

ASCENDION

Capstone Project



Personal BACKGROUND

(Name, Past Experience, Qualification, Career Summary)

Name : Swarna Sai Sreekar

Past Experience : Experience in previous Trainings with MERN stack, Spring Boot and GEN-AI Training

Qualification : Bachelor of Technology in CSE

Career Summary : To work in a dynamic environment that uses my skills and expertise in process of growth and development while allowing me to learn and enrich my competencies



Key Takeaways/Learnings from the Program (HTD)

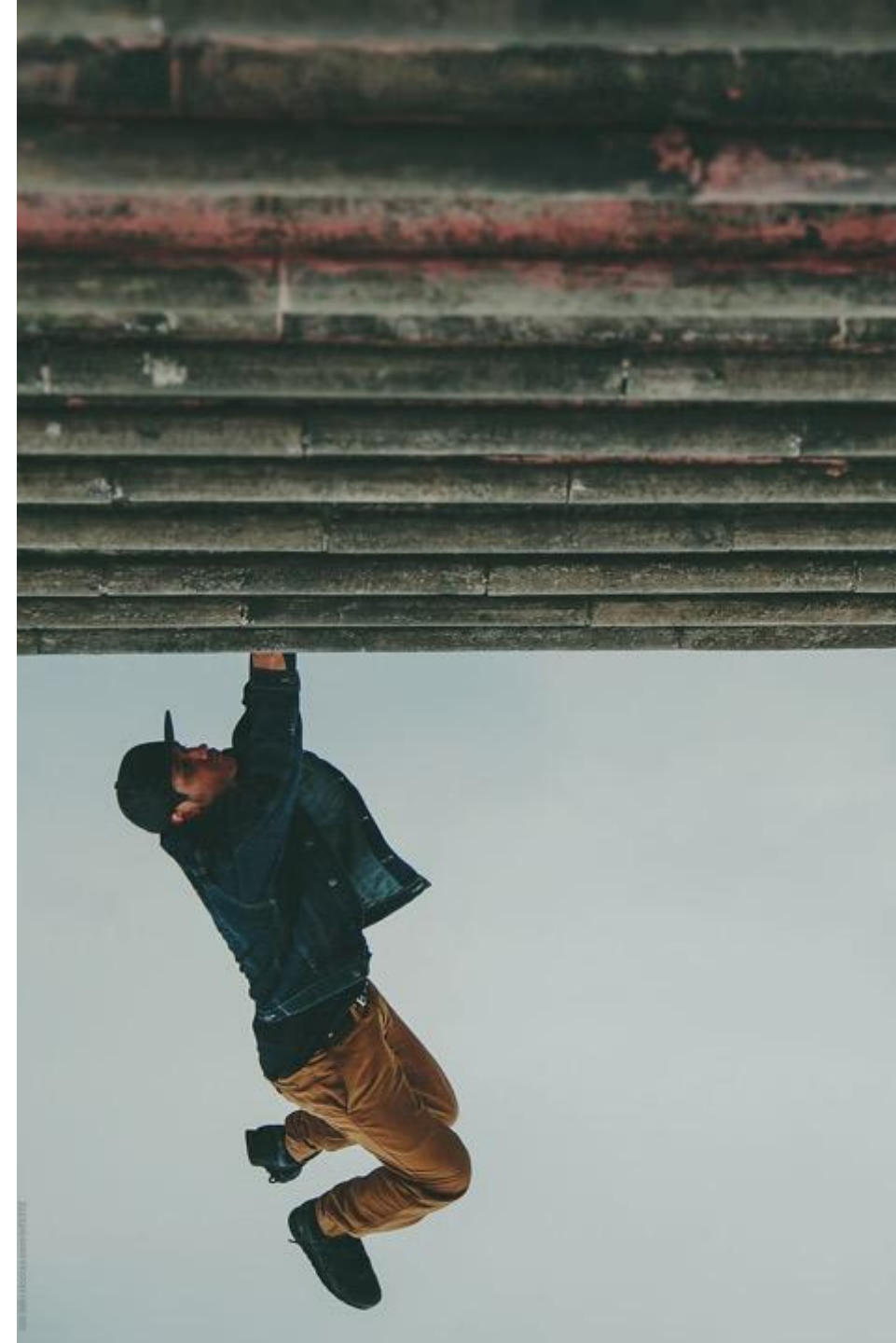
During my two-month training at Ascendion, I gained in-depth knowledge and practical experience in several key areas :

- **End-to-End Quality Engineering (QE) Expertise**
Skilled in manual and automation testing across functional, performance, and security domains, with strong test design and execution practices.
- **Full-Stack Test Automation**
Hands-on experience building scalable Selenium WebDriver frameworks with Java, TestNG, POM, Log4j, and Maven for CI/CD pipelines.
- **API Automation & Integration**
Proficient in REST API testing using Postman and Newman CLI, validating responses and integrating with Jenkins for automation.
- **Performance Testing**
Created JMeter scripts simulating realistic load scenarios with CLI-based execution and reporting for API and web performance.
- **Agile Testing with Jira & Xray**
Managed testing workflows in Jira Scrum boards, including sprint planning, bug tracking, and test executions using Xray.
- **DevOps & CI/CD Testing**
Implemented Shift-Left and Shift-Right strategies integrated with Jenkins, GitHub Actions, and SonarQube for continuous quality feedback.
- **Exploratory, Non-Functional & Mobile Testing**
Experienced in mobile automation using Appium on real devices and simulators.



