GRACE HOPPER CELEBRATION

ANITA B.ORG

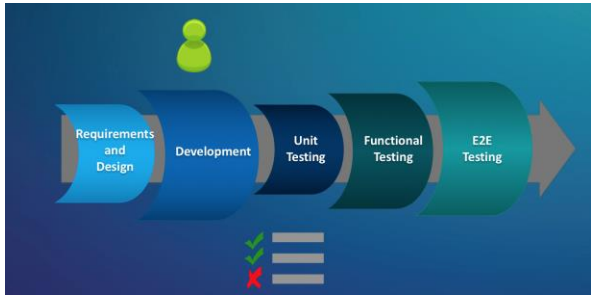
Fast and Fearless: Shift Left in Automated Testing

Sangy Santharam – Architect, Intuit

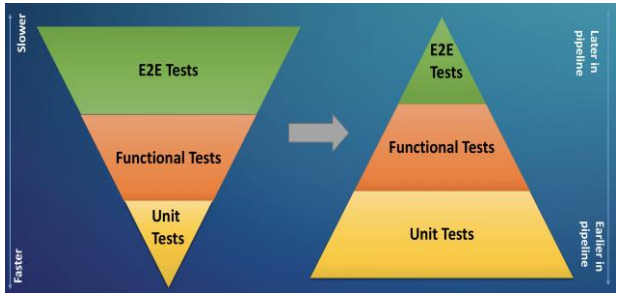
Sree Dasari – Staff Engineer, Intuit

Agenda

1. What is Shift Left?



2. How to achieve it?



3. Metrics

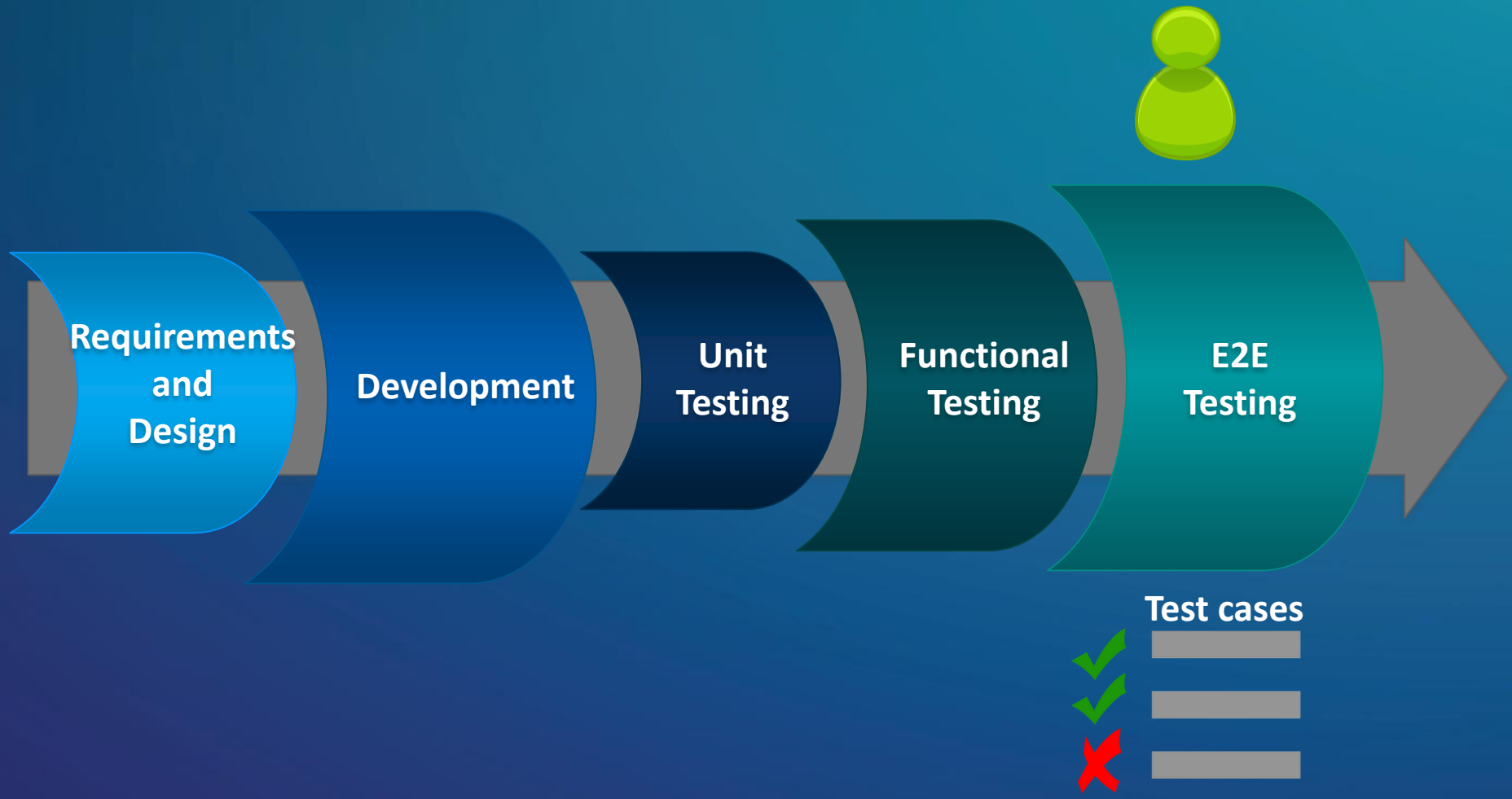
What is Shift Left?

