GitHub

GitHub Copilot - Part 2

By RaKeTe-Technology

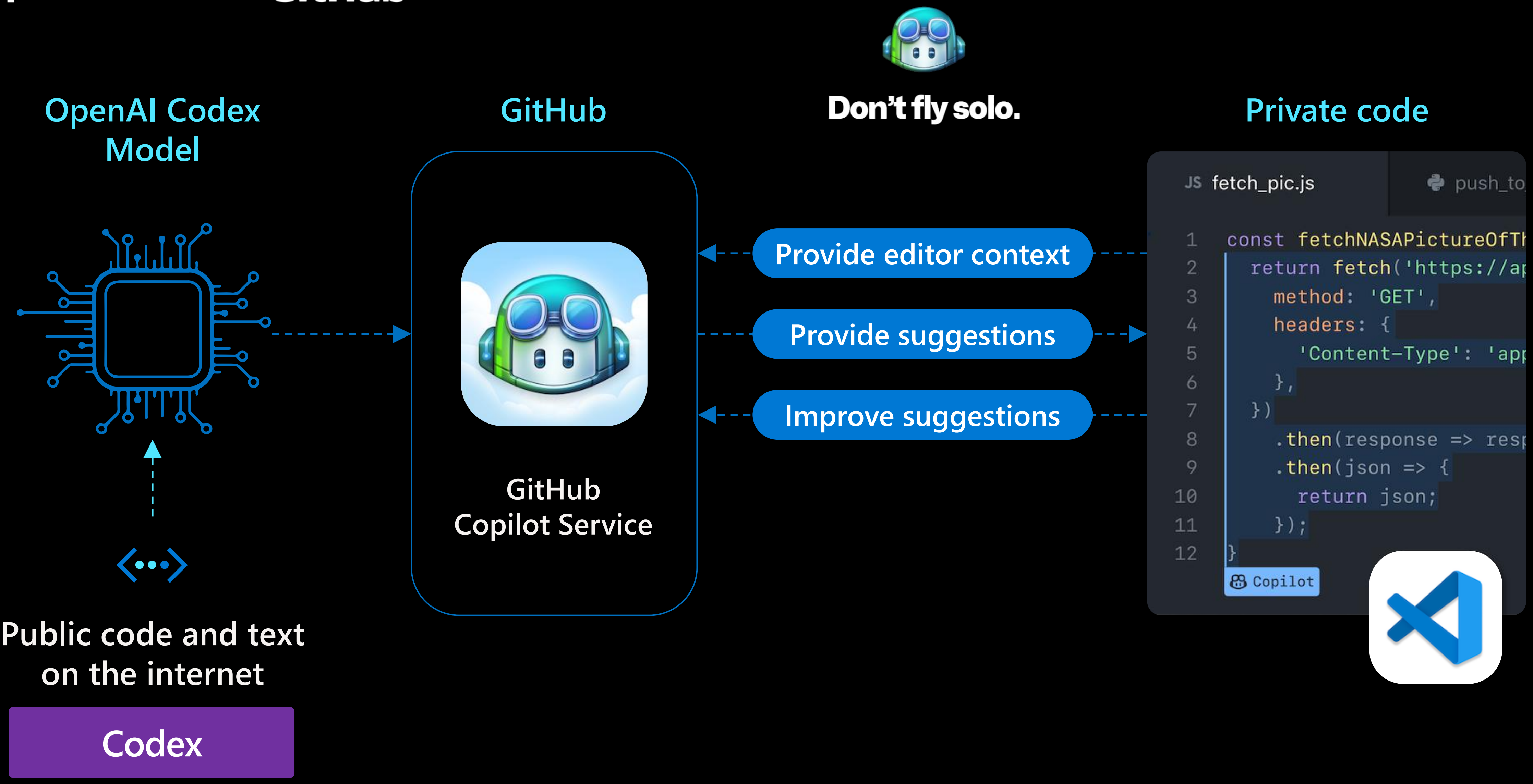
Ralph.Kemperdick@RaKeTeTechnology.com
Cloud Solution Architect, Advanced Analytics AI + Data & Analytics
LinkedIn www.linkedin.com/in/ralphke

Agenda

1. Quick Recap
2. Technology Stack
3. Considerations & Limitations
4. Responsibilities
5. Policy
6. Privacy, Data Protection & Security
7. Pricing
8. Next Steps
9. Q&A