Problem Statement of the Capstone Project

- Demonstrate end-to-end Quality Engineering skills with automated testing for web and APIs.
- Develop scalable Selenium WebDriver UI tests using Page Object Model, logging, and assertions.
- Create Postman API automation collections and run via Newman locally and in CI/CD.
- Manage test execution, reporting, and tracking using Jira and Xray.
- Optionally implement mobile automation (Appium) and performance testing (JMeter).
- Integrate tests and code into GitHub with branching strategies.
- Optionally configure Jenkins pipelines for automated test execution.
- Ensure robust, flexible, and maintainable automation supporting agile and DevOps workflows.



ASCENDION

Reference Links :

Links

Web Application

- <https://demoqa.com/>

API Base URL

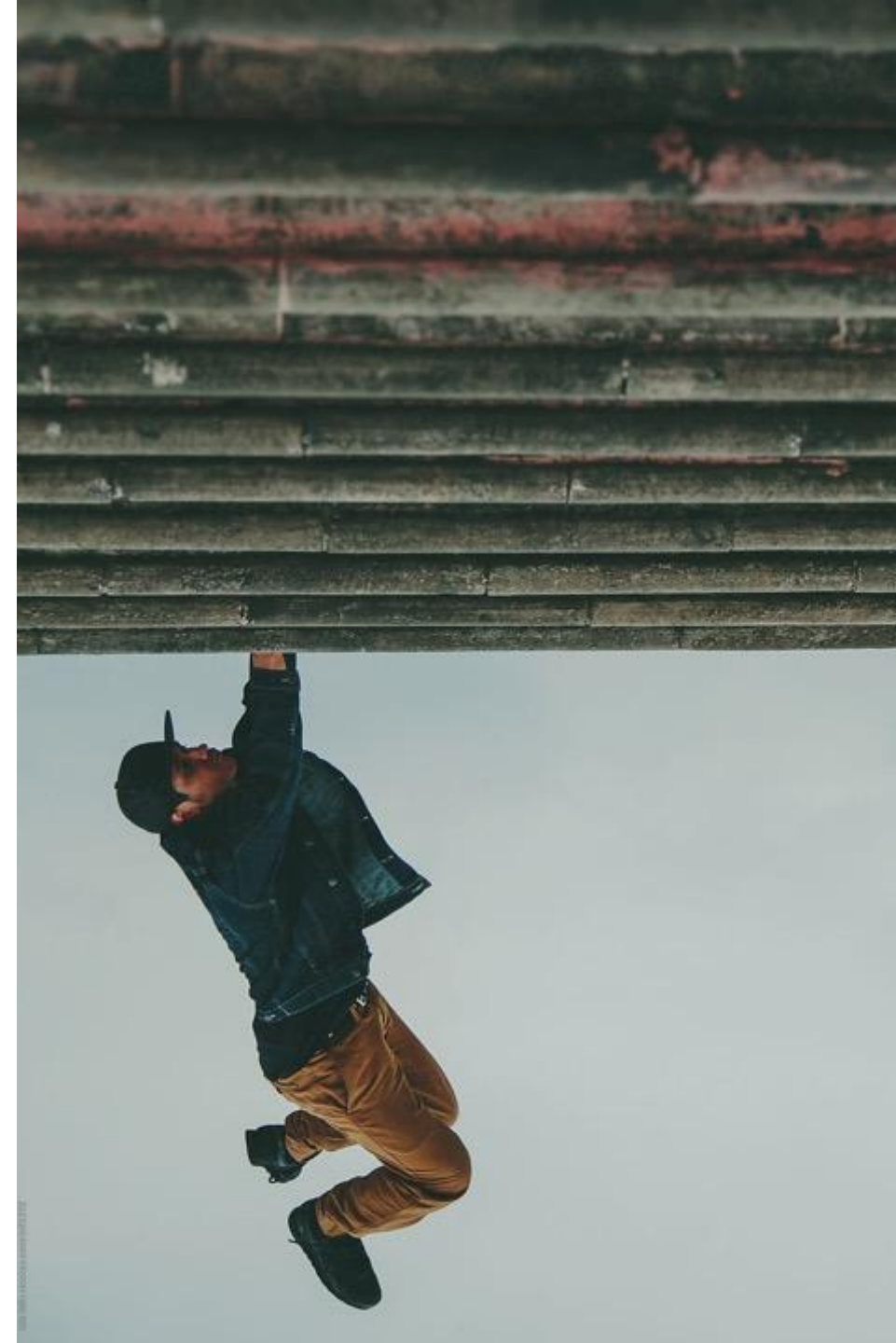
- <https://api.restful-api.dev/objects>

GitHub Repository

- https://github.com/sreekar0122/CAPSTONE_COGNIXIA_WEB_API_MAY2025

JIRA or Documentation (Optional)

