What is a CI/CD Pipe line?

A CI/CD pipeline is a deployment pipeline integrated with Automation tools and improved workflow.

If conducted properly ,it will minimize manual errors and enhance the feedback loops throughout the sdlc, allowing teams to deliver smaller chunks of releases within a shorter time.

The components and tools needed to build a pipeline depending on the teams particular needs and existing workflow.

|  |
| --- |
|  |

CI/CD Benfits :Advantages of continues Integration& delivery

Enhances” Quality at speed

For many teams ,product Quality and deployment speed are first come to mind when applying a new workflow. The old fashioned way of deploying updates stretches the time spent on identifying and reporting underlying issues .the more manual the feedback loop is , the longer it takes for the engineering team to fix the bugs.

A well -functioning CI/CD process is a productive solution to speed up deployment pace and make each release more valuable more valuable to the end users. It streamlines the deployment cycle by enable constant communication between teams and utilizing automated processes.

Automate tedious tasks

First ,Business resource are used in other more critical areas rather than invested in the repetitive, redundant testing tasks. When the process is automated and streamlined in a pipeline ,it effectively reduces the complexity and production costs for repetitive tests.

Reduce regression testing effort

The Quality Assurance teams job is very fragile to change .one small change can lead to many regression efforts. Therefore ,having access to all versions of the system is crucial to QA as well as other Stake holders. Moreover ,keeping the largest version updated will help improve the Quality and reliability of the QA feedback of bug logged.

Help developers stay focused

The CI/CD pipeline allows developers to stay focused on what they do best , writing code .without it ,developers have to handle all kinds of tasks from environment setup ,build making to issue investigation ,and even product delivery ,which is time -consuming and distracting from their primary focus.

Easier test logs archiving

Keeps logs of all code changes testing and deployments so that members can inspect at any time .it also allows rolling back to previous versions with a single routine push- button action

Effortless product updates

When the release process gets streamlined in the CI/CD process ,product updates are much less

Stressful for the development team.

More effective feedback loop

A rapid ,accurate and continuous feedback loop will effectively give shape to an organizational

Culture of learning and responsibility.

Adds business values to organizations

Having a CI/CD pipeline in place brings many benefits not only to the product team but also to the organizations business values .

Generate builds faster

Integrating CI and CD into your production line will establish a continuous and automated cycle in which deliveries are completed faster with more value all team members can stay on track of their projects and provide feedback in real time, thus any bugs or issues can be Quickly identified and resolved .

Over time ,your product will be refined thanks to the constant review from the team ,resulting in a more satisfying user experience.

Improve time to market

Deployment timing is one of the key elements that decide the success of your product release .timely deployment helps increase engagement with customers ,gain profit ,supports pricing ,and boost market goals for your business . with the right time to market ,the products ROI will significantly increase.

Improve the code Quality

The CI/CD pipeline provides a workflow that allows developers to integrate their codes more frequently and share with team members to avoid possible conflicts I future builds .this will help to reduce the cost of fixing defects and eventually improve the code Quality for all updates.

Better developer’s efficiency

Once the continuous and automated software delivery process has been established ,defects will have minimized .this means the developers will have more confidence in integrating their codes in smaller chunks daily An effective CI/CD process ensures that the team has all the tools needed to commit daily,

Stay on schedule ,and drive more product values.

Attract more talents

And as a cherry on top an integrated CI/CD pipeline makes you a more attractive employer to developers. Hiring talented developers is hard ; that is why having well -established CI/CD process will help you attract talents more easily.

Components of a CI/CD pipeline

A typical CI/CD pipeline must include these phases

Build phase

Testing phase

Deploy phase

Automated testing phase

Deploy to the phase

In such a pipeline , things started with the developer team to write the initial lines of code .the developers then commit these codes into a version control system, which is the first phase of the pipeline.

As a linear workflow ,the developers will commit new codes and push them to the version control system with an updated version tag

|  |
| --- |
|  |

The build phase is triggered when the new codes are pushed to a repository. Because the initial codes are stored in small branches of the repository. The compiler will gather all features of the codes and their dependencies ,then compile them into a new build .

|  |
| --- |
|  |

The testing phase comprises of multiple types of tests with the most crucial one being unit testing .unit

Testing will test the individual units of the product from its source code.

