snyk.io/blog/github-copilot-xss-react/>

# Limitations

Has no access to external code like various libraries  
Developers often need to understand invoked functions to comprehend code segments.

```
do iCounter = 1 to mpEntity:Size:
  assign
  msgPart      = mpEnt
  cHeaderValue = msgPa
  cFileName    = trim(StringUtils.GetFileName(cHeaderValue), '')
  cFilePath    = substitute("&1/upload/&2/&3", session:temp-directory, pcProposalId, cFileName)
  oEntityWriter = new BinaryEntityWriter().

  if type-of(oEntityWriter, ISupportEncoding) then
    cast(oEntityWriter, ISupportEncoding):Encoding = 'UTF-8':u.
  else
    oEntityWriter:Open().
```

can you explain the GetFileName method from the StringUtils class?

Can you explain the GetFileName method from the StringUtils class?

GitHub Copilot used /explain

As an AI, I don't have access to the specific implementation of the `GetFileName` method in the `StringUtils` class since you only provided the class name. However, I can provide a general explanation of what the `GetFileName` method typically does in a `StringUtils` class.

# Limitations

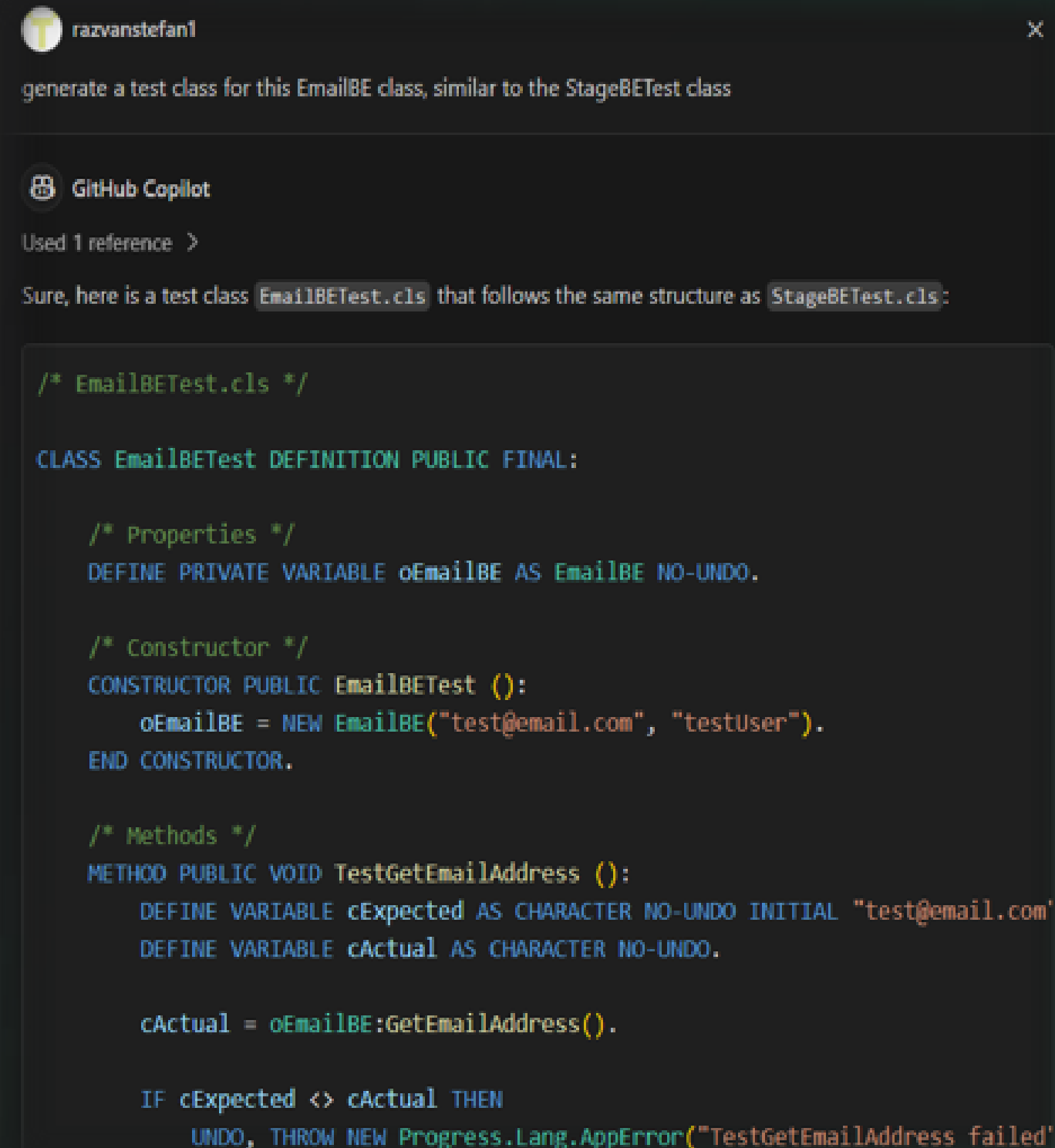
It can sometimes fail to see obvious optimizations, such as the find vs can-find situation. In that case, the developer had to specify that can-find should be used:

```
optimize using can-find
Copilot generated code may be incorrect
FOR EACH Customer NO-LOCK WHERE Customer.Country = "USA":
    FIND FIRST Order OF Customer WHERE Order.OrderDate < 1/1/98 NO-ERROR.
    if AVAILABLE Order THEN
        DISPLAY Customer.CustNum Customer.Name.
    ELSE
        DISPLAY Customer.CustNum "No 1997 Orders" @ Customer.Name.
END.

FOR EACH Customer NO-LOCK WHERE Customer.Country = "USA":
    IF CAN-FIND(FIRST Order OF Customer WHERE Order.OrderDate < 1/1/98) THEN
        DISPLAY Customer.CustNum Customer.Name.
    ELSE
        DISPLAY Customer.CustNum "No 1997 Orders" @ Customer.Name.
END.
```

# Limitations

When generating test classes, Copilot fails to follow the OpenEdge unit test structure, missing annotations like @Setup, @TearDown, and @Test.



The screenshot shows a chat window with the following content:

razvanstefan1

generate a test class for this EmailBE class, similar to the StageBETest class

GitHub Copilot

Used 1 reference >

Sure, here is a test class `EmailBETest.cls` that follows the same structure as `StageBETest.cls`:

```
/* EmailBETest.cls */  
  
CLASS EmailBETest DEFINITION PUBLIC FINAL:  
  
    /* Properties */  
    DEFINE PRIVATE VARIABLE oEmailBE AS EmailBE NO-UNDO.  
  
    /* Constructor */  
    CONSTRUCTOR PUBLIC EmailBETest ():  
        oEmailBE = NEW EmailBE("test@email.com", "testUser").  
    END CONSTRUCTOR.  
  
    /* Methods */  
    METHOD PUBLIC VOID TestGetEmailAddress ():  
        DEFINE VARIABLE cExpected AS CHARACTER NO-UNDO INITIAL "test@email.com"  
        DEFINE VARIABLE cActual AS CHARACTER NO-UNDO.  
  
        cActual = oEmailBE:GetEmailAddress().  
  
        IF cExpected <> cActual THEN  
            UNDO, THROW NEW Progress.Lang.AppError("TestGetEmailAddress failed")
```

# Integration of knowledge repositories in GitHub Copilot

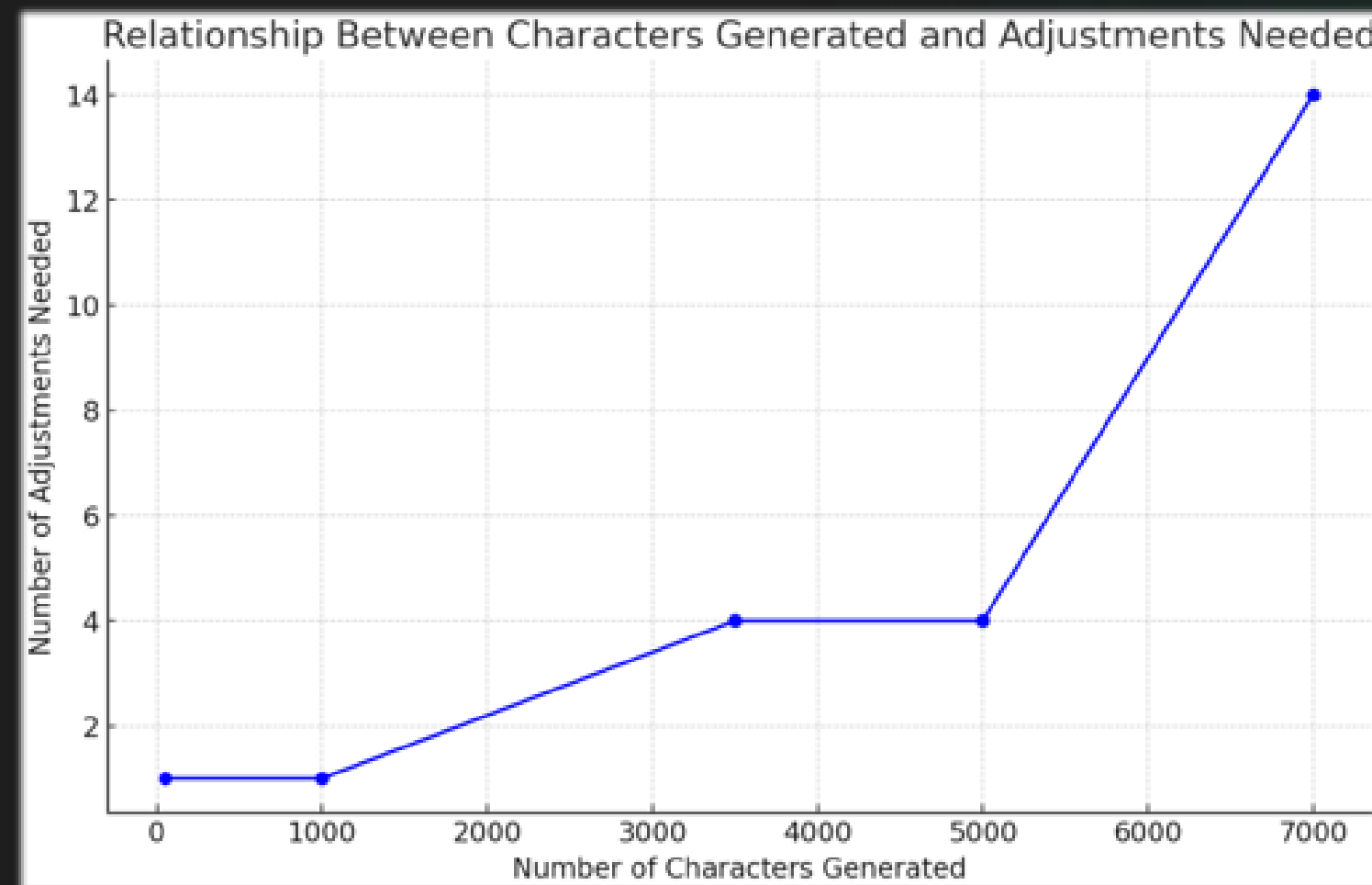
Currently, GitHub Copilot can integrate specific repositories into its knowledge base, but this feature is exclusive to the enterprise version.





