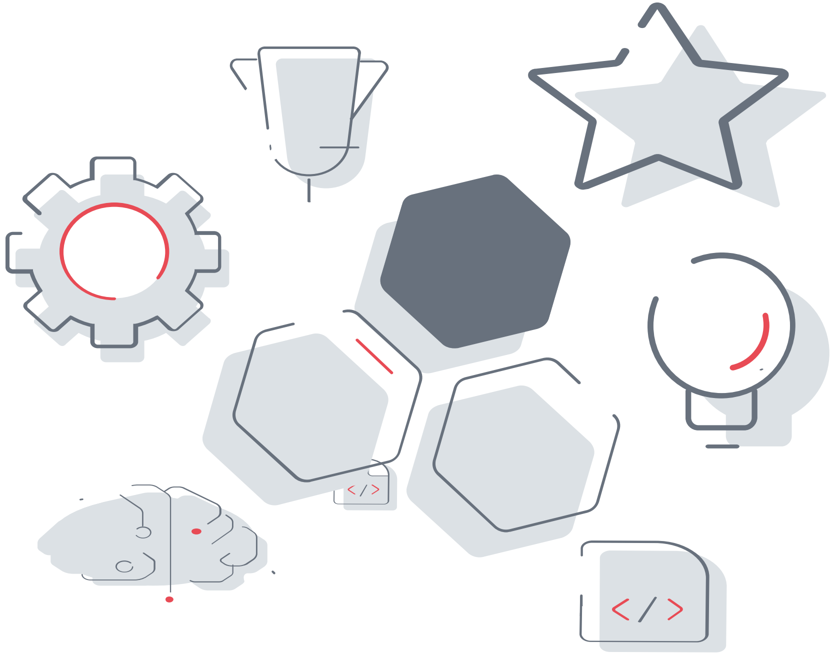
Accionlabs

QUALITY ENGINEERING PRACTICE - AN OVERVIEW

AGENDA

- INTRODUCTIONS
- QE PRACTICE - OVERVIEW
- AUTOMATION TESTING - TOOL STACK
- AI FOR QUALITY ENGINEERING
- METRICS
- CASE STUDIES
- Q & A
- NEXT STEPS

AN OVERVIEW OF THE QE PRACTICE



QUALITY ENGINEERING PRACTICE - OVERVIEW

Quality Engineers across All Geos

1000+

100+

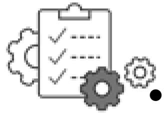
Total No of Customers engaged with QE teams

- All projects involve QA - Manual & Automation/SDETs
- QE Integrated with CI/CD & Release Cycles
- White box approach; all QA teams follow agile

200+

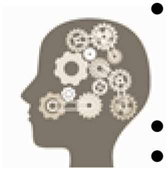
Total No of Projects Inflight

- Automation of legacy projects
- New product development
- Re-engineering products



Automation Testing

- Test Automation Frameworks
  - GenAI Based Automation
  - Keyword & Data Driven
  - Page Object Model
  - Low Code/No Code Approach
- Hybrid Framework (Web & Mobile)
- Data Quality
- Cloud Testing
- Continuous Testing



Functional Testing

- System Testing
- Mobile
- Integration Testing
- Device/Browser Compatibility
- DB Testing (Data Quality, Reports, etc)

