What is Copilot

**GitHub Copilot** is an AI (Artificial Intelligence) pair programmer that offers autocomplete-style suggestions as you code.

It is a cloud-based artificial intelligence tool developed by GitHub, OpenAI and Microsoft to assist users by autocompleting code. Trained on billions of lines of code, Copilot turns natural language prompts into coding suggestions across dozens of languages. It has been trained on natural language text and source code from publicly available sources, including code in public repositories on GitHub.

How it works

GitHub Copilot shares recommendations based on the project’s context and style conventions. It can quickly cycle though lines of code, complete function suggestions and decide which to accept, reject or edit. Spend less time creating boilerplate and repetitive code patterns, and more time on business logic. Write a comment describing the logic you want, and GitHub Copilot will immediately suggest code to implement the solution. GitHub Copilot shares recommendations based on the project's context and style conventions. Quickly cycle through lines of code, complete function suggestions, and decide which to accept, reject, or edit.

How to Use

You can receive suggestions from GitHub Copilot either by starting to write the code you want to use, or by writing a natural language comment describing what you want the code to do. GitHub Copilot integrates directly into your editor. So, no need to open any window or popup to get the code suggestions.

Supported IDE’s

GitHub Copilot is available as an extension in Visual Studio Code, Visual Studio, Vim, Neovim and the JetBrains suite of IDEs. It currently supports following JetBrains IDEs.

• IntelliJ IDEA (Ultimate, Community, Educational)

• Android Studio

• AppCode

• CLion

• Code With Me Guest

• DataGrip

• DataSpell

• GoLand

• JetBrains Client

• MPS

• PhpStorm

• PyCharm (Professional, Community, Educational)

• Rider

• RubyMine

• WebStorm

Supported Programming languages

• C

• C++

• C#

• Go

• Java

• JavaScript

• PHP

• Python

• Ruby

• Scala

• Swift

• TypeScript

• Kotlin

• HTML

• XML

Signup For Copilot

https://github.com/features/copilot

Getting started with GitHub Copilot

https://docs.github.com/en/copilot/getting-started-with-github-copilot?tool=vscode

GitHub Copilot Chat

https://docs.github.com/en/copilot/responsible-use-of-github-copilot-features/responsible-use-of-github-copilot-chat-in-your-ide

GitHub Copilot Privacy policy

https://docs.github.com/en/site-policy/privacy-policies/github-copilot-for-business-privacy-statement

GitHub Copilot Product Specific Terms

https://github.com/customer-terms/github-copilot-product-specific-terms

Github Copilot Chat

JetBrains and Visual Studio code IDEs support additional extension for Copilot which is Copilot chat. It helps developers in

• Analysing and explaining the code

• Generate method level, class/file level and project level documentation

• Optimising the code

• Creating unit test cases

Security when using Copilot

GitHub Copilot offers suggestions from a model that OpenAI built from billions of lines of open-source code. As a result, the training set for GitHub Copilot may contain insecure coding patterns, bugs, or references to outdated APIs or idioms. When GitHub Copilot produces suggestions based on this training data, those suggestions may also contain undesirable patterns.

You are responsible for ensuring the security and quality of your code. We recommend you take the same precautions when using code generated by GitHub Copilot that you would when using any code, you did not write yourself. These precautions include rigorous testing, IP scanning, and tracking for security vulnerabilities. GitHub provides several features to help you monitor and improve code quality, such as GitHub Actions, Dependabot, CodeQL and code scanning. All these features are free to use in public repositories.

Pros of using Copilot

• This tool will dramatically increase the programming speed and hence the productivity.

• This will facilitate problem solving in very less time.

• It can convert code from one programming language to another.

• Spend less time creating boilerplate and repetitive code patterns, and more time on what matters

• Even if you are working in a new language or framework, or just learning to code, GitHub Copilot can help you find your way. Tackle a bug or learn how to use a new framework without spending most of the time going through the docs or searching the web.

Cons of using Copilot

• It is not highly recommended for beginner and intermediate programmers. Because over time they will feel that their programming skills have weakened.

• It does not work in offline mode. Requires good internet speed to get the suggestions in very quick time.

• Developers should know how to phrase the requirements properly in natural language. Otherwise, Copilot may return some irrelevant code suggestions.

• GitHub Copilot will likely work less well in scenarios where natural language prompts provided by the developer are not in English and/or are grammatically incorrect.

• Github Copilot does not test the code it suggests so the code may not always work, or even make sense. Also, it may suggest old or deprecated uses of libraries and languages.

• GitHub Copilot is trained on public code. When a new library, framework, or API is released, there is less public code available for the model to learn from. That reduces GitHub Copilot’s ability to provide suggestions for the new codebase.

**Assignment 1 (server side)**

A news platform to publish articles

Minimum requirement: API to fetch these detail

• Provide list of categories

• Provide the list of stories for 0 or more category, for a page number, for a article type, for an author or a tag name with sort order as latest published date. Refer to homepage.json. categoryId, authorId and tags are added so that client side can filter without the need for actual API. These 3 keys are not mandatory.

• For a given article id, fetch the article details

• For an author fetch author details

You can do a CMS interface to manage Category, Article, Author

Requirement detail: Article is composed of these components

**Property**

**Description**

Title

String

Sub title - no

String

Article image

Image in 16:9 ratio

articleType - text , audio.video

Text, audio, video

Description- no

List of HTML snippet

Hyperlink to include Youtube video or X embed

mediaUrl - no

Video or audio URL

Mandatory category

Reference to a category

tags

Reference to multiple tags

Author

Reference to author

Publish date

Date of publish

CMS UI for adding article

Title

hero image URL

URL

Author name

Sub title

If articleType is text, description is displayed within CK Editor. For articleType audio/video mediaURL will be set.

Tag1

Tag2

Article type

Category

Title

hero image URL

URL

Author name

Sub title

If articleType is text, description is displayed within CK Editor. For articleType audio/video mediaURL will be set.

Tag1

Tag2

Article type

Category

Category contains category name that can be added or edited inline in category page.

Tag contains Tag name

Author contains author name, image and description.

Author list contains the list of authors and each can be edited.

**Assignment 2 (Client side)**

Screen to display the list of articles. These can be filtered specific to a category, article type, author or tag name. Provide pull to refresh feature.

Author

Category

Article type

Hero image

Title

Subtitle

Hero image

Title

Subtitle

Hero image

Title

Subtitle

Tag

Author

Category

Article type

Hero image

Title

Subtitle

Hero image

Title

Subtitle

Hero image

Title

Subtitle

Tag

Title

hero image

Author name – hovering, author image and description is displayed on floating view.

Sub title

If articleType is text, description is displayed in webview (on mobile)

If article type is video, native media player will play from the media URL

Tag1

Tag2

Title

hero image

Author name – hovering, author image and description is displayed on floating view.

Sub title

If articleType is text, description is displayed in webview (on mobile)

If article type is video, native media player will play from the media URL

Tag1

Tag2

On clicking the tag, article list with that tag name will be displayed.