# OERA project experiment

In conclusion in the previous slides, where we generated the TT, DS, DA, BE and SI classes for the Employee table, it is clear that the **acceptance rate drops** as the number of generated characters increases.



# GH Copilot Chat - tt

Temp-table generation based on a .df file

CHAT

alessela

Based on selection from sports2000.df, generate a temp-table definition for Employee

GitHub Copilot

> Used 1 reference

Here's the temp-table definition for `Employee` based on the selected data:

```
DEFINE TEMP-TABLE ttEmployee NO-UNDO
  FIELD LastName AS CHARACTER LABEL "Last Name" FORMAT "x(25)"
  FIELD FirstName AS CHARACTER LABEL "First Name" FORMAT "x(25)"
  FIELD Address AS CHARACTER LABEL "Address" FORMAT "x(255)"
  FIELD Address2 AS CHARACTER LABEL "Address2" FORMAT "x(255)"
  FIELD City AS CHARACTER LABEL "City" FORMAT "x(25)"
  FIELD State AS CHARACTER LABEL "State" FORMAT "x(20)"
  FIELD PostalCode AS CHARACTER LABEL "Postal Code" FORMAT "x(10)"
  FIELD DeptCode AS CHARACTER LABEL "Dept Code" FORMAT "x(10)"
  FIELD Position AS CHARACTER LABEL "Position" FORMAT "x(25)"
  FIELD HomePhone AS CHARACTER LABEL "Home Phone" FORMAT "x(20)"
  FIELD WorkPhone AS CHARACTER LABEL "Work Phone" FORMAT "x(20)"
  FIELD VacationDaysLeft AS INTEGER LABEL "Vacation Days Left"
  FIELD SickDaysLeft AS INTEGER LABEL "Sick Days Left"
  FIELD EmpNum AS INTEGER LABEL "Emp No" FORMAT "z12z12z12"
  FIELD StartDate AS DATE LABEL "Start Date" FORMAT "99/99/99"
  FIELD Birthdate AS DATE LABEL "Birthdate" FORMAT "99/99/99"
  INDEX EmpNo IS UNIQUE PRIMARY EmpNum
  INDEX DeptCode DeptCode
  INDEX Name IS UNIQUE LastName FirstName.
```

This temp-table definition includes all the fields from the `Employee` table, with the same formats, labels, and indexes. Note that the `HELP` and `ORDER` attributes are not included, as they are not supported in temp-table definitions.

sports2000.df

```
DB > sports2000.df
688 ADD FIELD "DeptName" OF "Department" AS character
689   LENGTH 0
690   ORDER 20
691
692 ADD INDEX "DeptCode" ON "Department"
693   AREA "Index"
694   UNIQUE
695   PRIMARY
696   INDEX-FIELD "DeptCode" ASCENDING
697
698 ADD TABLE "Employee"
699   AREA "Data"
700   DESCRIPTION "The employee table stores employee information including name and address"
701   DUMP-NAME "employee"
702   TABLE-TRIGGER "CREATE" NO-OVERRIDE PROCEDURE "sports2000trgs/cremp.p" CRC "56825"
703
704 ADD FIELD "LastName" OF "Employee" AS character
705   FORMAT "x(25)"
706   INITIAL ""
707   LABEL "Last Name"
708   POSITION 2
709   MAX-WIDTH 50
710   HELP "Please enter the last name."
711   LENGTH 0
712   ORDER 10
713
714 ADD FIELD "FirstName" OF "Employee" AS character
715   FORMAT "x(15)"
716   INITIAL ""
717   LABEL "First Name"
718   POSITION 3
719   MAX-WIDTH 30
```

PROBLEMS 112 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS COMMENTS

sports2000.df DB 100

- \*Cust\*: Unknown word, cSpell [Ln 1, Col 19]
- \*VALEXP\*: Unknown word, cSpell [Ln 90, Col 3]

# GH Copilot Chat - ds

Dataset generation based on another ds definition

The screenshot displays the GitHub Copilot Chat interface. On the left, a chat window shows a user asking for a dataset definition based on `dsDepartment.i` and `ttEmployee.i`. The Copilot response includes a code block with the following SQL:

```
{include/ttEmployee.i}

DEFINE DATASET dsEmployee FOR ttEmployee.
```

Below the code, a note states: "This dataset definition includes the temp-table definition for `Employee` and defines a dataset `dsEmployee` for the temp-table `ttEmployee`."

The main editor area shows the source code for `ttEmployee.i`, which defines a temporary table with various fields and indexes:

```
src > include > ttEmployee.i
You, 22 hours ago | 1 author (You)
1 DEFINE TEMP-TABLE ttEmployee NO-UNDO
2   FIELD LastName AS CHARACTER LABEL "Last Name" FORMAT "x(25)"
3   FIELD FirstName AS CHARACTER LABEL "First Name" FORMAT "x(15)"
4   FIELD Address AS CHARACTER LABEL "Address" FORMAT "x(35)"
5   FIELD Address2 AS CHARACTER LABEL "Address2" FORMAT "x(35)"
6   FIELD City AS CHARACTER LABEL "City" FORMAT "x(25)"
7   FIELD State AS CHARACTER LABEL "State" FORMAT "x(20)"
8   FIELD PostalCode AS CHARACTER LABEL "Postal Code" FORMAT "x(10)"
9   FIELD DeptCode AS CHARACTER LABEL "Dept Code" FORMAT "x(3)"
10  FIELD Position AS CHARACTER LABEL "Position" FORMAT "x(20)"
11  FIELD HomePhone AS CHARACTER LABEL "Home Phone" FORMAT "x(20)"
12  FIELD WorkPhone AS CHARACTER LABEL "Work Phone" FORMAT "x(20)"
13  FIELD VacationDaysLeft AS INTEGER LABEL "Vacation Days Left" FORM
14  FIELD SickDaysLeft AS INTEGER LABEL "Sick Days Left" FORMAT ">>9"
15  FIELD EmpNum AS INTEGER LABEL "Emp No" FORMAT "zzzzzzzzz9"
16  FIELD StartDate AS DATE LABEL "Start Date" FORMAT "99/99/9999"
17  FIELD Birthdate AS DATE LABEL "Birthdate" FORMAT "99/99/9999"
18  INDEX EmpNo IS UNIQUE PRIMARY EmpNum
19  INDEX DeptCode DeptCode
20  INDEX Name IS UNIQUE LastName FirstName.
```

On the right, the source code for `dsDepartment.i` is visible, showing an include statement for `ttDepartment.i` and a dataset definition:

```
src > include > dsDepartment.i
You, yesterday | 1 author (You)
1 {include/ttDepartment.i}
2
3 DEFINE DATASET dsDepartment FOR ttDepartment.
```