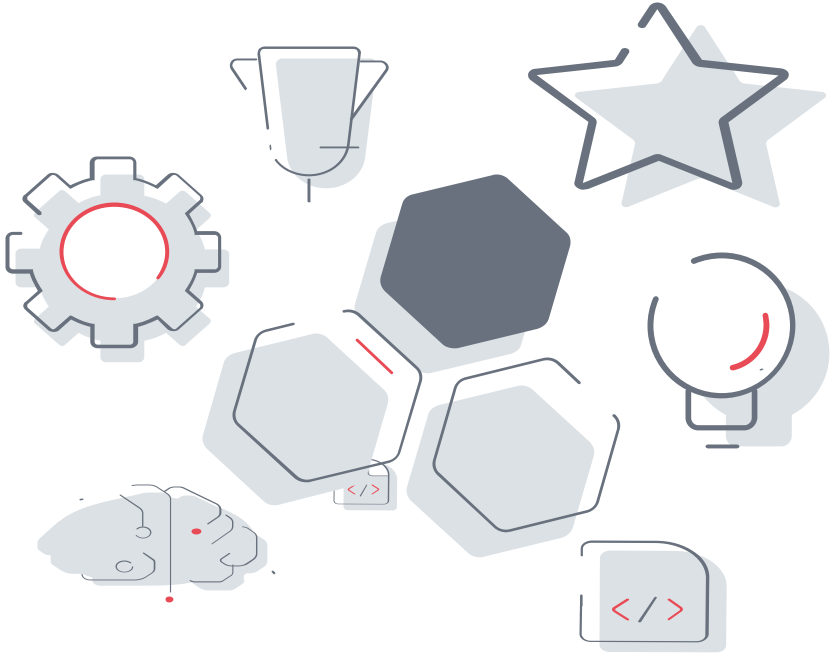
Non Functional Testing

- Usability
- Accessibility
- Security
- Performance
- Load
- Endurance
- Volume

QUALITY ENGINEERING COE

Development Cycle

Release Cycle



QUALITY ENGINEERING CENTER OF EXCELLENCE - FOCUS AREAS

Customer Value delivered

- Customer centric culture
- Shift left: reducing UAT, field defects
- Accelerate release cycles

Delivery Excellence

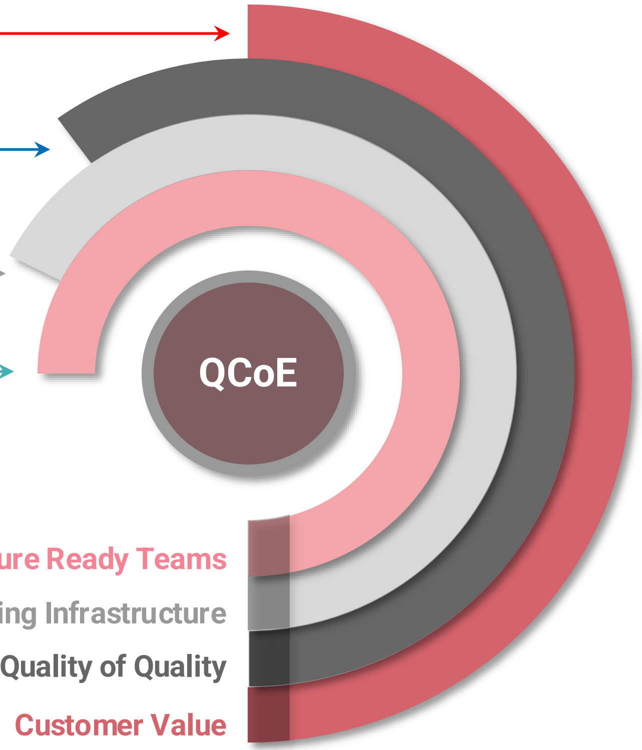
- Quantitative analysis driving process improvements
- Introducing state of the art frameworks, tools, best practices from the industry & across projects