Quick Recap

OpenAI + GitHub



```
const fetchNASAPictureOfTh
  return fetch('https://ap
    method: 'GET',
    headers: {
      'Content-Type': 'app
    },
  })
    .then(response => resp
    .then(json => {
      return json;
    });
}
```

Best practices

Single, Specific, Short

- Single responsibility
- Specific prompt
- Short response

Patterns

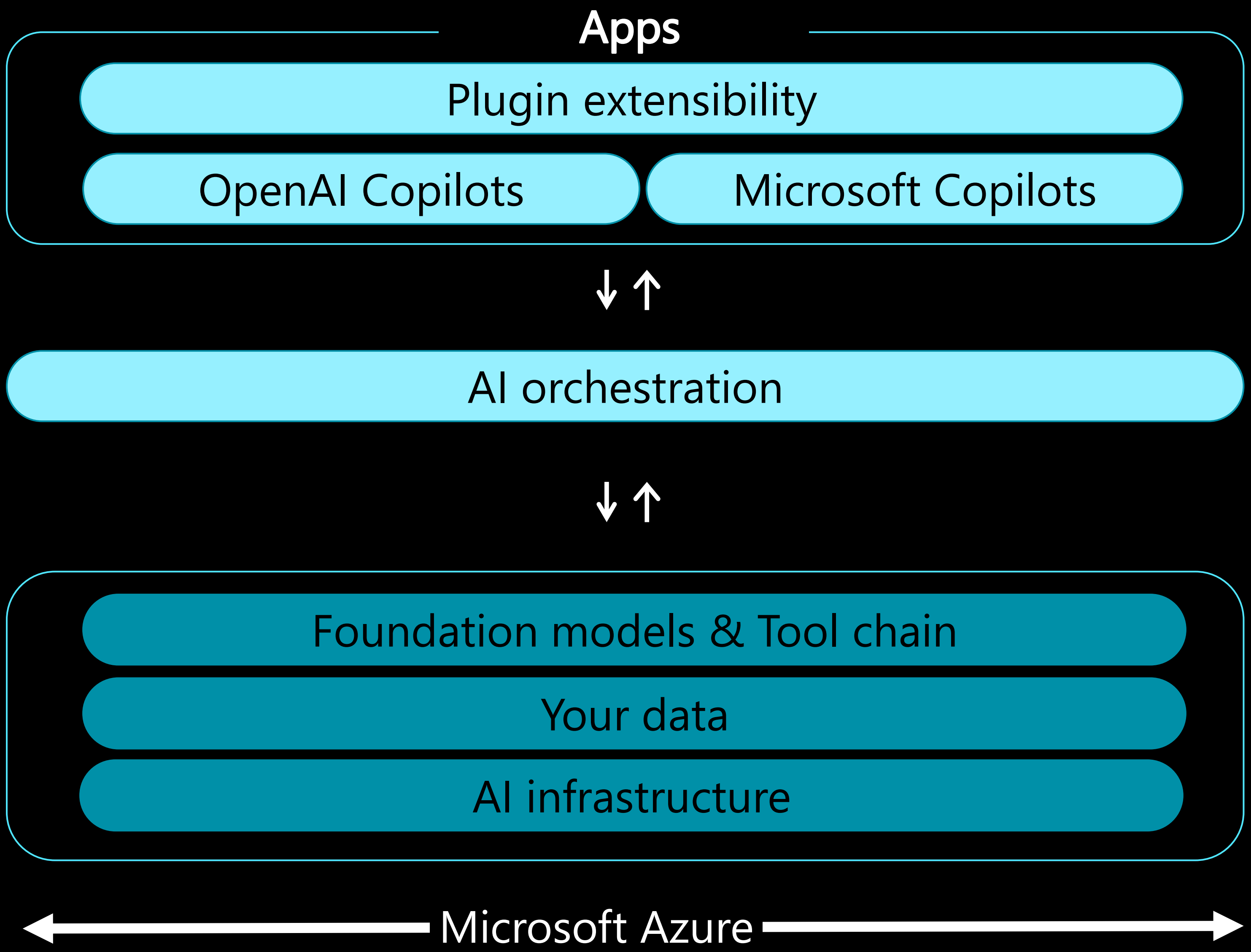
- Regex, CRON, PowerShell

Trust but verify

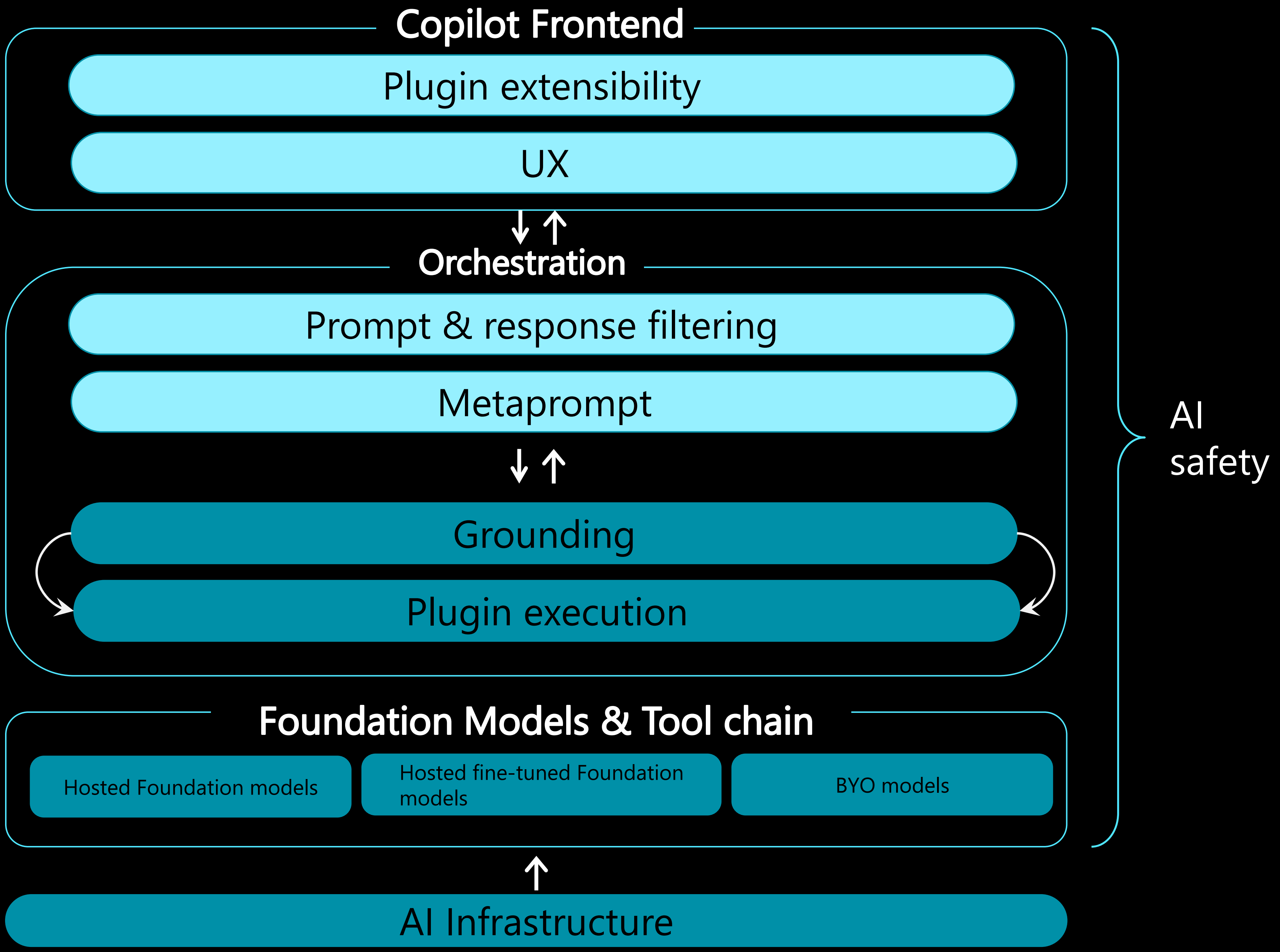
Context is important

Iterate, iterate, iterate

Copilot stack



Copilot stack



Copilot Frontend

Plugin extensibility

UX

Orchestration

Prompt & response filtering

Metaprompt

Grounding

Plugin execution

Foundation Models & Tool chain

Hosted Foundation models

Hosted fine-tuned Foundation models

BYO models

AI Infrastructure

AI safety

Limitations

Quality of Generated Code

While Copilot is impressive, it doesn’t always produce code that adheres to best practices. For instance, in JavaScript, it might generate var and == instead of const and ===, which can lead to subtle bugs and shadowing[1].

