

Agenda

Introductions

About Proton and Ionix AI

Product Overview

Demo

Q & A

The Enterprise Automation Challenge

Fragmented workflow intake

01

Business requests come from too many disconnected places, so teams struggle to understand, prioritize, and act on them consistently

Trapped enterprise knowledge

02

Critical business logic, automation scripts, process rules, data dependencies, and operational know-how remain buried across teams, tools, and repositories instead of becoming reusable enterprise skills.

Manual delta analysis and updates

03

Every enhancement, migration, policy update, or workflow change requires manual impact analysis, validation, updates, and review creating duplicated effort, delays, and inconsistent execution.

Limited execution intelligence

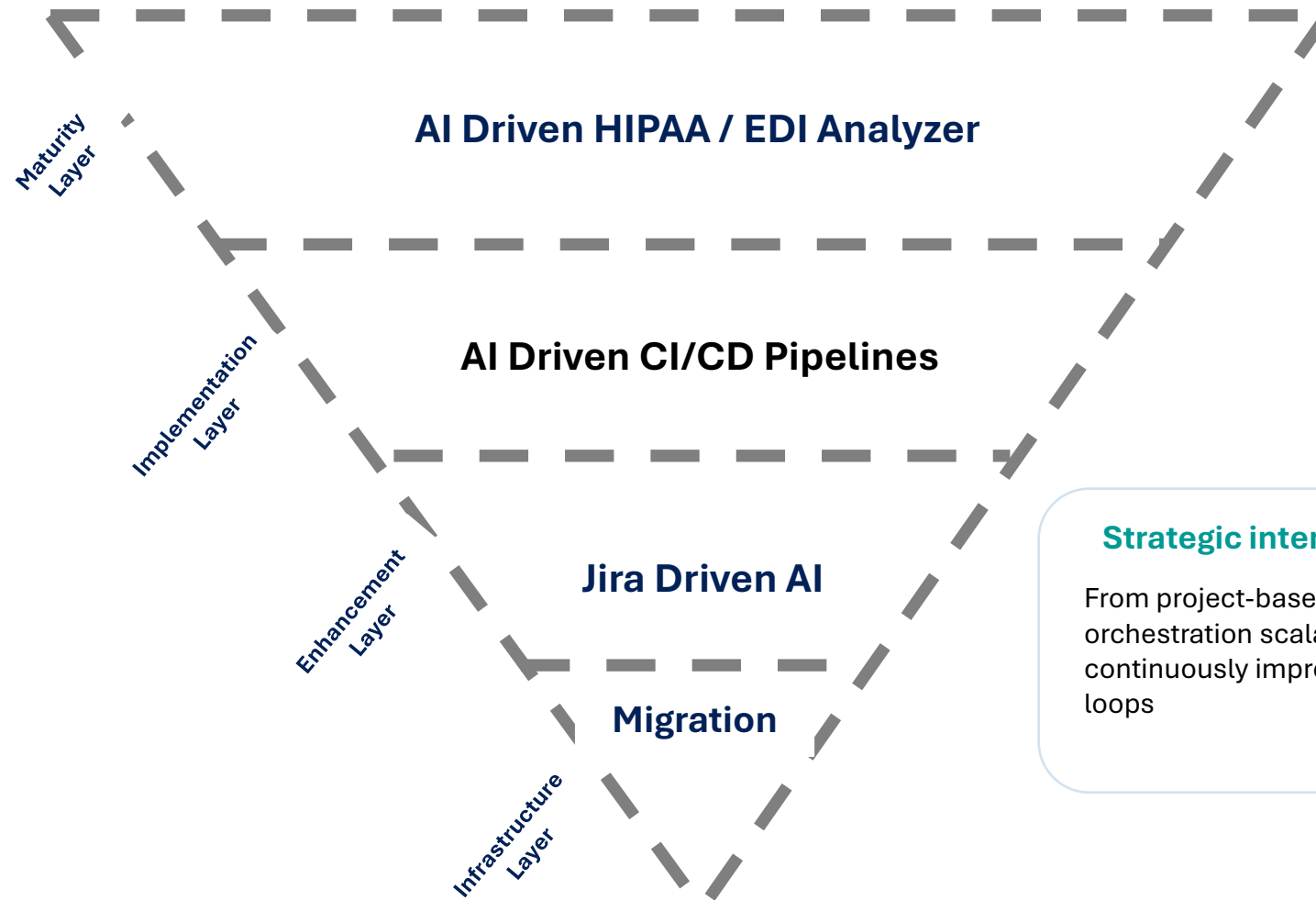
04

Execution outcomes, defects, exceptions, approvals, evidence, and optimization insights are not continuously fed back into future workflows, preventing automation from improving over time.

Business impact: slower delivery • higher operating cost • inconsistent governance • fragile automation • delayed modernization • missed AI ROI

IONIXAI WORKFLOW ORCHESTRATION

A layered operating model that brings autonomous workflow execution into migration, DevOps, EDI/HIPAA analysis, and production support.



Strategic intent

From project-based automation to governed orchestration scalable across Migration and EDI, continuously improving through AI skills and feedback loops

EDI Business Process Automation

IonixAI AI-Powered EDI Modernization Accelerator

EDI modernization is not only a file validation exercise — it is a business-process control and operational risk challenge.

The EDI operating risk

Current intake and validation flows rely on hard-coded checks, brittle matching, and manual review cycles.

- Every clean file still pauses for human review
- Validation failures show raw rule IDs with limited guidance
- Exact hash matching misses near-duplicate resubmissions

How IonixAI accelerates the EDI feedback loop

Core orchestration platform + EDI app accelerators



Register

Create correlation ID, fingerprint, lineage and file level execution state.



Infer