Traditional phases of testing are not scalable for Agile development

Agile + Speed

Not Scalable

Shifting Left in engineering culture and test cases provides continuous feedback on quality

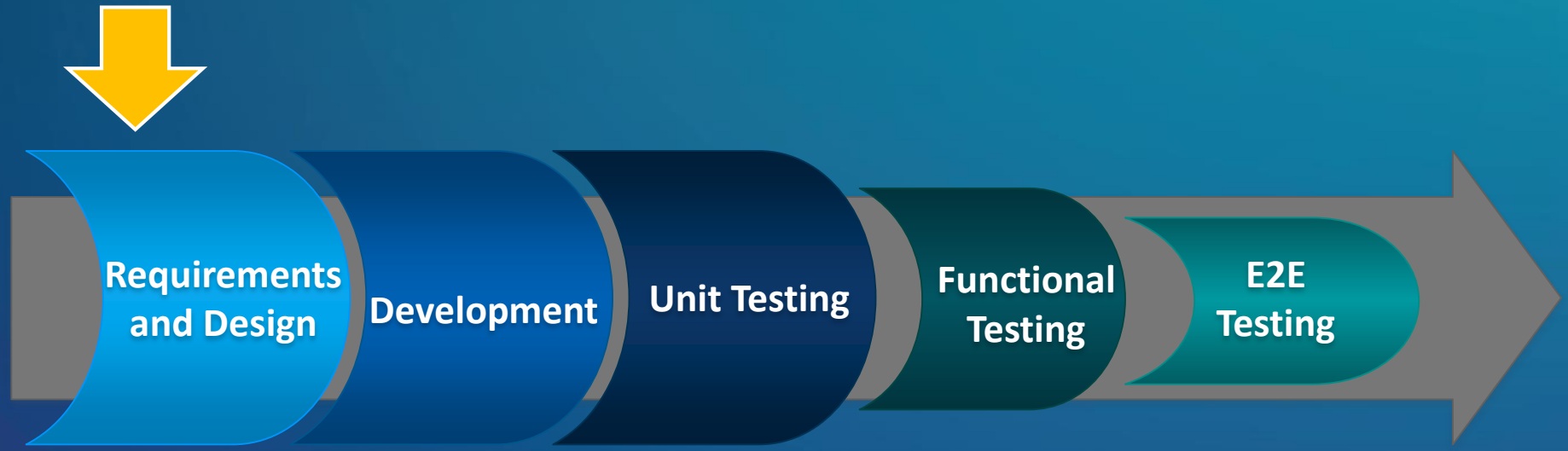


High quality earlier in the pipeline enables speed

Does it stop here?

Strong partnership with Product Manager and Architect enables the team to move faster with fewer defect emergencies