Build world-class Testing Infrastructure

- Solutions
- Proof of Concept
- Partnership with test tool vendors

Creating future ready teams

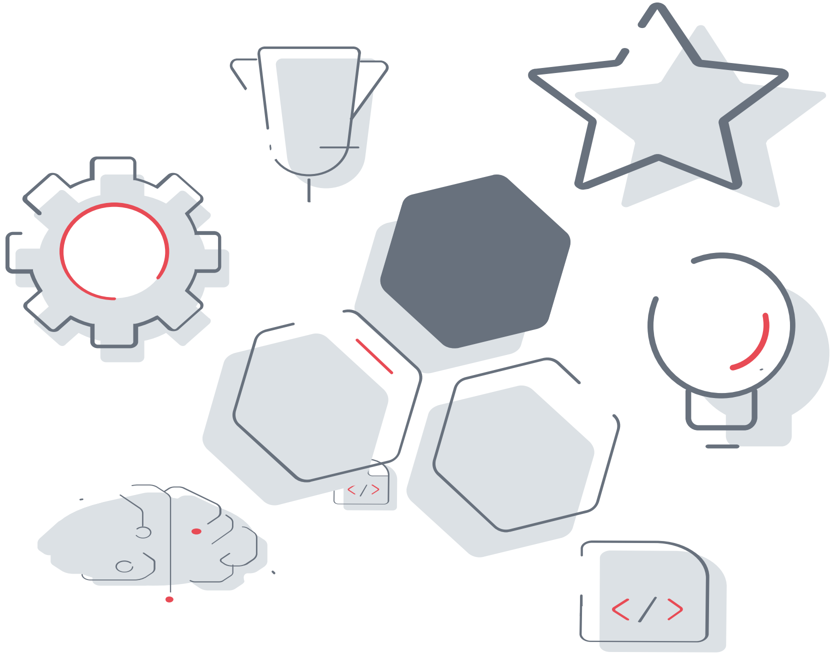
- Identify and groom SMEs in the team
- Train team members to meet future needs
- Identify, mentor SMEs & team leads



AUTOMATION TESTING - TOOL STACK

Development Cycle

Release Cycle



AUTOMATION TESTING TOOL STACK

Deep experience across Functional Testing, Performance testing using a variety of tools

Functional Automation	Web Applications	BDD, Cypress Protractor, Jasmine	Webdriver.io, Behave, Gherkin, Specflow	Test Complete	Selenium	Playwright, Cypress	Testim.io, Test Sigma
	APIs and Non-UI Resources	TestNG	Rest Assured	Postman	LoadUI	Nexial	
	Mobile Applications	Appium	Calabash	Robot Framework	Robotium	Selendroid, Watir	Nexial
	Desktop Applications	QTP, RFT, Squish	Robot Framework	Robot Window Tester	ZapTest	TestComplete, UFT & LeanFT	Nexial
	Data & Legacy Applications	UFT, LeanFT	Robot Framework	TestComplete	Nexial	Visual Studio	MS Excel
Performance Testing (API, UI)		Loadrunner	Apache JMeter	Perfaccion, Apache Benchmark, Cloudwatch	NewRelic	WinRunner, Silkperformer	
Continuous Integration Continuous Testing		Bamboo	Maven, Gradle	Jenkins	Ansible	BuildBOT	
Test Management		QAComplete	Test Collab, Zephyr	Jira	Test Link, TestRail	TestFLO	

AI Driven Quality Engineering

STAKEHOLDER'S DILEMMA

Best Practices Followed

Agile Methodology

Best Practices for QE Implemented

Tests are Automated

Measures & Metrics in Place

Am I Getting The Value?

Common Challenges

SDLC Rework Bottlenecks

- Traditional handoffs cause misinterpretations & quality issues.
- Changes in UX or architecture require significant rework.

Growing Complexity

- Substance Complexity: Feature-rich applications.
- Dynamic Complexity: Rapidly evolving requirements.
- Psychological Complexity: Increasing cognitive load on users.

Existing Solutions & Their Limitations

- SAFe, Spotify, etc models add overhead without solving core communication gaps.

BREEZE QE - AI-DRIVEN QE CAPABILITIES

AI-Driven Test Generation

- Auto-generates manual and automation test from user stories
- Converts acceptance criteria into robust, automated UI & API test scripts.