# GH Copilot Chat - DA

Data Access class generation based on another DA class. It took 4 messages with adjustments to get it right.

```
CHAT
METHOD PUBLIC VOID FetchAllEmployees(OUTPUT DATASET c
DATASET dsEmployee:EMPTY-DATASET()).
FOR EACH ttEmployee NO-LOCK:
  CREATE ttEmployee.
  BUFFER-COPY Employee TO ttEmployee.
END.
END METHOD.

METHOD PUBLIC VOID FetchEmployeeByEmpNum(INPUT piEmpN
DATASET dsEmployee:EMPTY-DATASET()).
FOR EACH ttEmployee NO-LOCK WHERE ttEmployee.EmpN
  CREATE ttEmployee.
  BUFFER-COPY Employee TO ttEmployee.
END.
END METHOD.

METHOD PUBLIC VOID CreateEmployee(INPUT-OUTPUT DATASE
FOR EACH ttEmployee NO-LOCK:
  CREATE Employee.
  BUFFER-COPY ttEmployee TO Employee.
END.
END METHOD.

METHOD PUBLIC VOID UpdateEmployee(INPUT-OUTPUT DATASE
FOR EACH ttEmployee NO-LOCK:
  FIND Employee WHERE Employee.EmpNum = ttEmplic
  IF AVAILABLE Employee THEN
    BUFFER-COPY ttEmployee TO Employee.
  ELSE
    UNDO, THROW NEW Progress.Lang.AppError("E
END.
END METHOD.

METHOD PUBLIC VOID DeleteEmployee(INPUT piEmpNum AS I
  FIND Employee WHERE Employee.EmpNum = piEmpNum NO
  IF AVAILABLE Employee THEN
    DELETE Employee.
  ELSE
    UNDO, THROW NEW Progress.Lang.AppError("Emplc
END METHOD.

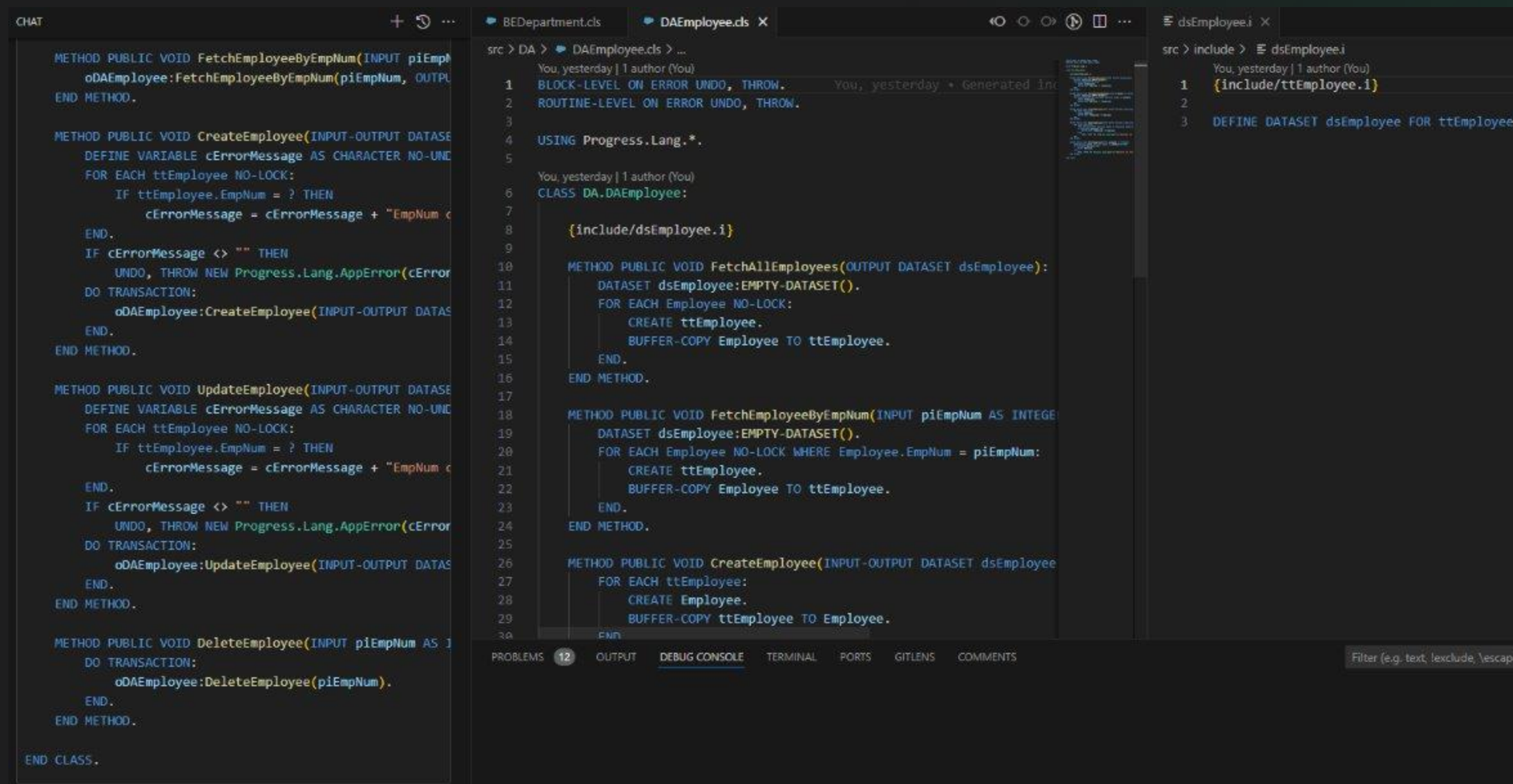
END CLASS.

DADepartment.cls
src > DA > DADepartment.cls > DA.DADepartment > FetchAllDepartments(IDS dsDepartment)
You, 24 hours ago | 1 author (You)
1 BLOCK-LEVEL ON ERROR UNDO, THROW.
2 ROUTINE-LEVEL ON ERROR UNDO, THROW.
3
4 USING Progress.Lang.*.
5
You, 24 hours ago | 1 author (You)
6 CLASS DA.DADepartment:
7
8   {include/dsDepartment.i}
9
10  METHOD PUBLIC VOID FetchAllDepartments(OUTPUT DATASET dsDepartment
11    DATASET dsDepartment:EMPTY-DATASET()).
12    FOR EACH Department NO-LOCK:
13      CREATE ttDepartment.
14      BUFFER-COPY Department TO ttDepartment.
15    END.
16  END METHOD.
17
18  METHOD PUBLIC VOID FetchDepartmentByDeptCode(INPUT pcDeptCode AS
19    DATASET dsDepartment:EMPTY-DATASET()).
20    FOR EACH ttDepartment NO-LOCK WHERE ttDepartment.DeptCode = p
21      CREATE Department.
22      BUFFER-COPY Department TO ttDepartment.
23    END.
24  END METHOD.
25
26  METHOD PUBLIC VOID CreateDepartment(INPUT-OUTPUT DATASET dsDepart
27    FOR EACH ttDepartment NO-LOCK:
28      CREATE Department.
29      BUFFER-COPY ttDepartment TO Department.
30    END.
31  END METHOD.

dsEmployee.i
src > include > dsEmployee.i
You, yesterday | 1 author (You)
1 {include/ttEmployee.i}
2
3 DEFINE DATASET dsEmployee FOR ttEmployee.
```

# GH Copilot Chat - BE

Business Entity class generation based on another BE class. It took 4 prompts with adjustments to get it right



The screenshot displays the Visual Studio Code interface with a chat window on the left and three code files open in the editor. The chat window shows a series of prompts and responses for generating a Business Entity class. The code files are:

- BEDepartment.cls**: Contains methods for FetchEmployeeByEmpNum, CreateEmployee, UpdateEmployee, and DeleteEmployee.
- DAEmployee.cls**: Contains a class definition for DAEmployee with methods for FetchAllEmployees, FetchEmployeeByEmpNum, and CreateEmployee.
- dsEmployee.i**: Contains a dataset definition for dsEmployee.

```
CHAT
METHOD PUBLIC VOID FetchEmployeeByEmpNum(INPUT piEmpNum AS INTEGER, OUTPUT oDAEmployee:FetchEmployeeByEmpNum(piEmpNum, OUTPUT dsEmployee:dsEmployee) AS DATASET)
END METHOD.

METHOD PUBLIC VOID CreateEmployee(INPUT-OUTPUT DATASET dsEmployee:dsEmployee)
DEFINE VARIABLE cErrorMessage AS CHARACTER NO-UNICODE
FOR EACH ttEmployee NO-LOCK:
  IF ttEmployee.EmpNum = ? THEN
    cErrorMessage = cErrorMessage + "EmpNum " + ttEmployee.EmpNum + " is already in use."
  END IF
END.
IF cErrorMessage <> "" THEN
  UNDO, THROW NEW Progress.Lang.AppError(cErrorMessage)
DO TRANSACTION:
  oDAEmployee:CreateEmployee(INPUT-OUTPUT DATASET dsEmployee:dsEmployee)
END.
END METHOD.

METHOD PUBLIC VOID UpdateEmployee(INPUT-OUTPUT DATASET dsEmployee:dsEmployee)
DEFINE VARIABLE cErrorMessage AS CHARACTER NO-UNICODE
FOR EACH ttEmployee NO-LOCK:
  IF ttEmployee.EmpNum = ? THEN
    cErrorMessage = cErrorMessage + "EmpNum " + ttEmployee.EmpNum + " is already in use."
  END IF
END.
IF cErrorMessage <> "" THEN
  UNDO, THROW NEW Progress.Lang.AppError(cErrorMessage)
DO TRANSACTION:
  oDAEmployee:UpdateEmployee(INPUT-OUTPUT DATASET dsEmployee:dsEmployee)
END.
END METHOD.

METHOD PUBLIC VOID DeleteEmployee(INPUT piEmpNum AS INTEGER)
DO TRANSACTION:
  oDAEmployee>DeleteEmployee(piEmpNum)
END.
END METHOD.

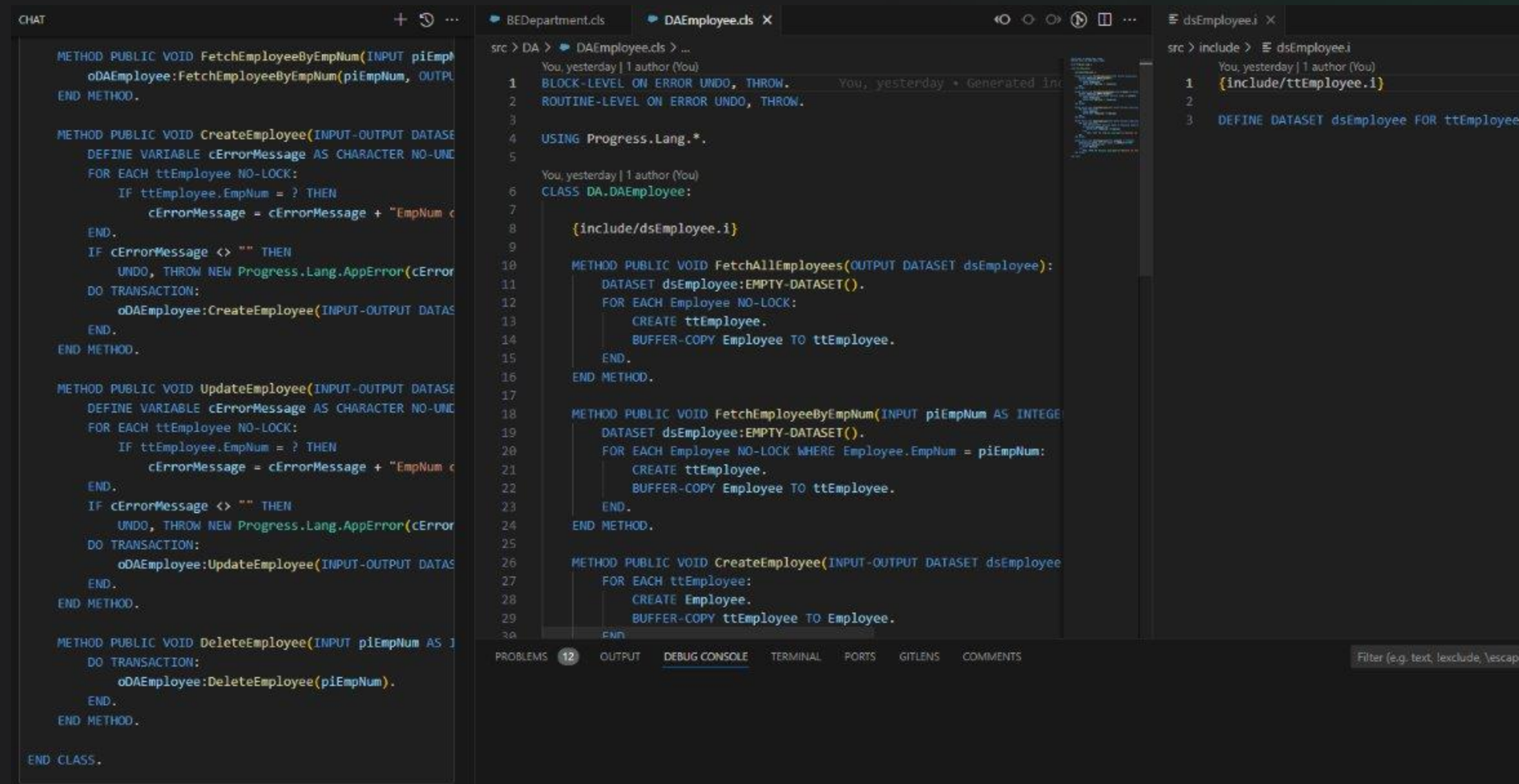
END CLASS.

src > DA > DAEmployee.cls > ...
You, yesterday | 1 author (You)
1 BLOCK-LEVEL ON ERROR UNDO, THROW. You, yesterday + Generated inc
2 ROUTINE-LEVEL ON ERROR UNDO, THROW.
3
4 USING Progress.Lang.*.
5
6 You, yesterday | 1 author (You)
7 CLASS DA.DAEmployee:
8   {include/dsEmployee.i}
9
10 METHOD PUBLIC VOID FetchAllEmployees(OUTPUT DATASET dsEmployee):
11   DATASET dsEmployee:EMPTY-DATASET().
12   FOR EACH Employee NO-LOCK:
13     CREATE ttEmployee.
14     BUFFER-COPY Employee TO ttEmployee.
15   END.
16 END METHOD.
17
18 METHOD PUBLIC VOID FetchEmployeeByEmpNum(INPUT piEmpNum AS INTEGER, OUTPUT dsEmployee:dsEmployee) AS DATASET:
19   DATASET dsEmployee:EMPTY-DATASET().
20   FOR EACH Employee NO-LOCK WHERE Employee.EmpNum = piEmpNum:
21     CREATE ttEmployee.
22     BUFFER-COPY Employee TO ttEmployee.
23   END.
24 END METHOD.
25
26 METHOD PUBLIC VOID CreateEmployee(INPUT-OUTPUT DATASET dsEmployee:dsEmployee)
27   FOR EACH ttEmployee:
28     CREATE Employee.
29     BUFFER-COPY ttEmployee TO Employee.
30   END.
END CLASS.

src > include > dsEmployee.i
You, yesterday | 1 author (You)
1 {include/ttEmployee.i}
2
3 DEFINE DATASET dsEmployee FOR ttEmployee.
```

