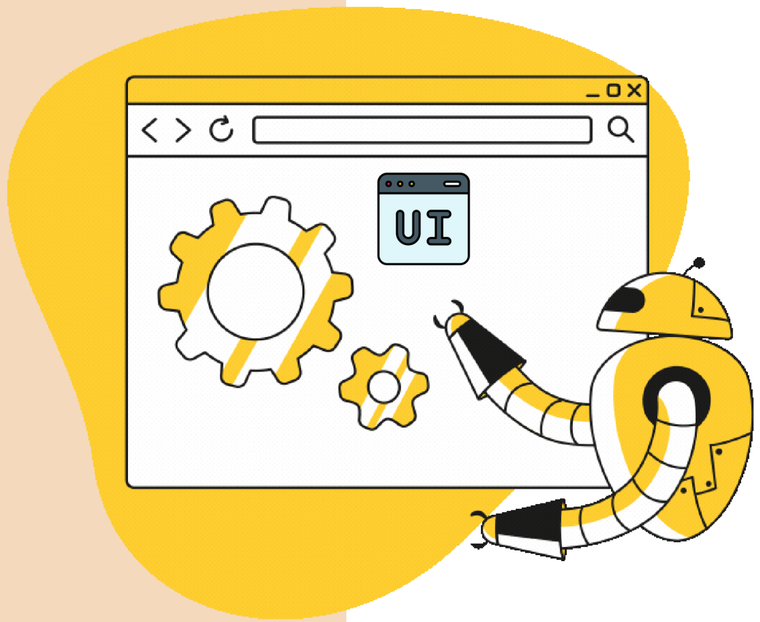


AUTOMATION OF UI TESTS

WHAT TO EXPECT

1. Why Automate UI Tests
2. Types of UI Test Automation
3. Key Benefits of UI Test Automation
4. Challenges in UI Test Automation
5. Popular UI Test Automation Tools
6. Continuous Integration and CI/CD
7. Continuous Integration & Testing

Why Automate UI Tests

Run this test 1000 times !

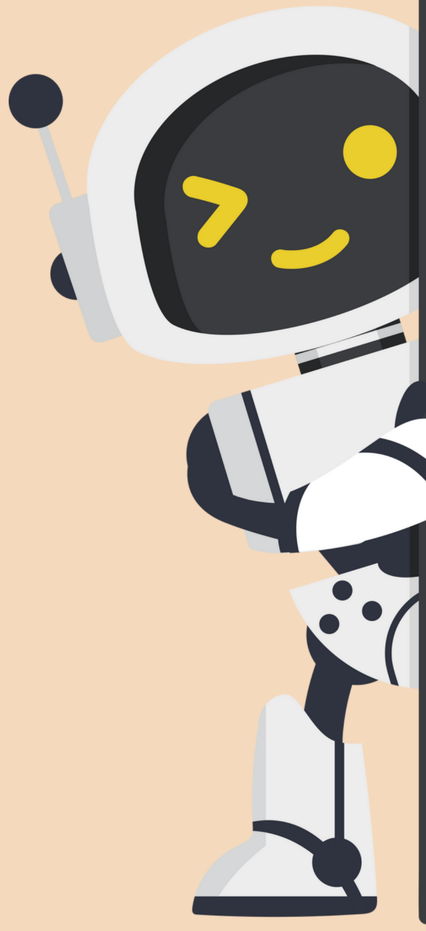
Testcase ID	Module	Testcase Description	Priority	Test Steps	Expected Result	Actual Result	Status	Overall Test Status
1	Login	On Providing valid credentials, User should be loggedin to the BookStore Application	P1	Step 1	Open https://bookstore.qacurry.com on any Browser			
				Step 2	Enter username - student@qacurry.com			
				Step 3	Enter Password - Q@curry			
				Step 4	Click on Login Button			

Why Automate UI Tests

- Improved Testing Efficiency
- Consistency and Reproducibility
- Early Bug Detection
- Regression Testing Benefits

