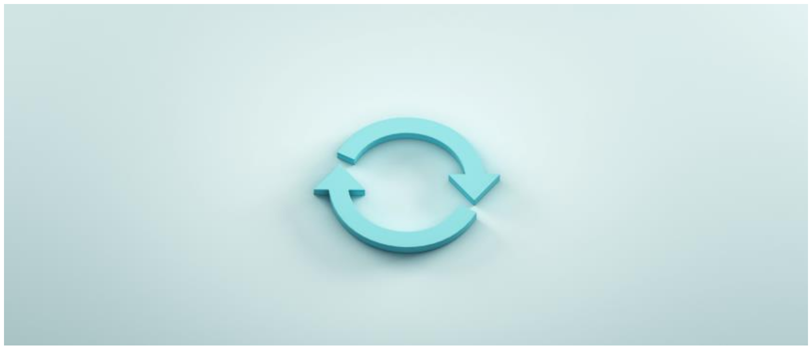
Success Story: Manual to Automated Testing

Test Automation Framework Integrated with Azure DevOps

Who is the client?

The client is the Department of Records for a major U.S. city, responsible for maintaining records and fulfilling report requests.



Client was having difficulty identifying repetitive test cases

What was the client's problem?

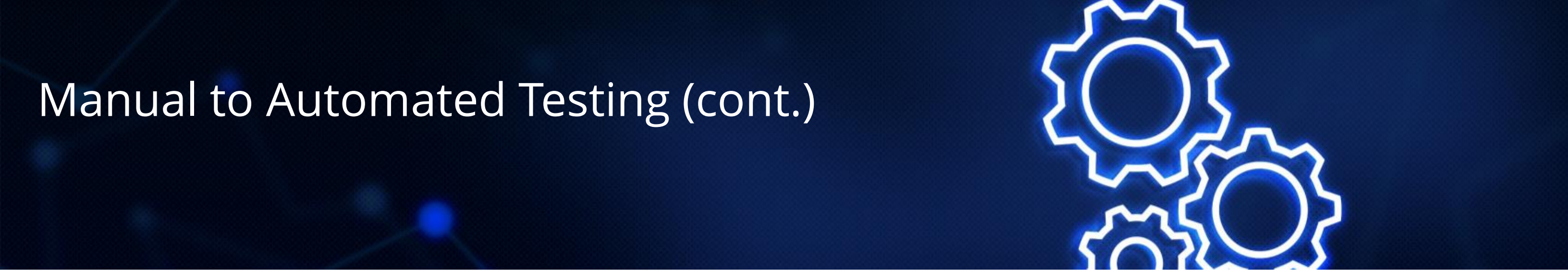
The client was having difficulty identifying repetitive legacy manual test cases and their migration to be able to adopt a test automation framework. They had a goal to reduce cost and increase productivity by integrating the automated testing process inside their CI/CD pipeline which would automate the testing, reporting and deployment of the application.

Challenges

1. Little to none documentation
2. Not enough visibility of the manual test cases
3. Lack of an end-to-end defined automation process
4. Higher learning curve for test automation

Solutions

1. Formal organizational project set up
2. Maintaining of current manual test cases with various attribute like priority, complexity, fitness to automation etc.
3. Workshop to identify the right applications for test automation
4. Resourced with dedicated and skilled automation engineers



Manual to Automated Testing (cont.)

What services did Data-Core provide to solve the problem?

