curl -X POST -u "kombaths:1133fbd358927c134f9d0f0d851c92a0ab" \

"https://ci.sms.us.caas.oneadp.com/job/AIM/job/risk-assessment-service-automation/job/master/buildWithParameters" \

-d "Task=NoHarmTests" \

-d "Version=v1" \

-d "Env=fit" \

-d "Branch=master"

curl -X POST -u "USERNAME:APITOKEN" "https://your.jenkins.server/job/JOBNAME/buildWithParameters" \

-d "PARAM1=value1" -d "PARAM2=value2"

curl -X POST -u "YOUR\_USERNAME:YOUR\_API\_TOKEN" \

"https://ci.sms.us.caas.oneadp.com/job/AIM/job/Nexo\_Parallel\_2/buildWithParameters" \

-d "Env=FIT" \

-d "Run\_NEXO\_APIs=true" \

-d "Run\_Test\_E2E=false" \

-d "Test\_E2E=oid-service" \

-d "Nexo\_No\_Harm\_Newman\_API\_Tests=false" \

-d "Run\_Watchmen\_UI\_Job=false" \

-d "test=@spInitiated" \

-d "envToRun=local" \

-d "Run\_Risk\_Assessment\_Job=false" \

-d "Version=v1" \

-d "Notification\_Email=stephan.raju.kombathula@adp.com"

build job: 'AIM\_QA/jenkins-automation-aim-qa-cg', parameters: [string(name: 'Pod', value: 'CL02'), string(name: 'Env', value: 'FIT'), booleanParam(name: 'UI\_Flows', value: false), extendedChoice(name: 'USERS', value: ''), booleanParam(name: 'JAX\_APIs', value: false), booleanParam(name: 'SCIM\_APIs', value: false), booleanParam(name: 'FedSSO\_CONSENT\_PAGE\_APIs', value: false), booleanParam(name: 'CG\_Newman\_APIs', value: false), extendedChoice(name: 'CG\_Newman\_API\_Tests', value: ''), booleanParam(name: 'OTP\_Flows', value: false), extendedChoice(name: 'OTP\_Logins', value: ''), string(name: 'Notification\_Email', value: 'stephan.raju.kombathula@adp.com')]

curl -X POST -u "YOUR\_USERNAME:YOUR\_API\_TOKEN" \

"https://ci.sms.us.caas.oneadp.com/job/AIM\_QA/job/jenkins-automation-aim-qa-cg/buildWithParameters" \

-d "Pod=CL02" \

-d "Env=FIT" \

-d "UI\_Flows=false" \

-d "USERS=" \

-d "JAX\_APIs=false" \

-d "SCIM\_APIs=false" \

-d "FedSSO\_CONSENT\_PAGE\_APIs=false" \

-d "CG\_Newman\_APIs=false" \

-d "CG\_Newman\_API\_Tests=" \

-d "OTP\_Flows=false" \

-d "OTP\_Logins=" \

-d "Notification\_Email=stephan.raju.kombathula@adp.com"

Please feel free to stay in touch. You can reach me at [Sanjoli@gmail.com](mailto:Sanjoli@gmail.com) or in LinkedIn <https://www.linkedin.com/in/sanjoli-ahuja/>

**AMAZON Q CLI—QA Prompts**

**Deploy the scripts** to a shared location accessible by all team members

2. **Create a central configuration** with all your Jenkins jobs

3. **Set up environment variables** for different team members' Jenkins credentials

4. **Document the prompt patterns** that team members can use

Trigger QA\_FPS\_Inception job with fit and v1 parameters

Trigger job old-2

Example of how to get output

> I'll trigger the QA\_FPS\_Inception Jenkins job with the parameters "fit" and "v1" using our simple-jenkins-bot project.

cd /Users/kombaths/aim\_qa\_qchat && ./start-qchat-advanced.sh

<https://bitbucket.es.ad.adp.com/scm/~kombaths/q_chat.git>

<https://bitbucket.es.ad.adp.com/projects/WM/repos/otg-ui-automation/browse>

<https://bitbucket.es.ad.adp.com/projects/LIV/repos/lifion-mobile-automation/browse>

# Jenkins Server Settings

jenkins.url=https://ci.sms.us.caas.oneadp.com

jenkins.username=kombaths

jenkins.token=1133fbd358927c134f9d0f0d851c92a0ab

regex = “^(?=.\*[0-9])(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[@#$%^&-+=()])(?=\\S+$).{8, 20}$”

**Explanation:**

* ^ used for starting character of the string.
* (?=.\*[0-9]) used for a digit must occur at least once.
* (?=.\*[a-z]) used for a lowercase alphabet must occur at least once.
* (?=.\*[A-Z]) used for an upper case alphabet that must occur at least once in the substring.
* (?=.\*[@#$%^&-+=()] used for a special character that must occur at least once.
* (?=\\S+$) white spaces don’t allow in the entire string.
* .{8, 20} used for at least 8 characters and at most 20 characters.

• • $ used for the end of the string.

**Playwright:**

Const [new page]=await promise.all({

Context.waitforevent(‘’page),

Page.clocl(‘a[target=‘b’lank]’))

Await page.waitforloatetate

Fixtures are test setup/ teardown utilities

Page.locator(‘’#username).fill(‘Stephen’)

Page.locator(‘’#username).click

Page.locator(‘’text=login).click

Button:hastext(submit)

getbyrole(button,{name:submit}).clkick

getbyrole(link,{name:home}).clkick

getbyplaceholder(‘entername’).fill

Getbylabel

Test.each

Const login datsa:

[

{user name and password

}]

Test.descript(‘’,()=>)

{

Test.each(logindata)}(‘’,asynchronous{page} data==>)

{

}

For of loop

For (const user of user)

Json files