Biased Language and Libraries

Copilot tends to favor certain libraries or frameworks. For example, in Python, it heavily leans toward using SQLAlchemy, which may not align with your preferred stack[2].

Single Context

You can only provide one piece of context when asking Copilot a question. This limitation can impact its ability to generate accurate code when dealing with complex scenarios[2].

Responsibilities

- Developers are taking full **responsibility** for implementing suggestions.
- The proposed code must meet the same strict **quality requirements** as **self-written** code.
- **Understanding** the code is a fundamental prerequisite to ensuring the highest possible quality and safety

Policy Framework - I

- Purpose
  - Ensure importance of maintaining data security and code integrity when using AI tools
- Scope
  - All developers using generative AI tools, should be aware of their responsibilities and the boundaries of the policy.
- Responsible use
  - Developers are accountable for the code generated with AI and must adhere to existing coding standards and practices.
- Intellectual property (IP)
  - Compliance with intellectual property laws and internal policies

Policy Framework - II

- Output validation
  - Developers must ensure code adheres to the company’s standards, doesn't introduce security risks, and aligns with project requirements, just like manually written code.
- Monitoring
  - Developers should track the quality of the code produced and impact on productivity by addressing issues or limitations.
- Documentation
  - Should cover the user, purpose, and manner of usage, to aid in tracking effectiveness and ensuring compliance with the policy.
- Training
  - Understanding the tool's capabilities, limitations, and the principles of in the policy, ensuring competent and responsible usage.

Policy Framework - III

- Policy review
  - Regular reviews and updates to the policy are necessary to adapt to evolving technologies and coding practices.
  - It's also important to communicate this to the employees.
- Governance and policy set the groundwork for fostering adoption of AI tooling!

Source: Empower developers with AI policy and governance - GitHub Resources

Handle Connections Gracefully

Here: Python + SQL Server

- **Close Connections**: Only establish connections when necessary, and close them promptly after use. Leverage connection pooling for efficiency.
- **Exception Handling**: Implement proper exception handling to gracefully handle connection errors.
- **Example (Python)**:
-
  ```
  import pyodbc
  def execute_sql(query):
      try:
          conn = pyodbc.connect("your_connection_string_here")
          cursor = conn.cursor()
          cursor.execute(query)
          result = cursor.fetchall()
          return result
      except pyodbc.Error as e:
          print(f"Error executing query: {e}")
      finally:
          conn.close()
  # Example usage
  query = "SELECT * FROM Sales.Orders WHERE OrderDate >= ‘2024-04-03’”
  results = execute_sql(query)
  ```

Privacy, Data Protection, Security and IP Protection

A. GitHub Copilot sends an encrypted prompt from Customer to GitHub to provide suggestions. Except as detailed below, Prompts are transmitted only to generate these suggestions in real-time, are deleted once the suggestions are generated, and are not used for any other purpose. Prompts are encrypted during transit and are not stored at rest without your permission.

B. Your Prompts are retained by GitHub in the following circumstances:

   A. CLI and Other Tools. If you use GitHub Copilot tools that operate outside of your code editor, such as Copilot for the Command Line Interface, GitHub Copilot retains your Prompts to those tools to provide the service.

   B. Private Language Models. If you have requested a customized private language model, GitHub Copilot retains your Prompts to fine-tune your private model.

   C. Customization. If you have customized GitHub Copilot to use alternative data handling, such as enabling interaction with third party extensions, GitHub Copilot will retain your Prompts based on that customization.

C. More Information. More detailed information on how data is processed by GitHub Copilot is in the GitHub Privacy Statement available at https://gh.io/privacy

Privacy: Data being collected

- Prompts
  - Bundle of contextual information Copilot sends when working on a file, Copilot pane is opened.
  - Prompts are retained unless you have disabled code snippet collection in settings.

- Suggestions
  - Lines of proposed text returned to Copilot after a Prompt is processed.
  - Suggestions are retained unless code snippet collection is disable in settings.