Use AI to classify partner, direction and ACK expectations from content.



Triage

Auto-approve clean files and escalate only exceptions or ambiguity.



Validate

Run deterministic X12/834 skills with rule IDs and evidence.



Explain

Generate root-cause summaries and partner-ready remediation



Learn

Use telemetry and history to detect recurrence and improve controls.

Value for EDI operations

- Exception-only human review
- Faster partner issue resolution
- Lower manual intake and triage effort
- Closed-loop ACK visibility
- Traceable rule, file, and decision history
- Reusable playbook across EDI partners

IonixAI turns EDI operations from manual file handling into a governed, AI-accelerated, self-learning automation factory.

AI accelerations: adaptive stability checks • partner inference • AI triage • failure explanation • near-duplicate detection • Splunk incident intelligence

Migration from Legacy to Modern Stack

IonixAI Agentic Migration Accelerator

Modernization is not only a technology upgrade—it is a business-process risk challenge.

The migration risk

Legacy platforms hold years of hidden business rules, exceptions, dependencies, and manual workarounds.

- Business workflows are hard to discover
- Downstream impacts are easy to miss
- Teams recreate knowledge manually
- Cutover risk increases without continuous validation

How IonixAI agents help

Discover

Understand current workflows, dependencies, and system behavior.

Convert

Turn tribal knowledge into reusable enterprise skills.

Adapt

Map current behavior to the modern stack and identify gaps.

Generate

Create migration-ready assets for validation and rollout.

Optimize

Update only changed scope instead of recreating everything.

Validate

Execute, self-correct, and improve with traceable outcomes.

Value for Customer

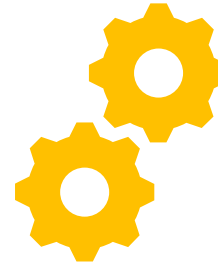
- Faster migration readiness
- Lower cutover and regression risk
- Reduced reliance on tribal knowledge
- Reusable playbook for future migrations
- Better traceability from requirement to outcome
- Scalable modernization across enterprise applications

IonixAI turns legacy modernization from a manual migration effort into a governed, repeatable, agentic migration factory



IonixAI should be able to:

- Read and understand the existing Business Partner creation process within Salesforce
- Analyze and interpret the existing automation script and its business intent
- Execute the test process based on the provided script steps
- Understand the enhancement story and identify impacted process areas
- Autonomously optimize and improve the existing test script
- Generate updated test scenarios aligned to the enhancement requirements
- Execute enhanced regression and integration testing workflows



Process Flow in IonixAI

- Ingest application, environment, and authentication details
- Read and analyze existing test automation scripts
- Understand the business process flow and testing intent
- Ingest enhancement stories from Word documents, Jira, or ServiceNow
- Identify impacted business processes and testing scenarios
- Automatically update and optimize test scripts
- Generate additional unit and integration test cases where required
- Execute automated test scenarios in the target environment
- Provide execution reports, defect insights, and optimization recommendations
- Continuously learn and improve test efficiency for future enhancements

Requirement Agent
Understand the enhancement story and identify impacted process areas

The screenshot shows a web browser window with the following elements:

- Browser Tabs:** "Agentic AI Platform" and "[TCM-231693] SalesforceLead...".
- Address Bar:** "Not secure uatkopqaapp02.corp.ybusa.net:3031/dashboard-2".
- Page Header:** "AI Assurance Test Execution Dashboard" with sub-headers "Requirement • Discovery • Test Execution • Auto-Heal".
- Input Field:** "Please Enter Jira Ticket ID".
- App Selection:** "Select app: Salesforce SAMI Client Dashboard" and a "Generate Test" button.
- Process Status Bar:** A row of eight cards, each with a title, an inception description, and a status: "Waiting".
 - Requirements (Inception: AC Analysis)
 - Recipe Matching (Inception: KB Patterns)
 - Scope Analysis (Inception: Sub-Agents)
 - PW Scanner (CDP Deep DOM Scan)
 - Code Generation (Java Framework)
 - Execute + Heal (Maven + Auto-Heal)
 - Push to Zephyr (Test Case Agent)
 - Done (Complete)
- Activity Log:** "LIVE ACTIVITY Waiting for pipeline".
- Main Content Area:** "AI SCANNER + TEST EXECUTION" with a sub-header "No page loaded" and a status "IDLE". Below this, it says "No test running — start a test to see live view".
- Log Panels:** "Pipeline Log" and "Test Execution Log", both showing "No pipeline activity yet".

Recipe Match Agent
Read and understand
the existing Business
Partner creation
process within
Salesforce

The screenshot shows a web browser window displaying the 'AI Assurance Test Execution Dashboard' for 'uatkopqaapp02.corp.ybusa.net:3031/dashboard-2'. The dashboard is for test case 'TCH-231693' and is currently in a 'Running...' state. A modal window titled 'Requirements — Complete' is open, showing a green checkmark and the text '✓ 20 acceptance criteria from TCM-231693'. Below this, a list of 13 acceptance criteria is displayed, each with a numbered description of user actions. A 'Recipe Matching' button is visible at the bottom right of the modal.

Requirements — Complete

✓ 20 acceptance criteria from TCM-231693

Salesforce:Leadconversion

1. AC-1: User opens App Launcher and selects "Sales Console"
2. AC-2: User clicks "Show Navigation Menu" and selects "Leads"
3. AC-3: User clicks the "New" button on the Leads list view
4. AC-4: When a record type dialog appears, user clicks "Next"
5. AC-5: User fills "Company" field with a generated company name
6. AC-6: User selects "Mr." from "Salutation" dropdown
7. AC-7: User fills "First Name" field with a test first name
8. AC-8: User fills "Last Name" field with a generated last name
9. AC-9: User selects "Customer Referral" from "Lead Source" dropdown
10. AC-10: User fills "Contact Email" field with a generated email
11. AC-11: User fills "Company Email" field with a generated email
12. AC-12: User fills "Phone" field with a generated phone number
13. AC-13: User fills "Street" field with a generated street address

