Red Hat Summit

Connect

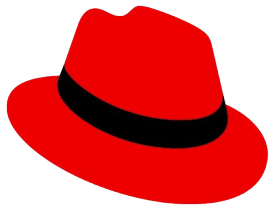
Legacy Workload migrieren mit AI

Wie Konveyor AI hilft, jeden Workload Cloud-fähig zu machen.

Georg Modzelewski
Specialist Solution Architect

Karsten Gresch
Specialist Solution Architect

Red Hat



Red Hat

Georg Modzelewski

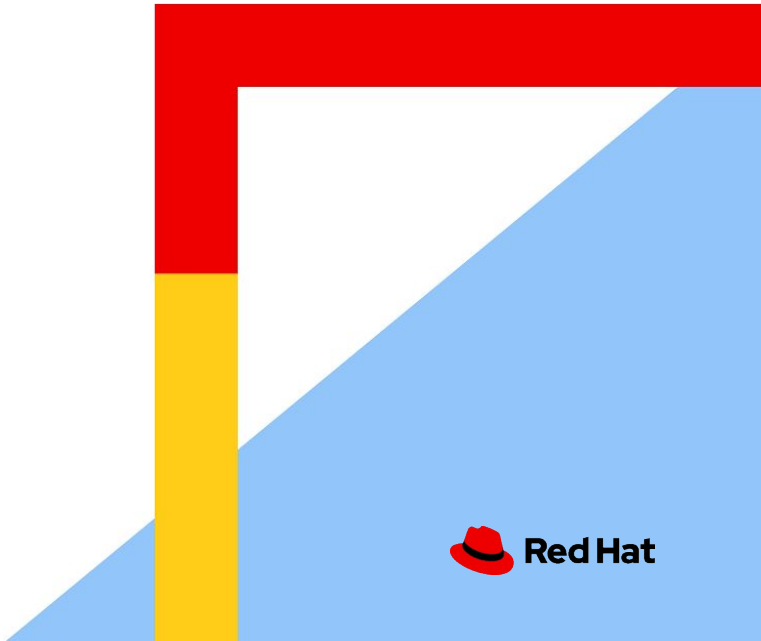
Specialist Solution Architect
Red Hat

Karsten Gresch

Specialist Solution Architect
Red Hat

Agenda

1. What?
2. Use case
3. Demo
4. How?
5. Roadmap
6. Q&A



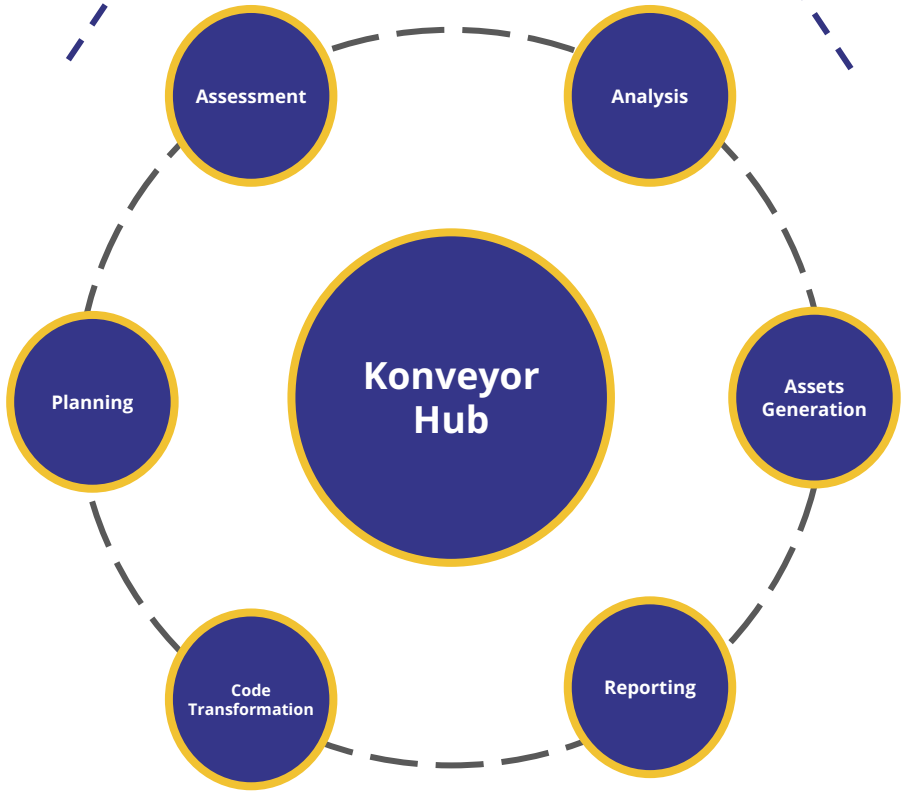
CLOUD NATIVE COMPUTING FOUNDATION

CNCF sandbox project

KONVEYOR

www.konveyor.io

A community of **people** passionate about **helping others modernize** and migrate their **applications** to Kubernetes by **building tools and best practices** on how to **accelerate the journey to Kubernetes**



Red Hat

IBM Research

claranet®

Additional contributors welcome

Red Hat Supported Tool

Available with OpenShift subscription

Migration Toolkit for Applications

Red Hat

Konveyor AI (Kai)

Upstream Early R&D Project

Why? - Goal

Goal: Improve the economics of re-platforming and refactoring applications to Kubernetes and cloud-native technologies by leveraging Generative AI

Approach:

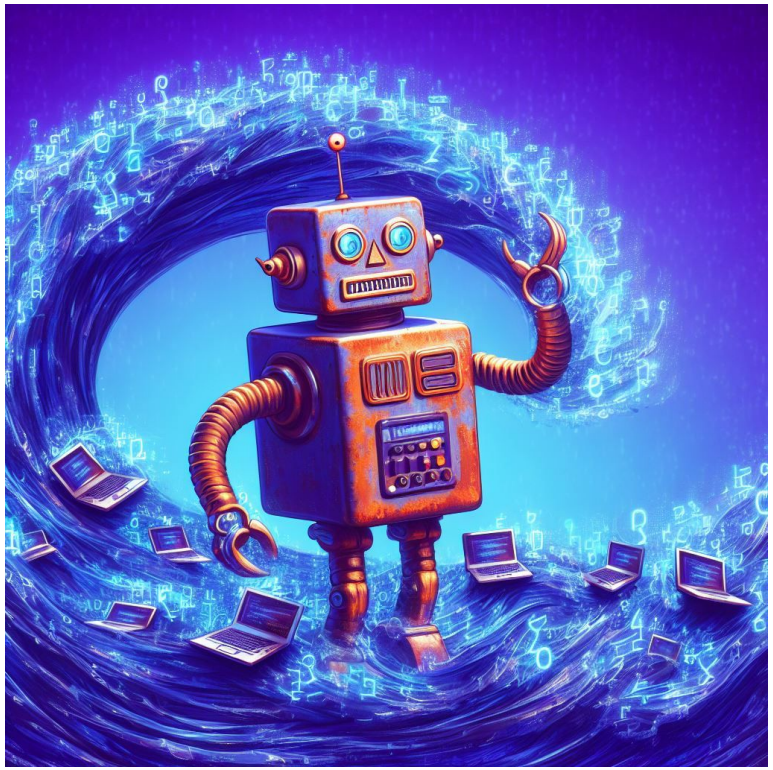
- Expand Konveyor MTA capabilities beyond surfacing information
- Generate code suggestions via LLMs for discovered migration issues
- Avoid fine-tuning each model by using Retrieval Augmented Generation (RAG)
  - Shape LLM results with examples of how Organization has solved similar problems in the past
- Model agnostic. And... ability to bring your own model

Red Hat

Konveyor AI (Kai)

Upstream Early R&D Project

How? Generative AI to automate source code changes between technologies



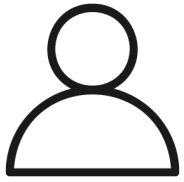
- Uses data in Konveyor to generate code suggestions
  - Source code analysis with Rules
    - Pinpoint issues to adopting a new technology
  - Changelog history from source repositories
    - Before/After of previously solved issues
- Crafts a tailored LLM prompt based on:
  - Knowledge of specific problems that need to be addressed
  - Knowledge of prior successful code changes
- Provides suggested code changes via IDE plugin

