F-15 - GitHub Copilot Knowledge Bases (ENTERPRISE)

SUMMARY

In this lesson, we explored **GitHub Copilot Knowledge Bases** and their role within enterprise and business-level subscriptions. Knowledge bases allow us to share consistent information across an organization, ensuring GitHub Copilot users have a unified understanding of the codebase. Here's a detailed step-by-step on how to leverage this feature:

1. **Access Requirements**: Knowledge bases are exclusive to **GitHub Copilot Enterprise and Business** subscriptions. Without these, the feature won't be available.

2. **Creating a Knowledge Base**:

    - Start by setting up a new repository specifically for the knowledge base.
    - Add relevant markdown files (`.md` or `.mdx`) that contain essential organizational information, API details, etc.
    - Navigate to the **settings** on the organization's GitHub account. This requires admin access.
    - In settings, locate `Copilot` , then select `Knowledge Bases` .
    - Add repositories to form the knowledge base. Only `.md` files are indexed.

3. **Using the Knowledge Base**:

    - Once the knowledge base is set up and indexed, it becomes accessible for use within Copilot Chat and VS Code.
    - To retrieve knowledge, initiate a chat with Copilot and ensure the context is set to the specific knowledge base.
    - When querying information, Copilot will pull data from the defined knowledge base.

4. **Practical Application**:

    - The knowledge base not only responds to queries but can also assist in code creation using the documented APIs, ensuring consistency.

WHAT WE LEARNED

- **Knowledge bases** are for business and enterprise subscriptions.

- They index markdown files only.
- Admin access is required for setup.
- They provide consistent organizational information.
- They aid in querying and code generation.

HOW WE CAN APPLY IT

- **Large Organizations**: Ensure all employees access consistent API documentation.
- **Centralized Information**: Replace scattered documentation with centralized markdown files.
- **Development Teams**: Facilitate cross-platform development (web, mobile, etc.) with unified API guidelines.
- **Code Consistency**: Boost consistent code implementation across teams using indexed documentation.

TIPS AND TRICKS

- **Use Markdown Wisely**: Only `.md` and `.mdx` files are indexed, so ensure all critical information is in these formats.
- **Keep Information Updated**: As your repositories get updated, the knowledge base does too, but regularly check for completeness.
- **Multiple Repos**: You can integrate multiple repositories into a single knowledge base for broader coverage.
- **Contextual Queries**: Always set your query context to the relevant knowledge base to ensure accurate information retrieval.

EXAMPLES

Setting up a Knowledge Base:

```
# Basic Setup for a Knowledge Base in GitHub

1. Own or access a GitHub Enterprise or Business account.
2. Create a dedicated repository for the knowledge base.
3. Populate the repository with `.md` files with necessary documentation.
4. Navigate to `Settings` -> `Copilot` -> `Knowledge Bases`.
5. Create a new knowledge base and link your repositories.
```

Example Query:

- **Question**: What is the API endpoint to create a new pet?
- **Response**: After setting context, Copilot pulls from the knowledge base.

```
POST /api/pets
Required fields: name, type, age
```

Python Script Generation:

```python
import requests

url = "https://example.com/api/pets"
payload = {
    "name": "Fluffy",
    "type": "Dog",
    "age": 3
}

response = requests.post(url, json=payload)
print(response.status_code)
```

This setup ensures a systematic approach to using GitHub Copilot for enterprise-level code management and development.