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  | |  |  |  |  |  | |

Once the builds have passed the tests ,they are moved to the deployment phase then pushed into a test server. This phase allows developers to simulate the product in a production equivalent environment to Environment to examine see the product features.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

The automation test phase will perform the final tests to Quality the built features before they are deployed to production. Automated and continuous testing is applied in the phase to utilize the builds and make sure there are no bugs remaining.

|  |  |
| --- | --- |
|  |  |
|  |  |

Throughout the pipeline ,whenever there is an error ,feedback will be instantly sent to the development team so that issues are immediately addressed. code changes to fix bugs will then go through the production pipeline once again.

After codes or the product passed all the tests without defects, they move on the production server in the final phase. the constant feedback loop helps make the pipeline a closed process where builds are continuously committed ,tested, and deployed to production.

Test Automation in CI/CD pipeline

It is important to maintain a quick and responsive feedback loop so that the whole team can “fail faster” thus resolves quicker and more efficiently. If your team does manual testing, kt is almost impossible to setup a continuous development environment when the test phase takes lots of time to test thoroughly.

However, since manual testing is still important for exploratory testing; software production teams shall adopt automated testing into their workflow.

By integrating test automation into the pipeline ,teams can enable faster builds and deployment by continuously generating feedback based on test results ideally, automated tests should be applied to as many test phases and as possible.

Choosing the right automation tools is the most important step that brings automation test practices into reality.

From functionality, security ,performance to regression tests, a powerful automation tool makes sure these fully tests fully cover what needs to be tested . additionally ,some automation tools are capable of conducting parallel testing .which is running many tests or test cases simultaneously. cross - browser and cross systems. parallel testing and end to end testing are the key capabilities that help increase test coverage and shorten the time to market the software.

The tests pre and post conditions .as well as infrastructure and test environment ,are also challenging factors for teams that still operate manually. It takes hours ,even days to fully build ,update and tear down test environments. An automated environment provisioning process can provide that in just a few clicks .

More importantly ,the powerful automated testing tools are those that can provide the latest versions Of browsers ,systems and resolution configurations; so the QA teams will not have to spin up maintain ,or tear down environment altogether.

Katalon is one of the most comprehensive solutions which can cover a wide range of automation requirements mentioned above .

Katalon supports all end to end testing types : web API mobile and desktop testing .The tool offers robust execution mechanism with parallel testing for the diversified environment from local to cloud services .katalon also provides automated environment infrastructure that eventually , you would not need to manage their own server.

What makes a good CI/CD pipeline

Ultimately, the purpose of employing CI/CD is that teams can generate Fast ,accurate ,reliable and comprehensive feedback for their development cycle .therefore , proper pipeline should cover these factors : speed ,accuracy ,reliability, and comprehension.

Speed

Continuous integration is meant to be rapid with instant feedback .if the developers have to wait longer than 10 minutes for their build to be verified in the QA environment, the flow will be considered disrupted .Because then, developers have to switch between contexts and wait for one build to pass before moving on to another .

In a CI/CD pipeline ,the time it takes for each commit will limit how many times developers can deploy new codes in a day.today,business require fast update pace and quick adaptation to changes ,which means the engineering team needs a CI/CD process that supports a fast moving workflow accordingly.

As the business continues to grow ,the CI/CD tools must scale just as quick to meet new demands .A powerful tool is the that is programmable and applicable to the existing development workflows. Plus ,

The CI/CD configuration needs to be stored as codes that allow reviewing ,versioning, and restoring

For future uses.

Accuracy

Apply automation to the deployment process is a great staring point .however automation is not enough to make a functional and beneficial CI/CD pipeline .its must accurately run and visualize the entire software delivery process. The pipeline ,combined with other tools have to precisely handle both simple and complex workflows, leaving no space for manual errors when performing repetitive tasks.

The more accurate the pipeline is, the closer it gets to being fully automated from continous integration to continuous deployment without any human interference needed.

Reliability

Having a reliable CI/CD pipe line improves the speed of building and deployment new commits significantly. the pipeline must ensure the output is always stabilized with the same input without oscillations in runtime.

As mentioned above ,the need for CI/CD infrastructure to operate and scale accordingly to the growing pace of the product team is high .when the number of teams and project s increase or the workflow changes ,the pipeline should stay reliable and resourceful to supports the increased workload .

Comprehension

A good CI/CD pipeline needs to cover as many as possible all the aspects of a seamless software delivery process .it takes just one phase left uncovered in the CI/CD tools to significantly affects the whole pipeline changes

Api Testing

An api is essentially the middle man of the layers and system within an application or software .