Blog article: https://www.konveyor.io/blog/kai-deep-dive-2024/

Red Hat

IDE Usage Walkthrough

Use case and target persona



Target Persona

- "**Migrator**": the developer who is updating the source code

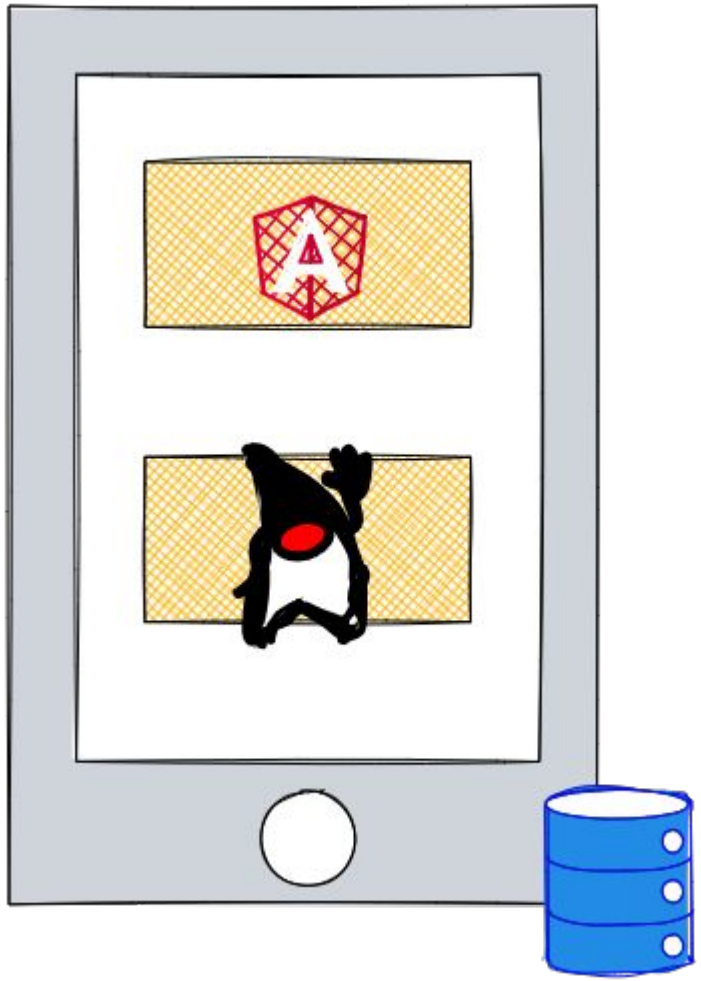
Use case

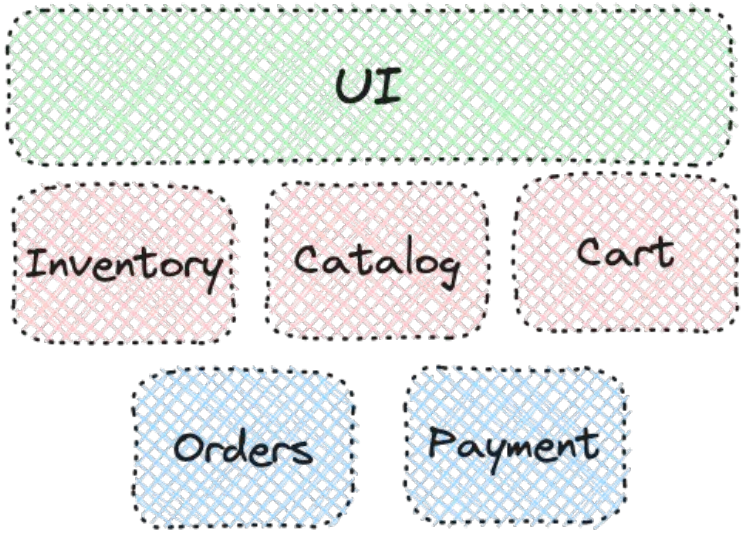
- Assisted migration of a Java EE app to Quarkus
  - Java EE app konveyor-ecosystem/coolstore
  - Uses community quarkus rules and custom-rules

Red Hat

Current Application Component Breakdown

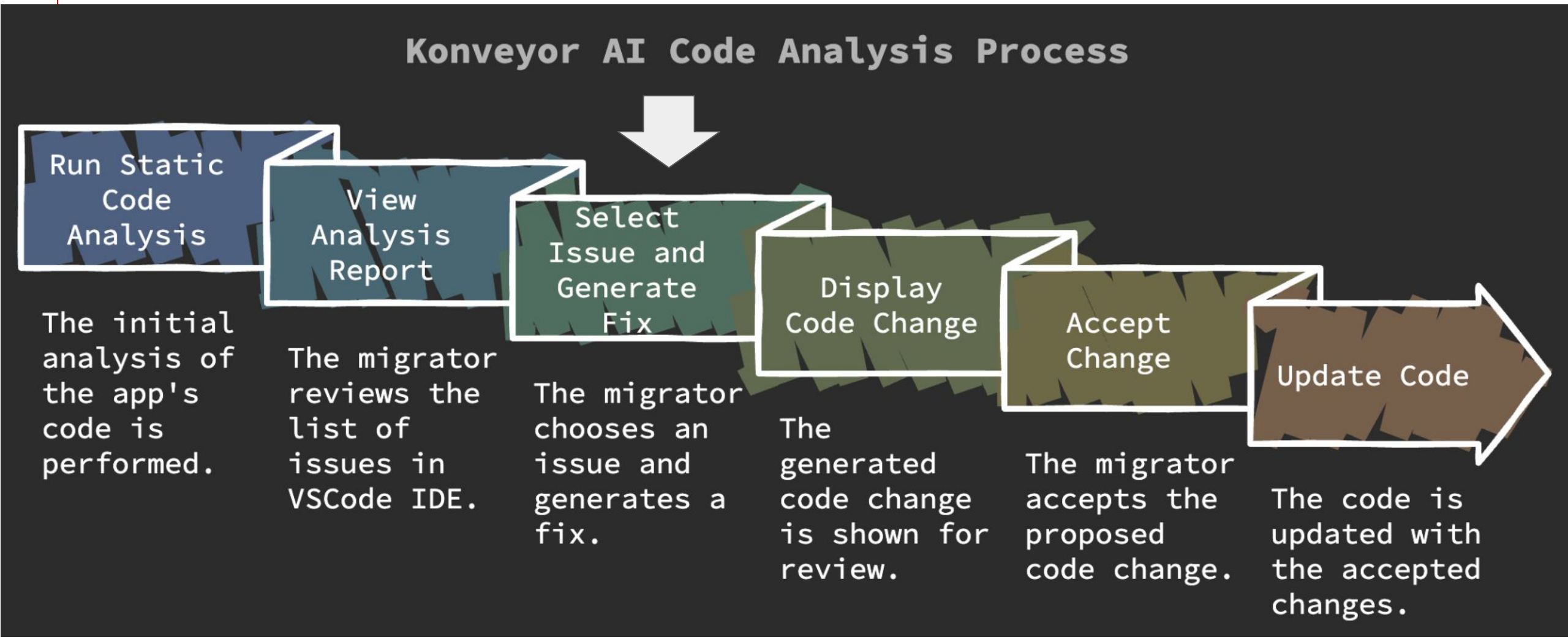
Coolstore Demo App (Legacy)





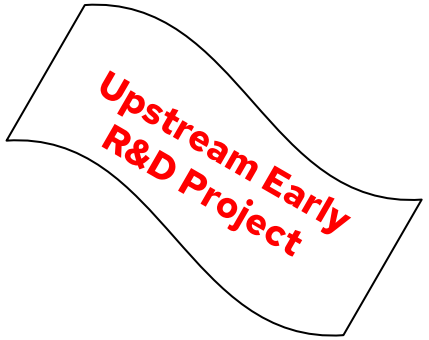
- ➔ Scaling challenges e.g. individual components or app
- ➔ Lifecycle is prolonged due to dependencies
- ➔ Everything runs in the same process
- ➔ Technical debt, hard to innovate

Red Hat


Konveyor AI Code Analysis Process
Run Static Code Analysis
View Analysis Report
Select Issue and Generate Fix
Display Code Change
Accept Change
Update Code
The initial analysis of the app's code is performed.
The migrator reviews the list of issues in VSCode IDE.
The migrator chooses an issue and generates a fix.
The generated code change is shown for review.
The migrator accepts the proposed code change.
The code is updated with the accepted changes.