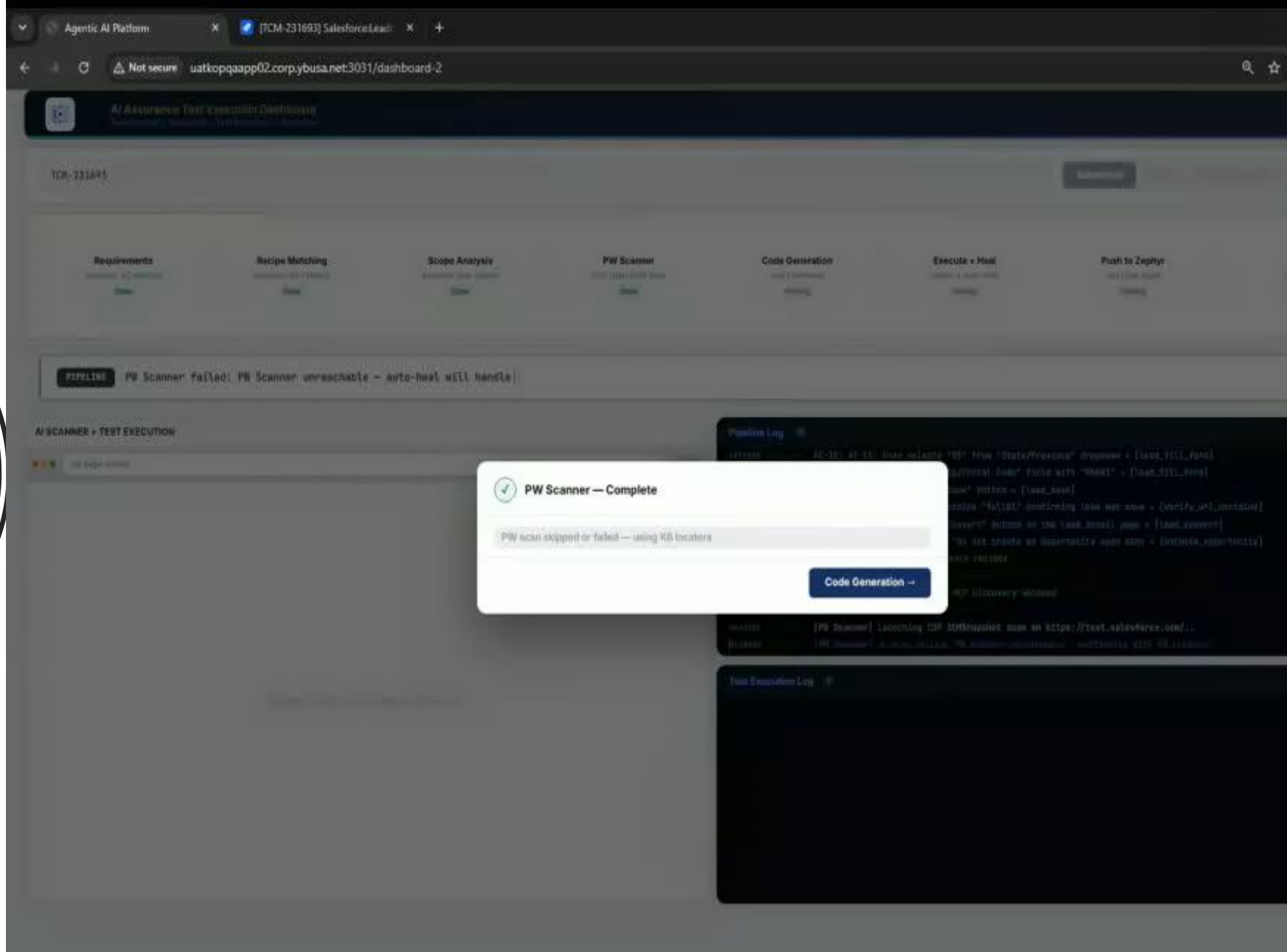
Recipe Matching →

Scope Analysis Agent
Analyze and
interpret the existing
automation script
and its business
intent