Api (application programming interface ) testing is performed at the message layer without GUI.it is a part of integration testing that determines whether the APIs meet the testers expectations ,reliability performance ,and security.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

There are two broad classes of web services web Api : soap and rest

SOAP is a standard protocol defined by the w3c standards for sending and receiving web service requests and responses.

REST is a web standard based architecture that uses HTTP. unlike soap based web services, there is no official standard for restful wed Apis

**Understand ApI requirements**

Before testing yours Apis ,you need to answer these questions to thoroughly understand the ApI requirements

WHAT IS THE API PURPOSE

Knowing the purpose of the api will set a firm foundation for you to well prepare your test data for input and output .this step also helps you define the verification approach. For some APIs you will verify response Against the database and the some others ,its better to verify the response against other APIs

Generally ,APIs of an application are used to manipulate its resources in reading (GET),creating (POST),updating (PUT)and deleting (DETELE).knowing the purpose of the API will set a firm foundation for you to well prepare your API testing data for input and output.

In addition ,this step also helps you define the verification approach . for some APIs you will verify the response against the database and for some others, it is better to verify the response against other APIs

The output of the create user API will be the input of the “ GET user” API for verification .the output of the “get user”API can be used as the input of the “update user’API and so on.

**SPECIFY THE API OUTPUT STATUS**

The most common API output you need to verify in the API testing is the response status code.

Verifying if the response code Equals to 200 or not to decide whether an API testing is passed or failed is familiar to new API testers. this is not a wrong verification. However ,it does it reflect all test scenarios of the API

All API response status codes are separated into five classes in a global standard .the first digit of the status code defines the class of response .the last two digit s do not have any class categorization role .

There are five for the first digit

\*1XX informational : the request is received and continues to be processed

\*2XX successful: the request is successfully received ,understood ,and accepted.

\*3XX redirection Furth action needs to be taken to complete the request

\*4xx client Error :the request contains the wrong syntax or cannot be fulfilled

\*5xx server :the server fails to fulfill an apparently valid request

However ,the actual response status code of an api is specified by the development team that build the API . so as a tester ,you need to verify whether:

\*The code follows global standard classes

\*the code is specified in the requirement.

What is API Testing

API testing is a software testing practice that tests the api directly from their functionality, reliability, performance ,to security. As part of integration testing .api testing aims to validate the logic of the build architecture within a short amount of time .the goal of api testing is not to check the individual software component itself ,but rather the connection between them.

API testing can be performed either manually or automatically with the help of API each comes with its own pros &cons .Generally automated API testing is preferred due to the repetitive and data -driven nature of the testing type ,also there are many aspects of API testing , but these are 3 most common

Categories

FUNTIONAL TESTING (checking whether the api performs its function as expected)

PERFORMANCE TESTING (checking how the api performs under different levels of usage)

SECURITY TESTING (checking if the apis have any vulnerability and or have been protected against security threats )

Test cases for API Testing for each category

Test cases for api functional Testing

Functionality is the core any Application under test ,their most basic and foundational functionality is data retrieval and data sending, and API functional testing should revolve around those 2 domains .check out the following functional test case and see how you can apply them to your own testing project

Status code validation for valid Requests

Verify that the API consistently returns the expected response status code ,such as “200k”, for valid and properly formatted requests.

Authentication Handing with invalid credentials

Test the API response when provided with invalid authentication credentials, ensuring it consistently returns a “401 unauthorized “ status code as expected .

Graceful handing of missing or invalid parameters

Verify that the API handles missing or invalid request parameters gracefully and returns clear and user friendly error messages that aid in troubleshooting.

Input Data validation with malformed data

Test the Api ‘s input validation by submitted various forms of malformed data ,such as invalid email formats , confirm that it properly rejects and response to those inputs.

Time out handing under load

Confirm that the API correctly handles timeous by simulating request that take longer to process ,ensuring that it remains responsive and does not hang.

Pagination functionality verification

Test the API pagination functionality by requesting specific pages of results and verifying that the responses contain the expected data and pagination information.

Concurrency Testing without data corruption

Verify that the API handles concurrent request from multiple users without data corruption of conflicts ,ensuring data integrity.

Response format adherence (json/xml)

Ensure that the API consistently returns responses in the specified format (e,g xml&json)and adheres to the defined schema for data structure.

Caching MeChanism evaluation with repeated requests

Evaluate the API ‘s caching mechanism by making repected requestsand verifying that the cache headers are correctly set and honored.

Rate limited assement

Test the API ‘S RATE limited by sending request at a rate that exceeds the defined limits and checikg for the expected rate limiting response, ensuring that limited are enforced.