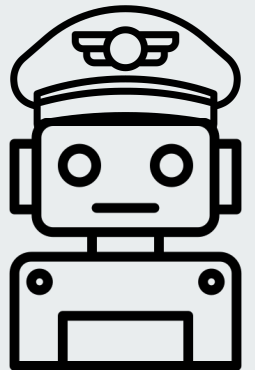




IDE-Integration of Github Copilot

Jetbrains IDEs





Introduction

! PLEASE DON'T LISTEN !

- Instead: use the time and start using / playing around with integrated AI capabilities for your DH coding.

Recommendations:

1. Enroll for the **student / teacher program** for **Github Copilot** OR apply for the **free trial**
2. Setup integration of **GitHub** Copilot plugins in **VSCo**de.
3. Test out with **Python**.



Useful starting points

- Zenodo community: <https://zenodo.org/communities/aiassistingdh>
- Get started workshop slides: <https://doi.org/10.5281/zenodo.10288101>



A small distinction

1. Standalone / “universal” AI tools (Chatbots)
 - a. ChatGPT
 - b. Claude
 - c. Perplexity AI
 - d. ...
2. Integrated AI tools (“AI copilots”)
 - a. Coding copilots (GitHub Copilot with plugin environment in various text editors, IDEs etc.)
 - b. AI text support (integration in Microsoft Word etc.)
 - c. ...



Structure

1. Showcase
 - 1.1. Capabilities of GitHub Copilot and the integration within JetBrains IDEs
2. News
 - 2.1. Context / history
 - 2.2. New developments
3. Integration of Github Copilot in IntelliJ Idea
 - 3.1. UI Integration - overview
 - 3.2. Autocomplete
 - 3.3. Code suggestions
 - 3.4. Integrated commands and prompts
 - 3.5. Integrated chat
4. Domain modeling?



Core question

A new style of (DH) programming?



GitHub Copilot in IDEs: JetBrains

- PyCharm : Python
- IntelliJ: Java
- Webstorm: HTML/CSS/JS
- ...



1. Showcase: integrated chat (I)

- “Solve the TODOs in **this file**” → context of the IDE-project.
 - Project as concept in IDEs.
 - Standardized structure of coding projects?


```
/**
 * Returns the full admin role
 * TODO test
 * @return the full admin role
 */
↳ Sebastian Schiller-Stoff
public static String getAdmin() {
    return GAMSAPISecurityRoles.ROLE_PREFIX.name + GAMSAPISecurityRoles.ADMINISTRATOR.name;
}

/**
 * Returns the full project admin role
 * TODO test
 * @param projectAbbr the project abbreviation
 * @return the full project admin role
 */
1 usage ↳ Sebastian Schiller-Stoff
public static String getProjectAdmin(String projectAbbr) {
    return GAMSAPISecurityRoles.ROLE_PREFIX.name + projectAbbr + GAMSAPISecurityRoles.ROLE_DELIMITER.name + GAMSAPISecurityRoles.PROJECT_ADMINISTRATOR.name;
}

/**
 * TODO test
 * Returns the full project editor role
 * @param projectAbbr the project abbreviation
 * @return the full project editor role
 */
1 usage ↳ Sebastian Schiller-Stoff
public static String getProjectEditor(String projectAbbr) {
```



GitHub Copilot

Hi @Stoff-Sebi, how can I help you?

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and [share feedback](#) so that we can learn and improve.



Stoff-Sebi

Please resolve the todos in the file



GitHub Copilot



The TODOs in the `GAMSAPISecurityRoles.java` file are related to testing the methods. To resolve these TODOs, you would need to write unit tests for each of the methods mentioned. Here's an example of how you could write these tests using JUnit and Mockito:

```
package org.zim.gamsapi.System.security;
```

The TODOs in the `GAMSAPISecurityRoles.java` file are related to testing the methods. To resolve these TODOs, you would need to write unit tests for each of the methods mentioned. Here's an example of how you could write these tests using JUnit and Mockito:

```
package org.zim.gamsapi.System.security;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.assertEquals;

public class GAMSAPISecurityRolesTest {

    @Test
    public void testGetAdmin() {
        String expectedRole = "ROLE_admin";
        assertEquals(expectedRole, GAMSAPISecurityRoles.getAdmin());
    }


    @Test
    public void testGetProjectAdmin() {
        String projectAbbr = "testProject";
        String expectedRole = "ROLE_testProject_admin";
        assertEquals(expectedRole, GAMSAPISecurityRoles.getProjectAdmin(projectAbbr));
    }
}
```


```
@Test
public void testGetProjectViewer() {
    String projectAbbr = "testProject";
    String expectedRole = "ROLE_testProject_viewer";
    assertEquals(expectedRole, GAMSAPISecurityRoles.getProjectViewer(projectAbbr));
}
}
```


This code creates a new test class `GAMSAPISecurityRolesTest` in the same package as `GAMSAPISecurityRoles`. It contains four test methods, one for each method in `GAMSAPISecurityRoles`. Each test method creates an expected role string, calls the corresponding method in `GAMSAPISecurityRoles`, and asserts that the returned role string matches the expected one.