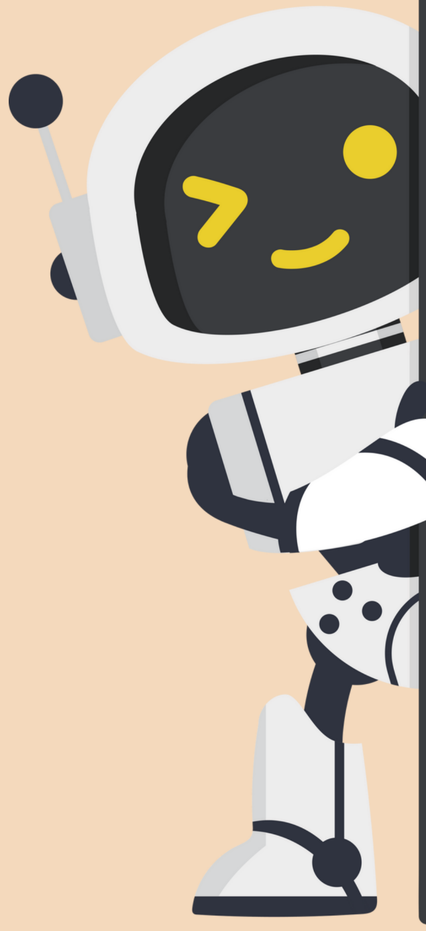
Types of UI Test Automation

- Scripted Automation
- Record and Playback
- Keyword-Driven Testing
- Data-Driven Testing



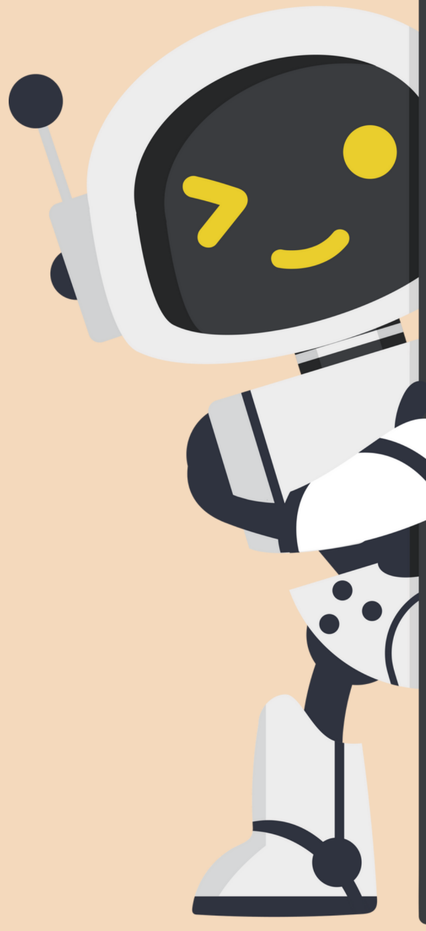
Key Benefits of UI Test Automation

- Faster Test Execution
- Cross-Browser and Cross-Platform Testing
- Improved Test Coverage
- Enhanced Regression Testing