- <https://ascendionlearning.atlassian.net/jira/software/c/projects/SR/summary>



Login Functionality Page & Test of the Selenium project with screenshots

```

10
11 public LoginPage(WebDriver driver) {
12     this.driver = driver;
13     PageFactory.initElements(driver, this);
14 }
15
16 @FindBy(id = "userName")
17 WebElement username;
18
19 @FindBy(id = "password")
20 WebElement password;
21
22 @FindBy(id = "login")
23 WebElement loginBtn;
24
25 @FindBy(id = "name")
26 WebElement loginMessage;
27
28 public void enterUsername(String user) {
29     username.clear();
30     username.sendKeys(user);
31 }
32
33 public void enterPassword(String pass) {
34     password.clear();
35     password.sendKeys(pass);
36 }
37
38 public void clickLogin() {
39     loginBtn.click();
40 }
41
42 public String getLoggedInUsername() {
43     return driver.findElement(By.id("userName-value")).getText();
44 }
45
46 public String getLoginMessage() {
47     return loginMessage.getText();
48 }

```

```

15 @Test(priority = 1)
    Run | Debug
16 public void loginWithInvalidCredentials() throws InterruptedException {
17     driver.get(constant.LOGIN_PAGE_URL);
18     LoginPage loginPage = new LoginPage(driver);
19
20     loginPage.enterUsername(constant.INVALID_USERNAME);
21     loginPage.enterPassword(constant.INVALID_PASSWORD);
22     loginPage.clickLogin();
23
24     Thread.sleep(1000);
25
26     Assert.assertTrue(
27         loginPage.getLoginMessage().contains(constant.INVALID_LOGIN_MSG),
28         "Invalid login warning not displayed."
29     );
30     log.info("Invalid login tested successfully.");
31 }
32
33 @Test(priority = 2)
    Run | Debug
34 public void loginWithWrongUsername() throws InterruptedException {
35     driver.get(constant.LOGIN_PAGE_URL);
36     LoginPage loginPage = new LoginPage(driver);
37
38     loginPage.enterUsername(constant.WRONG_USERNAME);
39     loginPage.enterPassword(constant.VALID_PASSWORD);
40     loginPage.clickLogin();
41
42     Thread.sleep(1000);
43
44     Assert.assertTrue(
45         loginPage.getLoginMessage().contains(constant.INVALID_LOGIN_MSG),
46         "Wrong username test failed."
47     );
48     log.info("Wrong username login tested successfully.");
49 }
50

```


Test Base Before & After Method, Config.properties, TestNG.xml of the Selenium project

```

112 public class TestBase {
113     public WebDriver driver;
114     public static Properties prop = new Properties(); // Ensure it's initialized
115     public Logger log = Logger.getLogger(TestBase.class);
116
117     // Load config just once
118     static {
119         try {
120             FileInputStream fis = new FileInputStream("testdata/config.properties");
121             prop.load(fis);
122             PropertyConfigurator.configure("testdata/log4j.properties");
123         } catch (IOException e) {
124             e.printStackTrace();
125             throw new RuntimeException("Failed to load config properties.");
126         }
127     }
128
129     @BeforeMethod
130     public void setup() {
131         String browser = prop.getProperty("browser", "chrome");
132         String headless = prop.getProperty("headless", "false");
133
134         if (browser.equalsIgnoreCase("chrome")) {
135             ChromeOptions options = new ChromeOptions();
136             if (Boolean.parseBoolean(headless)) {
137                 options.addArguments("--headless=new", "--disable-gpu");
138             }
139             driver = new ChromeDriver(options);
140         } else if (browser.equalsIgnoreCase("edge")) {
141             EdgeOptions options = new EdgeOptions();
142             if (Boolean.parseBoolean(headless)) {
143                 options.addArguments("--headless=new", "--disable-gpu");
144             }
145             driver = new EdgeDriver(options);
146         } else {
147             driver = new ChromeDriver(); // default
148         }
149

```

```

155     @AfterMethod
156     public void tearDown(ITestResult result) throws Exception {
157         if (ITestResult.FAILURE == result.getStatus()) {
158             takeScreenshot(result.getName());
159             log.error("Test Failed: " + result.getName());
160         }
161         driver.quit();
162     }
163
164     public void takeScreenshot(String name) throws IOException {
165         TakesScreenshot ts = (TakesScreenshot) driver;
166         File src = ts.getScreenshotAs(OutputType.FILE);
167         File dest = new File("screenshot/" + name + ".png");
168         FileUtils.copyFile(src, dest);
169     }
170 }