Please note that you need to have JUnit and Mockito libraries in your project dependencies to run these tests.

Using 4 References

 IUserPrincipalAuditorMapping.java

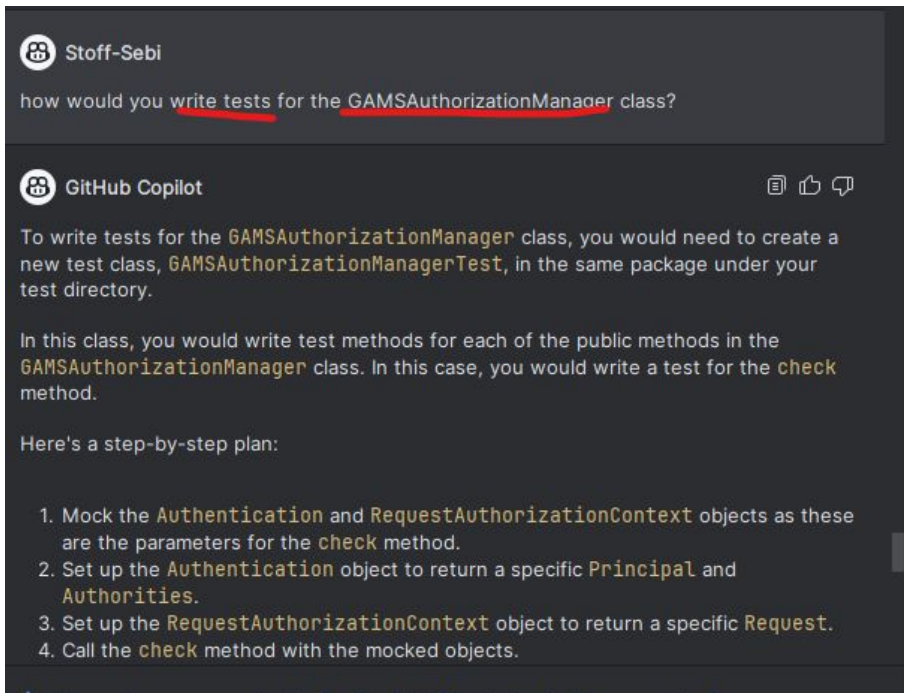
 docker-compose.yml

 application-dev.yml

 GAMSAPISecurityRoles.java

 [How can I add JUnit and Mockito libraries to my project dependencies?](#)

1. Showcase: Integrated chat (II)



The screenshot shows a chat interface with a dark background. At the top, a user named 'Stoff-Sebi' asks a question. Below, 'GitHub Copilot' provides a detailed answer with code snippets and a numbered list of steps.

Stoff-Sebi
how would you write tests for the GAMSAuthorizationManager class?

GitHub Copilot

To write tests for the `GAMSAuthorizationManager` class, you would need to create a new test class, `GAMSAuthorizationManagerTest`, in the same package under your test directory.

In this class, you would write test methods for each of the public methods in the `GAMSAuthorizationManager` class. In this case, you would write a test for the `check` method.

Here's a step-by-step plan:

1. Mock the `Authentication` and `RequestAuthorizationContext` objects as these are the parameters for the `check` method.
2. Set up the `Authentication` object to return a specific `Principal` and `Authorities`.
3. Set up the `RequestAuthorizationContext` object to return a specific `Request`.
4. Call the `check` method with the mocked objects.



1. Showcase: Autocomplete

```
// check if user is assigned to project
// TODO add nullpointer check?
// defined in request matcher in SpringSecurityConfiguration.java
String projectAbbr;
try {
    projectAbbr = authorizationContext.getVariables().get("projectAbbr");
} catch (NullPointerException e) {
    String msg = String.format("No project abbreviation found in request. Url: %s Method: %s", request.getRequestURL(), request.getMethod());
    log.trace(msg);
    throw new UserNotAssignedToProjectException(msg);
}
```




1.Showcase: Explicit context (I)

Hi, I'm new to the project and just cloned it via git. Can you explain to me the following things:

- What is this project about in general?
- What things do I need to know?
- How do I launch this?


 README.md ×  Makefile ×  pom.xml ×

 docker-compose.yml ×

 Stoff-Sebi

Hi, I'm new to the project and just cloned it via git. Can you explain to me the following things:

- What is this project about in general?
- What things do I need to know
- How do I launch this?

 GitHub Copilot



Sure, I'd be happy to help you understand the project.

1. What is this project about in general?

This project is a REST API for GAMS4+ (General Algebraic Modeling System). It's built using Spring Boot, a popular Java framework for creating stand-alone, production-grade Spring-based applications. The project uses several other technologies such as Maven for project management, SQL for data storage, and Docker for containerization. The project also includes OAuth2 for authentication and authorization.