Data-Core decided the best option was to create a custom framework that would benefit the client, not just from the test automation perspective, but also help in reducing the manual effort in test execution and deployment of the application. Hence, the solution built comprises the following:

1. Combine several open source automation tools/libraries to weave the benefit of each while circumventing their weaknesses
2. Develop commonly reusable features as much as possible, in order to reduce scripting effort
    - ❑ Initialization of application under test
    - ❑ Take screenshots
    - ❑ Explicit waits
    - ❑ Create automated reports
3. Automate the test execution process each time a new build is available
4. Publish the report in a dashboard
5. Deploy the build where necessary

Framework benefits: automation, execution & deployment

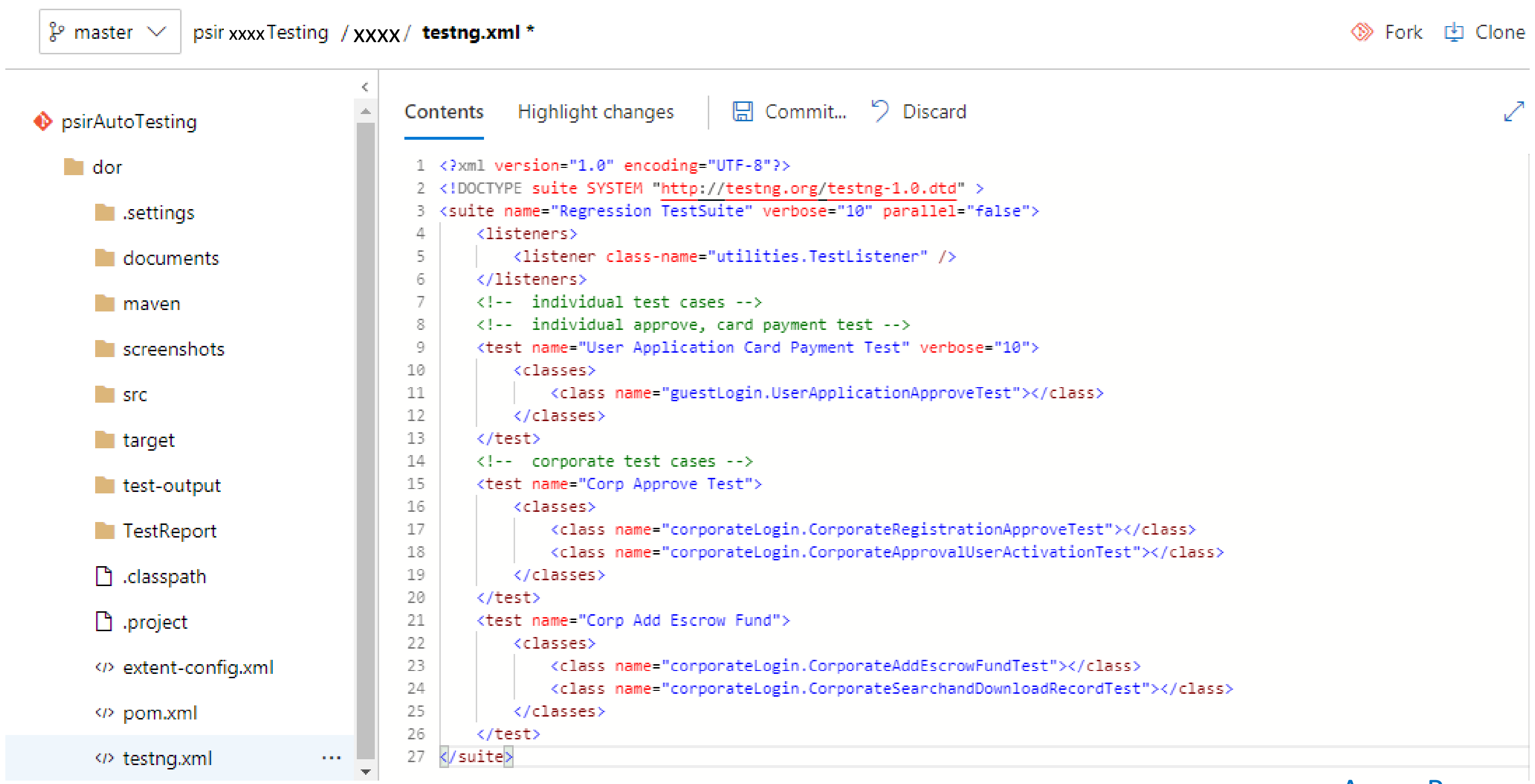
Tools and Technologies used:

- ✓ Java as a general programming language
- ✓ Selenium as a browser automation library
- ✓ TestNG as an automation framework
- ✓ Extent reports in conjunction with iTest Listeners for rich style reporting
- ✓ Jenkins server for continuous integration
- ✓ Microsoft Azure DevOps pipeline to automate the execution and deployment process

Manual to Automated Testing (cont.)