Quality Starts Here

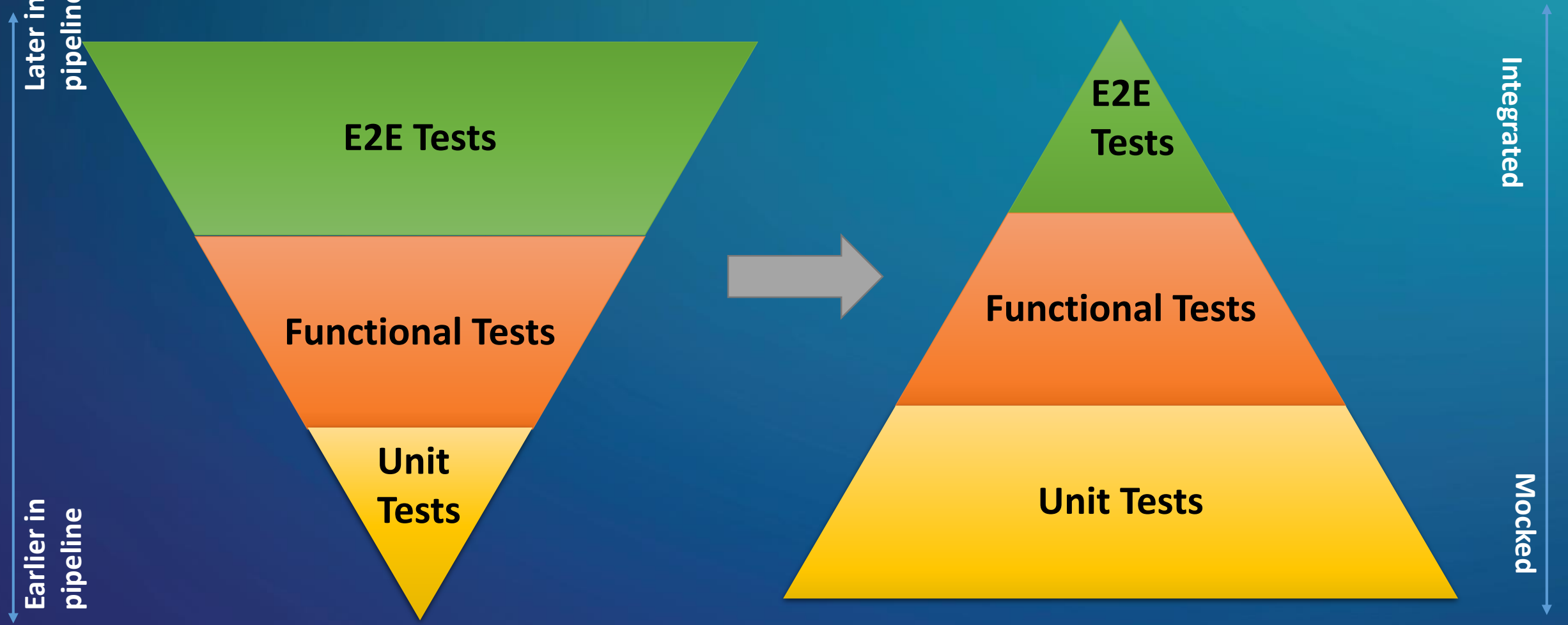


So …
What is Shift Left?

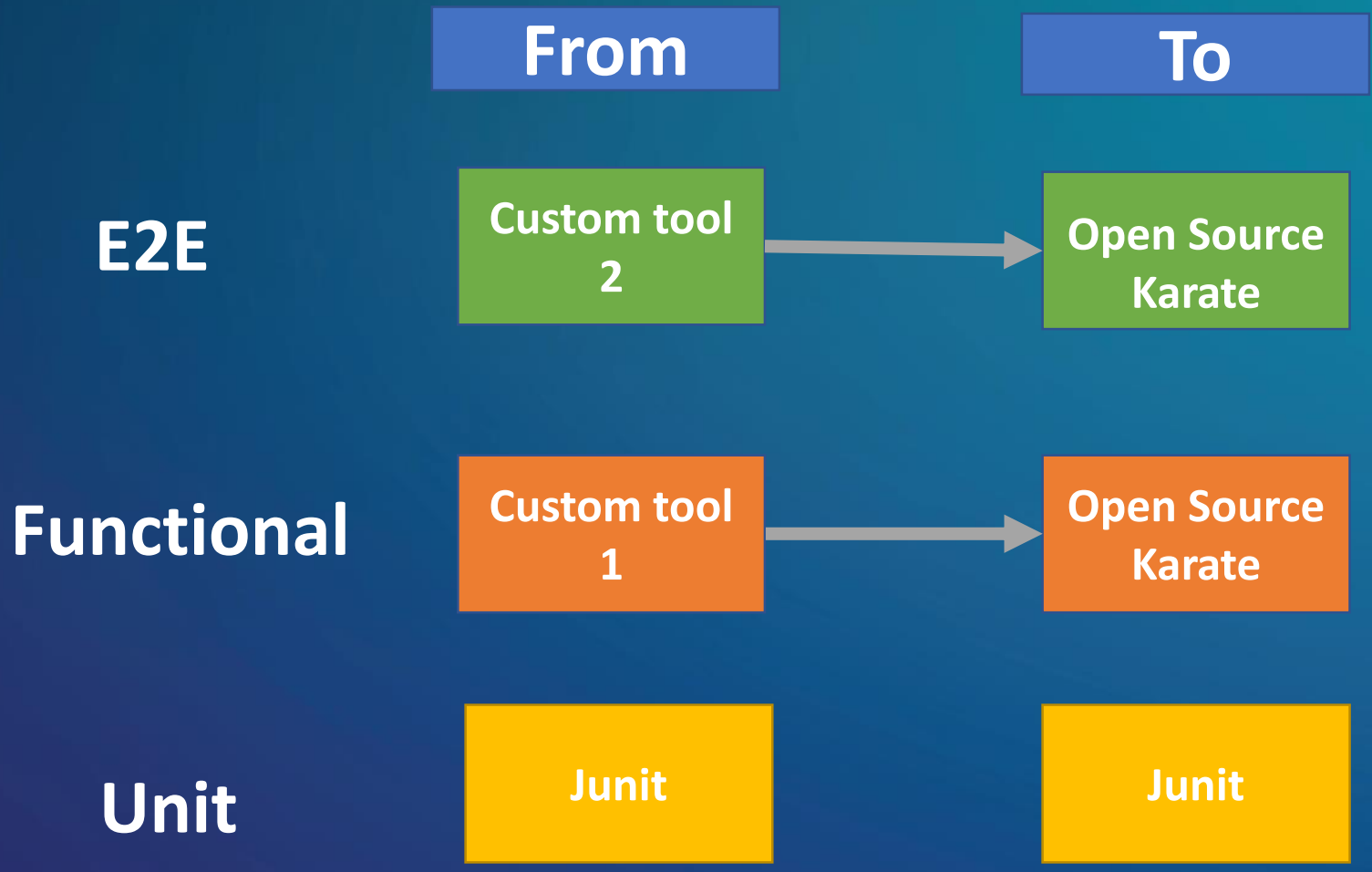
- Quality starts from requirements and design
- Move test cases closer to developer
- Developer takes ownership of quality

How did we achieve it?

180 degree shift in test cases enables code quality earlier in the pipeline



Right tools improve developer speed and enable faster releases



- Faster Automation
- Test case reuse
- Run locally

Scenario

Problem statement

Implement a conference registration system. (Imagine the registration process you went through for GHC.)