Red Hat

Demo - Video

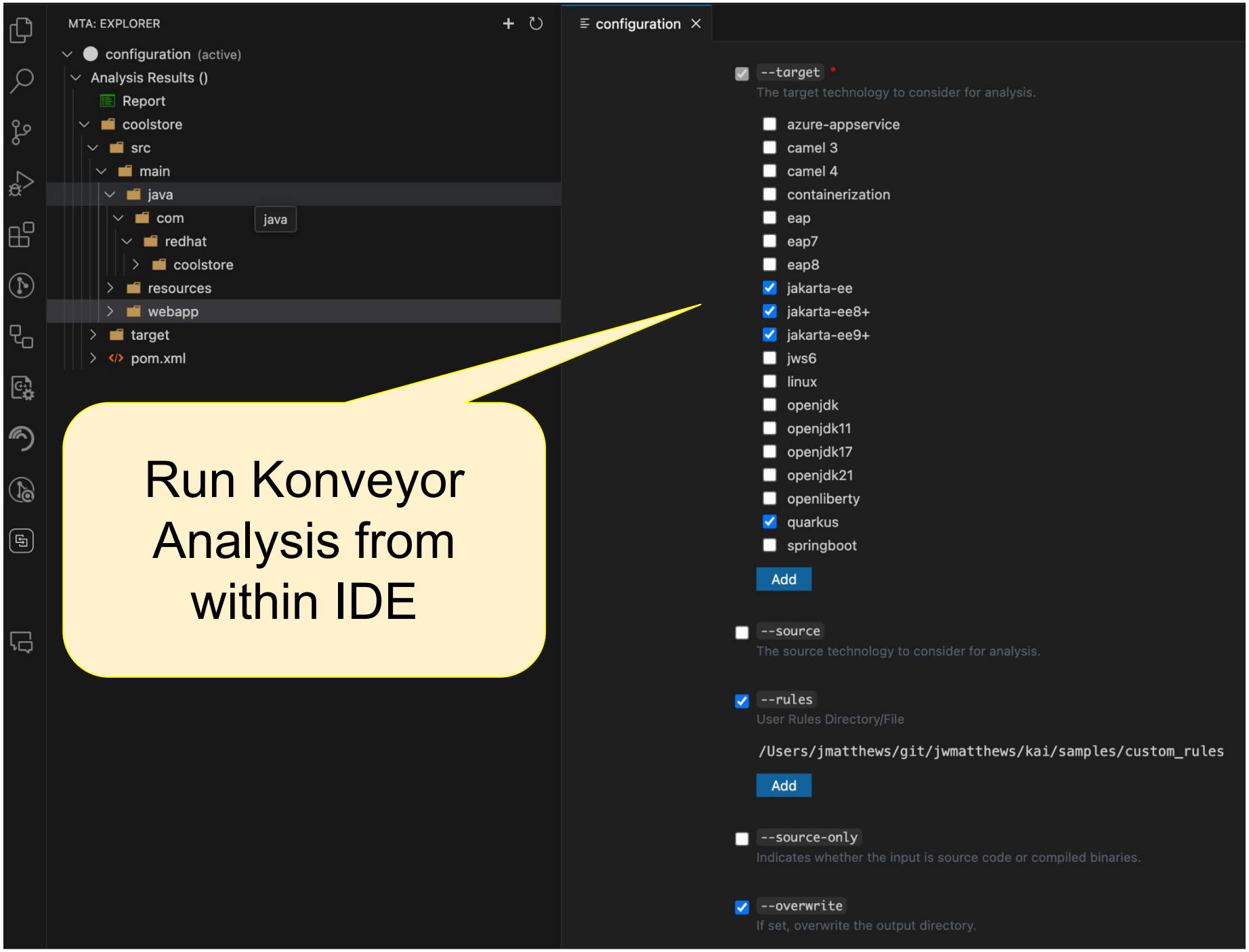


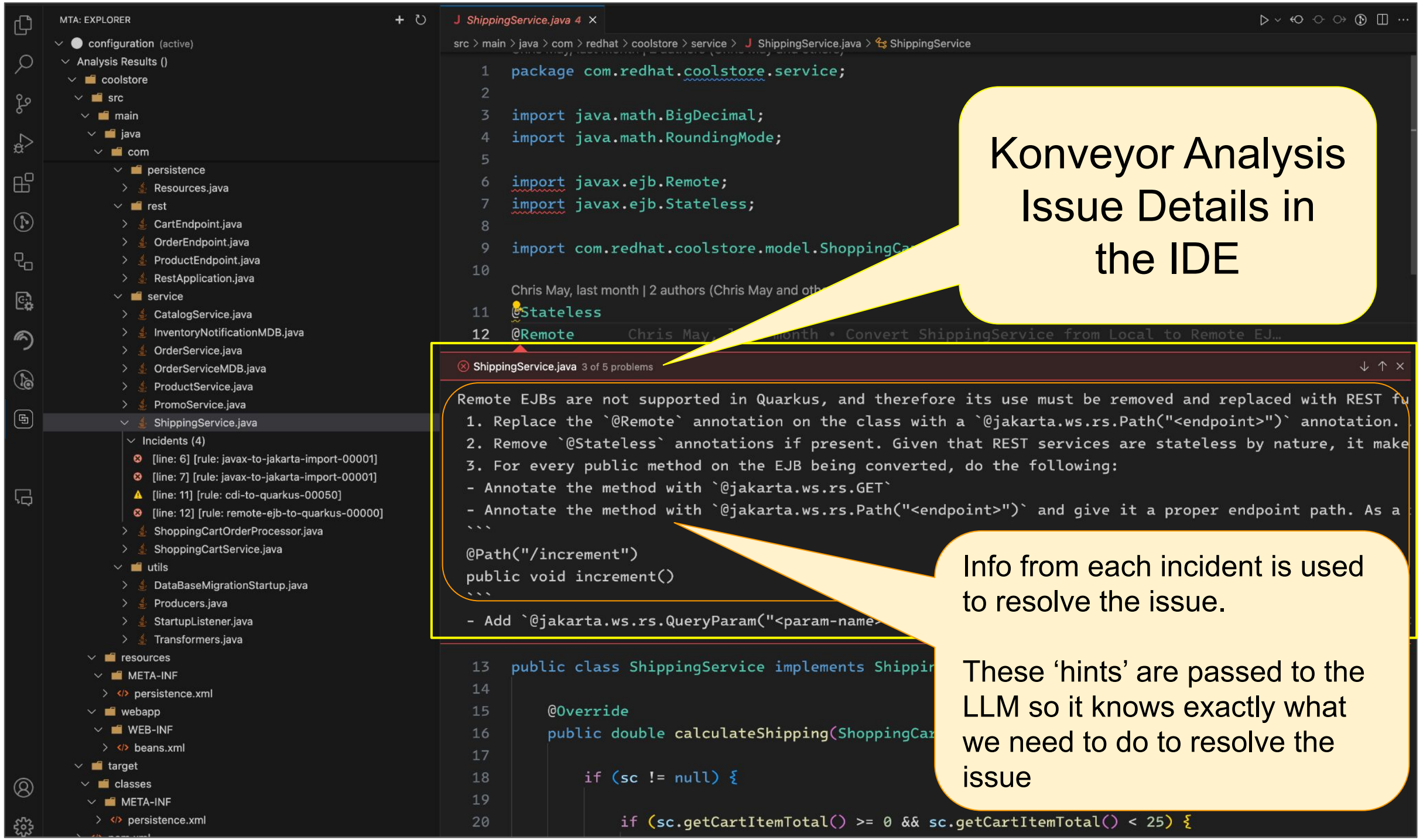
1. Clone git repository → cool Store app
2. Open IDE VSCode with **KAI extension*** (fork of Konveyor MTA)
3. Generate incident solution via LLM → **"Kai-Fix"**
    - Konveyor AI generates the code via LLMs and makes a suggestion
    - Incidents:
        - javax → jakarta
        - MDBs→reactive messaging
        - remote EJBs →REST endpoints (incl. logic and @Stateless→@ApplicationScopes)
        - JMS producers→reactive streams (Micrometer emitter)
        - RestApplication→REST endpoint (delete unnecessary code; change entire endpoint)
4. Start Quarkus in dev mode
    - Console works OOB, POM generated, bunch of extensions are migrated
    - Application runs locally (for further finetuning/testing)!