2. What things do I need to know?

Here are some key things you should know about:

- **Java and Spring Boot:** The project is primarily written in Java using the Spring Boot framework. Familiarity with these is essential for understanding the project's codebase.
- **Maven:** This is a build automation tool used primarily for Java projects. The project uses Maven for dependency management and build automation.
- **Docker:** Docker is used to create, deploy, and run applications by using containers. The project uses Docker to containerize the application and its dependencies.
- **PostgreSQL:** This is the database used in the project. Knowledge of SQL is necessary to interact with the database.
- **OAuth2:** This is an authorization framework that enables applications to obtain limited access to user accounts on an HTTP service. The project uses OAuth2 for authentication and authorization.
- **Git:** The project uses Git for version control. You should be

3. How do I launch this?

The project uses Docker for easy setup and deployment. Here are the steps to launch the project:

- First, navigate to the project root directory using the command line.
- Run the command `docker-compose up`. This will start all the services defined in the `docker-compose.yml` file.
- Once the Docker services are up and running, you can start the Spring Boot application. You can do this either from your IDE or from the command line. If you want to run it from the command line, use the following command:


```
<pre>./mvnw spring-boot:run -Dspring-boot.run.profiles=dev </pre>
```

This command will start the Spring Boot application with the 'dev' profile. The 'dev' profile is used for development and may have different configuration settings compared to the 'prod' (production) profile.

Please note that you need to have Docker, Java, and Maven installed on your machine to run these commands.



Showcase: Explicit context (II)

 Stoff-Sebi

and how is this app being put to production / deployed? What are the key things I need to know for this?



2. News - Context / History

1. **Development 2023:** From code whispering (autocomplete) → integrated chat (~ december 2023).
 - **Experimental** approach (Testing out useful UI)
 - **Microsoft environment:** Visual Studio Code as “trendsetter”, development of plugins and backend functionality.
2. **Development 2024:** Code copilots becoming more established in enterprise software
 - Integration in **Jetbrains IDEs** (Github Copilot Chat)
 - **Market competitors** getting established: Jetbrains AI Assistant , PIECES, Amazon Code Whisper, Tabnine, Codeium, ...
 - **Plugin environment** getting more divers.
 - Additional standalone software being developed aside from IDEs and text editors.



2. News / new developments

1. GitHub Copilot Enterprise
 - **Paid enhancement** of GitHub Copilot features
 - Adapted version of GitHub Copilot to own **enterprise's code**: Aimed at increasing productivity for developers.
 - Features: **Increased control over data protection** and integration to existing tools.
 - Aim: Enterprises should keep control over their code / their business secrets .
2. GitHub Workspaces (Technical Preview)
 - AI supported development environment
 - Aim: **AI suggestions** should be made available **directly in the project management platform** (WebUI of a GitHub repository). For example providing a suggestion on how to solve a specific issue.
 - Main difference to GitHub Copilot: **Focus on collaboration.**



3. Integration of GitHub Copilot in IntelliJ Idea (and Pycharm)

3.1 Overview

General UI features:

- code suggestions icon
- code chat icon
- tutorial icon
- result bar
- popup
- editor actions (commands) view(s)

Project

- DigitalObject
 - converter
 - exceptions
 - interfaces
 - DigitalObject
 - DigitalObjectBuilder
 - DigitalObjectCompactDTO
 - DigitalObjectController
 - DigitalObjectService
 - DigitalObjectRepository
 - Ingest
 - Integration
 - Project
 - exceptions
 - interfaces
 - IProjectRepository
 - IProjectService
 - Project
 - ProjectController
 - ProjectService
 - System
 - bootstrap
 - config
 - configproperties
 - security
 - exceptions
 - GAMSAPISecurityRoles
 - GAMSAuthorizationManager
 - IUserPrincipalAuditorMapping
 - SpringSecurityConfiguration
 - UserPrincipalAuditorMapping
 - utils
 - IProxyService

```

1 spring:
2   profiles:
3     # loads tests in dev profile.
4     active: dev
5   jpa:
6     hibernate:
7       # delete database on application startup during tests
8       ddl-auto: create
9       # format SQL statements
10      properties:
11        hibernate:
12          format_sql: true
13          # show_sql: true
14
15 # ensures that testcontainers reuses started containers
16 # (not sure if properties actually do something)
17 # https://www.testcontainers.org/test_framework_integration/manual_lif
18 testcontainers:
19   reuse:
20     enable: true

```

Document 1/1

GitHub Copilot

Accept solution

```

/manual_lifecycle_control/#reusing
-containers

```

Accept solution

```

# see: https://www.testcontainers
.org/features/reusing_containers.html

```

Accept solution

```

# https://www.testcontainers
.org/features/reusing_containers.html

```

Accept solution

```

.org/test_framework_integration
/manual_lifecycle_control/



```

Accept solution

```

.org/test_framework_integration
/manual_lifecycle_control/#reuse-containers