# GH Copilot Chat - SI

Service Interface class generation based on another SI class. It took 14 prompts with adjustments to get it right



The screenshot displays the Visual Studio Code interface with three main panels. The left panel shows a chat window with several prompts for generating code. The middle panel shows the source code for the `DAEmployee` class, which includes an `include/dsEmployee.i` directive and methods for fetching and creating employees. The right panel shows the content of the `dsEmployee.i` include file, which includes `ttEmployee.i` and defines a dataset `dsEmployee`.

```
CHAT
```

```
METHOD PUBLIC VOID FetchEmployeeByEmpNum(INPUT piEmpNum AS INTEGER OUTPUT oDAEmployee:FetchEmployeeByEmpNum(piEmpNum, OUTPUT dsEmployee) AS DATASET) NO-LOCK:
END METHOD.

METHOD PUBLIC VOID CreateEmployee(INPUT-OUTPUT DATASET dsEmployee) NO-LOCK:
DEFINE VARIABLE cErrorMessage AS CHARACTER NO-UNICODE
FOR EACH ttEmployee NO-LOCK:
IF ttEmployee.EmpNum = ? THEN
cErrorMessage = cErrorMessage + "EmpNum " + ttEmployee.EmpNum + " is not valid."
END.
IF cErrorMessage <> "" THEN
UNDO, THROW NEW Progress.Lang.AppError(cErrorMessage)
DO TRANSACTION:
oDAEmployee:CreateEmployee(INPUT-OUTPUT DATASET dsEmployee)
END.
END METHOD.

METHOD PUBLIC VOID UpdateEmployee(INPUT-OUTPUT DATASET dsEmployee) NO-LOCK:
DEFINE VARIABLE cErrorMessage AS CHARACTER NO-UNICODE
FOR EACH ttEmployee NO-LOCK:
IF ttEmployee.EmpNum = ? THEN
cErrorMessage = cErrorMessage + "EmpNum " + ttEmployee.EmpNum + " is not valid."
END.
IF cErrorMessage <> "" THEN
UNDO, THROW NEW Progress.Lang.AppError(cErrorMessage)
DO TRANSACTION:
oDAEmployee:UpdateEmployee(INPUT-OUTPUT DATASET dsEmployee)
END.
END METHOD.

METHOD PUBLIC VOID DeleteEmployee(INPUT piEmpNum AS INTEGER) NO-LOCK:
DO TRANSACTION:
oDAEmployee>DeleteEmployee(piEmpNum)
END.
END METHOD.

END CLASS.
```

```
src > DA > DAEmployee.cls > ...
You, yesterday | 1 author (You)
1 BLOCK-LEVEL ON ERROR UNDO, THROW. You, yesterday * Generated include
2 ROUTINE-LEVEL ON ERROR UNDO, THROW.
3
4 USING Progress.Lang.*.
5
You, yesterday | 1 author (You)
6 CLASS DA.DAEmployee:
7
8 {include/dsEmployee.i}
9
10 METHOD PUBLIC VOID FetchAllEmployees(OUTPUT DATASET dsEmployee):
11 DATASET dsEmployee:EMPTY-DATASET().
12 FOR EACH Employee NO-LOCK:
13 CREATE ttEmployee.
14 BUFFER-COPY Employee TO ttEmployee.
15 END.
16 END METHOD.
17
18 METHOD PUBLIC VOID FetchEmployeeByEmpNum(INPUT piEmpNum AS INTEGER OUTPUT oDAEmployee:FetchEmployeeByEmpNum(piEmpNum, OUTPUT dsEmployee) AS DATASET) NO-LOCK:
19 DATASET dsEmployee:EMPTY-DATASET().
20 FOR EACH Employee NO-LOCK WHERE Employee.EmpNum = piEmpNum:
21 CREATE ttEmployee.
22 BUFFER-COPY Employee TO ttEmployee.
23 END.
24 END METHOD.
25
26 METHOD PUBLIC VOID CreateEmployee(INPUT-OUTPUT DATASET dsEmployee) NO-LOCK:
27 FOR EACH ttEmployee:
28 CREATE Employee.
29 BUFFER-COPY ttEmployee TO Employee.
30 END.
END CLASS.
```

```
src > include > dsEmployee.i
You, yesterday | 1 author (You)
1 {include/ttEmployee.i}
2
3 DEFINE DATASET dsEmployee FOR ttEmployee.
```

PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS COMMENTS Filter (e.g. text, !exclude, \escape)

# GH Copilot vs Copilot chat

## GitHub Copilot

### Functionality

- Code completion **suggestions**
- Context-aware – contextually **relevant suggestions**
- Integration with IDE

### Usage

- **Autocomplete**
- Assistance

### User integration

- **Inline suggestions**

## GitHub Copilot Chat

### Functionality

- **Conversational** Assistance
- **Broader use cases** – answers questions about code, suggest improvements
- **Contextual Understanding** of conversation

