cypress

Empowering End-to-End Testing

PRICING PLAN PRESENTATION

Anjanay Raina	Garima Chopra	Heemank Verma	Vishnu Shon
2020494	2020376	2020064	2020414

List Content Presentation

Selection Criteria	03
What is Cypress?	04
Scope & Purpose	05
Key Features	06
Cypress vs Traditional Tools	07
Using Cypress	08
Real world applications	09
Limitations	10

OSS Selection Criteria

To guide our selection process, we set clear criteria:

- Focused on vital aspects of software development.
- Excluded commonly used languages, compilers, and mainstream IDEs.
- Emphasized areas such as testing, code analysis, integration, dependency management, and frameworks.

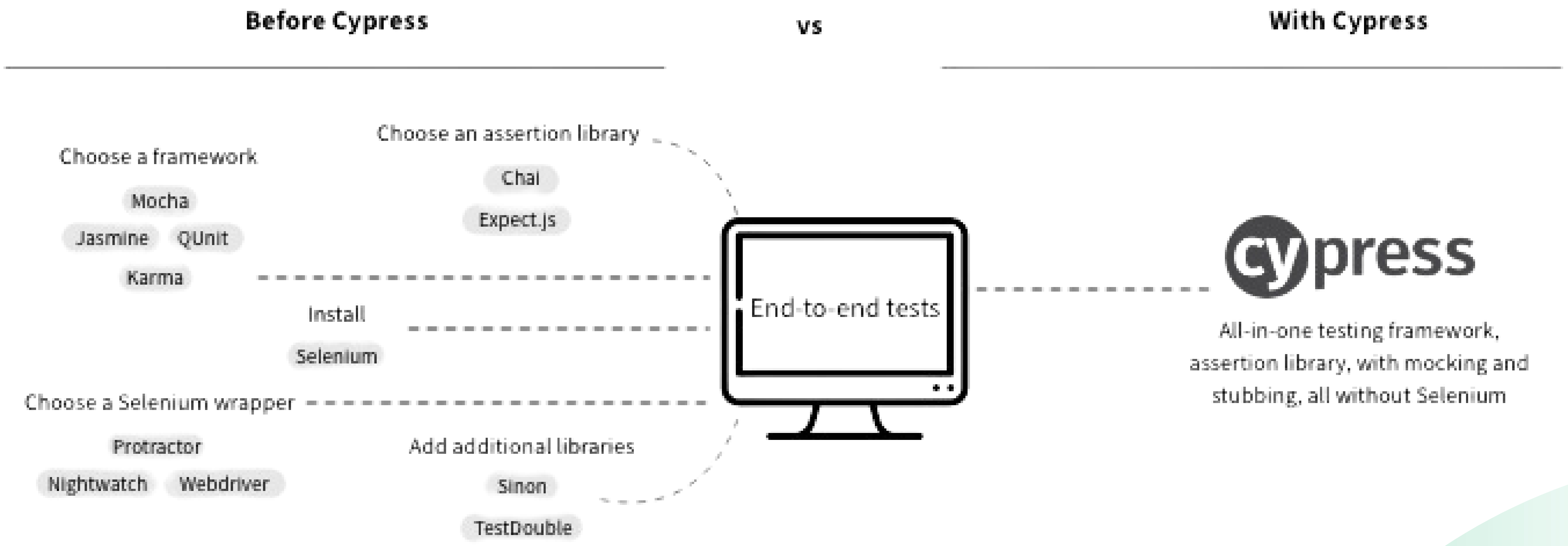
What is Cypress?

- Cypress is an open-source end-to-end testing framework for web applications.

- It's designed to enhance the testing phase of software development, making it more efficient and developer-friendly.

Scope and Purpose

Cypress is an open-source end-to-end testing framework for web applications. It's designed to enhance the testing phase of software development, making it more efficient and developer-friendly.



All-in-one testing framework, assertion library, with mocking and stubbing, all without Selenium

Key Features

Real-Time Reloads

Instantly see changes as you code without manual reloading, streamlining the testing process.

Automatic Waiting

Cypress intelligently waits for elements to appear, reducing the need for explicit waiting commands and enhancing test reliability.

Debuggability

Cypress offers a robust interactive test runner with in-browser debugging, allowing you to inspect and interact with your application's DOM at any point during test execution.