Source: About GitHub Copilot Individual - GitHub Enterprise Cloud Docs

IP Protection

- **Defense of Third-Party Claims**

Notwithstanding any other provision in Customer's volume licensing agreement, Microsoft’s obligation to defend Customer's use of GitHub Copilot under the Copilot Copyright Commitment will not apply if Customer has not set the **Duplicate Detection** filtering feature available in GitHub Copilot to its “**Block**” setting. You can learn how to enable the Duplicate Detection filter at https://gh.io/cfb-dd

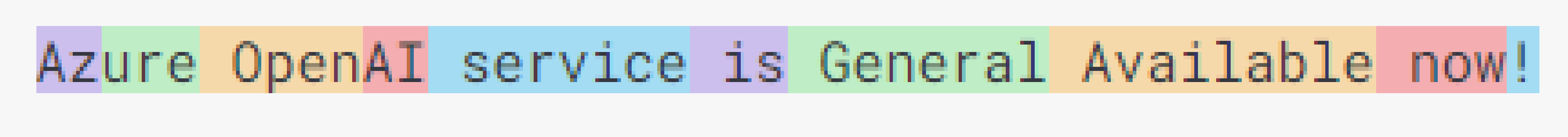
What is a token?

Sentence:

- Azure OpenAI service is General Available now!

Tokens:

- [AZ]-[ure] [Open]-[AI] [service] [is] [General] [Available] [now][!]



Azure OpenAI service is General Available now!

1 token is approximately 4 characters or 0.75 words.

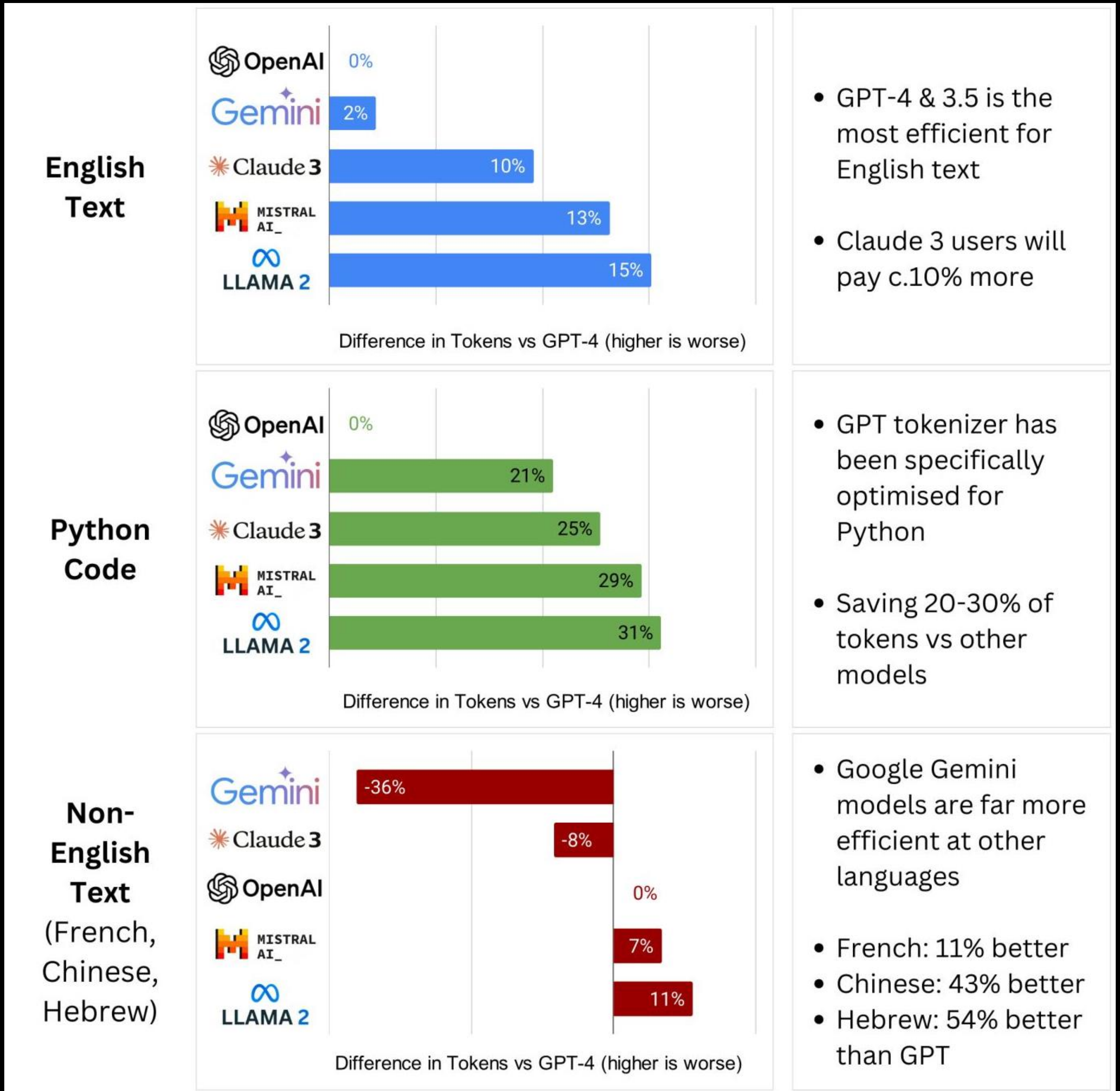
Do you know:
The collected works of Shakespeare are about 900,000 words or 1.2M tokens.
How many tokens are generated by each Tokenizer?
The Tokenizer Playground - a Hugging Face Space by Xenova