- **Create a login.**
- **Register for the conference by providing the payment information.**

<u>Required services</u>

1. **Login service**
2. **Registration service**

Use case #1: create a login

Use cases in Cucumber open source framework

Template

Given	Context
And	More Context
When	Event
And	Another Event
Then	Outcome
And	Another outcome

Given	User provides name
And	email address
And	password
When	User submits the request to create login
Then	System validates the email address
And	System creates the login for the user with email address as username
And	System responds with the username

Exercise #1

Implement a conference registration system. (Imagine the registration process you went through for GHC.)

- Create a login.
- Register for the conference by providing the payment information.

Given	Context
And	More Context
When	Event
And	Another Event
Then	Outcome
And	Another outcome

Exercise #1: solution

- Register for the conference by providing the payment information.

Given	User has already logged in
And	User provides username and valid credit card information
When	User submits the registration request
Then	System validates the credit card data
And	System creates the registration record for the user
And	System responds with the confirmation number

Demo

Exercise #2: create login negative path

Implement a conference registration system. (Imagine the registration process you went through for GHC.)

- **Create a login.**

Given	**Context**
And	**More Context**
When	**Event**
And	**Another Event**
Then	**Outcome**
And	**Another outcome**

One Possible negative test scenario

- User email already exists

Given url	
And request	
When method	
Then status	
And match response	

#GHC19

Exercise #2: solution

Given	User already has created login with email address Ex: “mary@xx.xom”
And	User provides the same email address “mary@xx.xom”
And	User provides password
When	User submits the request to create login
Then	System validates the email address
And	System responds with the error message saying that login already exists

Karate code sample

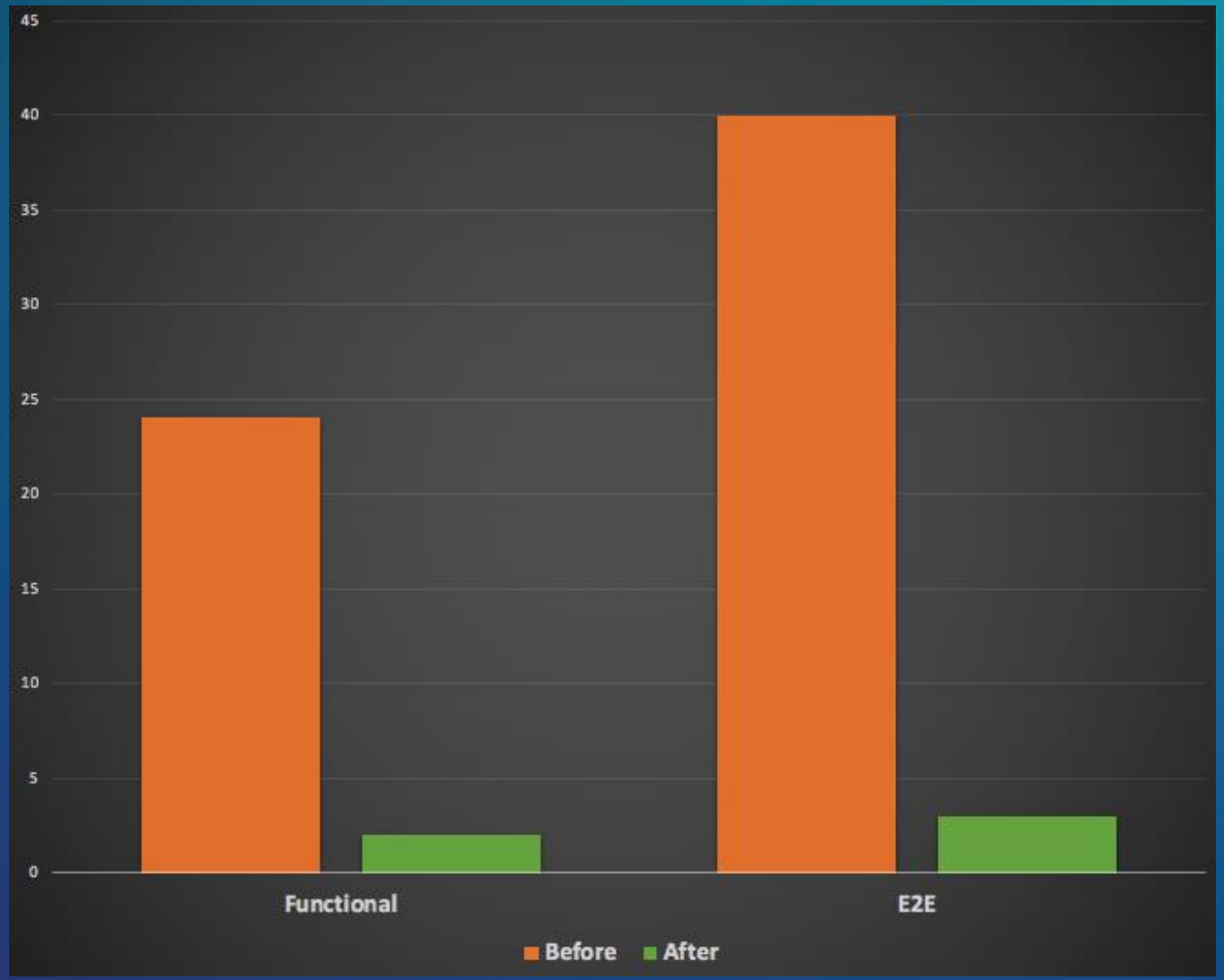
```
Given url localhost:8080/ghc/login
And request {“firstName”: “Mary”, “lastName”: “Doe”,
“email”: “mary@xx.com”,
“password”: “xyzAbc#12” }
 When method post
 Then status 400
 And match response == {errorCode: “LOGIN-001“ , errorMessage: “Login already exists with the given email”}
```

So …
How to achieve it?