The screenshot displays the 'AI Assurance Test Execution Dashboard' for a Salesforce lead. The dashboard includes a progress bar with stages: Requirements (Done), Recipe Matching (Done), Scope Analysis (Waiting), PW Scanner (Waiting), Code Generation (Waiting), Execute + Heal (Waiting), Push to Zephyr (Waiting), and Done (Waiting). A modal window titled 'Recipe Matching - Complete' is open, listing 20 acceptance criteria (AC-6 to AC-20) such as 'lead fill form', 'lead save', 'verify url contains', and 'lead convert'. A 'Scope Analysis -' button is located at the bottom right of the modal. The background shows an 'INCEPTION: DEDUP AGENT' notification and a code editor with automation script snippets.

Code Generation Agent

Execute the test process based on the provided script steps



Test Execution Agent
Autonomously optimize and improve the existing test script

The screenshot shows the Agentic AI Platform interface. At the top, the browser address bar displays 'uatkopqaapp02.corp.ybusa.net:3031/dashboard-2'. The dashboard header includes 'AI Assurance Test Execution Dashboard' and a 'LIVE' indicator. Below the header, a progress bar shows the status of various pipeline stages: Requirements (Done), Recipe Matching (Done), Scope Analysis (Done), PW Scanner (Done), Code Generation (Done), Execute + Heal (Waiting), Push to Zephyr (Waiting), and Done (Waiting). A 'PIPELINE' notification states: 'Pipeline ready - Inception - Code Generation complete. 28 steps ready for execution!'. The main section is titled 'AI SCANNER + TEST EXECUTION' and shows a '1st page loaded' status. A modal window titled 'Code Generation - Complete' is overlaid, displaying a green checkmark and the following information: 'Auto_TCM_231693.java generated successfully', '20 steps | 0 framework methods | 0 locator keys', 'Atomic Recipes used: app launcher sales console, nav menu leads, new lead, record type next, lead fill form, lead save, verify url contains, lead convert, uncheck opportunity', 'No user journeys matched - new journey path generated', 'Delta: 19 covered | 1 resolved by Phase 2', and 'Phase 2 (Delta Scope Code Generation) applied - delta resolved'. An 'Execute Test' button is located at the bottom right of the modal. In the background, a terminal window shows the command to run the generated test script: 'Ready to run: java -cp [TCM_231693].java - @pom folder D:/Frameworks/Agent-Repo/UI Tool/UITool/src/main/java/com/nibu/automation/script/test/Auto_TCM_231693.java'.

CICD Agent
Execute enhanced regression and integration testing workflows

The screenshot displays the 'AI Assurance Test Execution Dashboard' for project TCM-249608. The dashboard features a progress bar with seven steps: 1. Requirements (Inception: AC Analysis, Done), 2. Recipe Matching (Inception: KB Patterns, Done), 3. Scope Analysis (Inception: Sub-Agents, Done), 4. PW Scanner (CDP Deep DOM Scan, Done), 5. Code Generation (Java Framework, Done), 6. Execute + Heal (Maven + Auto-Heal, Running... Process), and 7. Push to Zephyr (Test Case Agent, Waiting). A 'RUNNER' section indicates 'Executing test via Maven with auto-heal (up to 3 iterations)'. Below this, there is a 'Test Execution Log' window showing the following text: 'Java test ready - click Run Test to compile and execute via Maven', 'Launching Auto_TCM_249608.java via Maven...', and 'Maven compile + run launched (auto-heal enabled)...'. The dashboard also includes navigation buttons for 'Salesforce', 'SAMI', 'Client Dashboard', and 'Generate Test'.

From Business Requirements to Executable Tests -- Acceptance criteria and user stories are converted into validated test steps and automation-ready code, reducing manual design effort and accelerating delivery.

