vFunction 4.3 Release Notes

Feature Highlight - Assisted Refactoring with new Agentic capabilities

Introducing vFunction's new agentic AI capabilities to automate refactoring of the generated TODOs.

The assisted refactoring capability enables organizations to complete refactoring and re-architecting projects within vFunction. Starting from AI driven analysis and automatic creation of TODOs, teams can now use GenAI driven code refactoring guided by the TODOs, automated validation of the TODOs, all the way to automated extraction of refactored modules from a monolithic application.

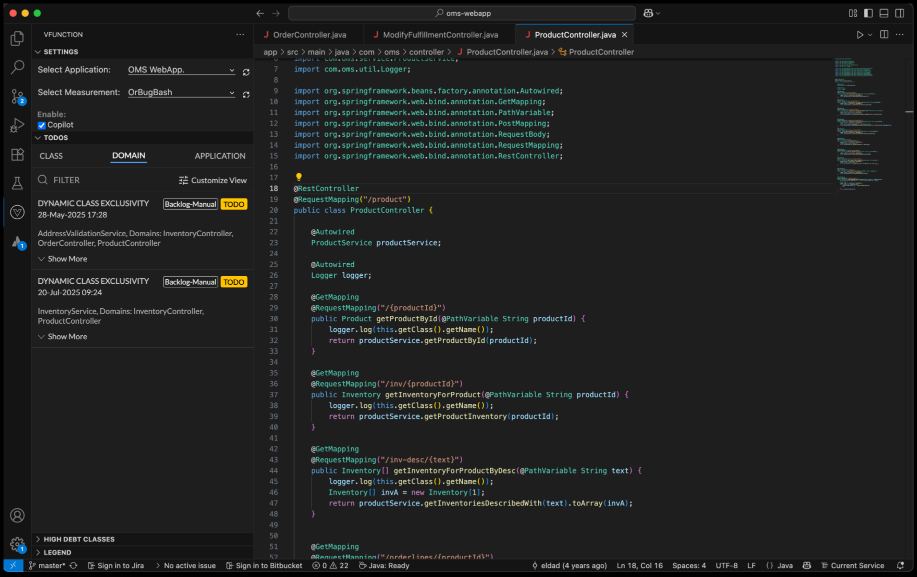
The refactoring is automated through integrating with code-assistants such as Github Copilot and Amazon Q, and can be applied in three ways:

- ### Visual Studio Code Extension

  A vFunction VSCode Extension is now available in the Marketplace.

  By connecting to the vFunction server, users can get architectural context directly on classes and methods. Engineers can view TODOs relevant to the open class in the extension's TODO panel, to the class' domains and to the entire application.

  The TODOs can either be sent directly to Copilot, or have a prompt copied into Amazon Q or other AI assistants.



Select Application: OMS WebApp.

Select Measurement: OrBugBash

Enable:
☑ Copilot

DYNAMIC CLASS EXCLUSIVITY Backlog-Manual TODO
28-May-2025 17:28
AddressValidationService, Domains: InventoryController, OrderController, ProductController

DYNAMIC CLASS EXCLUSIVITY Backlog-Manual TODO
20-Jul-2025 09:24
InventoryService, Domains: InventoryController, ProductController

```java
import com.oms.service.ProductService;
import com.oms.util.Logger;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;


@RestController
@RequestMapping("/product")
public class ProductController {

    @Autowired
    ProductService productService;

    @Autowired
    Logger logger;

    @GetMapping
    @RequestMapping("/{productId}")
    public Product getProductById(@PathVariable String productId) {
        logger.log(this.getClass().getName());
        return productService.getProductById(productId);
    }

    @GetMapping
    @RequestMapping("/inv/{productId}")
    public Inventory getInventoryForProduct(@PathVariable String productId) {
        logger.log(this.getClass().getName());
        return productService.getProductInventory(productId);
    }

    @GetMapping
    @RequestMapping("/inv-desc/{text}")
    public Inventory[] getInventoryForProductByDesc(@PathVariable String text) {
        logger.log(this.getClass().getName());
        Inventory[] invA = new Inventory[1];
        return productService.getInventoriesDescribedWith(text).toArray(invA);
    }


    @GetMapping
    @RequestMapping("/orderlines/{productId}")
```

- **Built-in MCP Server**

  Users can connect to the built-in vFunction MCP server, exposing tools for automated refactoring to any code assistant with MCP server support, among which are Github Copilot and Amazon Q.

  Configuration instructions to expose the tools to a code repository are accessible from the TODOs List, TODOs Dashboard and from the VSCode plugin.

- **Accessible GenAI Prompts for Each TODO**

  A new detailed GenAI prompt, with the context and instructions on how to automatically resolve the TODO is now available for monolithic applications.
  A new filter was added in the TODOdashboard to display only those TODOss that can be refactored in an assisted way.

New Features and Capabilities

- **New Manual TODO: "Remove from Common"**

  A manual refactoring TODO (a TODO specified by the user) for removing a domain class from the common library. The class is currently in the common library due to compile-time dependencies from other common classes.