```

```

1 browser=edge
2 headless=true

```

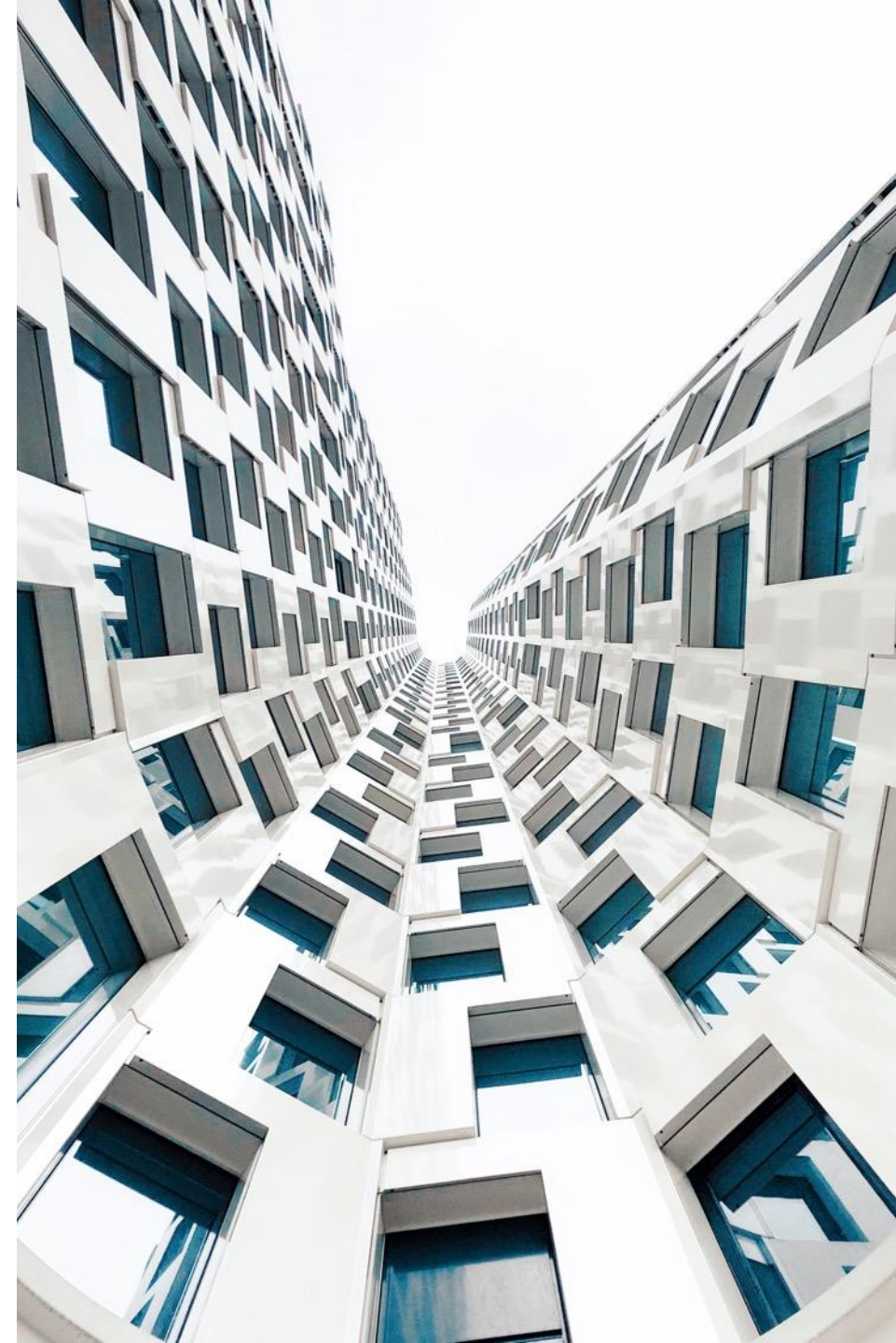
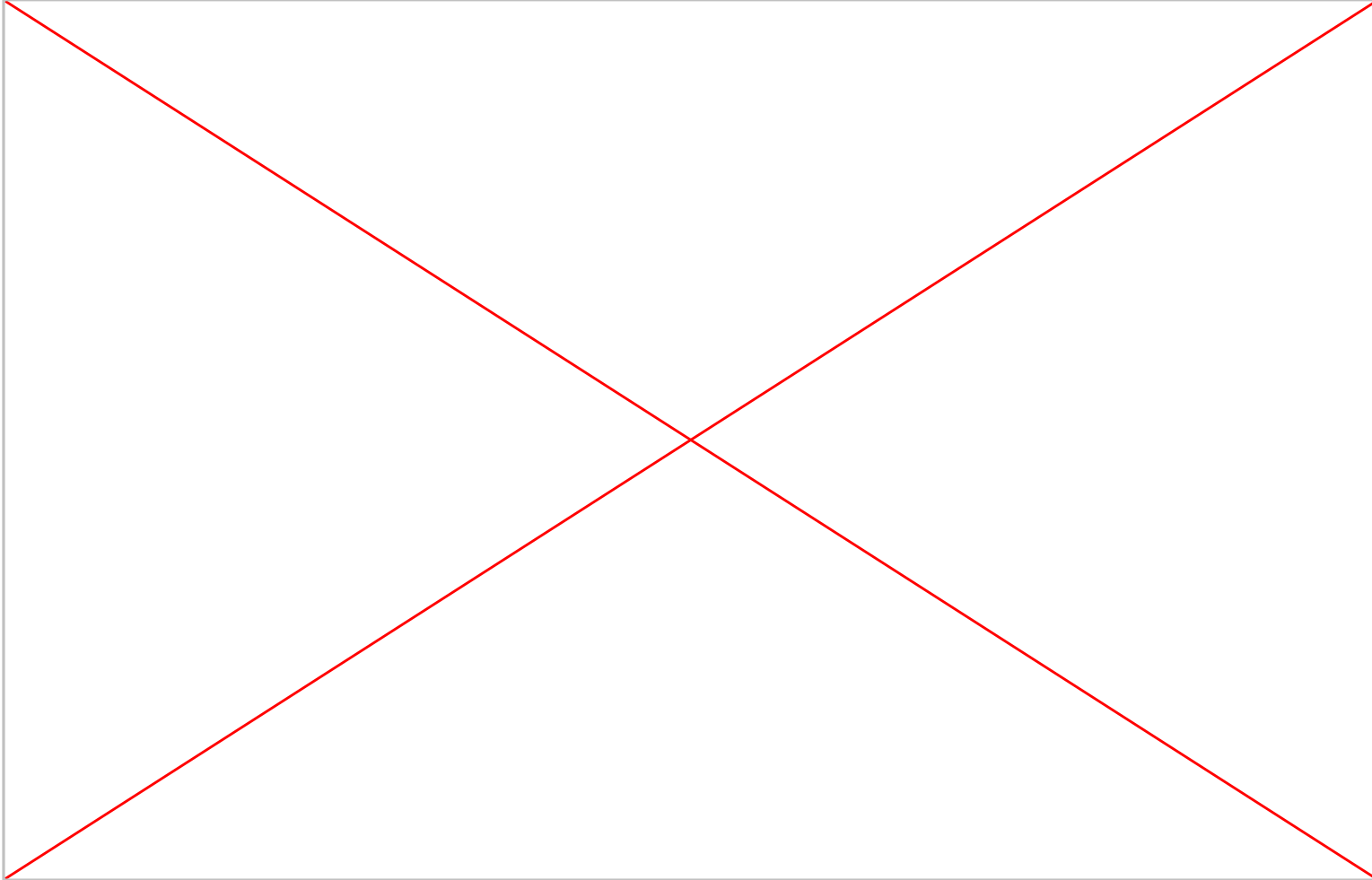
```

1 <?xml version="1.0" encoding="UTF-8"?>
   Bind to grammar/schema...
2 <suite name="Suite">
3   <test thread-count="5" name="Test">
4     <classes>
5       <class name="tests.LoginTest"/>
6       <class name="tests.SearchTest"/>
7       <class name="tests.FormSubmissionTest"/>
8       <class name="tests.ExtraFunctionalityTest"/>
9     </classes>
10  </test> <!-- Test -->
11 </suite> <!-- Suite -->
12