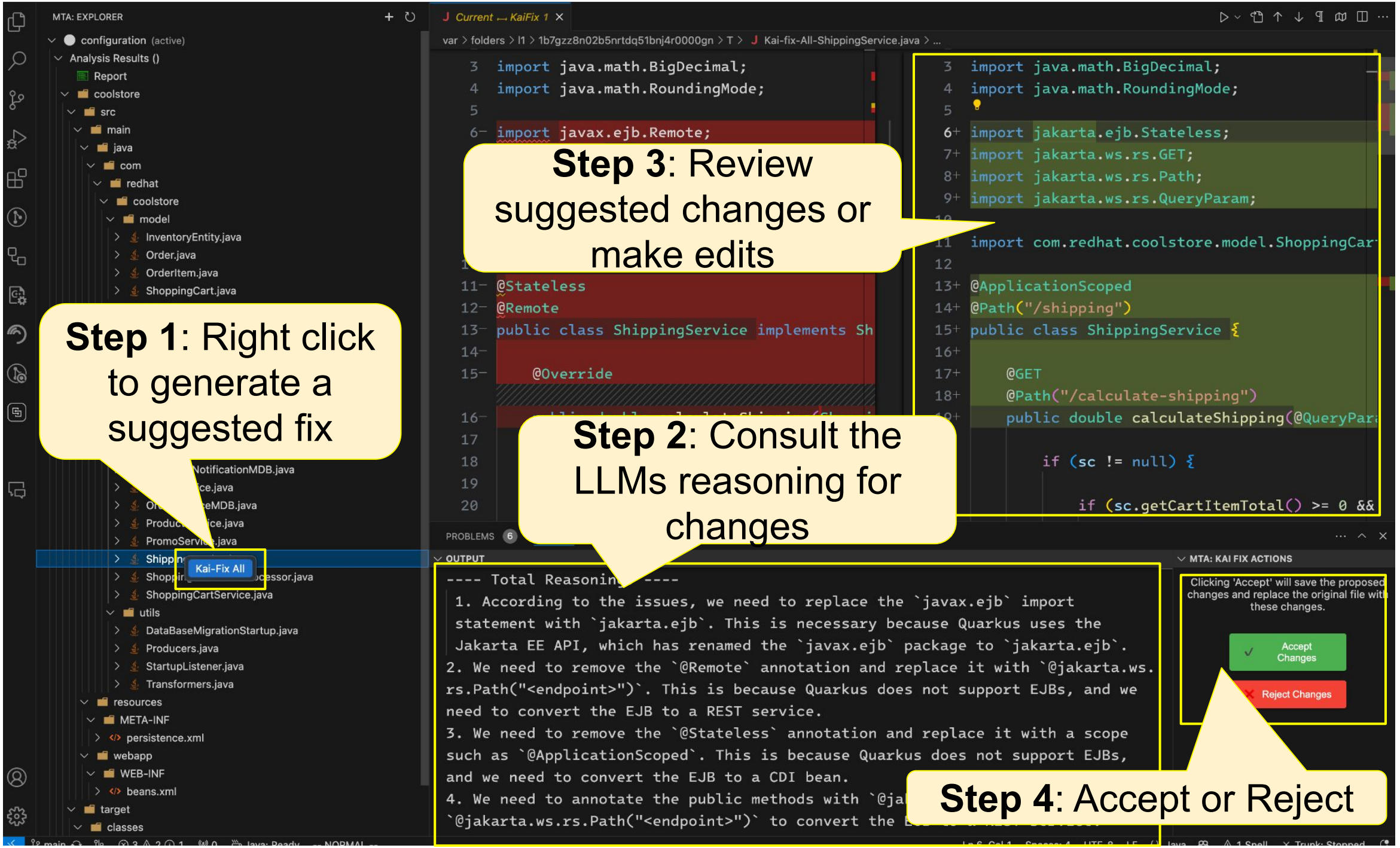
Red Hat

* https://github.com/konveyor-ecosystem/kai-vscode-plugin

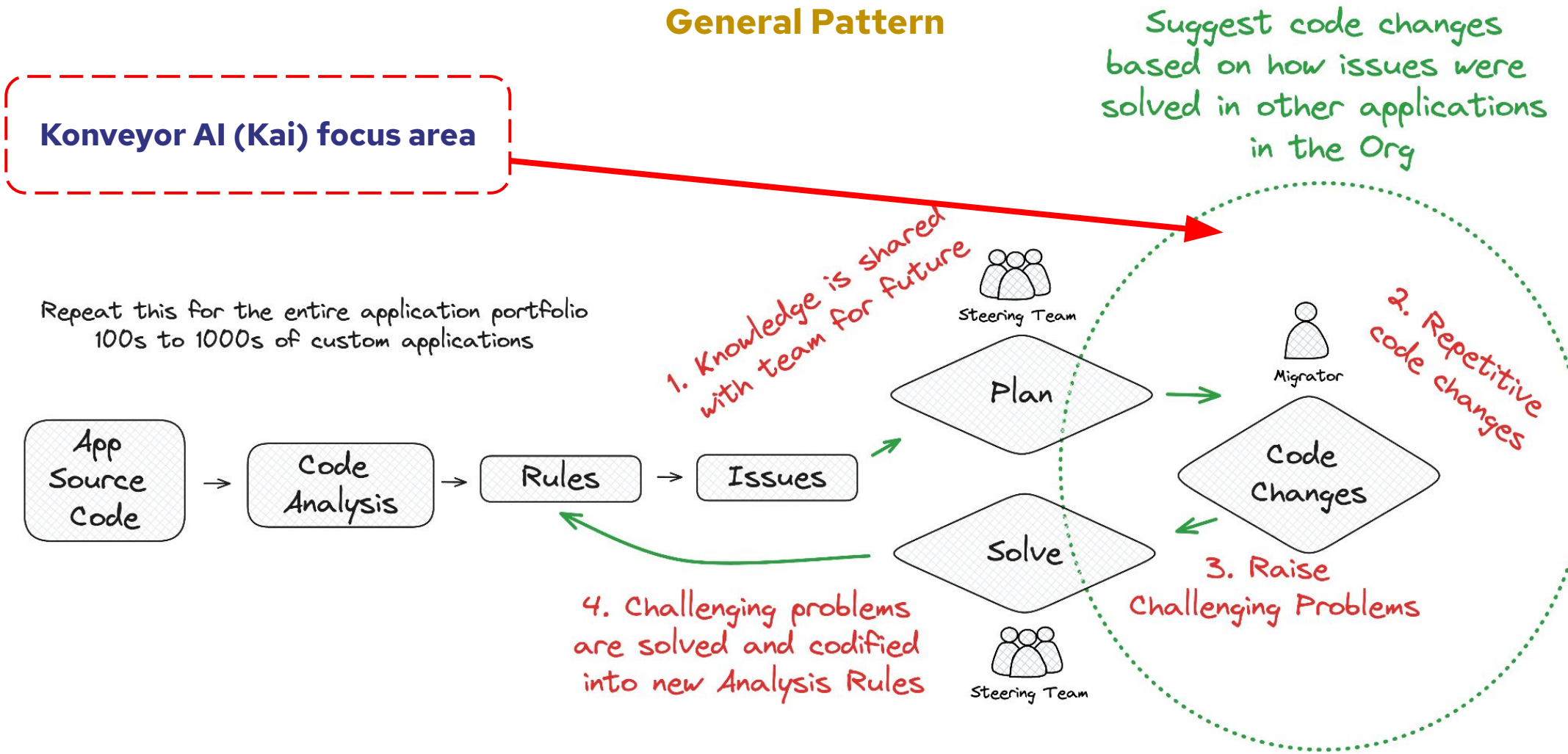

Accelerating Modernization with AI
Konveyor AI + Konveyor is static code analysis


MTA: EXPLORER
configuration (active)
Analysis Results ()
Report
coolstore
src
main
java
com
java
redhat
coolstore
resources
webapp
target
pom.xml
Run Konveyor Analysis from within IDE
configuration
--target *
The target technology to consider for analysis.
azure-appservice
camel 3
camel 4
containerization
eap
eap7
eap8
jakarta-ee
jakarta-ee8+
jakarta-ee9+
jws6
linux
openjdk
openjdk11
openjdk17
openjdk21
openliberty
quarkus
springboot
Add
--source
The source technology to consider for analysis.
--rules
User Rules Directory/File
/Users/jmatthews/git/jwmatthews/kai/samples/custom_rules
Add
--source-only
Indicates whether the input is source code or compiled binaries.
--overwrite
If set, overwrite the output directory.