  This TODO is available for automated refactoring.

- **Circular Flow Detection in Monolithic Applications**

  vFunction now highlights circular flows within monolithic applications, identifying circular execution paths between domains. Entry points involved in these flows are marked, enabling teams to pinpoint and refactor tightly coupled domains.

  This capability creates TODOs automatically on the baseline and on the latest measurement.

  This TODO is available for automated refactoring.

- **New TODO: Aging Frameworks Detection**

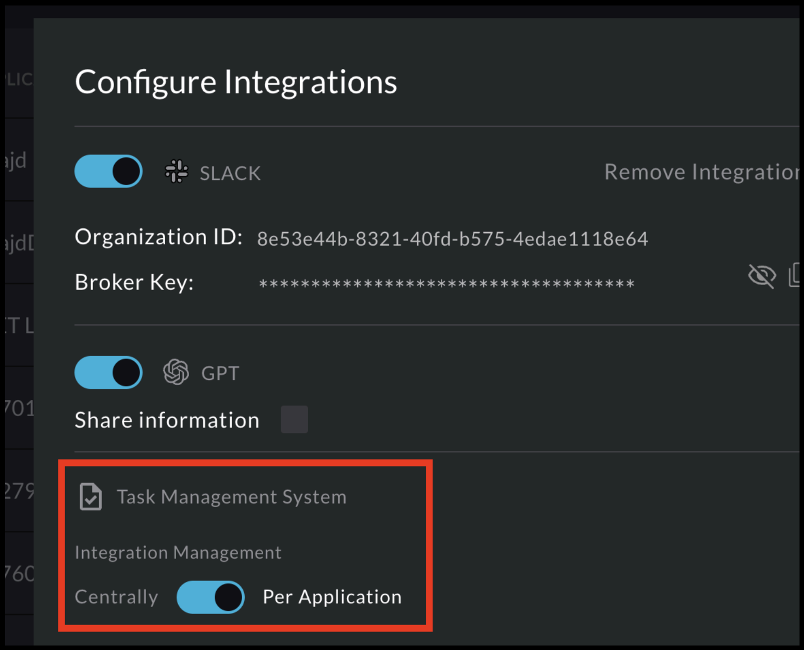
The aging framework TODO has been introduced to help identify monolithic services that rely on outdated libraries. Flagging aging frameworks is important because it can contribute to technical debt and security vulnerabilities if left unaddressed.
Note: The TODO becomes active only when the aging frameworks reference table has been updated within the last 30 days. If not, it remains disabled and a notification is shown in the Observability tab.

This TODO is available for automated refactoring.

- **Azure DevOps / Jira Integration: Per Application Access Control and Project Selection**

  Added support for users to sync between vFunction apps and JIRA/Azure DevOps projects, allowing users to sync TODOs only to specific projects.

  For users to view and manage access for their apps, admins should define access from *Settings › Apps* and turn “Integration Management” to “Per Application”.



Configure Integrations

SLACK

Remove Integration

Organization ID: 8e53e44b-8321-40fd-b575-4edae1118e64

Broker Key: *************************************

GPT

Share information □

Task Management System

Integration Management

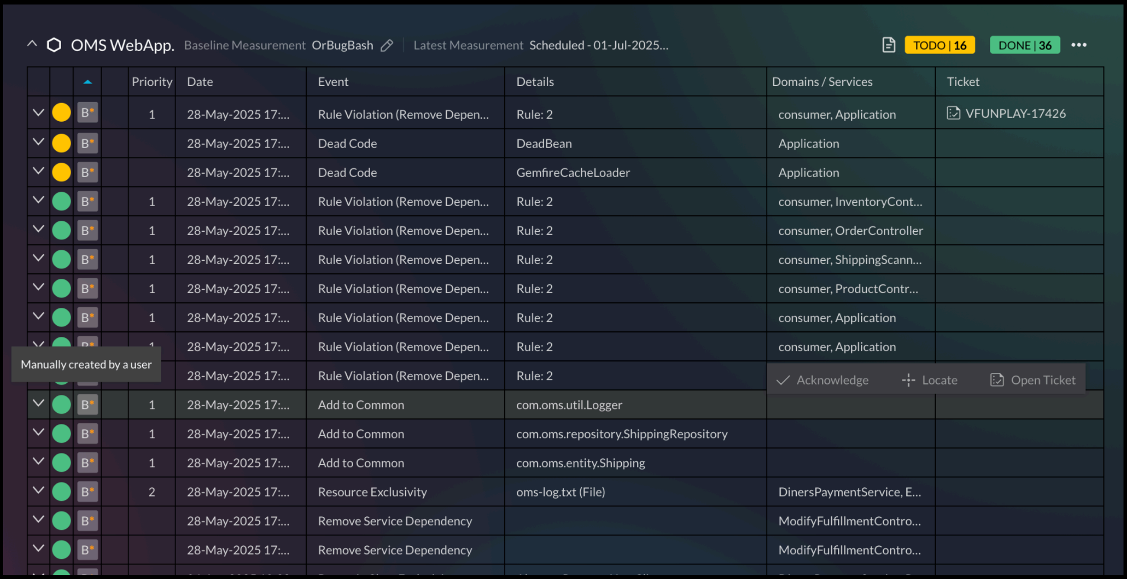
Centrally / Per Application

- **Jira Software Support**

  A new integration option, “Jira Software,” is now available to support organizations using on-prem or self-hosted versions of Jira (v9.12 and up). This extends our support for JIRA Cloud users.