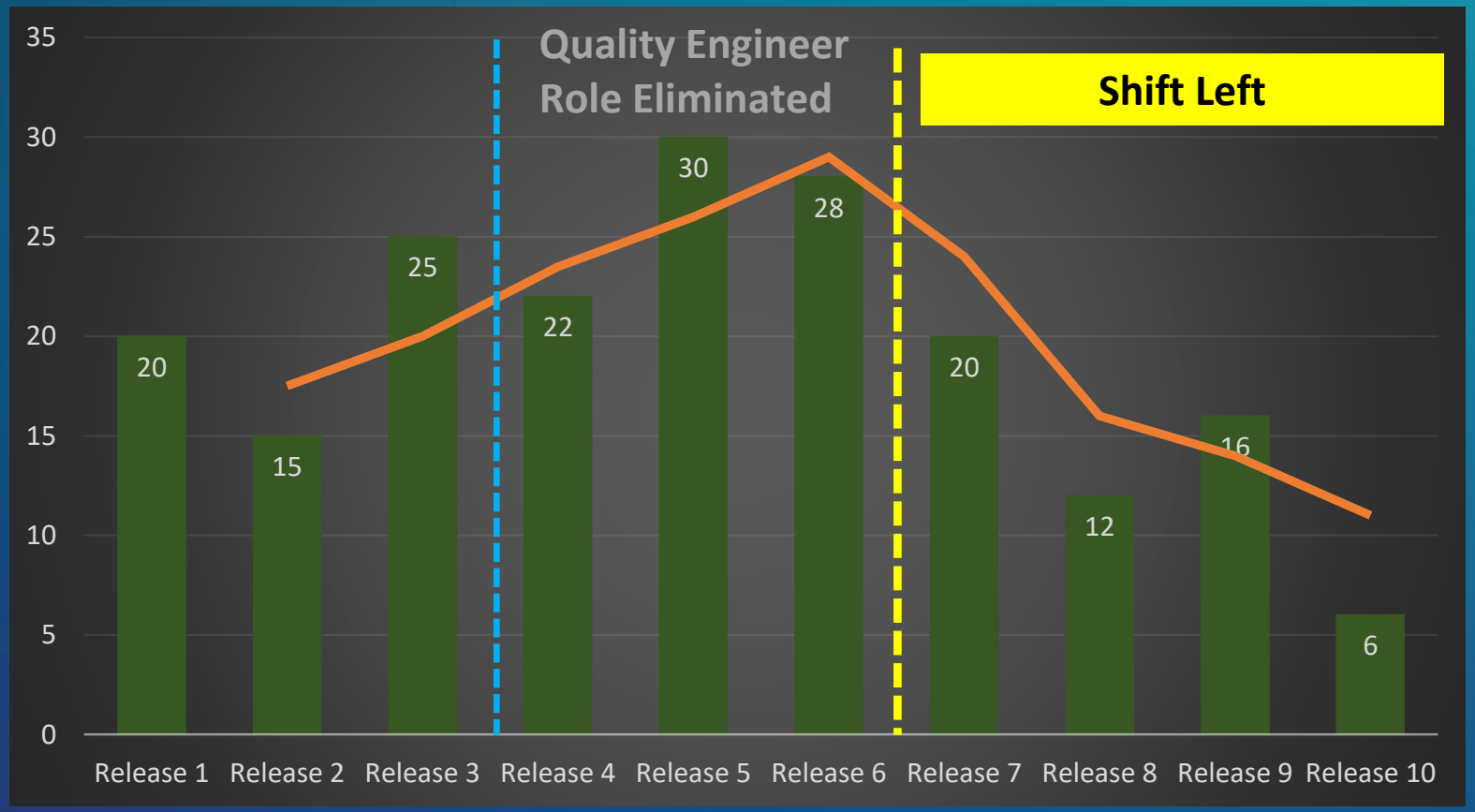
- Write use cases clearly using Cucumber open source framework
- Write tests for each context defined in use case using Karate open source framework
- Have higher test coverage early in the pipeline