MTA: EXPLORER
configuration (active)
Analysis Results ()
coolstore
src
main
java
com
persistence
Resources.java
rest
CartEndpoint.java
OrderEndpoint.java
ProductEndpoint.java
RestApplication.java
service
CatalogService.java
InventoryNotificationMDB.java
OrderService.java
OrderServiceMDB.java
ProductService.java
PromoService.java
ShippingService.java
Incidents (4)
[line: 6] [rule: javax-to-jakarta-import-00001]
[line: 7] [rule: javax-to-jakarta-import-00001]
[line: 11] [rule: cdi-to-quarkus-00050]
[line: 12] [rule: remote-ejb-to-quarkus-00000]
ShoppingCartOrderProcessor.java
ShoppingCartService.java
utils
DataBaseMigrationStartup.java
Producers.java
StartupListener.java
Transformers.java
resources
META-INF
persistence.xml
webapp
WEB-INF
beans.xml
target
classes
META-INF
persistence.xml
ShippingService.java 4
src > main > java > com > redhat > coolstore > service > ShippingService.java > ShippingService
1 package com.redhat.coolstore.service;
3 import java.math.BigDecimal;
4 import java.math.RoundingMode;
6 import javax.ejb.Remote;
7 import javax.ejb.Stateless;
9 import com.redhat.coolstore.model.ShoppingCa
Chris May, last month | 2 authors (Chris May and oth
11 @Stateless
12 @Remote
Chris May, month • Convert ShippingService from Local to Remote EJ…
ShippingService.java 3 of 5 problems
Remote EJBs are not supported in Quarkus, and therefore its use must be removed and replaced with REST fu
1. Replace the `@Remote` annotation on the class with a `@jakarta.ws.rs.Path("<endpoint>")` annotation.
2. Remove `@Stateless` annotations if present. Given that REST services are stateless by nature, it make
3. For every public method on the EJB being converted, do the following:
- Annotate the method with `@jakarta.ws.rs.GET`
- Annotate the method with `@jakarta.ws.rs.Path("<endpoint>")` and give it a proper endpoint path. As a
```
@Path("/increment")
public void increment()
```
- Add `@jakarta.ws.rs.QueryParam("<param-name>
13 public class ShippingService implements Shippir
15 @Override
16 public double calculateShipping(ShoppingCar
18 if (sc != null) {
20 if (sc.getCartItemTotal() >= 0 && sc.getCartItemTotal() < 25) {
Konveyor Analysis Issue Details in the IDE
Info from each incident is used to resolve the issue.
These ‘hints’ are passed to the LLM so it knows exactly what we need to do to resolve the issue


MTA: EXPLORER
configuration (active)
Analysis Results ()
Report
coolstore
src
main
java
com
redhat
coolstore
model
InventoryEntity.java
Order.java
OrderItem.java
ShoppingCart.java
Kai-Fix All
ShoppingCartService.java
utils
DataBaseMigrationStartup.java
Producers.java
StartupListener.java
Transformers.java
resources
META-INF
persistence.xml
webapp
WEB-INF
beans.xml
target
classes
Current ⟷ KaiFix 1
var > folders > l1 > 1b7gzz8n02b5nrtdq51bnj4r0000gn > T > Kai-fix-All-ShippingService.java > ...
import java.math.BigDecimal;
import java.math.RoundingMode;
import javax.ejb.Remote;
@Stateless
@Remote
public class ShippingService implements Sh
@Override
import jakarta.ejb.Stateless;
import jakarta.ws.rs.GET;
import jakarta.ws.rs.Path;
import jakarta.ws.rs.QueryParam;
import com.redhat.coolstore.model.ShoppingCar
@ApplicationScoped
@Path("/shipping")
public class ShippingService {
@GET
@Path("/calculate-shipping")
public double calculateShipping(@QueryPar
if (sc != null) {
if (sc.getCartItemTotal() >= 0 &&
PROBLEMS 6
OUTPUT
---- Total Reasonin ----
1. According to the issues, we need to replace the `javax.ejb` import statement with `jakarta.ejb`. This is necessary because Quarkus uses the Jakarta EE API, which has renamed the `javax.ejb` package to `jakarta.ejb`.
2. We need to remove the `@Remote` annotation and replace it with `@jakarta.ws.rs.Path("<endpoint>")`. This is because Quarkus does not support EJBs, and we need to convert the EJB to a REST service.
3. We need to remove the `@Stateless` annotation and replace it with a scope such as `@ApplicationScoped`. This is because Quarkus does not support EJBs, and we need to convert the EJB to a CDI bean.
4. We need to annotate the public methods with `@ja
`@jakarta.ws.rs.Path("<endpoint>")` to convert the
MTA: KAI FIX ACTIONS
Clicking 'Accept' will save the proposed changes and replace the original file with these changes.
Accept Changes
Reject Changes
Step 1: Right click to generate a suggested fix
Step 2: Consult the LLMs reasoning for changes
Step 3: Review suggested changes or make edits
Step 4: Accept or Reject

Modernization Engagement running at Scale



Red Hat

Large Language Model (LLM)

Concerns/Limitations

- **Limited context windows**
  - It's not feasible to include an entire applications full source code in a prompt
    - Generally models have a limited context window
    - Costs increase as more data is included in a request
- **Desire to augment a LLM's knowledge without fine-tuning**
  - Corporate internal frameworks are generally not part of an existing models 'training data'
  - We want to leverage a RAG approach to include 'solved examples' in the prompt to help shape the results
- **Integrate with multiple models**
  - Rapid advancements on models increases the desire for flexibility to explore new models as they become available

Red Hat

Migration

What does “Kai” do and why. What it does not.

- Refactor
  - Migrate JavaEE to JakartaEE and Quarkus
  - Migrate JMS to Reactive (MicroProfile)
  - Migrate EJB to REST
  - Remove unnecessary code, no longer required.

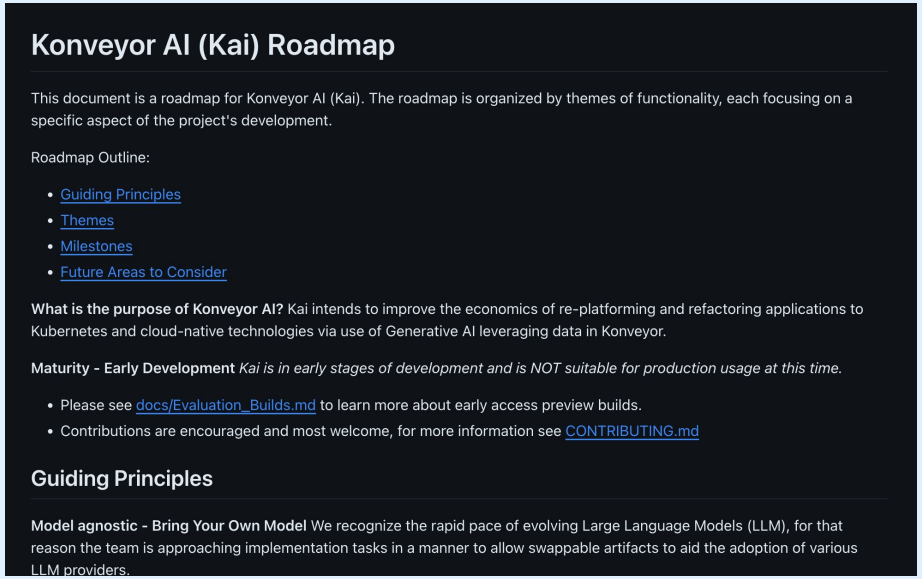
- Re-Architecture - **NO!**
  - Separation of Concern
  - Domain-Driven Design
  - Functional, Event-Driven, Serverless...

- Process
  - Automation
  - DevOps

Red Hat

Upstream Roadmap:

https://github.com/konveyor/kai/blob/main/ROADMAP.md



Konveyor AI (Kai) Roadmap

This document is a roadmap for Konveyor AI (Kai). The roadmap is organized by themes of functionality, each focusing on a specific aspect of the project's development.

Roadmap Outline:

- Guiding Principles
- Themes
- Milestones
- Future Areas to Consider

What is the purpose of Konveyor AI? Kai intends to improve the economics of re-platforming and refactoring applications to Kubernetes and cloud-native technologies via use of Generative AI leveraging data in Konveyor.