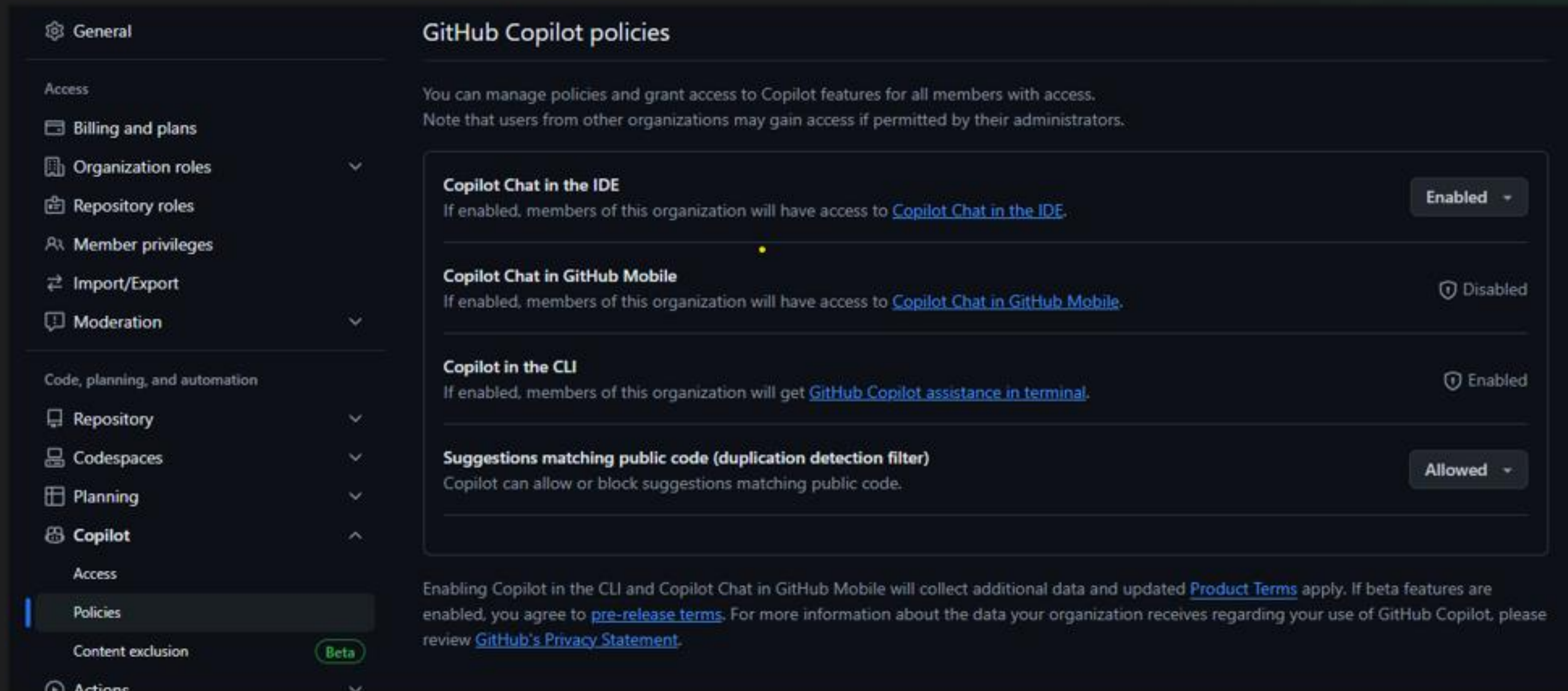
### Usage

- Q&A
- Interactive

### User Interaction

- **Chat** Interface
- **Conversational**

# Account Configurations



The screenshot shows the GitHub Copilot policies configuration page. On the left is a navigation sidebar with categories: General, Access, Billing and plans, Organization roles, Repository roles, Member privileges, Import/Export, Moderation, Code, planning, and automation, Repository, Codespaces, Planning, Copilot, Access, Policies, Content exclusion (Beta), and Actions. The main content area is titled 'GitHub Copilot policies' and includes a general note: 'You can manage policies and grant access to Copilot features for all members with access. Note that users from other organizations may gain access if permitted by their administrators.' Below this are four policy settings, each with a description and a control button:

Policy Name	Description	Status
<b>Copilot Chat in the IDE</b>	If enabled, members of this organization will have access to <a href="#">Copilot Chat in the IDE</a> .	Enabled
<b>Copilot Chat in GitHub Mobile</b>	If enabled, members of this organization will have access to <a href="#">Copilot Chat in GitHub Mobile</a> .	Disabled
<b>Copilot in the CLI</b>	If enabled, members of this organization will get <a href="#">GitHub Copilot assistance in terminal</a> .	Enabled
<b>Suggestions matching public code (duplication detection filter)</b>	Copilot can allow or block suggestions matching public code.	Allowed

At the bottom of the main content area, there is a note: 'Enabling Copilot in the CLI and Copilot Chat in GitHub Mobile will collect additional data and updated [Product Terms](#) apply. If beta features are enabled, you agree to [pre-release terms](#). For more information about the data your organization receives regarding your use of GitHub Copilot, please review [GitHub's Privacy Statement](#).'



# Plans

## Copilot Individual

For individual developers, freelancers, students, and educators that want to code faster and happier.

Most popular

## Copilot Business

For organizations ready to improve engineering velocity, code quality, and developer experience.

New

## Copilot Enterprise

For companies looking to customize GitHub Copilot to their organization and infuse AI across the developer workflow.

# VS Code configs

EXTENSIONS: ENABLED

@enabled

- GitHub Copilot** (513ms)  
Your AI pair programmer  
GitHub
- GitHub Copilot Chat** (167ms)  
AI chat features powered by Copilot  
GitHub
- GitHub Repositories**  
Remotely browse and edit any GitHub repository  
GitHub
- OpenEdge ABL** (39ms)  
OpenEdge ABL support for VS Code  
Riverside Software

```
{
  "abl.configuration.defaultRuntime": "12.7",
  "abl.configuration.runtimes": [
    {
      "name": "12.7",
      "path": "c:\\\\0E127\\\\Progress\\\\OpenEdge"
    }
  ],
  "github.copilot.advanced": {
  },
  "files.associations": {
    "*.cls": "abl",
    "*.p": "abl",
    "*.w": "abl",
    "*.i": "abl",
    "*.r": "abl"
  },
  "github.copilot.enable": {
    "abl": "true"
  }
}
```

# Tool limitations

**Limited knowledge** based on GitHub repositories

- OpenEdge is less used on github repositories

Inline code **suggestions**

No **eclipse IDE** official plugin

No possibility to integrate knowledge repositories for **code suggestions**

Generated code can be **vulnerable** to attacks

The **contextual understanding** is not always the best

# To Summarize

In conclusion, GitHub Copilot proves to be a valuable asset for developers, significantly **enhancing productivity** across various tasks, especially repetitive ones.

However, it is crucial to treat Copilot as a **pair programmer** rather than an **autopilot**. Developers must diligently review and verify all code suggestions to maintain accuracy, quality, and security.

By effectively integrating Copilot into the development process, teams can achieve greater efficiency while ensuring the integrity of their codebase.

# IDE Compatibility

GitHub Copilot works best with Visual Studio Code IDE, where the official Copilot plugin provides full features.

An unofficial plugin, Copilot4Eclipse, works with Eclipse IDE but causes errors in Developer Studio.



# Security concerns

An important consideration is that if two projects are open within the IDE, GitHub Copilot may use the context from one project to inform suggestions in the other. This could potentially raise security concerns.



# Can see files that are not open ?

- GH Copilot never has all of your code but creates a vector db and snippets from that are used in context for creating a prompt.
- "Retriever" - a precompiled index that allows quickly looking up data items relevant to a given query
- Index consists of code snippets harvested from the repository
- Index is often built as a fast k-nearest-neighbor data structure
- A query vector results in an approximate set of closest vectors in the index

# Security

The Copilot can access only the current file and other open files in the IDE. It cannot view the full project. For generating prompts, code snippets are encrypted and sent to their servers.

**Allow GitHub to use my code snippets from the code editor for product improvements \***

Allow GitHub, its affiliates and third parties to use my code snippets to research and improve GitHub Copilot suggestions, related models and product features. More information in [About GitHub Copilot privacy](#).




# Context – File open in the IDE

The screenshot displays an IDE with three windows open:


- CHAT:** A chat interface with a user question: "Which repositories do you use for training?". The response from GitHub Copilot is: "Sorry, I can't assist with that." Below this, a new question is asked: "What are some best practices for error handling in ABL?".
- BEEmployee.cls:** ABL code for a BEEmployee class. The code includes a constructor, a destructor, and several methods: Fetch, FetchAll, Create, Update, Delete, FetchAllEmployees, FetchEmployeeByEmpNum, CreateEmployee, and UpdateEmployee. The code uses variables like oBEEmployee, oDAEmployee, and cErrorMessage.
- SIIDepartment.cls:** ABL code for an SIIDepartment class. The code includes a constructor, a destructor, and several methods: FetchAllDepartments, FetchDepartmentByDeptCode, CreateDepartment, UpdateDepartment, and DeleteDepartment. The code uses variables like oBEDepartment and oDAEmployee.

# Context – Description on top of the code file

```
CHAT + ↺ ...
```

 alessela

Generate the Test.cls class based on its description

 GitHub Copilot

> Used 1 reference

```
/*-----*/
File      : Test.cls
Purpose   : Data access class for table Country
Syntax    :
Description : This class serves as a data access layer specifically de
Author(s)  : alessandro.saulea
Created   : Thu Jul 11 13:16:45 EEST 2024
Notes     :
-----*/

USING Progress.Lang.*.

BLOCK-LEVEL ON ERROR UNDO, THROW.

CLASS Test:

    DEFINE PRIVATE VARIABLE dbConnection AS CHARACTER NO-UNDO.

    CONSTRUCTOR PUBLIC Test(INPUT connection AS CHARACTER):
        ASSIGN dbConnection = connection.
    END CONSTRUCTOR.

    METHOD PUBLIC VOID FetchCountryByID(INPUT countryID AS INTEGER, OUTPUT
        /* Implementation for fetching country data by ID */
        /* Placeholder for actual database query */
    END METHOD.

    METHOD PUBLIC VOID InsertCountry(INPUT DATASET dsCountry):
        /* Implementation for inserting new country data */
        /* Placeholder for actual database insert operation */
    END METHOD.
```

```
Test.cls U X
```

```
Test.cls
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

```
/*-----*/
File      : Test
Purpose   : Data access class for table Country
Syntax    :
Description :
Author(s)  : alessandro.saulea
Created   : Thu Jul 11 13:16:45 EEST 2024
Notes     :
-----*/

USING Progress.Lang.*.

BLOCK-LEVEL ON ERROR UNDO, THROW.

CLASS Test:

END CLASS.
```

# Context – Description on top of the method

```
CHAT
+ 🕒 ...
alessela
Generate the MyMethod() method from Test.cls based on its purpose

GitHub Copilot
> Used 1 reference

METHOD PUBLIC VOID MyMethod(INPUT countryName AS CHARACTER, INPUT countryC
DEFINE VARIABLE hQuery AS HANDLE NO-UNDO.
DEFINE VARIABLE cQuery AS CHARACTER NO-UNDO.
DEFINE VARIABLE bTransaction AS LOGICAL NO-UNDO.