Metrics

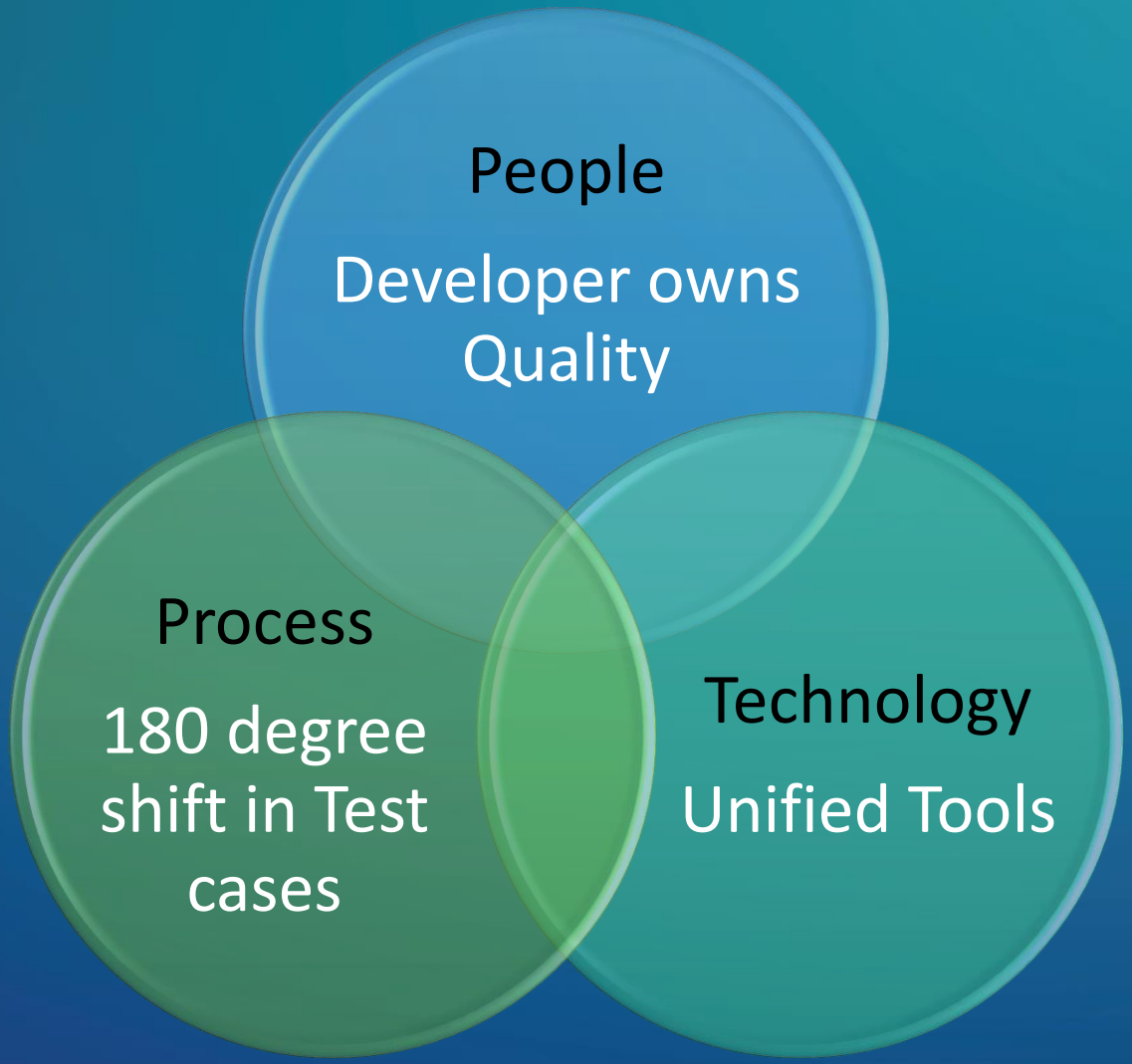
Test execution time reduced from days to hours for 50+ APIs



Defects in E2E before and after Shift Left



So …
What makes shift left successful?



Shift Left will enable development teams to deliver fast, frequent and fearless releases to customers.

Questions?

You can reach us @ 

https://www.linkedin.com/in/sangysantharam/

https://www.linkedin.com/in/sree-dasari

Please provide your feedback by taking the session survey in GHC mobile app

#GHC19

Appendix

Exercise #2 – TestCase2 Solution

Given	User provides a valid email address
And	User provides password with less than 8 characters
When	User submits the request to create login
Then	System validates the email address and password
And	System responds with the error message saying that password should be al least 8 characters or more

Karate code sample

```
Given url localhost:8080/ghc/login
And request {“firstName”: “Mary”, “lastName”: “Doe”,
“email”: “mary@xx.com”,
“password”: “xyz” }
 When method post
 Then status 400
 And match response == {errorMessage: “Password should be at least 8 characters or more.”}
```