```

Project

- DigitalObject
 - converter
 - exceptions
 - interfaces
 - DigitalObject
 - DigitalObjectBuilder
 - DigitalObjectCompactDTO
 - DigitalObjectController
 - DigitalObjectService
 - IDigitalObjectRepository
- Ingest
- Integration
- Project
 - exceptions
 - interfaces
 - IProjectRepository
 - IProjectService
 - Project
 - ProjectController
 - ProjectService
- System
 - bootstrap
 - config
 - configproperties
 - security
 - exceptions
 - GAMSAPISecurityRoles
 - GAMSAuthorizationManager
 - IUserPrincipalAuditorMapping
 - SpringSecurityConfiguration
 - UserPrincipalAuditorMapping
 - utils
 - IProxyService

```
application.yml
1 spring:
2   profiles:
3     # loads tests in dev profile.
4     active: dev
5   jpa:
6     hibernate:
7       # delete database on application startup during tests
8       ddl-auto: create
9     # format SQL statements
10    properties:
11      hibernate:
12        format_sql: true
13      # show_sql: true
14
15 # ensures that testcontainers reuses started containers
16 # (not sure if properties actually do something)
17 #
18 testcontainers:
19   reuse:
20     enable: true
```

GitHub Copilot Chat

New Conversation May 14, 2024 08:58

GitHub Copilot

Hi @Stoff-Sebi, how can I help you?

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and share feedback so that we can learn and improve.

Ask Copilot a question or type '/' for commands

+

GA gams-api @sebi/oauth2 gams-api-dev

Welcome to GitHub Copilot

application.yml

```
1 spring:
2   profiles:
3     # loads tests in dev profile.
4     active: dev
5   jpa:
6     hibernate:
7       # delete database on application startup during tests
8       ddl-auto: create
9       # format SQL statements
10      properties:|
11        hibernate:
12          format_sql: true
13          # show_sql: true
14
15      # ensures that testcontainers reuses started containers
16      # (not sure if properties actually do something)
17      testcontainers:
18        reuse:
19          enable: true
```

Welcome to GitHub Copilot for IntelliJ IDEA

GitHub Copilot is your AI pair programmer, now in IntelliJ IDEA. Get autocomplete-style suggestions as you code, fix code, generate tests, and much more!

Next: Get code suggestions in your editor

Document 1/1 > spring > jpa > properties:

Project

- exceptions
- interfaces
 - IUserRepository
 - IUserService
 - User
 - UserController
 - UserService
 - GamsApiApplication
 - MetadataBaseEntity
 - MetadataBaseEntityBuilder
- resources
 - static
 - templates
 - application.yml
 - application-dev.yml
- test
 - java
 - org.zim.gamsapi
 - Datastream
 - DatastreamBuilderTest
 - DatastreamControllerIT
 - DatastreamRepositoryIT
 - DatastreamServiceIT
 - DatastreamTest
 - DigitalObject
 - ConstraintViolationTest
 - DigitalObjectBuilderTest
 - DigitalObjectControllerIT
 - DigitalObjectRepositoryIT
 - DigitalObjectServiceIT
 - DigitalObjectTest
 - enums
 - TestDatastream

application.yml

```

1  spring:
2    profiles:
3      # loads tests in dev profile.
4      active: dev
5    jpa:
6      hibernate:
7        # delete database on application startup during tests
8        ddl-auto: create
9      # format SQL statements
10     properties:
11       hibernate:
12         format_sql: true
13         # show_sql: true
14
15 # ensures that testcontainers
16 # (not sure if properties active)
17 testcontainers:
18   reuse:
19     enable: true

```

- Show Context Actions Alt+Enter
- Paste Ctrl+V
- Copy / Paste Special
- Column Selection Mode Alt+Shift+Insert
- Find Usages Alt+F7
- Go To
- Folding
- Analyze
- Refactor
- Generate... Alt+Insert
- Open In
- Local History
- Git
- Compare with Clipboard
- Diagrams
- GitHub Copilot

- Generate Docs
- Explain This
- Fix This
- Simplify This
- Generate Tests
- Reference File in Chat

Project

- exceptions
- interfaces
 - IUserRepository
 - IUserService
 - User
 - UserController
 - UserService
 - GamsApiApplication
 - MetadataBaseEntity
 - MetadataBaseEntityBuilder
- resources
 - static
 - templates
 - application.yml
 - application-dev.yml
- test
 - java
 - org.zim.gamsapi
 - Datastream
 - DatastreamBuilderTest
 - DatastreamControllerIT
 - DatastreamRepositoryIT
 - DatastreamServiceIT
 - DatastreamTest
 - DigitalObject
 - ConstraintViolationTest
 - DigitalObjectBuilderTest
 - DigitalObjectControllerIT
 - DigitalObjectRepositoryIT
 - DigitalObjectServiceIT
 - DigitalObjectTest
 - enums
 - TestDatastream

application.yml

```

1 spring:
2   profiles:
3     # loads tests in dev profile.
4     active: dev
5   ...