```

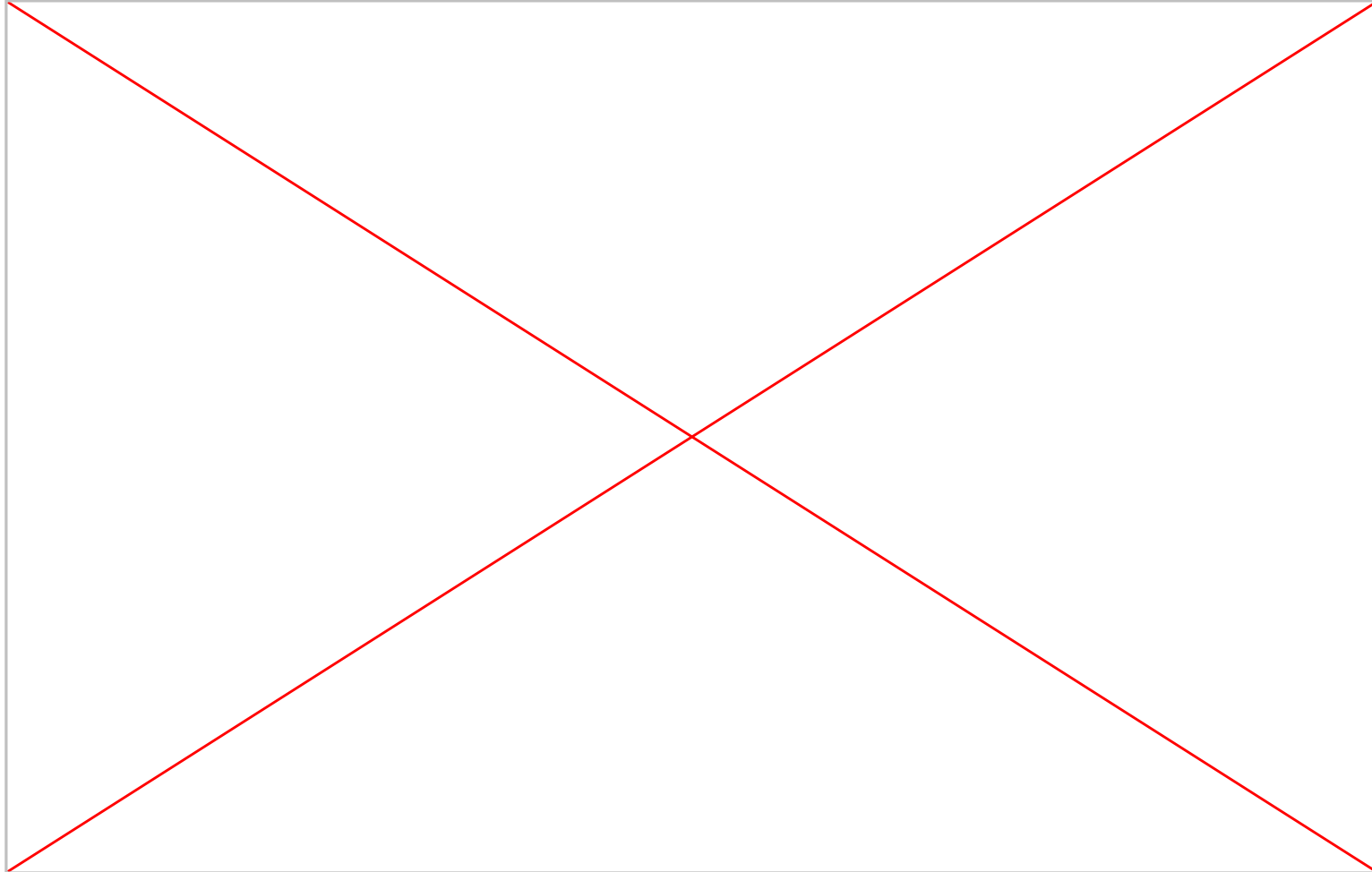
ASCENDION

Selenium Form Submission Test Video



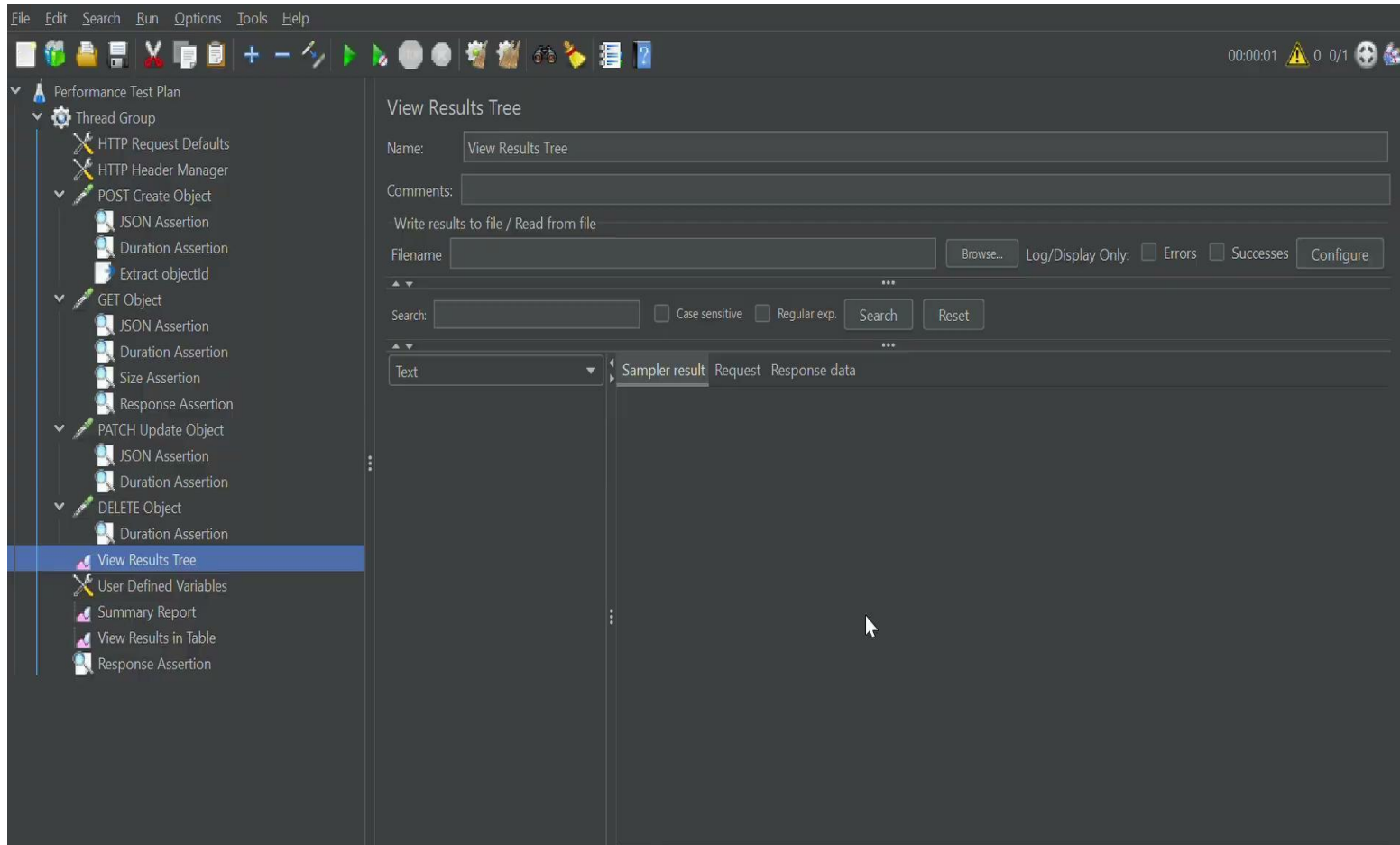
ASCENDION

API Postman Test Video



ASCENDION

JMeter api.restful-dev Test Video



Postman Request Methods, Responses of the API(api.restful-dev) with screenshots

The screenshot displays a Postman REST client interface. At the top, the request method is set to **POST** and the URL is `{{baseUrl}} /objects`. The **Body** tab is selected, showing a JSON payload with placeholders for device information. The response section at the bottom shows a **200 OK** status with a response time of 544 ms. The response body is a JSON object containing an ID, name, creation timestamp, and device details.

Request:

```
1 {
2   "name": "{{deviceName}}",
3   "data": {
4     "color": "{{deviceColor}}",
5     "capacity": "{{deviceCapacity}}"
6   }
7 }
```

Response:

```
1 {
2   "id": "ff80818196f2a23f0196fc8a0b4419f5",
3   "name": "Apple iPhone 14",
4   "createdAt": "2025-05-23T09:47:07.460+00:00",
5   "data": {
6     "color": "Blue",
7     "capacity": "128GB"
8   }
9 }
```


Postman Request Methods, Validations of the API(api.restful-dev) with screenshots

Environmental Variables

	Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/>	objectId	default		ff80818196f2a23f0196fc8a0b4419f5
<input checked="" type="checkbox"/>	baseUrl	default	https://api.restful-api.dev	https://api.restful-api.dev

Collection Variables

	Variable	Initial value	Current value
<input checked="" type="checkbox"/>	deviceName	Apple iPhone 14	Apple iPhone 14