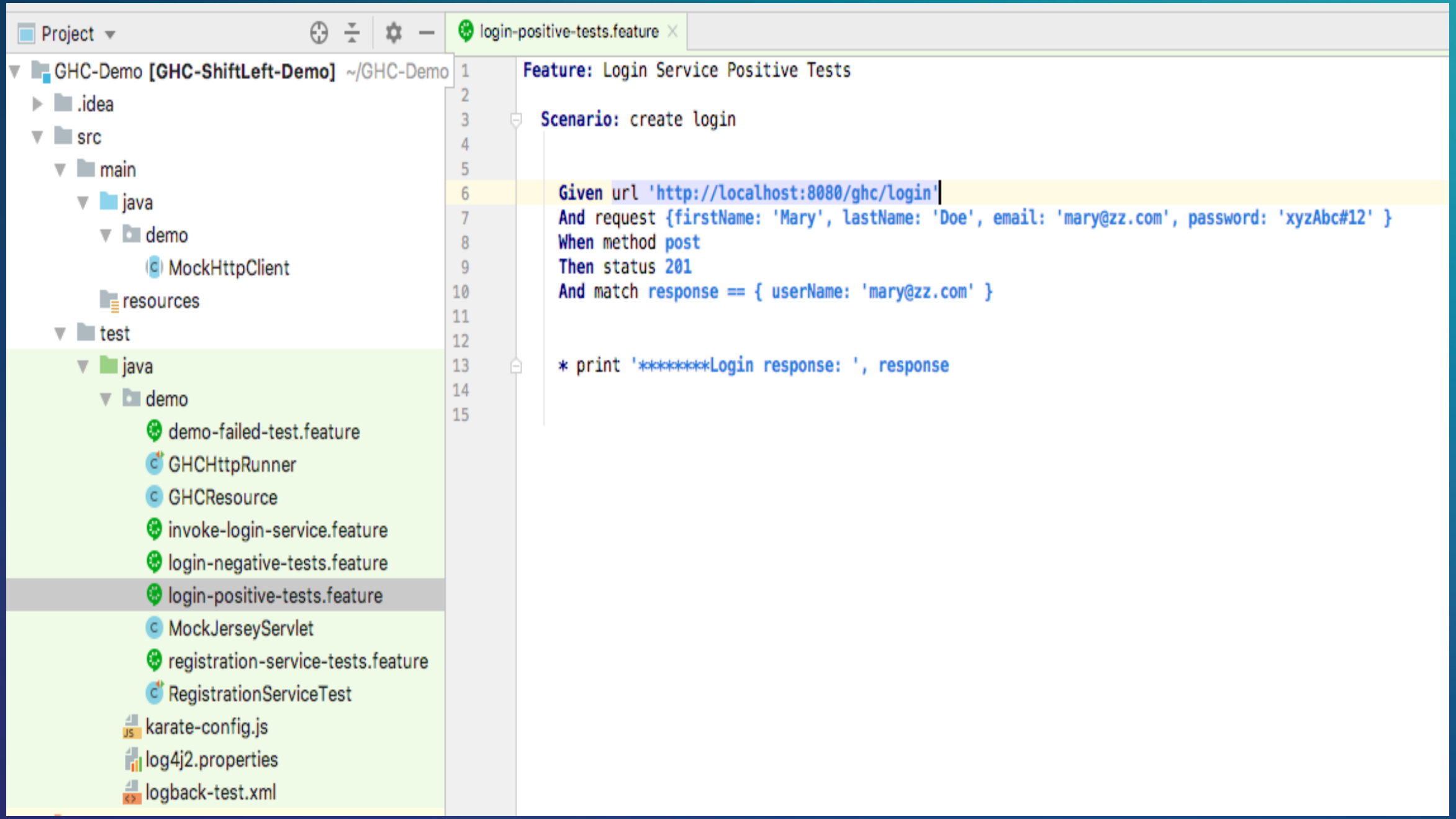
Exercise #2 – TestCase3 Solution

Given	User provides a invalid email address
And	User provides valid password
When	User submits the request to create login
Then	System validates the email address and password
And	System responds with the error message saying that email address is not valid.

Karate code sample

```
Given url localhost:8080/ghc/login
And request {“firstName”: “Mary”, “lastName”: “Doe”,
“email”: “mary@xx.com.com”,
“password”: “xyzABc#12” }
 When method post
 Then status 400
 And match response == {errorMessage: “Email address is invalid.”}
```

Demo


Project
GHC-Demo [GHC-ShiftLeft-Demo] ~/GHC-Demo
.idea
src
main
java
demo
MockHttpClient
resources
test
java
demo
demo-failed-test.feature
GHCHttpRunner
GHCResource
invoke-login-service.feature
login-negative-tests.feature
login-positive-tests.feature
MockJerseyServlet
registration-service-tests.feature
RegistrationServiceTest
karate-config.js
log4j2.properties
logback-test.xml
login-positive-tests.feature
1 Feature: Login Service Positive Tests
2
3 Scenario: create login
4
5
6 Given url 'http://localhost:8080/ghc/login'
7 And request {firstName: 'Mary', lastName: 'Doe', email: 'mary@zz.com', password: 'xyzAbc#12' }
8 When method post
9 Then status 201
10 And match response == { userName: 'mary@zz.com' }
11
12
13 * print '********Login response: ', response
14
15

registration-service-tests.feature

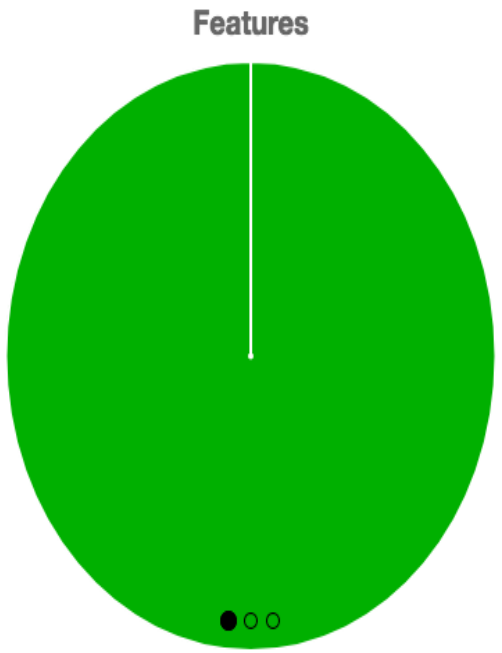
RegistrationServiceTes

registration-service-tests.feature

```
Feature: Registration Service Tests

  Scenario: Register for GHC - successful

    * def registrationRequest = {userName: 'mary@xx.com', nameOnCard: 'Mary Doe', cardType: 'Visa', cardNumber: '1234 5678 9101 6789', cvv: '1234', expirationMonth:'10', expirationYear: '2021'}

    * call read('classpath:demo/invoke-login-service.feature')

    * print '********Registration request: ', registrationRequest

    Given url demoBaseUrl
    And path 'ghc/register'
    And request registrationRequest
    When method post
    Then status 200
    And match response == { confirmationNumber: '#uuid' }

    * print '********Registration response: ', response
```