Maturity - Early Development *Kai is in early stages of development and is NOT suitable for production usage at this time.*

- Please see docs/Evaluation_Builds.md to learn more about early access preview builds.
- Contributions are encouraged and most welcome, for more information see CONTRIBUTING.md

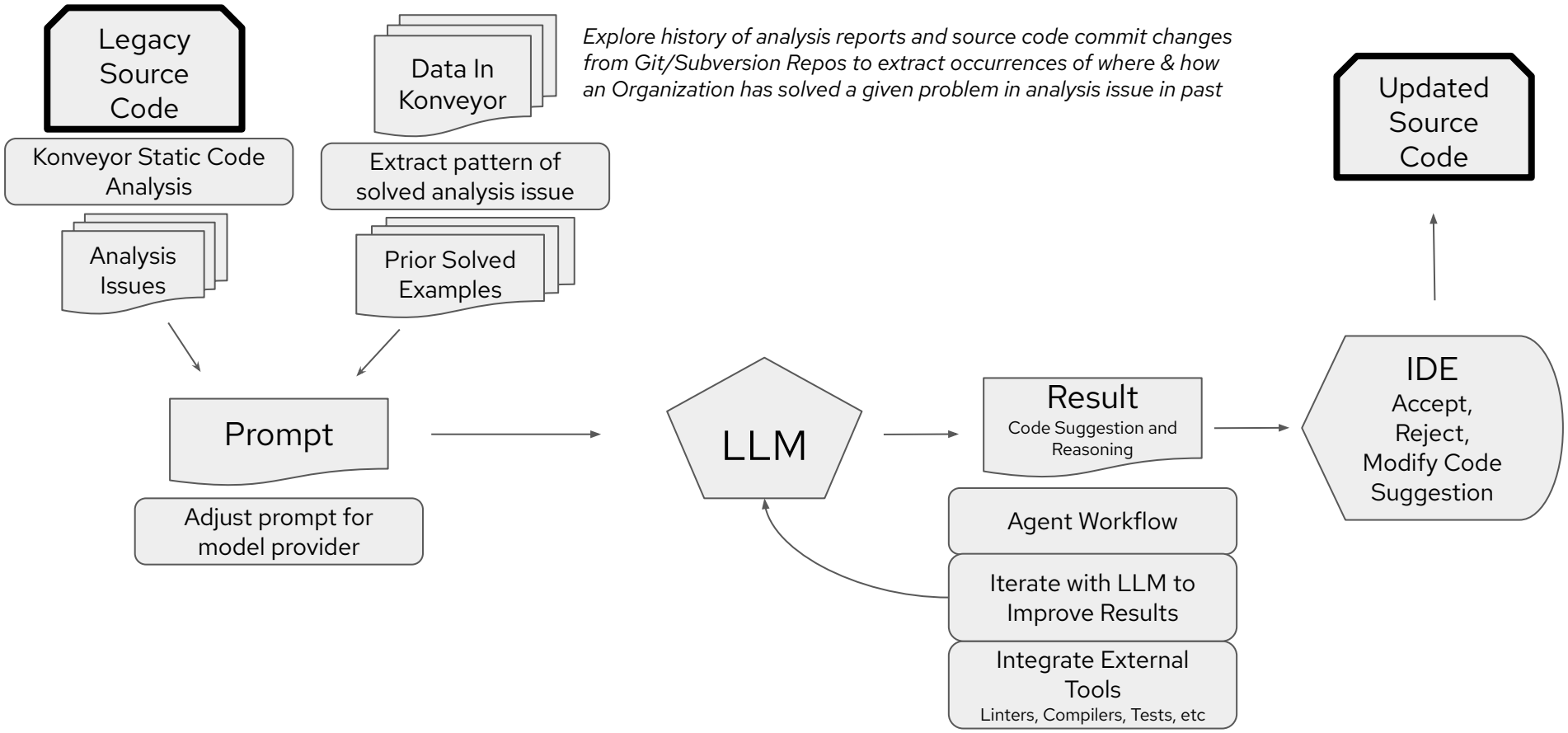
Guiding Principles

Model agnostic - Bring Your Own Model We recognize the rapid pace of evolving Large Language Models (LLM), for that reason the team is approaching implementation tasks in a manner to allow swappable artifacts to aid the adoption of various LLM providers.

Red Hat

Roadmap

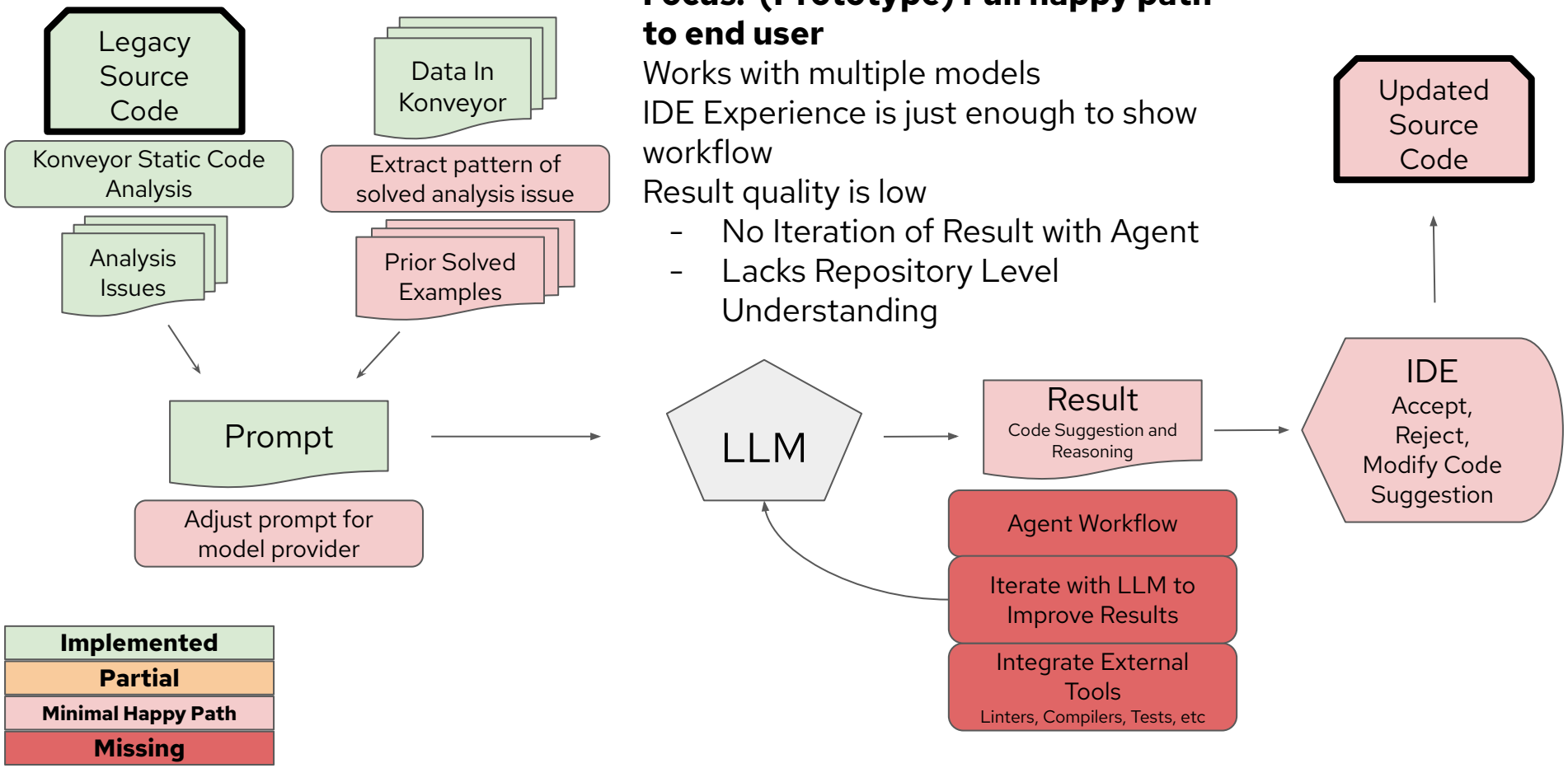
Vision



Red Hat

Roadmap

Implemented as of Oct 2024 (before refactoring)