Smart Performance & Load Testing

- AI-powered performance modeling and analysis for web, mobile, and APIs
- Detects anomalies and bottlenecks

Synthetic Data Generation

- Intelligent tools create test data
- Enhances fest coverage and ensure data adequacy

Healing of Tests

- Ease of managing changes to UI and APIs
- improves test stability including flaky tests

Auto Triage of Defects

- Use of ML Models to auto triage defects

QE Dashboards & Analytics

- Ease of integration with dashboarding solutions of choice

The Breeze QE Framework

- **Collaborative Foundation**
  - SME-driven design align with goals.
- **KPI-Driven Implementation**
  - Focus on key outcomes like test coverage, detect density, and time-to-market
- **Tailored Solutions**
  - AI based solutions tailored for each client's tech stack and domain for maximum impact
- **Continuous Improvement**
  - Feedback loops and learning models drive ongoing optimization

95% **Test Coverage**

80% **Faster Test Case Generation**

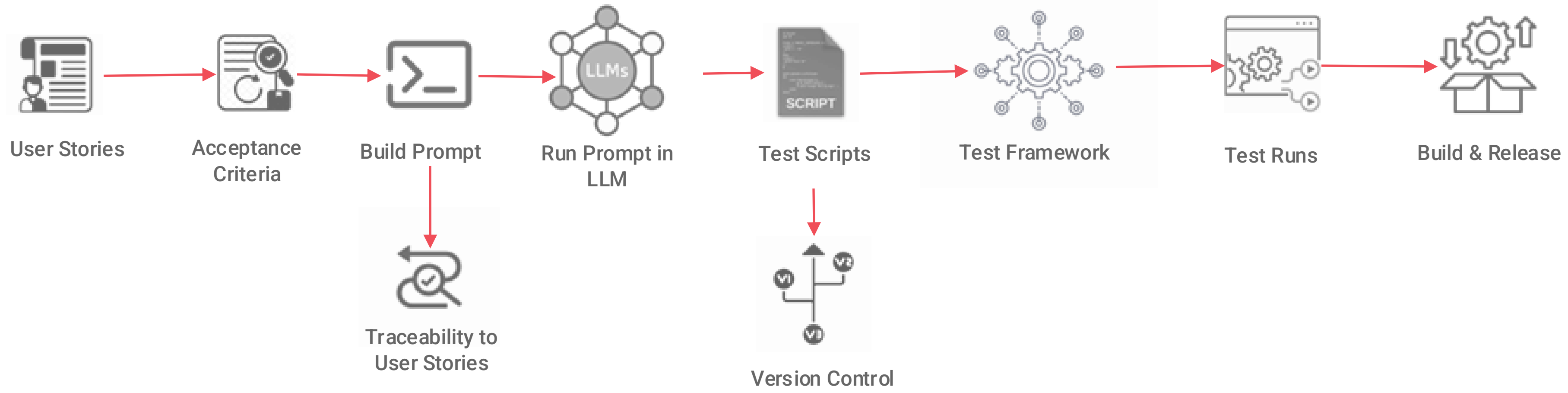
30% **Faster To Market**

BREEZE QE IN ACTION – CUSTOMER IMPLEMENTATIONS ACROSS DOMAINS

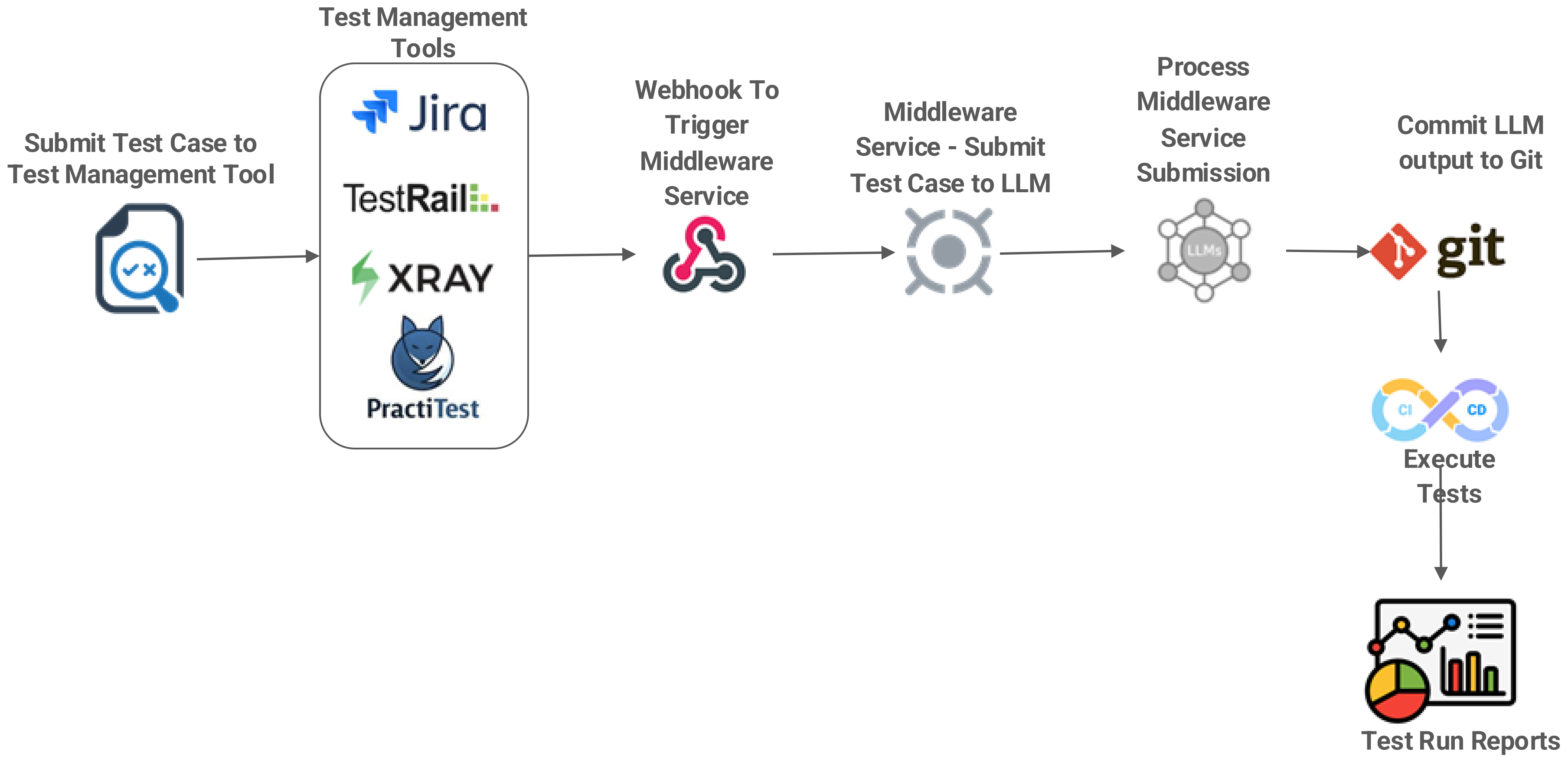
Trusted by Global Enterprises to Transform QE with AI. Below are Some of the Key Implementations