TestGen Agent

Generate updated test scenarios aligned to the enhancement requirements

The screenshot displays the 'AI Assurance Test Execution Dashboard' in a web browser. The dashboard features a header with the title and a subtitle: 'AI-powered test migration, live browser discovery, and self-healing test execution'. Below the header, there are five main sections, each with an icon, a title, and a brief description:

- Test Execution Dashboard** (Status: **UPGRADE**): Migrate Java/Selenium tests to Playwright with AI-powered discovery, locator generation, and auto-fix.
- Jira-Driven Test Generator** (Status: **ACTIVE**): Generate executable Playwright tests from Jira tickets — AI creates test steps from acceptance criteria, discovers live locators, and produces runnable code.
- Test Case Management** (Status: **ACTIVE**): AI-powered test case generation from Jira tickets — NLP analysis, acceptance criteria validation, and Zephyr integration.
- Agent Monitor** (Status: **UPGRADE**): Real-time health and status of all pipeline agents — Script Analyzer, MCP Discovery, Guide Gen, Auto-Fix.
- Artifact Factory** (Status: **UPGRADE**): Screenshots and recordings with step-by-step training prompts — create guided walkthroughs from test execution captures for user training.

| Capability | Most relevant video | Why it aligns |
|--|---|--|
| Ingest application, environment, and authentication details | Code Generation Agent- Slide 5 | Agent uses application configuration, framework paths, base classes, locator strategies, environment settings, and authentication patterns to create runnable tests. The Execution Agent video also supports this by showing tests running in the selected target environment. |
| Read and analyze existing test automation scripts | Recipe Match Agent- Slide 3 | The knowledge layer scans existing locator files, utility classes, workflow sequences, and proven automation patterns. It converts existing framework knowledge into reusable agent skills rather than rebuilding tests from scratch. |
| Understand the business process flow and testing intent | Recipe Match Agent- Slide 3 | This is the strongest video for business-process understanding. The agent interprets acceptance criteria, identifies the closest end-to-end journey, and connects the requirement to proven business workflows. The Scope Analysis Agent provides additional detail by decomposing criteria into actions, targets, outcomes, and validations. |
| Ingest enhancement stories from Word documents, Jira, or ServiceNow | Requirements Agent / Jira Ingestion- Slide 1 | This video demonstrates requirement ingestion and structured acceptance-criteria extraction. The current demo specifically shows Jira. Word and ServiceNow can be positioned as additional supported intake channels, but they are not visibly demonstrated in the current recording. |
| Identify impacted business processes and testing scenarios | Scope Analysis Agent- Slide 4 | The sub-agents analyze the validated requirements, identify the affected workflow, expand missing steps, and determine the complete testing scope. The Enhancement Test Scenario Generation video also reinforces the creation of enhancement-aligned scenarios. |

| Capability | Most relevant video | Why it aligns |
|--|---|--|
| Automatically update and optimize test scripts | Code Generation Agent- Slide 5 | This video directly supports the capability. Phase 1 creates vetted baseline code, delta analysis isolates uncovered enhancement scope, and the CLI applies only the required changes through reusable skills. This also demonstrates token optimization because the agent processes only the new delta instead of regenerating the full script. |
| Generate additional unit and integration test cases where required | TestcaseGen Agent- Slide 8 | This video is most relevant because it shows new test scenarios being generated from enhancement requirements and coverage gaps. However, the current demos mainly emphasize functional and end-to-end scenarios. A separate overlay or recording would strengthen the specific claim around unit and integration testing. |
| Execute automated test scenarios in the target environment | Execution and Auto-Healing Agent- Slide 6 | This video shows generated tests being executed in the live target environment, including orchestration, monitoring, parallel test pods, and completion tracking. |
| Provide execution reports, defect insights, and optimization recommendations | Execution and Auto-Healing Agent- Slide 6 & 7 | The video shows execution status, logs, pass/fail results, failure analysis, healing activity, and publishing results to the enterprise test repository—in the demo, Zephyr. It is the strongest proof point for reporting and defect insights. |
| Continuously learn and improve test efficiency for future enhancements | Execution and Auto-Healing Agent- Slide 6 & 7 | Auto-healed locators and corrected interactions improve future execution reliability. The Recipe Match Agent also supports this message because validated workflows and reusable knowledge patterns become available for future enhancements. |