Pricing

Comparison of price per token across common LLM-Models

Tokens cost money and their efficiency varies widely across models

OpenAI is leading on English and Python, but Gemini in non-English content



- GPT-4 & 3.5 is the most efficient for English text
- Claude 3 users will pay c.10% more

- GPT tokenizer has been specifically optimised for Python
- Saving 20-30% of tokens vs other models

- Google Gemini models are far more efficient at other languages
- French: 11% better
- Chinese: 43% better
- Hebrew: 54% better than GPT

Source: https://www.linkedin.com/in/peter-gostev/

Pricing

- **GitHub Offerings at Microsoft Product Terms and GitHub's plans - GitHub Docs**
- **For GitHub Codepilot for Business an Agreement must be in place:**

- **Enterprise Agreement** Enterprise Agreement | Microsoft Volume Licensing
- **Microsoft Customer Agreement** Licensing Documents (microsoft.com)

- GitHub Copilot Business, $19 USD per user per month.
- GitHub Copilot Enterprise, $39 USD per user per month.
- A connected Azure subscription by your organization or enterprise account will enable metered billing via Azure
- About billing for GitHub Copilot - GitHub Docs
- Upgrading from Business to Enterprise will upgrade all seats!
- GitHub Advanced Security licenses About billing for GitHub Advanced Security - GitHub Docs

Functionality by Copilot Service

	Copilot Individual	Copilot Business	Copilot Enterprise
Pricing	$10 USD per month $100 USD per year	$19 USD per user per month	$39 USD per user per month
Types of GitHub accounts	Personal accounts	Organization or enterprise accounts	Enterprise accounts on GitHub Enterprise Cloud
Copilot Chat	✓	✓	✓
GitHub Copilot in the CLI	✓	✓	✓
Code snippet collection	✓ (Unless disabled)	🚫	🚫
Blocks suggestions matching public code	✓	✓	✓
Plugs right into your editor	✓	✓	✓
Offers multi-line function suggestions	✓	✓	✓
Organization-wide policy management	🚫	✓	✓
Exclude specified files	🚫	✓	✓
Audit logs	🚫	✓	✓
Copilot Chat in GitHub.com	🚫	🚫	✓
Copilot pull request summaries	🚫	🚫	✓

Next Steps

- Decide where to purchase your service
  - OpenAI or Azure OpenAI
- **Get started**
- Learning
  - Emphasizing continuous learning and upskilling in a fast-changing environment is key to able to push boundaries
- Discovering
  - Encourage experimenting to uncover what really works and strive for meaningful impact
- Deploying
  - Applying models to real-life and create value.
    Monitor and finetune, learn and repeat the process.

Resources

- Learn more about Visual Studio Copilot
- Getting started with GitHub Copilot - GitHub Enterprise Cloud Docs
- Essentials of GitHub Copilot - GitHub Resources
- Microsoft Learn AI Skills Challenge
- Welcome | Learn how to interact with OpenAI models (microsoft.github.io)

Thank you!