  Jira Software integration uses Personal Access Tokens (instead of OAuth).

- **Clear Labeling of Manually Created TODOs**
  Manually created TODOs are now explicitly labeled in the UI. This allows users to quickly distinguish between automated and manually entered TODOs.



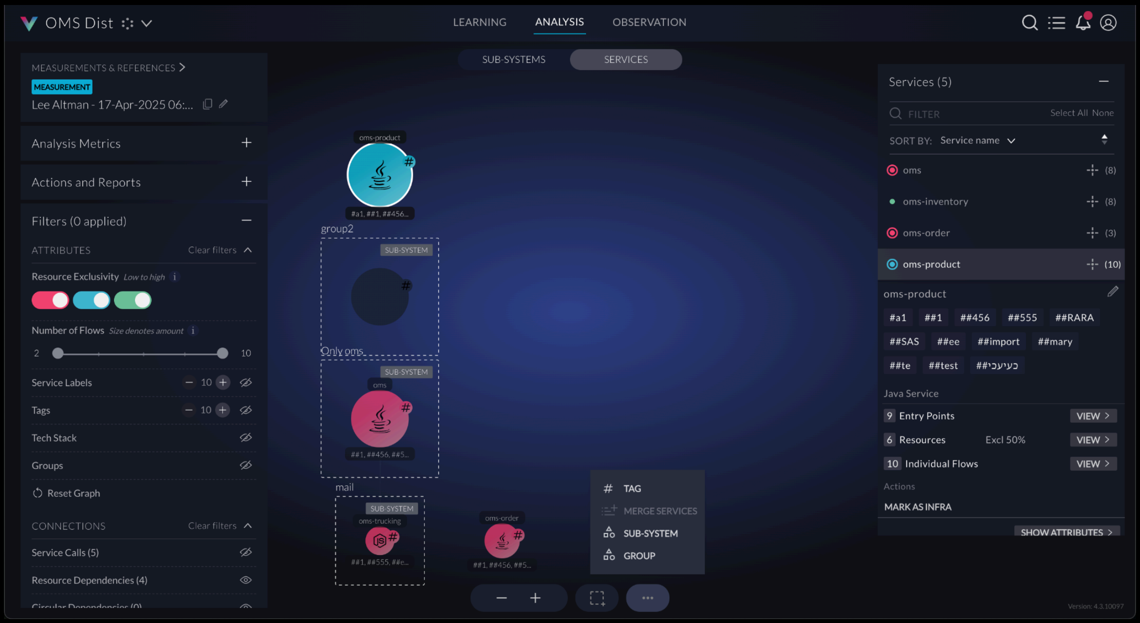
				Priority	Date	Event	Details	Domains / Services	Ticket
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, Application	VFUNPLAY-17426
⌄	●	B*			28-May-2025 17:...	Dead Code	DeadBean	Application	
⌄	●	B*			28-May-2025 17:...	Dead Code	GemfireCacheLoader	Application	
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, InventoryCont...	
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, OrderController	
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, ShippingScann...	
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, ProductContr...	
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, Application	
⌄	●	B*		1	28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2	consumer, Application	
					28-May-2025 17:...	Rule Violation (Remove Depen...	Rule: 2		
⌄	●	B*		1	28-May-2025 17:...	Add to Common	com.oms.util.Logger		
⌄	●	B*		1	28-May-2025 17:...	Add to Common	com.oms.repository.ShippingRepository		
⌄	●	B*		1	28-May-2025 17:...	Add to Common	com.oms.entity.Shipping		
⌄	●	B*		2	28-May-2025 17:...	Resource Exclusivity	oms-log.txt (File)	DinersPaymentService, E...	
⌄	●	B*			28-May-2025 17:...	Remove Service Dependency		ModifyFulfillmentContro...	
⌄	●	B*			28-May-2025 17:...	Remove Service Dependency		ModifyFulfillmentContro...	

Manually created by a user

New Capabilities for Distributed Applications supported by OTEL

- **Unified Services Navigation Experience**
  This UX change ensures consistent interactions when managing microservices and subsystems. All filtering, locating and bulk action tools now reside in one unified view. This includes replacing the “Selected Service” menu with a more intuitive “Actions” button. Which includes: tagging, merging, grouping and creating Sub-systems for services.

vFunction



- **Preserve and Display Subsystems in Imported C4 Diagrams**
  Architects can now maintain the integrity of original system designs when importing C4 diagrams into vFunction. Subsystems are no longer flattened into individual services, they are preserved and visually differentiated in the architecture view.

vfunction.com To learn more about vFunction Inc. click here.