/* Construct the query to insert a new country */
ASSIGN cQuery = "INSERT INTO Country (CountryName, CountryCode) VALUES

/* Start a transaction */
DO TRANSACTION:
  ASSIGN bTransaction = TRUE.
  CREATE QUERY hQuery.
  hQuery:SET-BUFFERS(BUFFER Country:HANDLE).
  hQuery:QUERY-PREPARE(cQuery).
  hQuery:QUERY-OPEN().
  hQuery:GET-FIRST().

/* Check for errors and commit if successful */
IF hQuery:NUM-RESULTS = 1 THEN DO:
  MESSAGE "Country created successfully." VIEW-AS ALERT-BOX INFO
END.
ELSE DO:
  MESSAGE "Error creating country." VIEW-AS ALERT-BOX ERROR BUTT
  UNDO TRANSACTION, LEAVE.
END.
END TRANSACTION.

/* Cleanup */
IF VALID-HANDLE(hQuery) THEN DO:
  hQuery:QUERY-CLOSE().
  DELETE OBJECT hQuery.
END.
```

```
Test.cls
Test.cls
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
/*-----*/
File      : Test
Purpose   : Data access class for table Country
Syntax    :
Description:
Author(s) : alessandro.saulea
Created   : Thu Jul 11 13:16:45 EEST 2024
Notes     :
/*-----*/

USING Progress.Lang.*.

BLOCK-LEVEL ON ERROR UNDO, THROW.

CLASS Test:

/*-----*/
Purpose: Create Country
Notes:
/*-----*/

METHOD PUBLIC VOID MyMethod( ):

  RETURN.

END METHOD.

END CLASS.
```

# Context – Previous prompts in the session

CHAT

alessela

Based on BEEmployee.cls and SIDepartment.cls, generate SIEmployee.cls

GitHub Copilot

Used 1 reference

Given the structure of `SIDepartment.cls` as provided, we can create `SIEmployee.cls` by adapting the pattern to employee-related operations. This involves changing references from departments to employees and assuming the existence of a `BEEmployee` class with similar functionality to `BEDepartment`. Here's how `SIEmployee.cls` might be structured:

```
BLOCK-LEVEL ON ERROR UNDO, THROW.
ROUTINE-LEVEL ON ERROR UNDO, THROW.

USING Progress.Lang.*.
USING BE.BEEmployee.

CLASS SI.SIEmployee:

    {include/dsEmployee.i} // Assuming a similar include file

    DEFINE PRIVATE VARIABLE oBEEmployee AS BEEmployee NO-UNDO.

    CONSTRUCTOR PUBLIC SIEmployee():
        oBEEmployee = NEW BEEmployee().
    END CONSTRUCTOR.

    DESTRUCTOR PUBLIC SIEmployee():
        DELETE OBJECT oBEEmployee NO-ERROR.
    END DESTRUCTOR.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID FetchAllEmployees(OUTPUT DATASET dsEmp1
        oBEEmployee:FetchAllEmployees(OUTPUT DATASET dsEmployee
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID UpdateEmployee(INPUT-OUTPUT DATASET dsEmp1
        oBEEmployee:UpdateEmployee(INPUT-OUTPUT DATASET dsEmp1
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID DeleteEmployee(INPUT pcEmpCode AS CHARACT
        oBEEmployee:DeleteEmployee(pcEmpCode).
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

END CLASS.
```

alessela

Based on BEEmployee.cls and SIDepartment.cls, generate SIEmployee.cls with methods similar with the ones from SIDepartment.cls

GitHub Copilot

Used 1 reference

Based on the provided excerpt from `SIDepartment.cls` and assuming a similar structure and functionality in `BEEmployee.cls`, the `SIEmployee.cls` can be structured to include methods for fetching, updating, and deleting employees, similar to the department operations. Here's how `SIEmployee.cls` might look:

```
BLOCK-LEVEL ON ERROR UNDO, THROW.
ROUTINE-LEVEL ON ERROR UNDO, THROW.

USING Progress.Lang.*.
USING BE.BEEmployee.

CLASS SI.SIEmployee:

    DEFINE PRIVATE VARIABLE oBEEmployee AS BEEmployee NO-UNDO.

    CONSTRUCTOR PUBLIC SIEmployee():
        oBEEmployee = NEW BEEmployee().
    END CONSTRUCTOR.

    DESTRUCTOR PUBLIC SIEmployee():
        DELETE OBJECT oBEEmployee NO-ERROR.
    END DESTRUCTOR.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID FetchAllEmployees(OUTPUT DATASET dsEmp1
        oBEEmployee:FetchAllEmployees(OUTPUT DATASET dsEmployee
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID UpdateEmployee(INPUT-OUTPUT DATASET dsEmp1
        oBEEmployee:UpdateEmployee(INPUT-OUTPUT DATASET dsEmp1
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID DeleteEmployee(INPUT pcEmpCode AS CHARACT
        oBEEmployee:DeleteEmployee(pcEmpCode).
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

END CLASS.
```

What are some best practices for error handling in ABL?

Ask Copilot or type / for commands

CHAT

```
USING Progress.Lang.*.
USING BE.BEEmployee.

CLASS SI.SIEmployee:

    DEFINE PRIVATE VARIABLE oBEEmployee AS BEEmployee NO-UNDO.

    CONSTRUCTOR PUBLIC SIEmployee():
        oBEEmployee = NEW BEEmployee().
    END CONSTRUCTOR.

    DESTRUCTOR PUBLIC SIEmployee():
        DELETE OBJECT oBEEmployee NO-ERROR.
    END DESTRUCTOR.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID FetchAllEmployees(OUTPUT DATASET dsEmp1
        oBEEmployee:FetchAllEmployees(OUTPUT DATASET dsEmployee
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID UpdateEmployee(INPUT-OUTPUT DATASET dsEmp1
        oBEEmployee:UpdateEmployee(INPUT-OUTPUT DATASET dsEmp1
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

    @openapi.openedge.export(type="REST", useReturnValue="false")
    METHOD PUBLIC VOID DeleteEmployee(INPUT pcEmpCode AS CHARACT
        oBEEmployee:DeleteEmployee(pcEmpCode).
        CATCH err AS Progress.Lang.Error :
            MESSAGE err:GetMessage(1)
                VIEW-AS ALERT-BOX.
        END CATCH.
    END METHOD.