Features Statistics

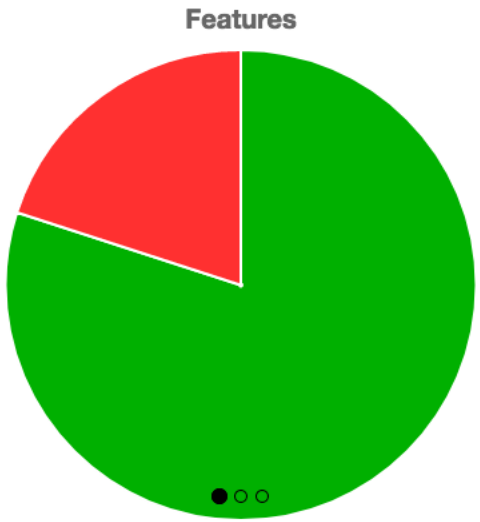
The following graphs show passing and failing statistics for features



	Steps						Scenarios			Features	
Feature	Passed	Failed	Skipped	Pending	Undefined	Total	Passed	Failed	Total	Duration	Status
demo/login-negative-tests.feature	27	0	0	0	0	27	3	0	3	604ms	Passed
demo/login-positive-tests.feature	6	0	0	0	0	6	1	0	1	241ms	Passed
demo/registration-service-tests.feature	20	0	0	0	0	20	1	0	1	510ms	Passed
demo/invoke-login-service.feature	9	0	0	0	0	9	2	0	2	065ms	Passed
4	**62**	**0**	**0**	**0**	**0**	**62**	**7**	**0**	**7**	**1s 421ms**	
	100.00%	**0.00%**	**0.00%**	**0.00%**	**0.00%**		**100.00%**	**0.00%**			**100.00%**

Features Statistics

The following graphs show passing and failing statistics for features



	Steps						Scenarios			Features	
Feature	Passed	Failed	Skipped	Pending	Undefined	Total	Passed	Failed	Total	Duration	Status
demo/login-negative-tests.feature	27	0	0	0	0	27	3	0	3	242ms	Passed
demo/login-positive-tests.feature	6	0	0	0	0	6	1	0	1	224ms	Passed
demo/demo-failed-test.feature	3	1	2	0	0	6	0	1	1	208ms	Failed
demo/registration-service-tests.feature	20	0	0	0	0	20	1	0	1	296ms	Passed
demo/invoke-login-service.feature	9	0	0	0	0	9	2	0	2	136ms	Passed
5	**65**	**1**	**2**	**0**	**0**	**68**	**7**	**1**	**8**	**1s 109ms**	
	95.59%	**1.47%**	**2.94%**	**0.00%**	**0.00%**		**87.50%**	**12.50%**			**80.00%**

Feature Report

	Steps						Scenarios			Features	
Feature	Passed	Failed	Skipped	Pending	Undefined	Total	Passed	Failed	Total	Duration	Status
demo/login-test-failure.feature	3	1	2	0	0	6	0	1	1	185ms	Failed

Feature demo/login-test-failure.feature

Login Service Positive Tests

Scenario create login 185ms

Steps

Given url 'http://localhost:8080/ghc/login' 008ms

And request {firstName: 'Mary', lastName: 'Doe', email: 'mary@zz.com', password: 'xyzAbc#12' } 000ms

When method post 176ms

Then status 200 000ms

com.intuit.karate.exception.KarateException: login-test-failure.feature:9 - status code was: 201, expected: 200, response time: 86, url: http://localhost:8080/ghc/login, response: {"userName":"mary@zz.com"}

```
com.intuit.karate.exception.KarateException: login-test-failure.feature:9 - status code was: 201, expec
```

Doc string

And match response == { userName: 'mary@zz.com' } 000ms

* print '********Login response: ', response 000ms

References

1. Karate - https://github.com/intuit/karate
2. Cucumber - https://docs.cucumber.io/guides/10- minute-tutorial/
3. Robin Beck – https://www.lynda.com/Cucumber- tutorials/Behavior-Driven-Development
4. https://github.com/sree-dasari/GHC-SHIFTLEFT-DEMO

Speakers

Sangy Santharam

Sangy Santharam is an Architect at Intuit leading functional end to end teams and driving technology adoption. She is a thought leader and drives innovation and builds out organization wide capabilities. Prior to Intuit, she was an engineer at Qualcomm with research experience in indoor positioning systems and fleet management system. She led an ERP systems team at Oracle. She has an Master of Science in computer science from University of Southern California.

Sree Dasari

Sree Dasari is a Lead Software Engineer at Intuit. She led several successful projects to solve Intuit's global business needs. She is passionate about delivering high quality software. She led the Shift Left Quality transformation for product development teams consisting of 85+ Engineers. Prior to Intuit, she worked at Apple as a software engineer, designing and implementing authentication services to support iTunes and the Apple Store. She earned a Master of Science degree in computer science from Santa Clara University.