Cypress vs Traditional Tools

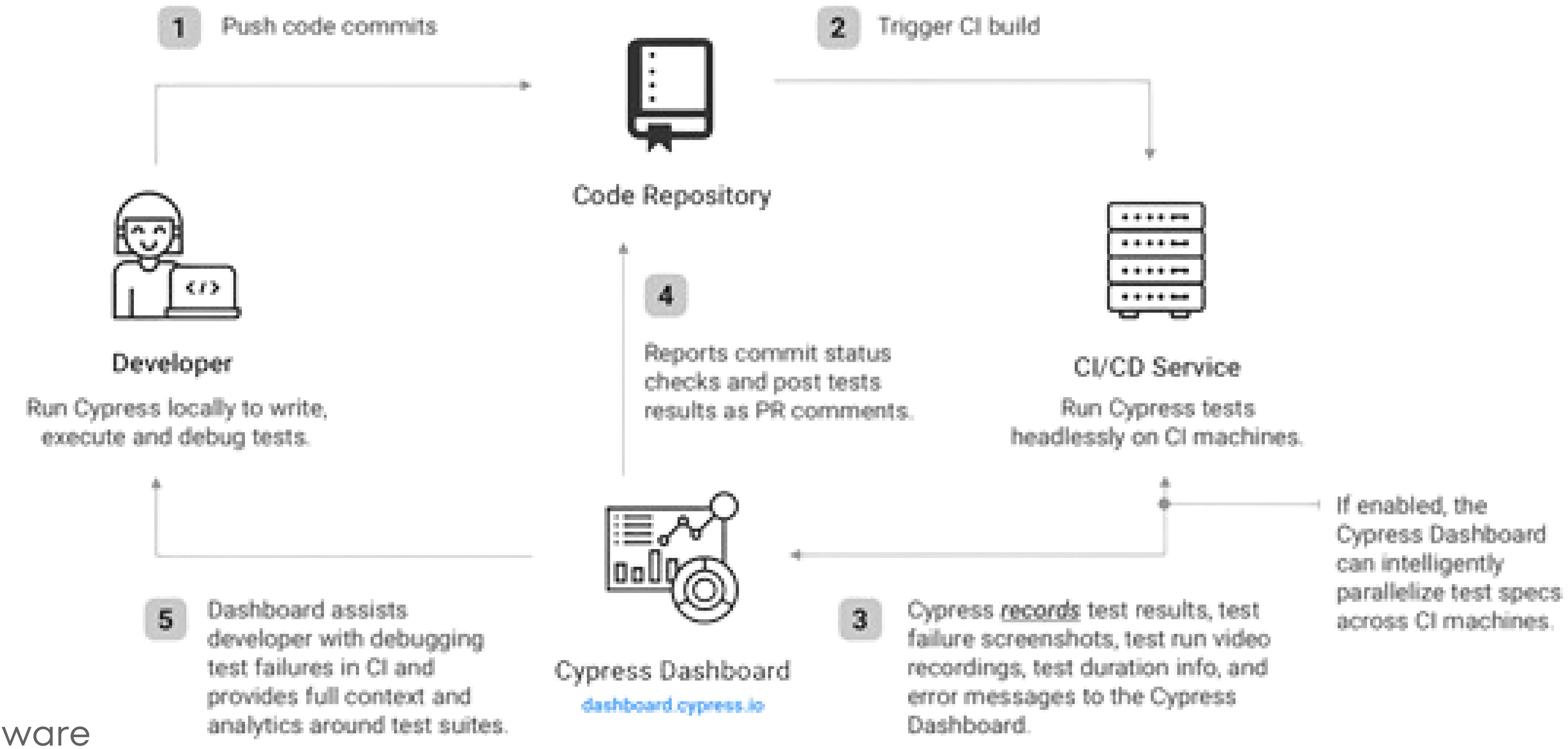
Traditional Tools

- ✓ Real-time reloading
- ✓ Automatic waiting
- ✓ Debugging capabilities
- ✓ Cross-browser testing
- ✓ REST API testing
- ✓ Custom commands
- ✓ CI/CD integration
- ✓ Community-driven

Cypress

- ✓ Real-time reloading
- ✓ Automatic waiting
- ✓ Debugging capabilities
- ✓ Cross-browser testing
- ✓ REST API testing
- ✓ Custom commands:
- ✓ CI/CD integration
- ✓ Community-driven

Using Cypress in Software Development



1 Push code commits

2 Trigger CI build

Code Repository

Developer

Run Cypress locally to write, execute and debug tests.

4 Reports commit status checks and post tests results as PR comments.

CI/CD Service

Run Cypress tests headlessly on CI machines.

If enabled, the Cypress Dashboard can intelligently parallelize test specs across CI machines.

5 Dashboard assists developer with debugging test failures in CI and provides full context and analytics around test suites.

Cypress Dashboard

dashboard.cypress.io

3 Cypress records test results, test failure screenshots, test run video recordings, test duration info, and error messages to the Cypress Dashboard.

- Cypress seamlessly integrates into the software development lifecycle.
- Witness how it empowers developers to write efficient, reusable tests, becoming a cornerstone of the development process.

Real world applications that have benefitted from Cypress

Microsoft

Microsoft uses Cypress for efficient testing, catching bugs early in web apps, and enhancing software quality.

Trulia

Cypress reduced test flakiness, ensuring better software quality for Trulia's real estate platform.

CircleCI

CircleCI improved CI/CD reliability with Cypress, boosting software release confidence.

GitLab

GitLab automated complex test scenarios with Cypress, resulting in faster test execution and improved platform quality.