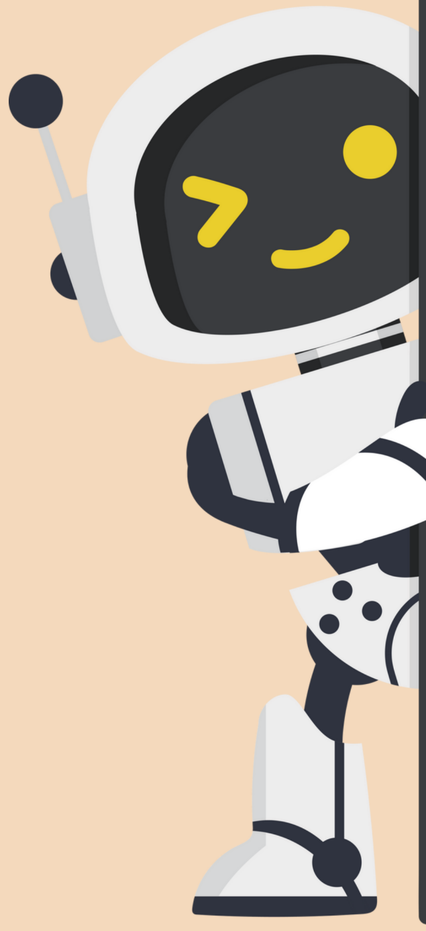
Challenges in UI Test Automation

- Maintenance of Test Scripts
- Handling Dynamic Elements
- Mobile Test Automation Challenges
- Test Data Management



Scripting and Test Design

- Writing Efficient Test Scripts
- Identifying Test Cases for Automation
- Test Data Generation and Management