Focus: (Prototype) Full happy path to end user

Works with multiple models

IDE Experience is just enough to show workflow

Result quality is low

- No Iteration of Result with Agent
- Lacks Repository Level Understanding

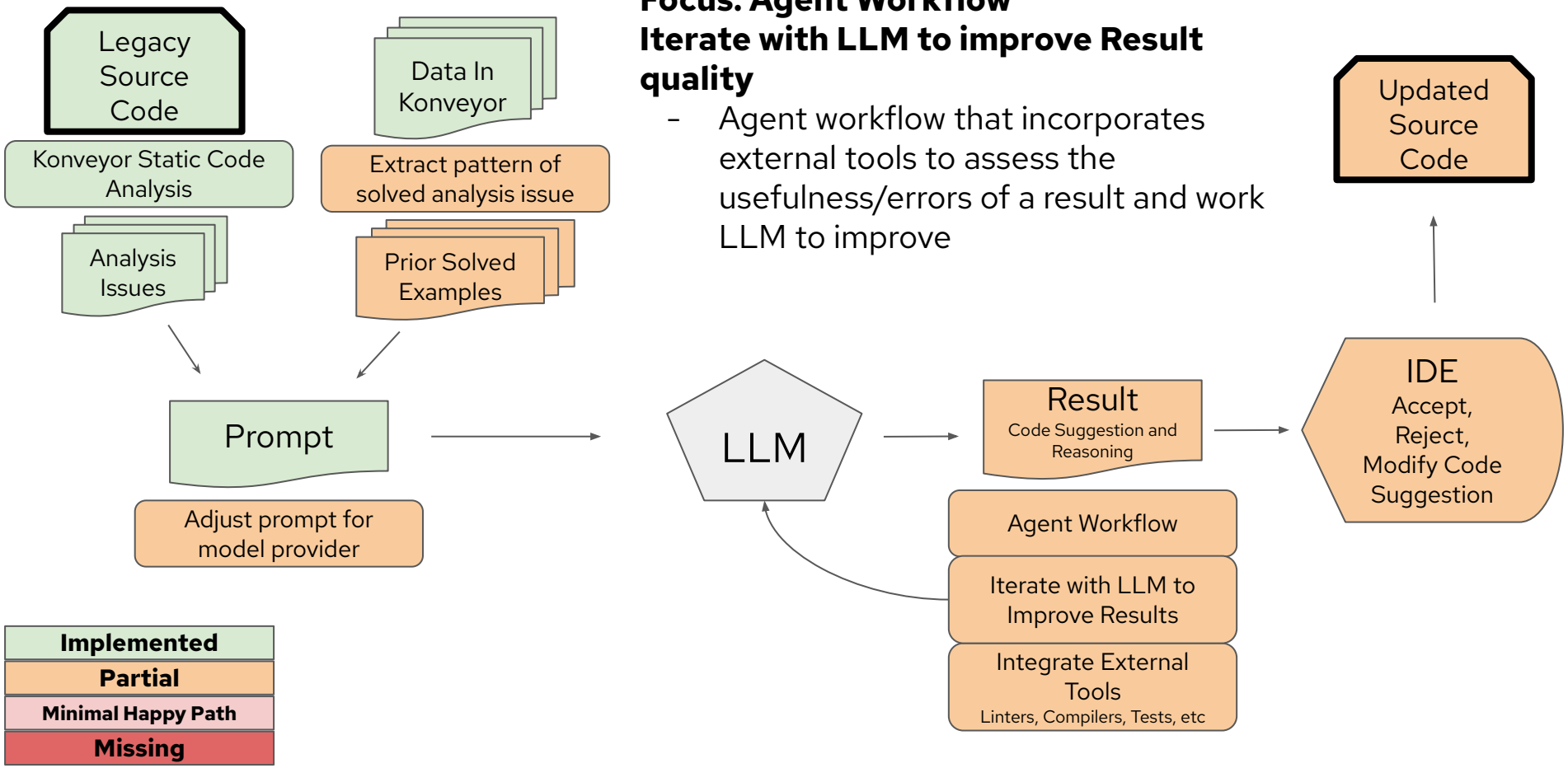
Red Hat

Roadmap

Late Q4 2024

Focus: Agent Workflow
Iterate with LLM to improve Result quality

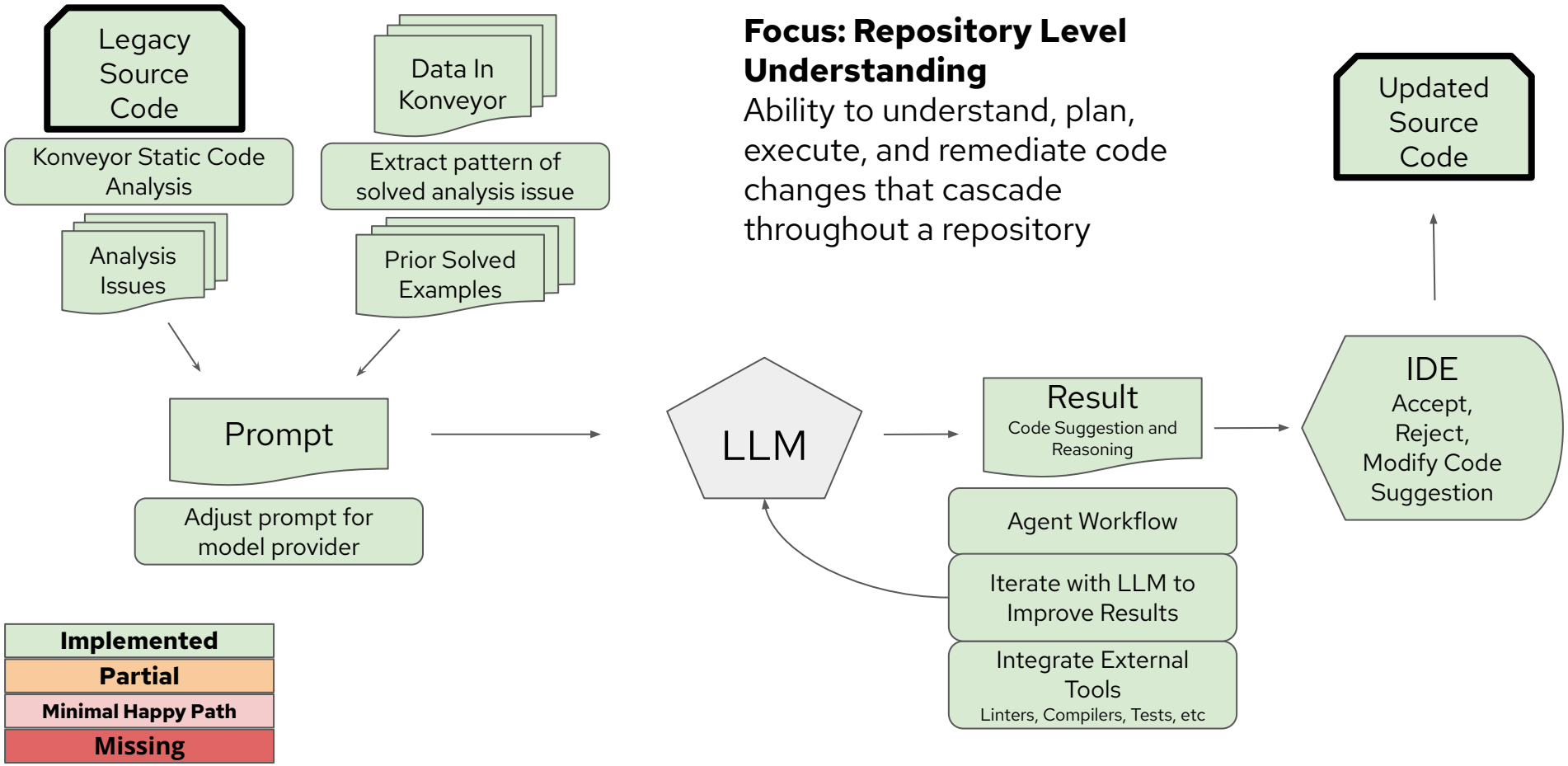
- Agent workflow that incorporates external tools to assess the usefulness/errors of a result and work LLM to improve