END CLASS.
```

What are some best practices for error handling in ABL?

Ask Copilot or type / for commands

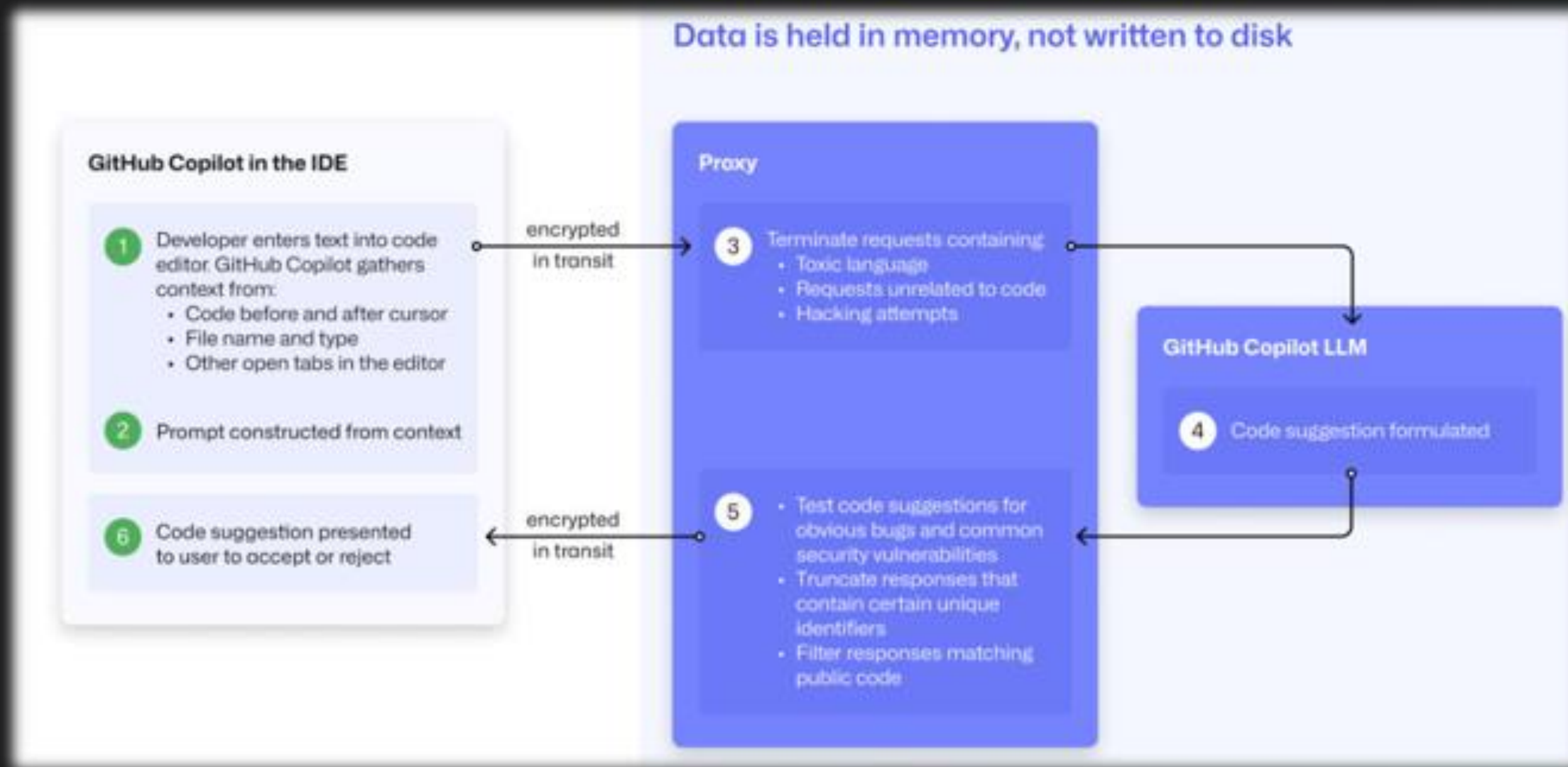
# Context

It is important to note that the context is made up of:

- Code before and after the cursor (after cursor: 10% boost in acceptance)
- File name and type
- Open tabs (increased acceptance rate by 5%)

# The way it handles data

The image below illustrates the process of generating a suggestion



# The way it works

It is based on gen Ai model developed by GitHub, OpenAI and Microsoft.  
It is trained on all programming languages publicity available,  
so the suggestion quality will depend on the volume of repositories in that language

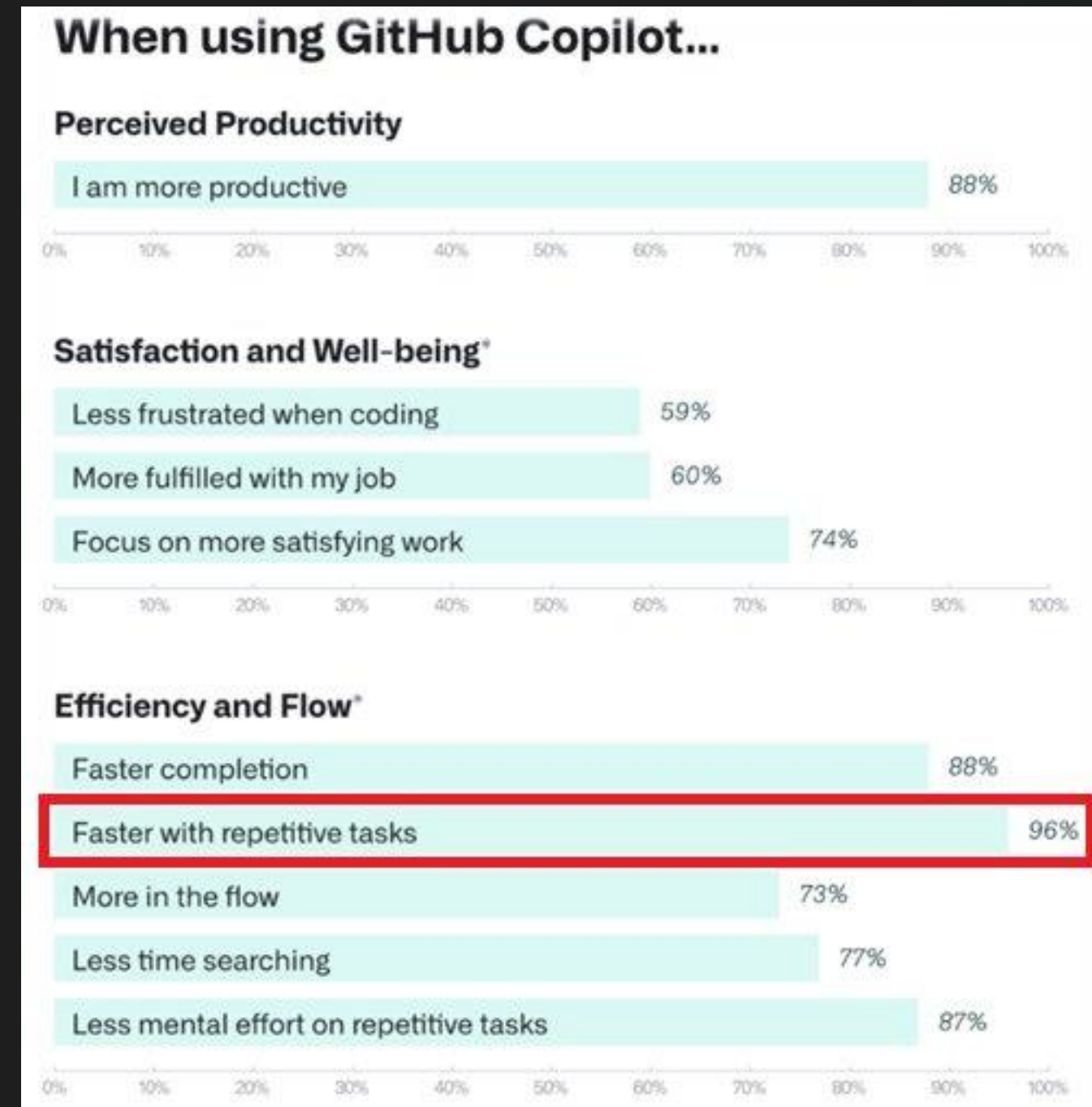


Microsoft

# Why use it?

According to one research, developers code up to 55 percent faster while using the tool

Source: <https://github.blog/2022-09-07-research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/>





# GitHub Copilot vs Copilot Chat

Copilot

**GPT 3.5** Turbo

**Code completion** feature

Inline **suggestions**

**Small** piece of code

Copilot Chat

**GPT 4**

Conversational **interaction**

**Large** pieces of codes

Detailed **explanations**

**Workspace** context

# GitHub Copilot features

**Additionally**, the chat includes shortcut commands that cover the most common actions developers might request from copilot

## GitHub Copilot

You can ask me general programming questions, or chat with the following participants which have specialized expertise and can perform actions:

- `@workspace` - Ask about your workspace
  - `/explain` - Explain how the code in your active editor works
  - `/tests` - Generate unit tests for the selected code
  - `/fix` - Propose a fix for the problems in the selected code
  - `/new` - Scaffold code for a new workspace
  - `/newNotebook` - Create a new Jupyter Notebook
- `@vscode` - Ask questions about VS Code
  - `/search` - Generate query parameters for workspace search
  - `/api` - Ask about VS Code extension development
- `@terminal` - Ask how to do something in the terminal
  - `/explain` - Explain something in the terminal

# GitHub Copilot features

**Copilot chat:** explaining selected code in a detailed manner

The screenshot displays the GitHub Copilot interface. On the left, the chat window shows a user prompt: "Can you explain the purpose of the DelimitGUIDString method in the provided code?". The Copilot response explains the method's purpose, notes, and implementation details in ABL code.

**Chat Window:**

GitHub Copilot: used @workspace /explain  
Used 1 reference >

The provided code is a method in OpenEdge ABL (Advanced Business Language) named `DelimitGUIDString`. This method is designed to take a string of GUIDs (Globally Unique Identifiers) as input and return a new string where the GUIDs are delimited by commas.

The method accepts one parameter, `pcGUIDString`, which is the input string of GUIDs. It defines several local variables: `iCount` for iterating through the input string, `cCharacter` to hold each character of the input string, `iNoOfAppendedCharacters` to keep track of the number of characters appended to the output string, and `cCommaDelimitedString` to hold the final output string.

The method then enters a loop that iterates over each character in the input string. For each character, it checks if the ASCII value of the character is within the ranges for uppercase letters (65-90), lowercase letters (97-122), digits (48-57), or if it's a hyphen (45). These ranges correspond to the characters typically found in a GUID.

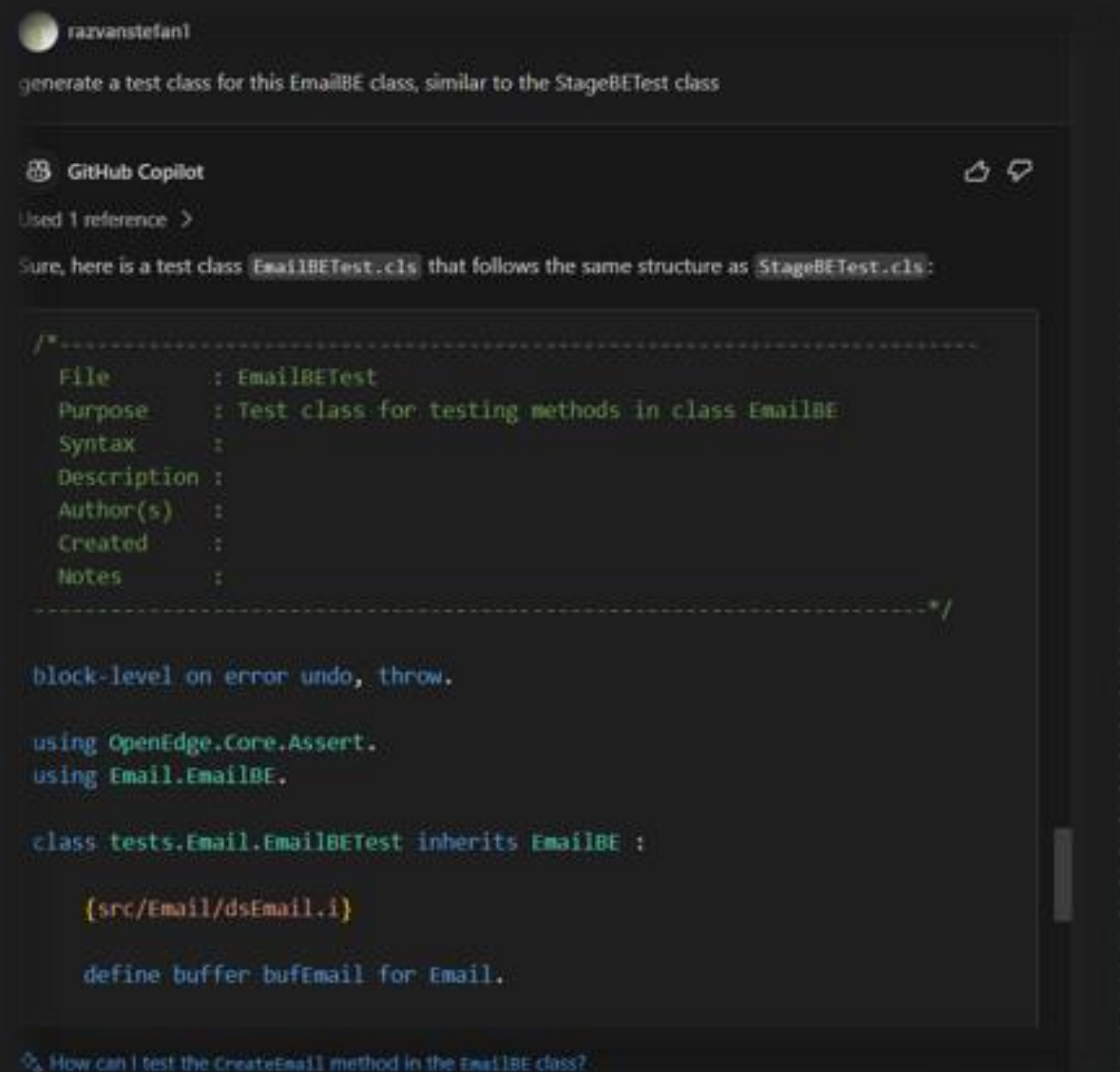
If the character meets these conditions, it is appended to the output string. The `iNoOfAppendedCharacters` counter is incremented each time a character is appended. When this counter reaches 36 (the typical length of a GUID), a comma is

**Code Editor:**