```

All Classes Files Symbols **Actions** Git Text Include disabled actions

copilot


- GitHub Copilot Chat
- GitHub Copilot
 - Copilot: Edit Keyboard Shortcuts... Tools | GitHub Copilot
 - Copilot: Log CA Certificates Tools | GitHub Copilot
 - Copilot: Show GitHub Copilot Status
 - Copilot: View GitHub Documentation... Tools | GitHub Copilot
 - Copilot: Logout from GitHub Tools | GitHub Copilot
 - Copilot: Login to GitHub
 - Copilot: Show Completions Alt+I Tools | GitHub Copilot
 - Copilot: Log Diagnostics Tools | GitHub Copilot
 - Copilot: View Feedback Forum... Tools | GitHub Copilot
 - Copilot: Disable Completions Ctrl+Alt+Shift+O Tools | GitHub Copilot
 - Copilot: Open Settings... Tools | GitHub Copilot
 - Copilot: Disable Completions for YAML Tools | GitHub Copilot
 - Copilot: Open Completions Tools | GitHub Copilot
 - Copilot: View Welcome Guide... Tools | GitHub Copilot
 - Open GitHub Copilot Chat Ctrl+Shift+C Tools | GitHub Copilot
- GitHub Copilot
 - GitHub Copilot Settings > Plugins
 - GitHub Copilot notifications Settings > Notifications
 - Plugins: GitHub Copilot
 - Show GitHub Copilot
- Welcome to GitHub Copilot

Activate GitHub Copilot Chat window Assign Shortcut Alt+Enter



3.2. Autocomplete

- Fill in the **most accurate generated suggestion** for the developer.
- Let the **developer decide** if he / she accepts the generated result.



3.2. Autocomplete - simple functions

👤 Sebastian Schiller-Stoff *

```
def calculate_average():|
    """
    Calculates the average of the given list of numbers.
    """
    numbers = [1, 2, 3, 4, 5]
    return sum(numbers) / len(numbers)
```



3.2. Autocomplete - changed parameters

Sebastian Schiller-Stoff *

```
def calculate_average(num1, num2):  
    return (num1 + num2) / 2
```

Sebastian Schiller-Stoff *

```
def calculate_average(numbers):  
    return sum(numbers) / len(numbers)
```

Sebastian Schiller-Stoff *

```
def calculate_average(numbers1, numbers2):  
    """  
    Calculates the average of two lists of numbers.  
    """  
    return (sum(numbers1) + sum(numbers2)) / (len(numbers1) + len(numbers2))
```



3.3. Suggestions

- Let the developer choose between different solutions.
- The developer should be able to **review** generated **results**.

3.3. Suggestions

```
13  
14  
15 def calculate_average(numbers1, numbers2):  
16
```

Sebastian Schiller-Stoff *

GitHub Copilot x

```
Accept solution  
def calculate_average(numbers1, numbers2): # type: ignore  
    """  
    Calculates the average of the two given lists of numbers.  
    """  
    return (sum(numbers1) + sum(numbers2)) / (len(numbers1) + len(numbers2))
```

Accept solution

```
def calculate_average(numbers1, numbers2):  
    """  
    Calculate the average of two lists of numbers.  
    """  
    average1 = sum(numbers1) / len(numbers1)  
    average2 = sum(numbers2) / len(numbers2)  
  
    return (average1 + average2) / 2
```

Accept solution

```
def calculate_average(numbers1, numbers2):  
    """  
    Calculate average of two lists of numbers.  
    """  
    return (sum(numbers1) + sum(numbers2)) / (len(numbers1) + len(numbers2))
```

Accept solution

```
def calculate_average(numbers1, numbers2):  
    """  
    Calculates the average of two lists of numbers.  
    """  
    return (sum(numbers1) + sum(numbers2)) / (len(numbers1) + len(numbers2))
```



3.4. Integrated commands and prompts (actions)

- Trigger **certain predefined tasks** for the current / **marked code or file**.
 - /doc
 - /test
 - /fix
 - /test
 - /simplify
- Use **optimized prompts** (prompt engineering) in the background.
 - “You are an expert for software documentation...”

gams5-client main

```
class Pyriilo:
    return self.integration_service.integrate(project_abbr, id)

def disintegrate_project_object(self, project_abbr: str, id: str):
    """
    Disintegrates a single object of a project in gams-integration services.
    """
    return self.integration_service.disintegrate(project_abbr, id)

def ingest(self, project_abbr: str):
    """
    Performs a complete ingest operation based on the definition of a project.
    This method is responsible for the entire process of ingesting a project.
    It includes transforming SIPs to BagIt format, ingesting all bags from the local bag structure, deleting all objects first,
    deleting all indices from dependent services, and finally integrating the project objects.
    """
    # demo for transforming local SIPs to bagit format
    self.transform_sips_to_bags(project_abbr)

    # optionally delete all objects first
    self.delete_objects(project_abbr)

    # delete all indices from dependent services
    self.disintegrate_project_objects(project_abbr)

    # ingesting all bags from the local bag structure
    self.ingest_bags(project_abbr)

    # demo index all
    self.integrate_project_objects(project_abbr)
```