Domain	Scope of Testing	The Need	Key Outcomes
Home Services Platform	UI & API	• Large volume of test scenarios for automation (2000+) • Very short duration for release (2.5 months) • Budget constraints	95% test coverage, 30% faster release, 40% QA cost reduction
Supply Chain AI	UI, API, Edge Case Handling	• Weekly releases • Complex scenarios related to data	AI-driven test generation from user stories + ML-based defect triage. 80% reduction in test creation time, 95% coverage, 30% faster time-to-market
Digital Wealth Management	UI & Performance	• Complex UI and background process impacting performance of UI of investment dashboards	Derive UI & Performance script using one prompt. Easy and automated deployments Improved response time insights, 25% drop in critical defect leakage
Healthcare SaaS	End to End Test Automation	• Large volume of legacy tests • Required a full-stack QE setup incl. test healing, Jira integration, API validation	Healing tests, 70% reduction in triage time, real-time QA dashboards
Logistics Automation	API Test Automation	• Validation of APIs after reengineering and meet aggressive timelines for release • Swagger-based API test generation + synthetic data for tests	API regression test time reduced by 60%, improved coverage of negative scenarios
Education	UI & API Testing	• Needed improved test coverage while not taking away the focus of the QE team in while they focus on priority deliverables	95% test coverage, 40% faster release with ease in managing change, 30% QA cost reduction
Health Insurance	End to End Test	• Use of local LLMs for data protection and used for	Data protection, 70% reduction in QE

THE PROCESS ALIGNMENT



TECHNICAL IMPLEMENTATION


Submit Test Case to Test Management Tool
Test Management Tools
Jira
TestRail
XRAY
PractiTest
Webhook To Trigger Middleware Service
Middleware Service - Submit Test Case to LLM
Process Middleware Service Submission
LLMs
Commit LLM output to Git
git
CI
CD
Execute Tests
Test Run Reports

ENABLERS

Core Team / SMEs

Custom Copilots

QE SMEs are at the forefront of modernizing software quality engineering by combining their expertise with advanced AI tools to deliver innovative, efficient solutions.

- **Enhanced Quality**: SMEs bridge the gap between traditional QE expertise and AI-driven innovation, ensuring high-quality solutions.
- **Efficiency and Speed**: Tailored solutions accelerate test automation, defect resolution, and overall QE processes.
- **Domain Relevance**: SMEs' domain-specific knowledge ensures solutions align with client-specific goals and compliance requirements.
- **Scalability**: Solutions designed by SMEs are flexible and adaptable, catering to diverse industries and evolving client needs.

Customization Per Needs: Custom Copilots are built per customer specific requirements, ensuring relevance and precision.

LLM-Powered Test Generation: Automatically generates manual test cases from user stories, ensuring comprehensive coverage of functional requirements & Converts manual test scripts into automation scripts

ML-Based Defect Triage and Analysis Analyzes defect logs and historical data to prioritize and classify issues for faster resolution.

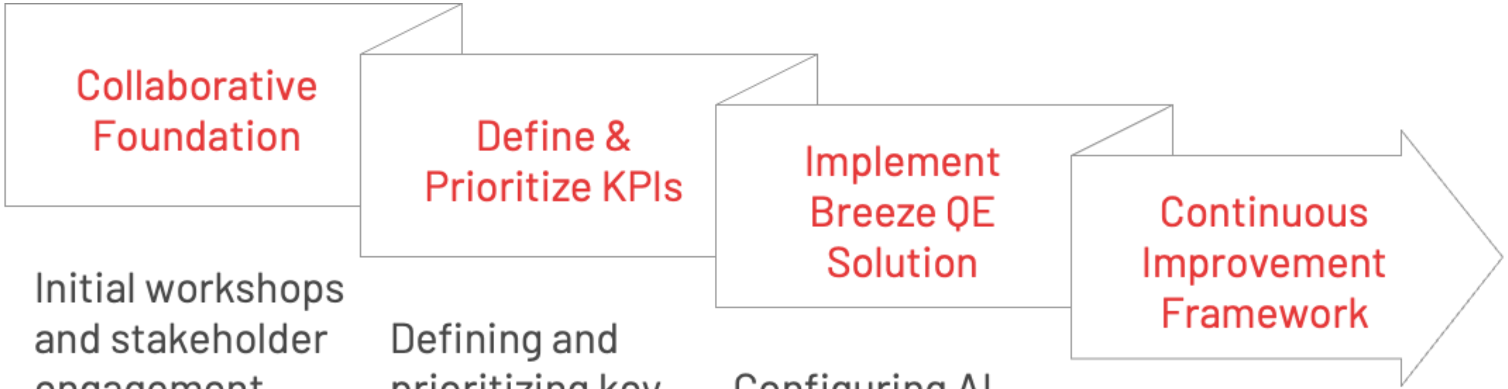
Benefits:
- Reduces test creation and maintenance effort with intelligent automation.
- Accelerates defect resolution by providing intelligent triaging and analysis.
- Aligns with client-specific goals to deliver maximum efficiency and quality improvements.