<input checked="" type="checkbox"/>	deviceColor	Blue	Blue
<input checked="" type="checkbox"/>	deviceCapacity	128GB	128GB

Validation Scripts

The screenshot shows a Postman interface for a GET request to the endpoint `{{baseUrl}}/objects/{{objectId}}`. The 'Scripts' tab is active, displaying the following validation scripts:

```

1 pm.test("Status code is 200", () => pm.response.to.have.status(200));
2 pm.test("Body includes device name", () => {
3   pm.expect(pm.response.text()).to.include(pm.collectionVariables.get("deviceName"));
4 });
5 pm.test("Response time < 2000ms", () => pm.expect(pm.response.responseTime).to.be.below(2000));
6 pm.test("Content-Type is JSON", () => pm.response.to.have.header("Content-Type"));
7

```

The 'Test Results' tab is also active, showing four passed tests:

- PASSED** Status code is 200
- PASSED** Body includes device name
- PASSED** Response time < 2000ms
- PASSED** Content-Type is JSON

The overall status at the top right is **200 OK**.

Responses of the API(api.restful-dev) of the JMeter with Result Table

Performance Test Plan

Thread Group

HTTP Request Defaults

HTTP Header Manager

POST Create Object

JSON Assertion

Duration Assertion

Extract objectId

GET Object

JSON Assertion

Duration Assertion

Size Assertion

Response Assertion

PATCH Update Object

JSON Assertion

Duration Assertion

DELETE Object

Duration Assertion

View Results Tree

User Defined Variables

Summary Report

View Results in Table

Response Assertion

View Results in Table

Name:

View Results in Table

Comments:

Write results to file / Read from file

Filename

Browse...