Popular UI Test Automation Tools

Selenium

OpenSource

UI Testing

TestComplete

Commercial

UI Testing

Katalon

Commercial

UI Testing

UFT

Commercial

UI Testing

Tosca

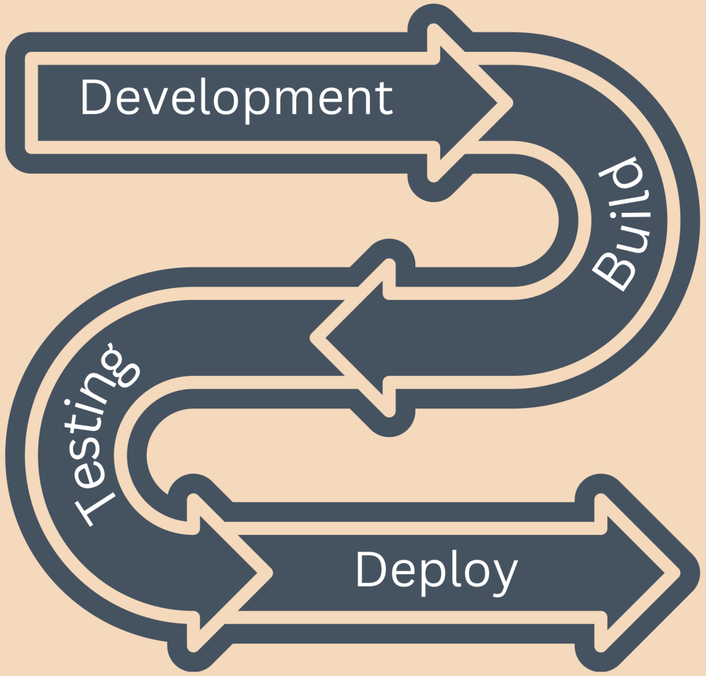
Commercial

UI Testing

See you in Next Video

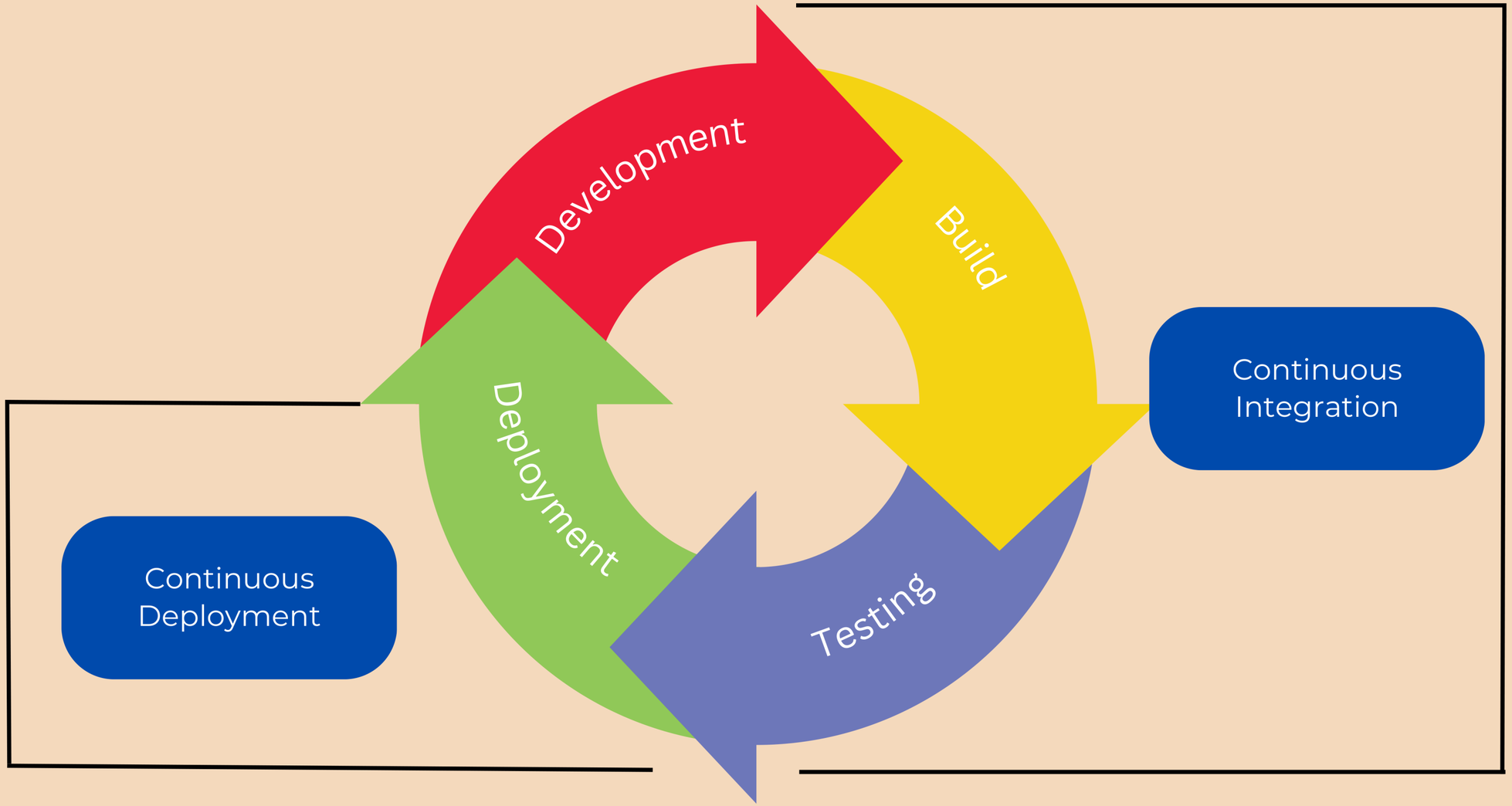
Continuous Integration and CI/CD

Continuous Integration (CI) and Continuous Integration/Continuous Delivery (CI/CD) are practices in software development aimed at automating and streamlining the process of integrating code changes, testing, and delivering software to production.



These practices are essential in modern software development to ensure code quality, reduce manual errors, and deliver software more frequently and reliably.