1 usage Sebastian Schiller-Stoff

13°C Bewölkt

Search

GitHub Copilot Chat

Creating a ContentModelsService in Python

```
def ingest(self, project_abbr: str):
    """
    Performs a complete ingest operation based on the definition of a project.
    This method is responsible for the entire process of ingesting a project.
    It includes transforming SIPs to BagIt format, ingesting all bags from the local bag structure, deleting all objects first,
    deleting all indices from dependent services, and finally integrating the project objects.
    """
    # demo for transforming local SIPs to bagit format
    self.transform_sips_to_bags(project_abbr)

    # optionally delete all objects first
    self.delete_objects(project_abbr)

    # delete all indices from dependent services
    self.disintegrate_project_objects(project_abbr)

    # ingesting all bags from the local bag structure
    self.ingest_bags(project_abbr)

    # demo index all
    self.integrate_project_objects(project_abbr)
```

Show Context Actions Alt+Enter

Cut Ctrl+X

Copy Ctrl+C

Paste Ctrl+V

Copy / Paste Special >

Column Selection Mode Alt+Shift+Insert

Find in Files >

Find Usages Alt+F7

Go To >

Folding >

Search with Google >

Refactor >

Generate... Alt+Insert

Run 'Pyriilo' Ctrl+Shift+F10

Debug 'Pyriilo' ⚙

Modify Run Configuration... >

Open In >

Local History >

Git >

Execute Selection in Python Console Alt+Shift+E

Run File in Python Console >

Compare with Clipboard >

GitHub Copilot >

Generate Docs

Explain This

Fix This

Simplify This

Generate Tests

Reference File in Chat

Stoff-Sebi

/doc

GitHub Copilot



```
def ingest(self, project_abbr: str):
```

```
    """
```

```
    Performs a complete ingest operation based on the defined SIP folders.
```

```
    This method is responsible for the entire process of ingesting SIPs.
```

```
    The process includes transforming SIPs to BagIt format, deleting ex
```

```
    Args:
```

```
        project_abbr (str): The abbreviation of the project name. This
```

```
    The method performs the following steps:
```

1. Transform SIPs to BagIt format: This is done by calling the 'tr
2. Delete all objects: This is an optional step that deletes all d
3. Delete all indices from dependent services: This is done by call
4. Ingest all bags from the local bag structure: This is done by c
5. Integrate all objects of a project: This is done by calling the

Stoff-Sebi

/simplify

GitHub Copilot



```
def ingest(self, project_abbr: str):
```

```
    """
```

```
    Performs a complete ingest operation based on the defined SIP folders.
```

```
    """
```

```
    self.transform_sips_to_bags(project_abbr)
```

```
    self.delete_objects(project_abbr)
```

```
    self.disintegrate_project_objects(project_abbr)
```

```
    self.ingest_bags(project_abbr)
```

```
    self.integrate_project_objects(project_abbr)
```

The simplified version of the code removes the comments. The comments were removed because the method names are self-explanatory and the docstring at the beginning of the method provides a high-level overview of what the method does. This makes the code cleaner and more concise without changing its behavior.

> Using 1 Reference



3.5. Integrated chat

- Provide **chatbot like integration** in the IDE.
- Allow the developer to ask any question and to combine it with the **context controlled by the IDE.**
 - Context generation: Use cursor position, open files in tabs, use predefined prompts, use git change tracking etc.
- **Simplify user input** (for coding related questions) AND **simplify** the process to actually **use generated results** (text and code)
 - “Insert this code at cursor position”
 - “Copy the code block”
 - ...




3.5. Integrated chat

I want you to analyze the code and make some suggestions what can be improved. Add these suggestions as TODOs to the code




Sure, here are some suggestions for improvements that can be added as TODOs to the code:

```
def ingest(self, project_abbr: str):  
    """  
    Performs a complete ingest operation based on the defined SIP folders.  
    """  
    # TODO: Consider adding error handling or checks to ensure the SIP folders exist and are accessible  
  
    # demo for transforming local SIPs to bagit format  
    self.transform_sips_to_bags(project_abbr)  
  
    # TODO: Consider adding a confirmation prompt before deleting all objects, as this is a destructive operat  
  
    # optionally delete all objects first  
    self.delete_objects(project_abbr)  
  
    # delete all indices from dependend services  
    self.disintegrate_project_objects(project_abbr)  
  
    # TODO: Add error handling for the ingest_bags method to handle any issues that might occur during ingesti  
  
    # ingesting all bags from the local bag structure  
    self.ingest_bags(project_abbr)
```


 Stoff-Sebi

Please resolve all todos in `Pyrilo.py`

 GitHub Copilot



Sure, here's how you might resolve the TODOs in `Pyrilo.py`:

1. Add error handling or checks to ensure the SIP folders exist and are accessible.
2. Add a confirmation prompt before deleting all objects, as this is a destructive operation.
3. Add error handling for the `ingest_bags` method to handle any issues that might occur during ingestion.
4. Add logging statements to provide more visibility into the progress of the ingest operation.