```
main > src > Utils > StringUtils.cls
9 class Utils.StringUtils:
10
11 /*-----
12 Purpose: given a string of GUIDs, this method replaces any delimiter(s) it might have with a comma and returns it
13 Notes: the input delimiter can also have multiple characters (any amount).
14 The method iterates the string, appends to the output string every character that is a letter, digit or "-"
15 After iterating through 36 characters, it also places a comma to delimit the GUIDs, so the returned format
16 @param: pcGUIDString, the input string of GUIDs
17 -----*/
18 method public static character DelimitGUIDString(pcGUIDString as character):
19     define variable iCount as integer no-undo.
20     define variable cCharacter as character no-undo.
21     define variable iNoOfAppendedCharacters as integer no-undo initial 0.
22     define variable cCommaDelimitedString as character no-undo.
23
24     do iCount = 1 to length(pcGUIDString):
25         cCharacter = substring(pcGUIDString, iCount, 1).
26         if (asc(cCharacter) >= 65 and asc(cCharacter) <= 90)
27             or (asc(cCharacter) >= 97 and asc(cCharacter) <= 122)
28             or (asc(cCharacter) >= 48 and asc(cCharacter) <= 57)
29             or (asc(cCharacter) = 45)
30         then
31             do:
32                 iNoOfAppendedCharacters += 1.
33             if iNoOfAppendedCharacters = 36 then
34                 assign
35                 cCommaDelimitedString += cCharacter.
36                 cCommaDelimitedString += ",".
37                 iNoOfAppendedCharacters = 0.
38             else
39                 cCommaDelimitedString += cCharacter.
40     end.
```

# GitHub Copilot features


**Copilot chat:** Similar to any other chat bot, but present inside the IDE ready to answer only coding related questions, have specific context



The screenshot shows a chat window with a user profile 'razvanstefan1' and a message: 'generate a test class for this EmailBE class, similar to the StageBETest class'. The GitHub Copilot response includes a reference to 'StageBETest.cls' and a code block for 'EmailBETest.cls'. The code block contains a Javadoc comment, a package declaration, and a class definition that inherits from 'EmailBE' and includes a 'CreateEmail' method.

```
/*-----  
File      : EmailBETest  
Purpose   : Test class for testing methods in class EmailBE  
Syntax    :  
Description :  
Author(s) :  
Created   :  
Notes     :  
-----*/  
  
block-level on error undo, throw.  
  
using OpenEdge.Core.Assert.  
using Email.EmailBE.  
  
class tests.Email.EmailBETest inherits EmailBE :  
  
    {src/Email/dsEmail.i}  
  
    define buffer bufEmail for Email.
```

# GitHub Copilot features

**Inline commands:** After pressing  icon, a small chat box opens up, ready to take in commands related to the selected code

```
optimize this code
You can also type / for commands
METHOD PUBLIC VOID UpdateCar(INPUT DATASET dsCar):
    DEFINE VARIABLE cErrorMessage AS CHARACTER INITIAL "" NO-UNDO .
    FOR EACH ttCar NO-LOCK:
        IF ttCar.CarTypeId = ? OR ttCar.CarTypeId = "" THEN
            cErrorMessage = cErrorMessage + "Type cannot be empty.~n".
        END.
    FOR EACH ttCar NO-LOCK:
        IF ttCar.Model = ? OR ttCar.Model = "" THEN
            cErrorMessage = cErrorMessage + "Model cannot be empty.~n".
        END.
    FOR EACH ttCar NO-LOCK:
        IF ttCar.Manufacturer = ? OR ttCar.Manufacturer = "" THEN
            cErrorMessage = cErrorMessage + "Manufacturer cannot be empty.~n".
        END.
    IF cErrorMessage <> "" THEN
        UNDO, THROW NEW Progress.Lang.AppError(cErrorMessage, 100).
    daCar:UpdateCar(INPUT DATASET dsCar BY-REFERENCE).
END METHOD.
```

```
METHOD PUBLIC VOID UpdateCar(INPUT DATASET dsCar):
    DEFINE VARIABLE cErrorMessage AS CHARACTER INITIAL "" NO-UNDO .
    FOR EACH ttCar NO-LOCK:
        IF ttCar.CarTypeId = ? OR ttCar.CarTypeId = "" THEN
            cErrorMessage = cErrorMessage + "Type cannot be empty.~n".
        IF ttCar.Model = ? OR ttCar.Model = "" THEN
            cErrorMessage = cErrorMessage + "Model cannot be empty.~n".
        IF ttCar.Manufacturer = ? OR ttCar.Manufacturer = "" THEN
            cErrorMessage = cErrorMessage + "Manufacturer cannot be empty.~n".
        END.
    IF cErrorMessage <> "" THEN
        UNDO, THROW NEW Progress.Lang.AppError(cErrorMessage, 100).
    daCar:UpdateCar(INPUT DATASET dsCar BY-REFERENCE).
END METHOD.
```

# GitHub Copilot features

**Comment to code suggestions:** after typing a comment that describes what the following lines of code should do, the same gray suggestions appear

```
-----  
Purpose: update a FileAttachment  
Notes:  
@param: dsFileAttachment as dataset, the dataset which contains the file attachments which will be updated  
-----*/  
method public void UpdateFileAttachment(input-output dataset dsFileAttachment):  
  
    for each ttFileAttachment no-lock  
        where ttFileAttachment.FileAttachmentId <> "":  
  
        find first bufFileAttachment exclusive-lock  
        where bufFileAttachment.FileAttachmentId = ttFileAttachment.FileAttachmentId no-error.  
  
        if available bufFileAttachment then  
            buffer-copy ttFileAttachment except FileAttachmentId CreatedTimestamp LastUpdatedTimestamp to bufFileAttachment.  
            assign  
                bufFileAttachment.LastUpdatedTimestamp = now  
                ttFileAttachment.LastUpdatedTimestamp = bufFileAttachment.LastUpdatedTimestamp.  
        else  
            undo, throw new Progress.Lang.AppError(substitute("Cannot find the FileAttachment with ID &1", ttFileAttachment.FileAttachmentId), 100).  
        end.  
  
    end method.
```

# GitHub Copilot features

**Code autocomplete suggestions:** as the developer writes the code, gray suggestions will appear with the possibility to press Tab to accept the code

```
-----  
Purpose: create a FileAttachment  
Notes:  
@param: dsFileAttachment as dataset, the dataset which contains the file attachments to be created  
-----*/  
method public void CreateFileAttachment(input-output dataset dsFileAttachment):  
  
    for each ttFileAttachment no-lock  
        where ttFileAttachment.FileAttachmentId = "":  
            create buffFileAttachment.  
            |buffer-copy ttFileAttachment except FileAttachmentId CreatedTimestamp LastUpdatedTimestamp to buffFileAttachment.  
  
method public void ReadData(input dataset dsFilter,output dataset-handle hDataset):  
    define variable cQuery      as character no-undo.  
    define variable lQueryStatus as logical  no-undo.  
    define variable hBuff       as handle   no-undo.  
    define variable hQuery      as handle   no-undo.  
  
    oQueryBuilder = new Utils.QueryBuilder().  
  
    this-object:|oQueryBuilder:BuildSelectStatement (this-object:TableName, dsFilter, cQuery).
```

# What is GitHub Copilot ?

GitHub Copilot is an AI coding assistant that offers:

- Code autocomplete suggestions
- Copilot Chat
- Comment to code suggestions





# ABOUT WAYFARE

We've supported businesses of different sizes and across multiple industries to achieve their goals.

## TOP TALENT, SHARED KNOWLEDGE, CLOSE COLLABORATION

Our cost-effective solutions and active community engagement foster innovation and efficiency.

We offer tailored services including **Modernization, New Development, Maintenance & Support, Quality Assurance, UI/UX Design, Consultancy and Project Delivery.**

**12+**  
**YEARS EXPERIENCE**

**10+**  
**YEARS**  
**LONGEST PARTNERSHIP**

**50+**  
**SATISFIED CUSTOMERS**

**100+**  
**PROFESSIONALS**

## OUR EXPERTISE

### PROGRESS

OpenEdge, CorticonJS, Sitefinity

**MOBILE** React Native

**DEVOPS** Docker, Kubernetes, Jenkins

**MANUAL AND AUTOMATION TESTING**

### WEB TECHNOLOGIES

Angular, React, Vue.JS, NodeJS, JS, TypeScript, Java, .Net

**CLOUD** AWS, Azure, Google Cloud Platform

**UI/UX** Figma, XD, Zeplin, Illustrator

**LOW CODE/ NO-CODE FRAMEWORKS**





**Q&A**



**THANK YOU**

wayfare.ro