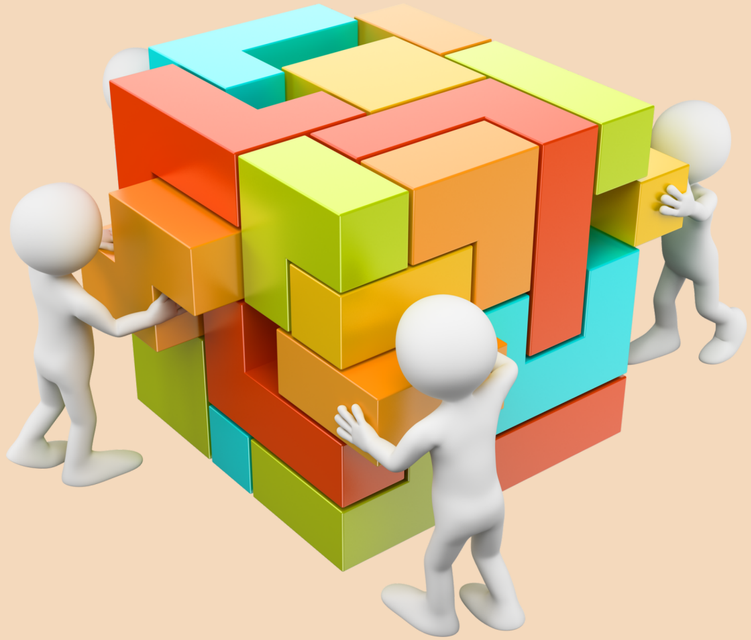
Continuous Integration and CI/CD



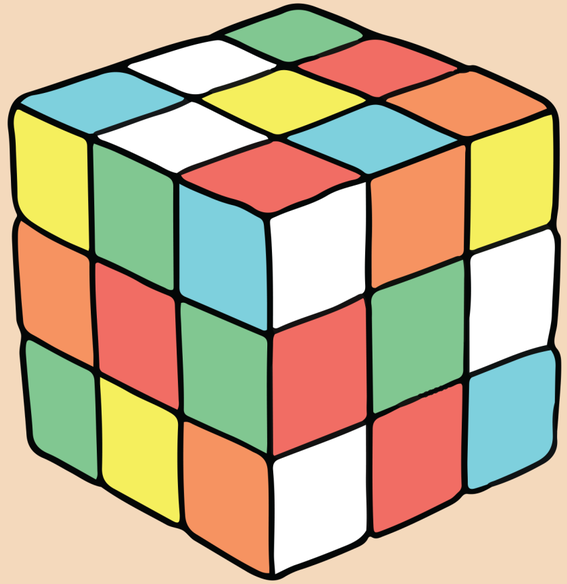
A typical CI/CD pipeline can be broken down into the following stages:

1. **Development :** This phase is where the coding takes place, and the code is integrated into a version control repository and subsequently verified for correctness.
2. **Build :** The validated code is employed to construct the application, and the resulting artifact serves as the basis for testing.
3. **Testing :** Typically, the constructed artifact is installed in a test environment, where comprehensive tests are conducted to verify the application's functionality.
4. **Deploy :** This marks the ultimate phase of the pipeline, during which the thoroughly tested application is released to the production environment.

Continuous Integration

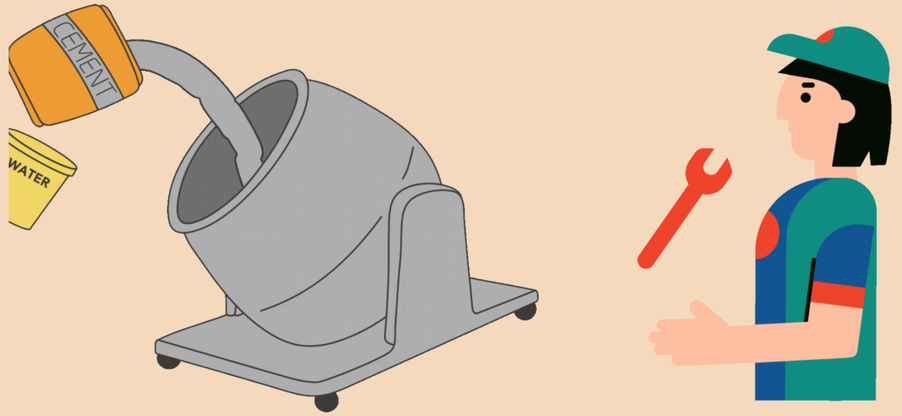


Final Outcome



Continuous Integration & Testing

Continuous Integration (CI) in testing enhances software quality by identifying and addressing issues early, preventing the disruption of existing features, and promoting developer collaboration. Leading CI tools like Jenkins, Travis CI, CircleCI, and GitHub Actions are widely adopted in the software development industry.



Continuous Integration & Testing