Implementation:

Data-Core designed a data driven framework in Selenium that had all test data managed from Microsoft Excel. This allowed execution of same test cases for various types of data without having to execute each of them manually. The commonly reusable features encourages reusability of test codes by reducing scripting time and effort. Consequently, plugging new features into the framework was greatly reduced.



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="Regression TestSuite" verbose="10" parallel="false">
    <listeners>
        <listener class-name="utilities.TestListener" />
    </listeners>
    <!--  individual test cases -->
    <!--  individual approve, card payment test -->
    <test name="User Application Card Payment Test" verbose="10">
        <classes>
            <class name="guestLogin.UserApplicationApproveTest"></class>
        </classes>
    </test>
    <!--  corporate test cases -->
    <test name="Corp Approve Test">
        <classes>
            <class name="corporateLogin.CorporateRegistrationApproveTest"></class>
            <class name="corporateLogin.CorporateApprovalUserActivationTest"></class>
        </classes>
    </test>
    <test name="Corp Add Escrow Fund">
        <classes>
            <class name="corporateLogin.CorporateAddEscrowFundTest"></class>
            <class name="corporateLogin.CorporateSearchandDownloadRecordTest"></class>
        </classes>
    </test>
</suite>
```

Azure Repos

Framework:

Azure Repos

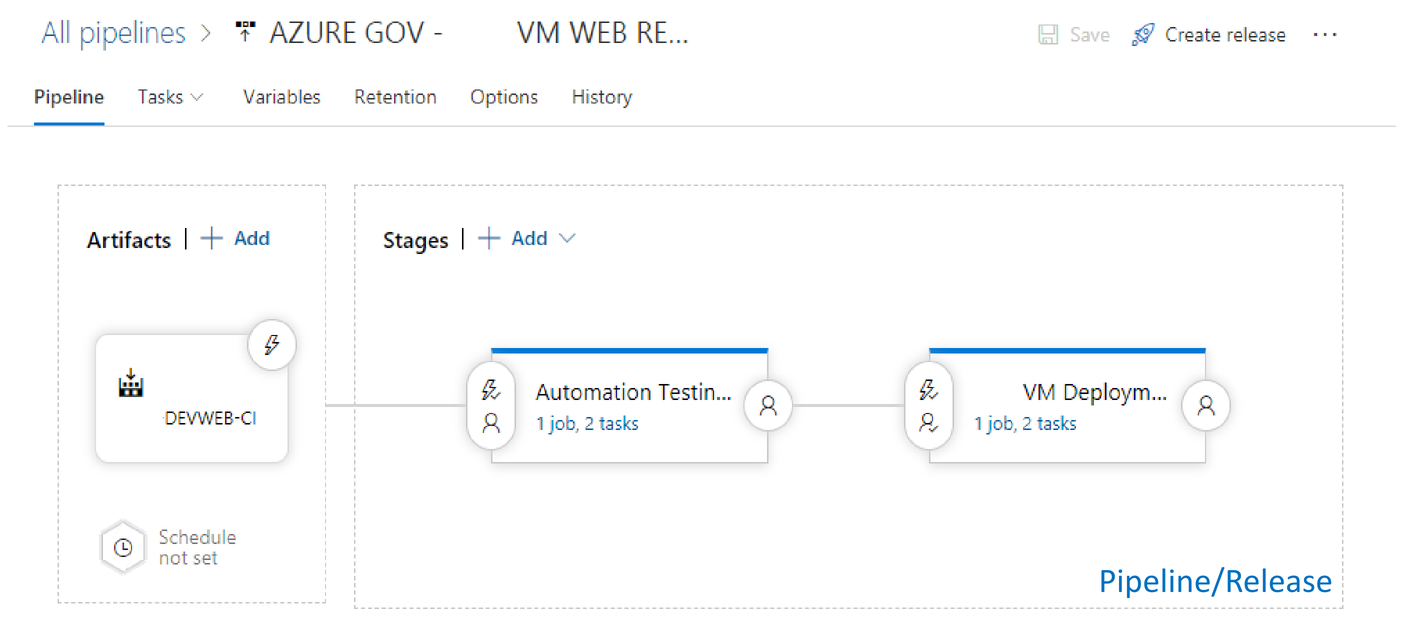
In this project, we have utilized Azure Repos Git to store our project and test source codes. Both codes are located in their respective branches. The screen shows the automation codes residing in master branch of Azure repo.

Manual to Automated Testing (cont.)

Pipeline/Release

The next step was to create a release pipeline in order to implement the CI/CD architecture. The basic steps in implementing the release pipeline are as follows:

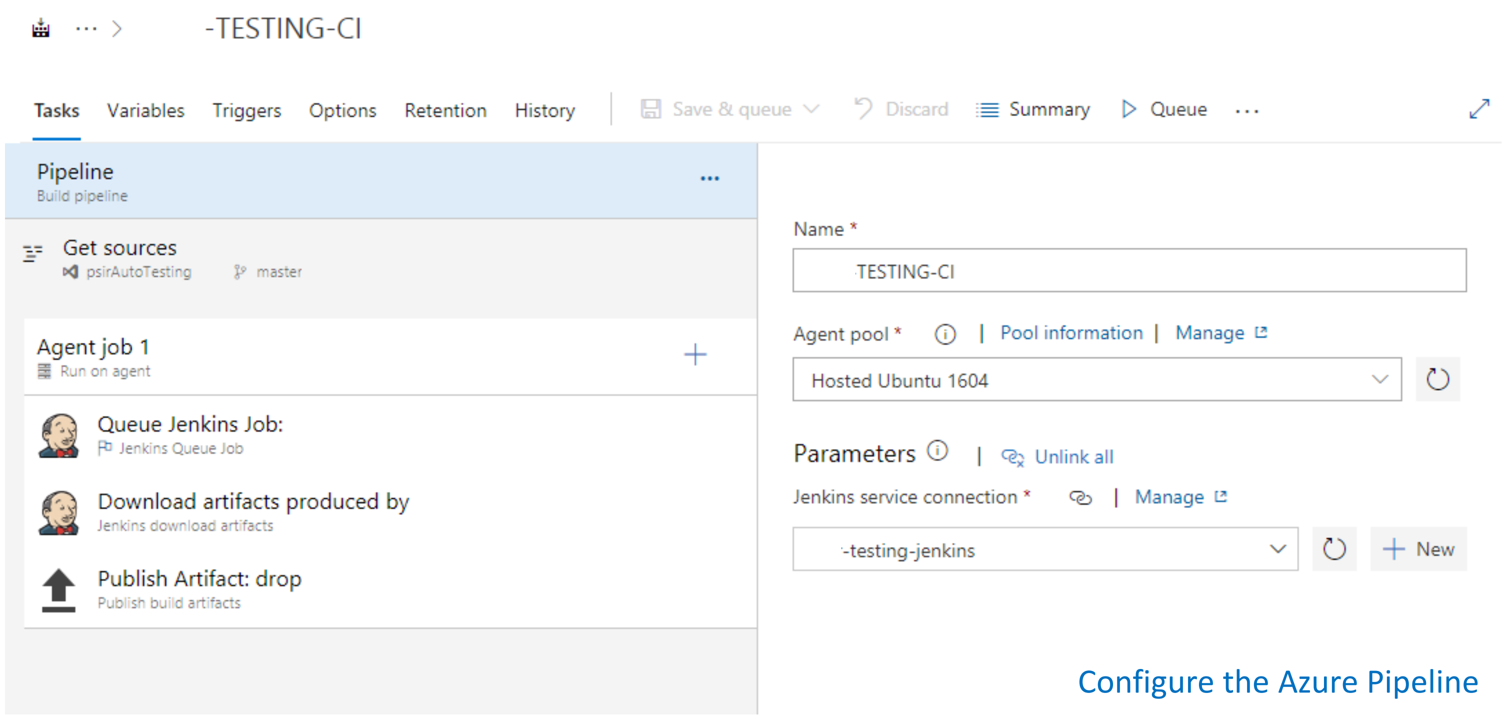
1. Configure the Azure Pipeline to use the Azure Repo where the automation codes reside
2. Define the build process – either automated (triggered whenever new code is pushed to Repos) or manual
3. Create Release – automation testing is triggered, result published in desired directory
4. Deploy build to any target of choice – depending on test result, project source code is then tested and packaged to be deployed to any target, i.e. to production branch, etc.



Configure the Azure Pipeline

The first job of this build process is to configure a Pipeline in Azure. When it comes to automated testing, the pipeline consists of source codes for automation residing in Azure Repos. The next step is to queue the codes for execution in the CI pipeline and produce results. Once the results are published, the same is saved as an artifact to automatically trigger the next step, i.e. shipping application code to production branch.