Red Hat

Roadmap

Q2 2025



Focus: Repository Level Understanding

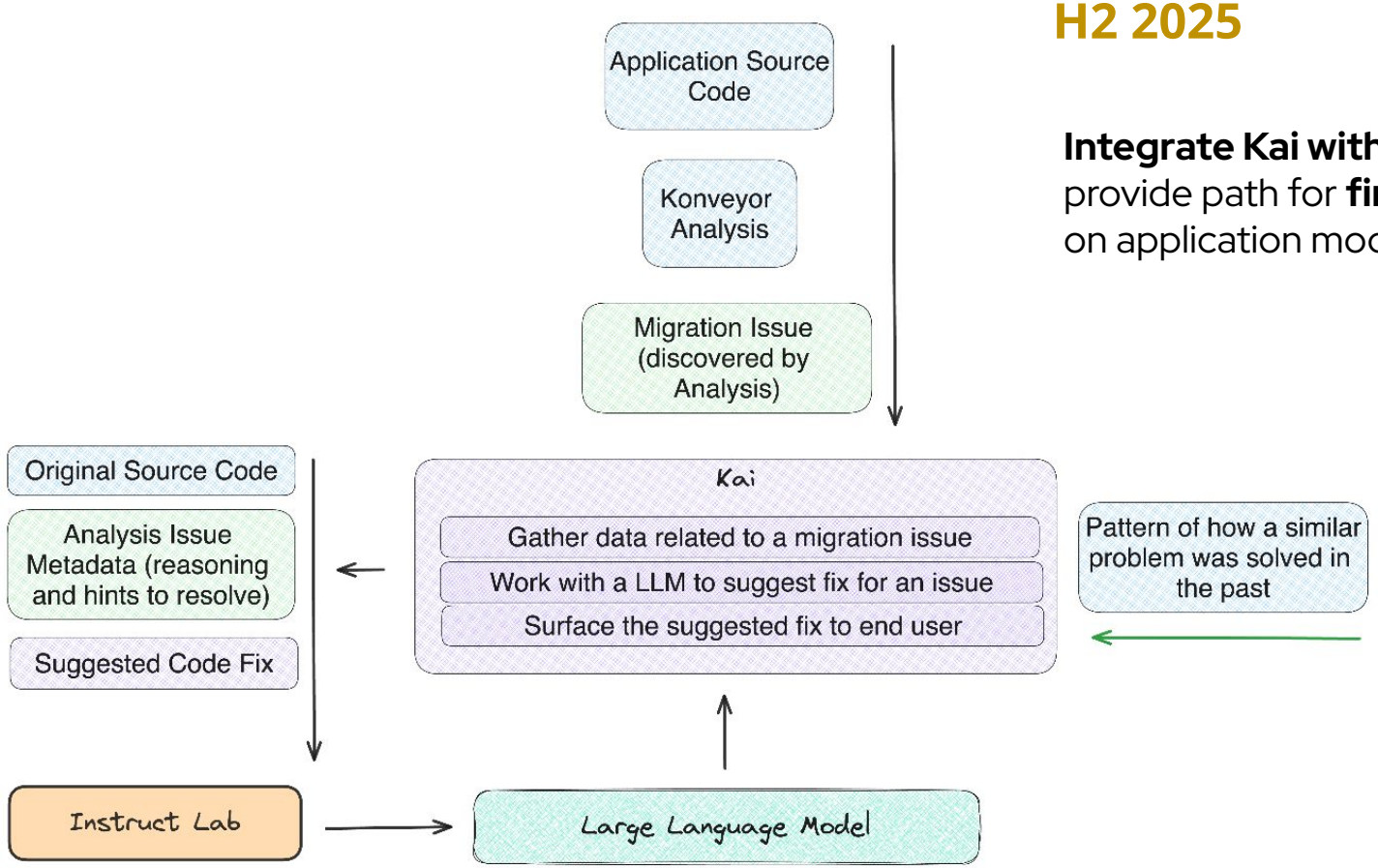
Ability to understand, plan, execute, and remediate code changes that cascade throughout a repository

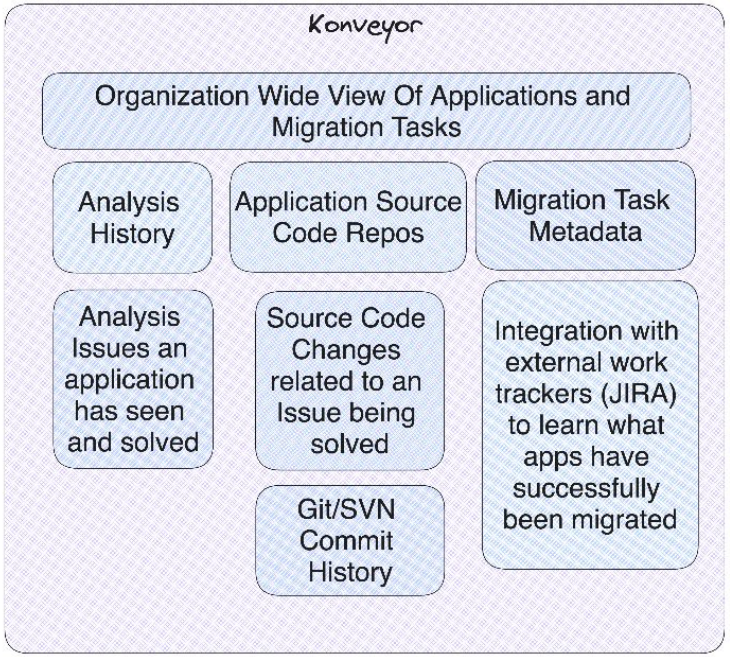
Red Hat

Roadmap

H2 2025

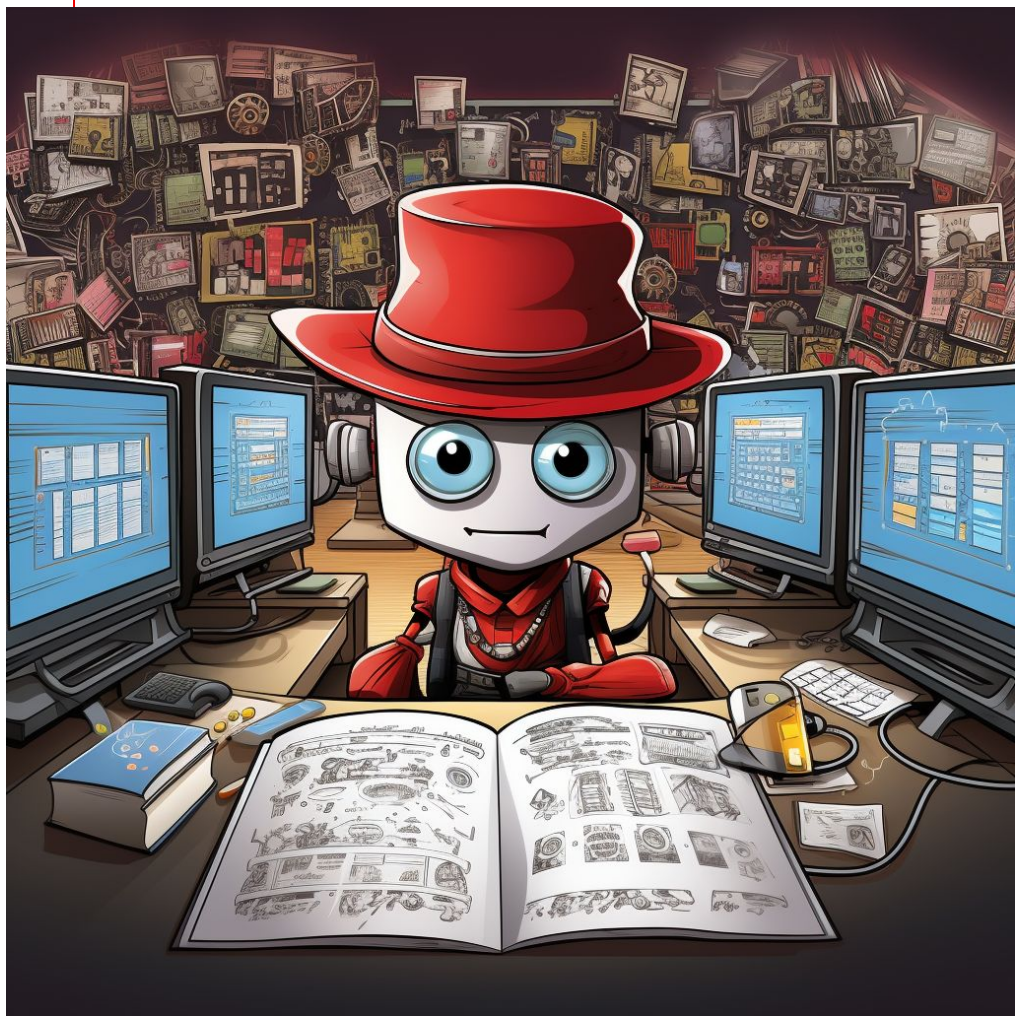
Integrate Kai with Instruct Lab to provide path for **fine-tuning models** on application modernization data





Training cycle to improve LLMs on App Modernization tasks
App Mod knowledge is mined from Konveyor and added to InstructLab's Knowledge base

Red Hat



Questions

Red Hat

Session: 15:50 - 16:20

red.ht/rhsc24-de-s6

Jetzt Session bewerten!

Einfach QR-Code scannen, Session wählen und bewerten.
Vielen Dank!

Red Hat

Red Hat Summit

Connect

Thank you

 linkedin.com/company/red-hat

 facebook.com/redhatinc

 youtube.com/user/RedHatVideos

 twitter.com/RedHat

Red Hat