Here's the updated code:

```
def ingest(self, project_abbr: str):
    """
    Performs a complete ingest operation based on the defined SIP folders.
    """
    # Check if the SIP folders exist and are accessible
    if not os.path.exists(project_abbr):
        logging.error(f"SIP folders for project {project_abbr} do not exist or are not accessible.")
        return

    # Transform local SIPs to bagit format
    self.transform_sips_to_bags(project_abbr)

    # Confirmation prompt before deleting all objects
    confirm = input("Are you sure you want to delete all objects? This is a destructive operation. (yes/no): ")
    if confirm.lower() != 'yes':
        return


    # Delete all objects first
    self.delete_objects(project_abbr)


    # Delete all indices from dependent services
    self.disintegrate_project_objects(project_abbr)


    try:
        # Ingest all bags from the local bag structure
        self.ingest_bags(project_abbr)
    except Exception as e:
```


Please note that this code now includes a user prompt which may not be suitable for all use cases, especially for automated scripts or services. You might want to replace this with a parameter or configuration option. Also, the error handling for `ingest_bags` is very generic and might need to be adjusted based on the specific exceptions that can be thrown by this method.

✓ Using 4 References

 main.py

 DigitalObject.py

 Pyrilo.py

 DigitalObjectService.py



3.5. Chat history

- Help the developer to “train” **specialized contexts** for certain scenarios.
 - Ask questions in context of certain conversations.
- **Save and maintain** reference to past solutions and questions.



GitHub Copilot Chat








2 ^ v

← Back

+ ...



-  Analyzing and Improving Code with TODOs 5 minutes ago
-  Code Documentation and Unit Testing 36 minutes ago
-  Improving code and solving TODOs in Pырilo.py 11 days 21 hours ago
-  Understanding the setup.py file in Python packages 11 days 21 hours ago
-  New Conversation May 2, 2024 13:01 11 days 21 hours ago


ders ex

cts, as

sues th


GitHub Copilot Chat

Code Documentation and Unit Testing


 GitHub Copilot

Hi @Stoff-Sebi, how can I help you?

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and [share feedback](#) so that we can learn and improve.

 Stoff-Sebi

I want to create a ContentModels service as I did with DigitalObjectService for DigitalObject and SipService for SIP. Please generate the required code


 GitHub Copilot

Sure, I can help you create a `ContentModelsService` similar to your `DigitalObjectService` and `SipService`.

First, let's define the `ContentModelsService` class. This class will contain methods that perform operations related to content models. The exact methods will depend on your specific needs, but for now, I'll create a few placeholder methods based on the methods in your `DigitalObjectService` class.

```
ContentModelsService:
    __init__(self):
        pass

    save_content_model(self, id: str, project_abbrev: str):
        """
        Saves a content model
        """
        pass
```

 What are some best practices for writing unit tests?

I need to add a section about unittesting ...





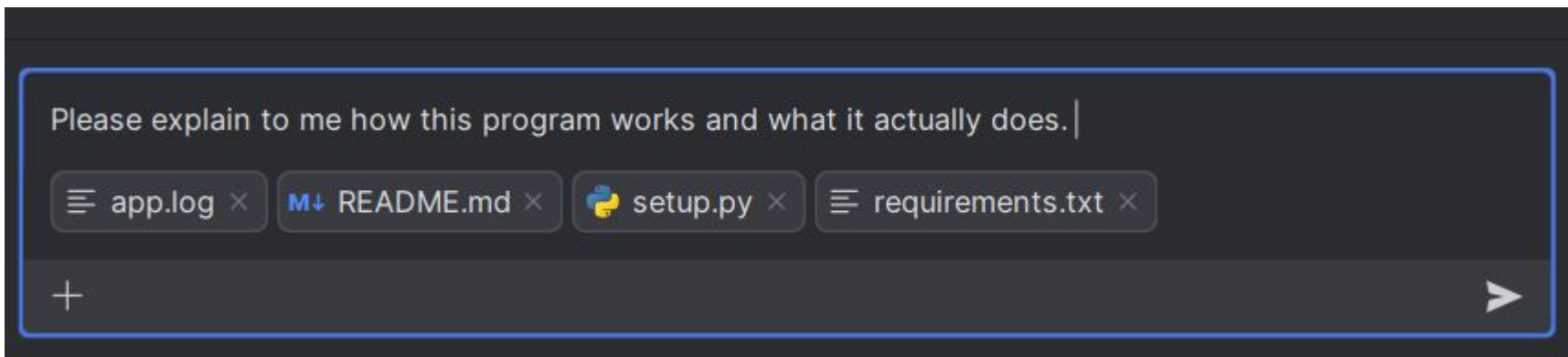
3.5. Integrated Chat - Explicit context


- Provide **transparency** for which files / code was used as context for the generated response.
- Let the developer **influence the generated context** (from the IDE) via **explicitly stating** which **files** should be used as **reference**.




3.5. Integrated Chat - Explicit context

- Might also include non-code files that are also relevant for certain questions, like:
 - Log files
 - Readmes
 - Documentation files etc.



 Stoff-Sebi

Please explain to me how this program works and what it actually does.

 GitHub Copilot



The program you're referring to is a Python-based client application for GAMS5, named Pyrilo. It's designed to act as a proxy against the REST-API and imitates the behavior of Cirilo in Python.