Manual to Automated Testing (cont.)



-TESTING-CI

Pipeline
Build pipeline

Get sources
psirAutoTesting master

Agent job 1
Run on agent

Queue Jenkins Job:
Jenkins Queue Job

Download artifacts produced by
Jenkins download artifacts

Publish Artifact: drop
Publish build artifacts

Name *
TESTING-CI

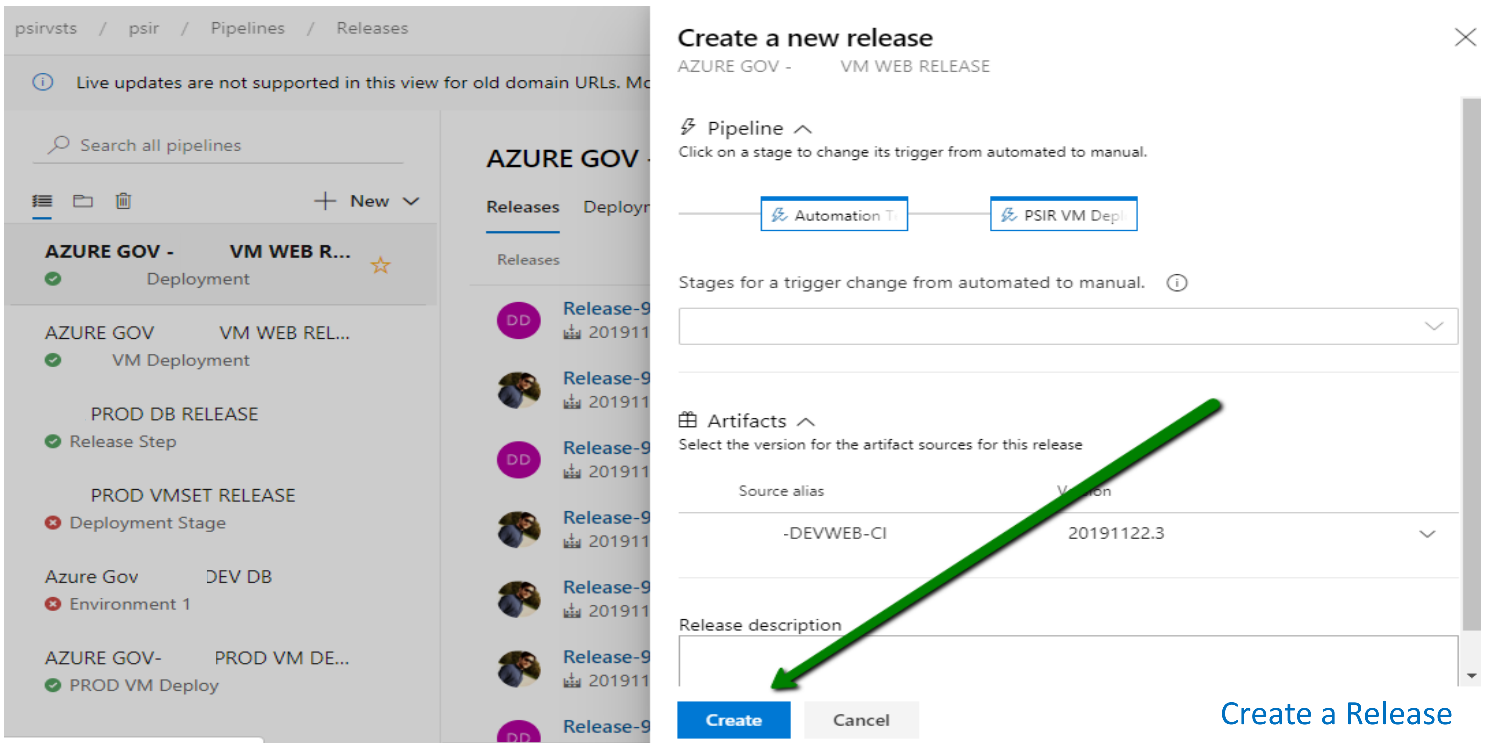
Agent pool * | Pool information | Manage
Hosted Ubuntu 1604

Parameters | Unlink all
Jenkins service connection * | Manage
-testing-jenkins

Configure the Azure Pipeline

Create a Release

The next step in this automated CI/CD pipeline is to create a release in order to test application code and deploy the same in production branch.



psirvsts / psir / Pipelines / Releases

Live updates are not supported in this view for old domain URLs. Mc

AZURE GOV - VM WEB R...
Deployment

AZURE GOV VM WEB REL...
VM Deployment