CAPABILITIES

- User Story & Test Scenario Generation
- Generation of UI & API Automation Scripts
- Synthetic Data Generation Tools
- Performance Test Builder & Analyzer
- Machine Learning Based QE Dashboards
- AI-Powered Static Code Analysis Tools
- Computer Vision based Test Case Generation

BREEZE QE IMPLEMENTATION APPROACH

Following are the various stages or phases involved in setting up Breeze QE


Collaborative Foundation
Define & Prioritize KPIs
Implement Breeze QE Solution
Continuous Improvement Framework

Initial workshops and stakeholder engagement

Defining and prioritizing key performance indicators

Configuring AI Agents per testing needs and setting up pipelines

Monitoring and optimizing testing processes

MAPPING BREEZE QE CAPABILITIES TO KPIs

S. No	Impact Areas	KPIs
1	**Test Efficiency**	• Test Case Creation Time • Automation Script Creation Time
2	**Test Coverage**	• Functional Test Coverage • Automation Coverage • Edge Case Coverage
3	**Execution**	• Execution Time per Build • Defect Detection Rate
4	**Quality of Quality**	• Defect Leakage Rate • Defect Density
5	**Productivity**	• Resource Efficiency • Cost Savings
6	**Reliability**	• Test Script Stability • Flaky Test Rate
7	**Continuous Improvement**	• Model Accuracy • Feedback Loop Effectiveness
8	**Business Impact**	• Time-to-Market Reduction • Customer Satisfaction

ROI BASED QE SERVICE

Scope For Test Automation

Functional Tests

Accessibility Tests

Localization Tests

Assessment
1 - 2 Month

- Consolidate manual tests
- Publish roadmap
- Review test automation strategy
- Categorize tests per complexity
- Package regression and smoke tests

Steady State
2 - 4 Months

- Finalize test strategy
- Prioritize test automation for the best ROI
- Build the automation test framework
- Automate tests (backlog & in-sprint)
- Focus on regression and smoke test packages
- Publish reports (dashboards)
- Review progress on a bi-weekly basis

Transform
3-6 Months

- Complete development of automation test suites
- Ongoing maintenance of test assets

Success Stories

CASE STUDY: JOBBER

Custom Copilots based Automating Web and Mobile apps using Webdriver.io and typescript

Customer Profile

- Jobber is an end-to-end business management system for home service company. The software handles everything from customer relationship management, to quoting, scheduling, job tracking, invoicing, and a whole lot more.
- It provides a platform for home services professionals to book customers and manage all of their workload around those jobs.

Challenges Faced

- There are web and mobile applications developed for android and iOS using react native to handle the complete workflow like service initiation, billing and payment.
- The key challenge was to develop the framework and automate 2000 manual test cases to run in CI/CD pipeline.
- All these activities required to complete in just 2 months

User Interface
React Native, GraphQL, MongoDB

Testing Tools
UI testing (Web & Mobile)-> Selenium, Appium, WebDriver.io, TypeScript

Cloud Device Lab for Desktop and Mobile Devices
Browesrstack

Custom Copilots
OpenAI

Solution Delivered

- Rapid Team Mobilization: Accion TCoE quickly assembled a team of subject matter experts (SMEs) equipped with the necessary skills to address the project requirements.
- Knowledge Transition and Parallel Development: A well-planned knowledge acquisition session with the Jobber's team enabled the capture of all requirements. Simultaneously, the team began developing the testing framework.
- Custom Copilots Built:
  - To systematically convert manual test cases into automated scripts
  - To generate integrate automation scripts to test framework, enabling remote and parallel execution on iOS and Android devices.
- Timely Delivery and Efficiency Gains: The entire process was completed within the planned timeline and successfully used for regression testing. The structured approach resulted in an effort saving of approximately 60%-70% of the total QA effort.