Log/Display Only:

☐ Errors
 ☐ Successes

Configure

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	16:02:08.101	Thread Group...	POST Create ...	591		791	264	571	345
2	16:02:08.754	Thread Group...	GET Object	154		738	189	153	0
3	16:02:08.909	Thread Group...	PATCH Updat...	349		784	244	348	169
4	16:02:09.259	Thread Group...	DELETE Object	407		723	211	406	220

Responses of the API(api.restful-dev) of the Thread Group & Result Tree

Thread Group

Name:

Comments:

Action to be taken after a Sampler error

☒ Continue
 ☐ Start Next Thread Loop
 ☐ Stop Thread
 ☐ Stop Test
 ☐ Stop Test Now

Thread Properties

Number of Threads (users):

Ramp-up period (seconds):

Loop Count: ☒ Infinite

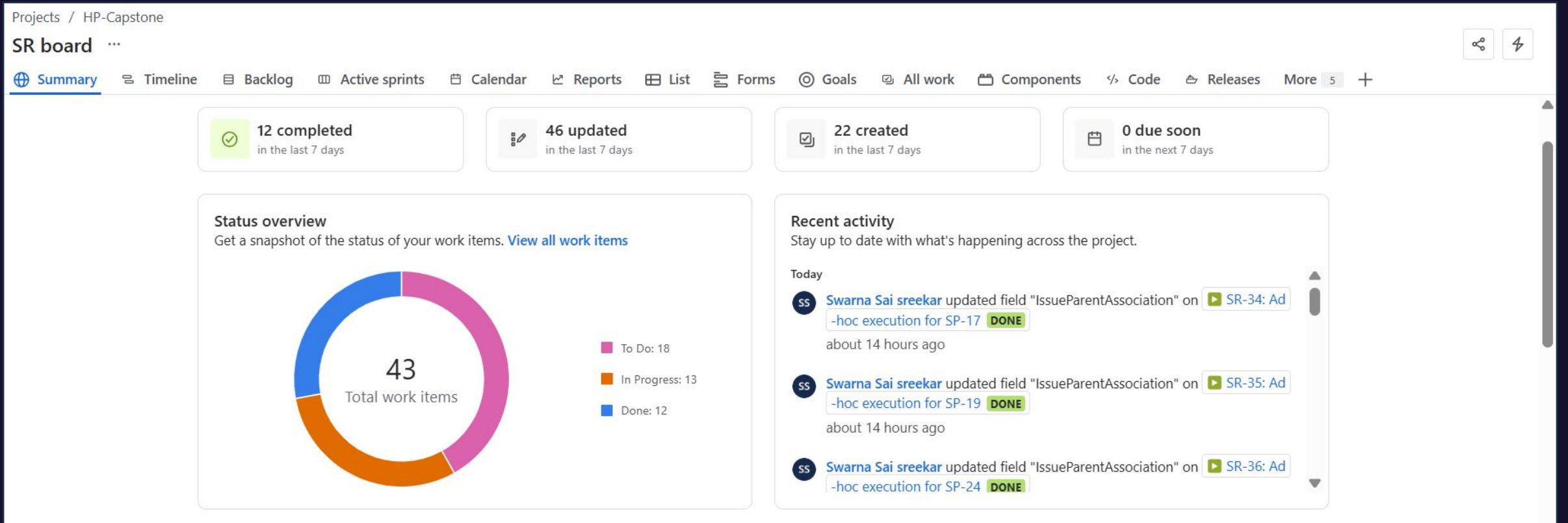
☒ Same user on each iteration
☐ Delay Thread creation until needed
☒ Specify Thread lifetime

Duration (seconds):

Startup delay (seconds):

Text	Sampler result	Request	Response data
POST Create Object	Thread Name:Thread Group 1-1 Sample Start:2025-05-23 16:02:08 IST Load time:591 Connect Time:345 Latency:571 Size in bytes:791 Sent bytes:264 Headers size in bytes:644 Body size in bytes:147 Sample Count:1 Error Count:0 Data type ("text" "bin" ""):text Response code:200 Response message:OK HTTPSampleResult fields: ContentType: application/json DataEncoding: null		
GET Object			
PATCH Update Object			
DELETE Object			

HP-Capstone(SR) Summary in SR - JIRA - Board



Conclusion

- Gained comprehensive Quality Engineering skills across manual, automation, API, mobile, and performance testing.
- Developed and maintained scalable test automation frameworks using industry tools and best practices.
- Integrated testing solutions within agile workflows and CI/CD pipelines for continuous quality assurance.
- Managed end-to-end test execution, reporting, and defect tracking effectively using Jira and Xray.
- Applied DevOps testing strategies like Shift-Left and Shift-Right for early and continuous feedback.
- Built readiness to deliver robust, maintainable, and scalable testing solutions in real-world projects.
- Strengthened practical experience with mobile automation and performance testing tools.

ASCENDION

**Thank
You**