PROD DB RELEASE
Release Step

PROD VMSET RELEASE
Deployment Stage

Azure Gov DEV DB
Environment 1

AZURE GOV- PROD VM DE...
PROD VM Deploy

AZURE GOV

Create a new release
AZURE GOV - VM WEB RELEASE

Pipeline
Click on a stage to change its trigger from automated to manual.

Stages for a trigger change from automated to manual.

Artifacts
Select the version for the artifact sources for this release

Source alias	Version
-DEVWEB-CI	20191122.3

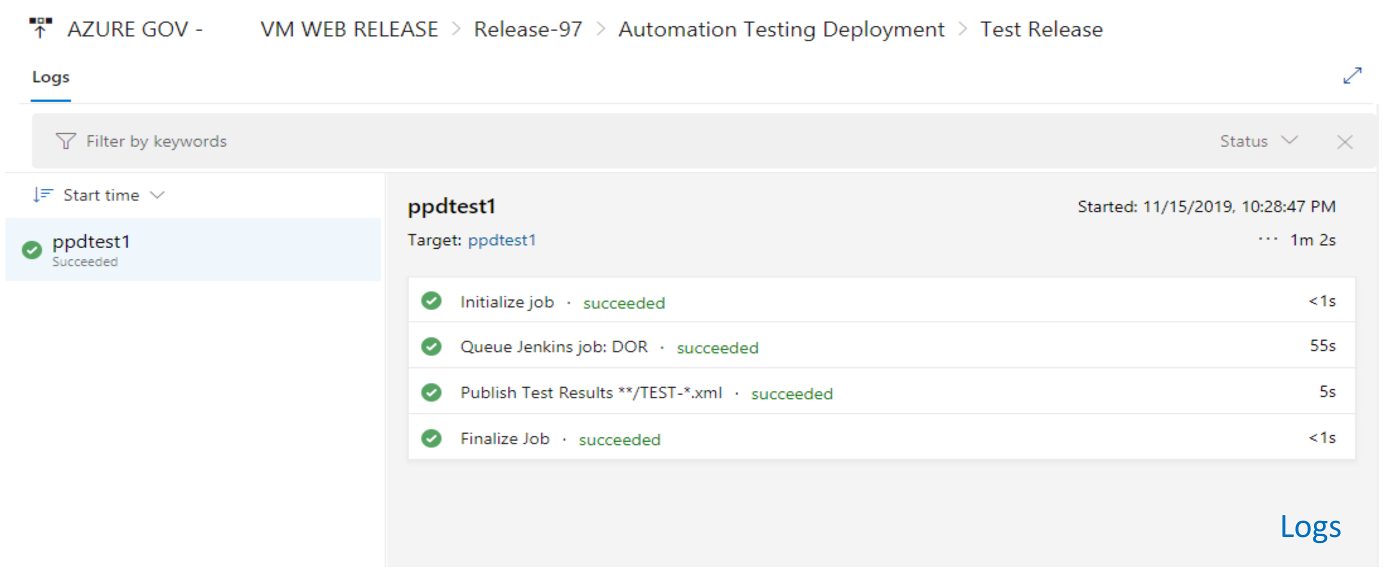
Release description

Create a Release

Manual to Automated Testing (cont.)

Logs

While the Pipeline jobs are running, Data-Core could go in and see the list of jobs and their respective logs.


AZURE GOV - VM WEB RELEASE > Release-97 > Automation Testing Deployment > Test Release

Logs

Filter by keywords | Status

Start time

ppdtest1
Succeeded

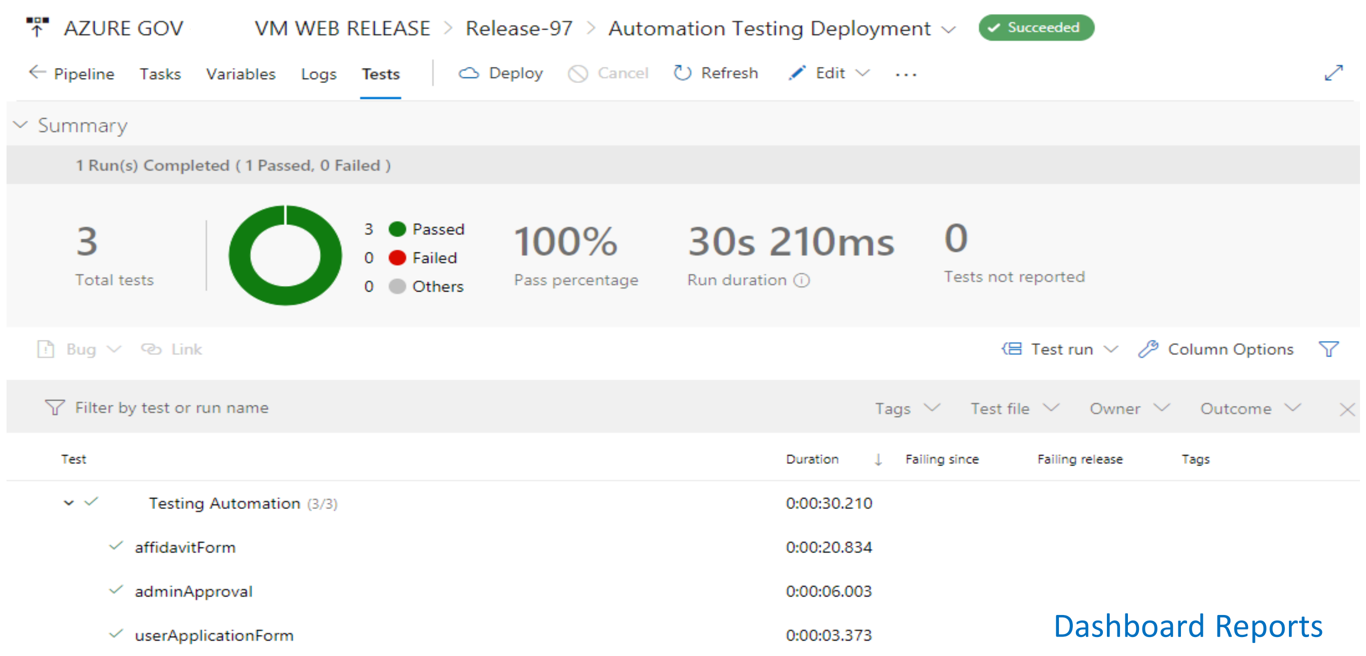
ppdtest1 — Started: 11/15/2019, 10:28:47 PM
Target: ppdtest1 — 1m 2s