JOBBER: OUTCOME & KEY BENEFITS

Metric	Previous Process	AI Assisted QE Process
Test Case Creation Time	3 Hrs/Feature	1 Hr/Feature
Test Coverage	60%	95%
Automation Coverage	50%	95%
QA Costs	High	Reduced by 40%
Time to Market	Standard	Accelerated by 30%

Key Benefits

- **Increased Efficiency**: Delivered automation scripts for 2000 test scenarios and integrated with CI/CD pipeline in 2 Months
- **Productivity Increase**: Saved 60%-70% of developer time for testing by owning the QE responsibility
- **Enhanced Coverage**: Achieved 95% coverage across UI and APIs, ensuring higher reliability.
- **Time To Market**: Reduced Time to Market by a Month
- **Scalability**: The automation approach is scalable to new features and updates with minimal manual intervention.

CASE STUDY: noodle.ai

Implementation of Custom Copilots in Testing of AI-Driven Supply Chain Planning Software

Customer Profile

- The client is a software product development company developing a cutting-edge, AI-driven supply chain planning software. Their platform leverages machine learning and AI algorithms to optimize inventory, forecast demand, and streamline logistics.
- The client aimed to enhance the quality assurance (QA) process for their software by automating test case generation and impr oving test coverage while reducing manual effort.

Challenges Faced

- Time-Consuming Manual Test Case Creation: Writing manual test cases based on acceptance criteria consumed significant time and resources.
- Inadequate Test Coverage: The client struggled to maintain high test coverage for their complex UI and API workflows.
- Scaling Automation: Existing automation efforts were ad hoc and lacked consistency, resulting in gaps in testing critical user journeys.
- High Cost of Testing: Manual efforts and fragmented automation were driving up QA costs without commensurate improvements in efficiency or reliability.

APIs & User Interface
Reactjs, Python, Postgres DB

Testing Tools
UI & API testing -> Playwright, Pytest & Python

Infrastructure
AWS

Custom Copilots
OpenAI, Gemini & CodeLama

Solution Delivered

- Adopted Custom Copilots QE Services Framework to collaborate and identify areas for value generation.
- Custom Copilots were built to in:
  - Automated Conversion of Acceptance Criteria into detailed manual test cases, ensuring clarity and precision in test coverage.
  - Identify gaps in testing and fillin additional edge case scenarios for improved robustness.
  - Machine learning-based reporting was employed to auto-triage defects and provide insightful dashboards for faster decision-making.
- Timely Delivery and High Efficiency: The entire solution was delivered within the planned timeline and effectively used for regression testing. This approach achieved an overall QA effort saving of approximately 70%-80%.

NOODLE: OUTCOME & KEY BENEFITS

Metric	Before Implementation	After Implementation
Test Case Creation Time	5 Hrs/Feature	1 Hr/Feature
Test Coverage	70%	95%
Automation Coverage	50%	95%
QA Costs	High	Reduced by 40%
Time to Market	Standard	Accelerated by 30%

Key Benefits

- **Increased Efficiency**: Manual test case creation time reduced by 80%.
- **Enhanced Coverage**: Achieved 95% coverage across UI and APIs, ensuring higher reliability.
- **Cost Savings**: QA efforts streamlined, reducing overall testing costs.
- **Scalability**: The automation approach is scalable to new features and updates with minimal manual intervention.

Accionlabs

Thank You!