OpenAI

OpenAI's GPT-3 Explorer relies on Cypress for a responsive UI and an enhanced user experience.

Example
Video

Video Link

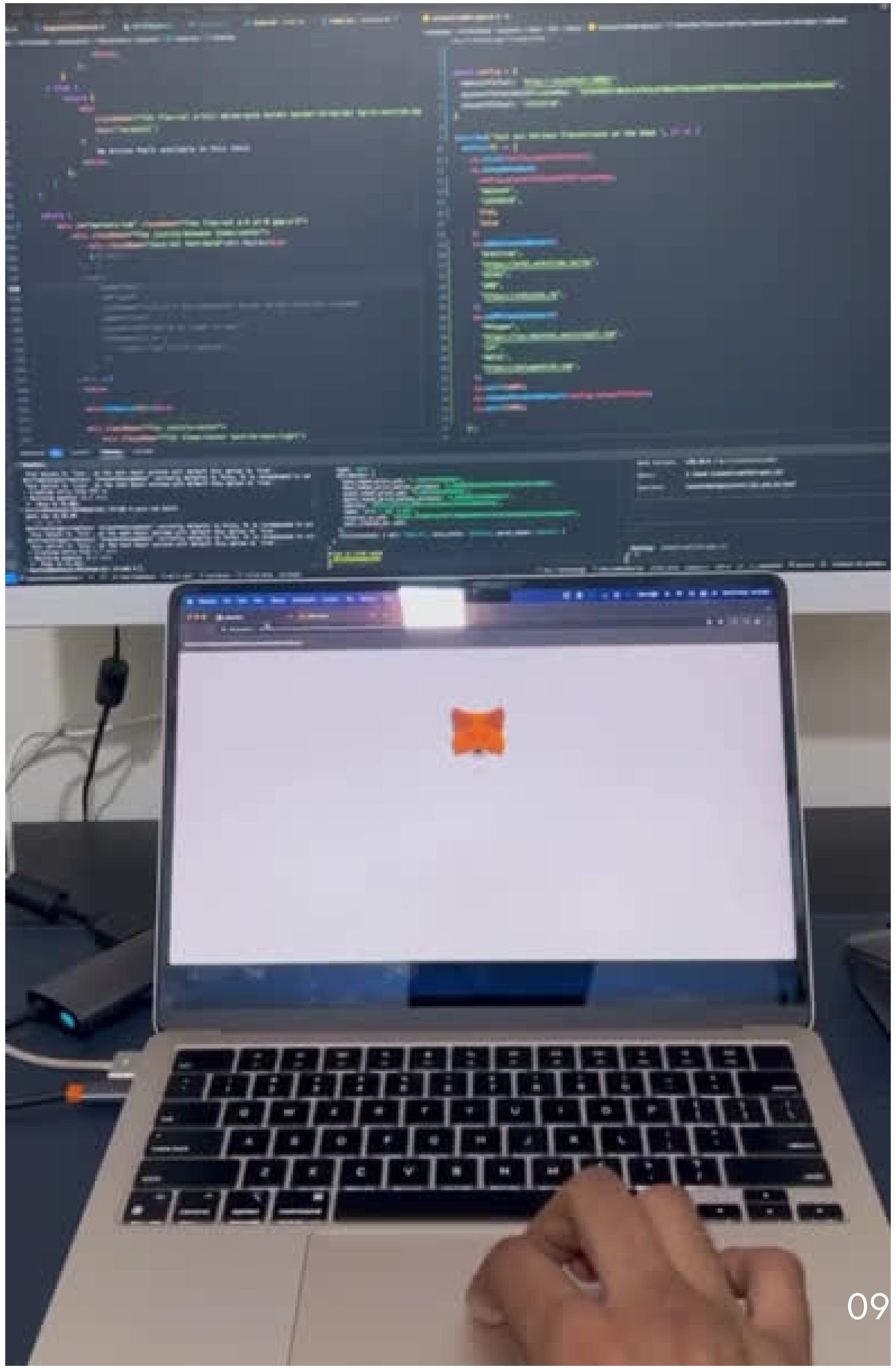
The video features a Cypress E2E testing bot, emphasizing wallet and website testing. The bot creates a new wallet account, simulating a real-world scenario.

Integration Testing:

- Seamless integration with the associated website demonstrates a comprehensive E2E testing approach.

Authentication Test:

- Primary testing includes logging in with the newly created wallet, validating user authentication methods.



Challenges and Limitations

Limited Cross-Browser Support:

- Cypress primarily supports Chrome, requiring extra setup for other browsers.

No Multi-Tab/Window Testing:

- Cypress lacks built-in support for testing across multiple tabs or windows.

Handling External Domains:

- Testing network requests to external domains may require additional effort.

Resource Intensiveness:

- Large test suites can slow down Cypress; efficient organization and parallelization are essential.

Thank You

Floor open for Q&A