```
Initialize job · succeeded                              <1s
Queue Jenkins job: DOR · succeeded                      55s
Publish Test Results **/TEST-*.xml · succeeded           5s
Finalize Job · succeeded                                <1s
```

Logs

Dashboard Reports

Azure DevOps has built-in dashboards to display the test execution result. Various graphical charts and presentations are available and can be shared with concerned stakeholders.


AZURE GOV VM WEB RELEASE > Release-97 > Automation Testing Deployment — Succeeded

Pipeline | Tasks | Variables | Logs | Tests | Deploy | Cancel | Refresh | Edit

Summary

1 Run(s) Completed (1 Passed, 0 Failed)

3 Total tests — 3 Passed, 0 Failed, 0 Others — 100% Pass percentage — 30s 210ms Run duration — 0 Tests not reported

Bug | Link | Test run | Column Options

Filter by test or run name | Tags | Test file | Owner | Outcome

Test	Duration	Failing since	Failing release	Tags
Testing Automation (3/3)	0:00:30.210			
affidavitForm	0:00:20.834			
adminApproval	0:00:06.003			
userApplicationForm	0:00:03.373			

Dashboard Reports

Benefits

1. **Excel for test data management:** The test data is managed in Microsoft Excel which enables it to be easy to use, read and maintain, even for resources without knowledge of the automation tool used.
2. **Selective test execution:** With the help of TestNG, test cases can be executed as a batch and can also be executed based on various attributes, e.g. Category, Priority etc.
3. **Execution in multiple nodes:** The test cases are executed on various target machines with various software/hardware combinations, as deemed best for the project.
4. **HTML reporting:** The framework allows logging of each test step and publishing the same as a rich html style report that can be shared with stakeholders.
5. **Easy maintenance of automation code:** Data-Core used Microsoft Azure Repos as a version control system for maintaining automation codes. This allows the team of testers to continuously merge their automation codes in a single repository and on a regular basis.
6. **Automation of test execution and publishing report:** With the implementation of Azure CI/CD pipeline, the entire process of test execution, the results published in dashboards and the deployment of the application has been automated within a single workflow.

Cut test cycles by at least 50%

Conclusion

The out-of-the-box approach Data-Core & the client embraced has allowed for verification with graphs and charts. This would have been impossible with a single automation tool alone. The automation effort covers areas that would have been difficult to test manually, considering the combinations of data, and thereby has resulted in at least 50% lesser testing cycle per release of the application.

Discover the Data-Core Advantage.
Learn how we can help your company be more relevant, effective and efficient.

USA Headquarters
1500 JFK Blvd., Suite 624
Philadelphia, PA 19102
Tel: 215 243 1990
www.datacoresystems.com

Bristol, PA
111 Sinclair Road,
Bristol, PA 19007
Toll Free: 877 300 9529
Tel: 267 569 0800

Las Vegas, NV
1771 East Flamingo Road Suite B100
Las Vegas, NV 89119
Toll Free: 877 300 9529
Tel: 702 795 9559