Here's a brief overview of its functionality:

1. **Initialization:** The program is set up and run in a standard Python virtual environment (venv). This involves creating a local venv and installing the necessary dependencies from a `requirements.txt` file.
2. **Running the Program:** The main entry point of the program is `main.py`. When this script is run, it starts the Pyrilo application.
3. **Interacting with GAMS5:** The application interacts with a GAMS5 instance, which needs to be available for Pyrilo to function. GAMS5 is a high-level modeling system for mathematical optimization. Pyrilo acts as a client, making requests to the GAMS5 server and handling the responses.
4. **Data Ingestion:** The logs you provided show that the program is ingesting data from various folders (e.g., `vor231`, `vor232`, etc.) into a project named `demo`. This is done by zipping the folder, then making a POST request to the GAMS5 server to ingest the zipped data.

The specific details of what the program does with the ingested data, and how it interacts with the GAMS5 server, would be found in the code within `main.py` and other Python scripts in the project.



References (I)

GitHub CoPilot Workspace

- Dohmke, Thomas. 2024. „GitHub Copilot Workspace: Welcome to the Copilot-Native Developer Environment“. *The GitHub Blog* (blog). 29. April 2024. <https://github.blog/2024-04-29-github-copilot-workspace/>.
- Heise online. 2024. „GitHub veröffentlicht KI-gestützte Entwicklungsumgebung Copilot Workspace“. *c't Magazin*. 29. April 2024. <https://www.heise.de/news/GitHub-veroeffentlicht-KI-gestuetzte-Entwicklungsumgebung-Copilot-Workspaces-9698383.html>.

GitHub CoPilot Enterprise

- Dohmke, Thomas. 2024. „GitHub Copilot Enterprise Is Now Generally Available“. *The GitHub Blog* (blog). 27. Februar 2024. <https://github.blog/2024-02-27-github-copilot-enterprise-is-now-generally-available/>.



References (II)

JetBrains IDEs and Copilot Chat

- Ramel, By David, und 03/08/2024. o. J. „JetBrains IDEs Finally Get GitHub Copilot Chat -“. Visual Studio Magazine. Zugegriffen 7. Mai 2024. <https://visualstudiomagazine.com/Articles/2024/03/08/jetbrains-chat.aspx>.
- „GitHub Copilot Chat General Availability in JetBrains IDE“. 2024. *The GitHub Blog* (blog). 7. März 2024. <https://github.blog/changelog/2024-03-07-github-copilot-chat-general-availability-in-jetbrains-ide/>.

Infinity AI und Copilot

- Lück, Bernhard. 2024. „Infinity AI Copilot wird mit Microsoft Azure OpenAI Service erweitert“. Security-Insider. 29. April 2024. <https://www.security-insider.de/infinity-ai-copilot-wird-mit-microsoft-azure-openai-service-erweitert-a-14b37d3068d9155165a923b6664e2aac/>.



References (III)

Alternatives for GitHub Copilot

- „Best Free and Paid GitHub Copilot Alternatives“. o. J. Zugegriffen 7. Mai 2024.
<https://code.pieces.app/blog/best-free-and-paid-github-copilot-alternatives>.
<https://code.pieces.app/blog/best-free-and-paid-github-copilot-alternatives>.
- Conner Ardman, Reg. 2023. *I Tried Every AI Coding Assistant*.
<https://www.youtube.com/watch?v=TXtnFw9eDmM>.
- Grishina, Anastasia. 2024. „GitHub Copilot Alternatives for Developers to Choose From“. *SoftTeco* (blog). 4. April 2024. <https://softteco.com/blog/github-copilot-alternatives-for-developers>.
- Kopilot. o. J. „15 Best GitHub Copilot Alternatives to try in 2024 (free and paid)“. Zugegriffen 7. Mai 2024.
<https://www.copilot.live/de/blog/github-copilot-alternative>.



References (IV)

Use-Cases

- Kerr, Kedasha. 2024. „10 Unexpected Ways to Use GitHub Copilot“. *The GitHub Blog* (blog). 22. Januar 2024. <https://github.blog/2024-01-22-10-unexpected-ways-to-use-github-copilot/>.
- Scarlett, Rizel. 2022. „8 Things You Didn't Know You Could Do with GitHub Copilot“. *The GitHub Blog* (blog). 14. September 2022. <https://github.blog/2022-09-14-8-things-you-didnt-know-you-could-do-with-github-copilot/>.

AI and software development

- Heise Online. 2023. „Universe 2023: GitHub sieht KI als wichtigsten Treiber für Softwareentwicklung“. *c't Magazin*. 9. November 2023. <https://www.heise.de/news/Universe-2023-GitHub-sieht-KI-als-wichtigsten-Treiber-fuer-Softwareentwicklung-9356924.html>.

Additional

- „Github-COO Kyle Daigle: ‚Programmieren wird eine grundlegende Fähigkeit bleiben‘ - Golem.de“. 2024. 30. April 2024. <https://www.golem.de/news/github-coo-kyle-daigle-programmieren-wird-eine-grundlegende-faehigkeit-bleiben-2404-184631.html>.
- Shipper, Dan. 2024. „The Next Big Programming Language Is English“. 3. Mai 2024. <https://every.to/chain-of-thought/i-spent-24-hours-with-github-copilot-workspaces>.