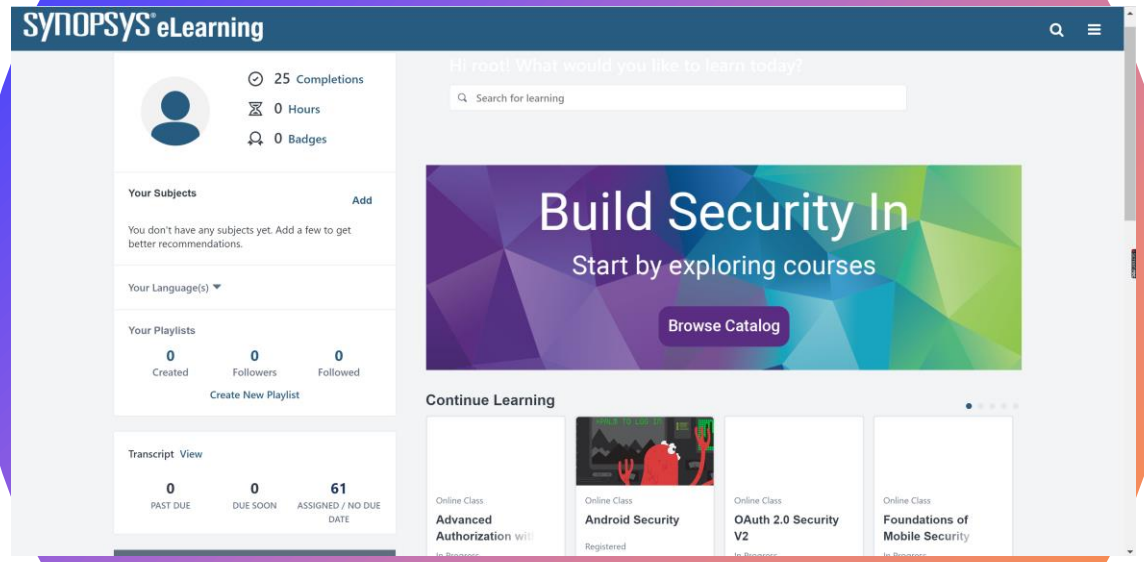
ELEARN

D.Prudhvi Teja



Introduction

- ELearn is a for-profit Massive Open Online Course (MOOC) provider aimed at professionals (companies).
- Professionals take courses primarily to improve job-related skills. All courses generate credit toward technical certification.
- ELearn has made a special effort to attract corporate trainers seeking to create coursework for employees of their company, work culture, harassment at the work place etc.
- ELearn is just a platform for the courses.
- ELearn is E-learning Where CSOD is Cornerstone On Demand

TEST AUTOMATION FRAMEWORK

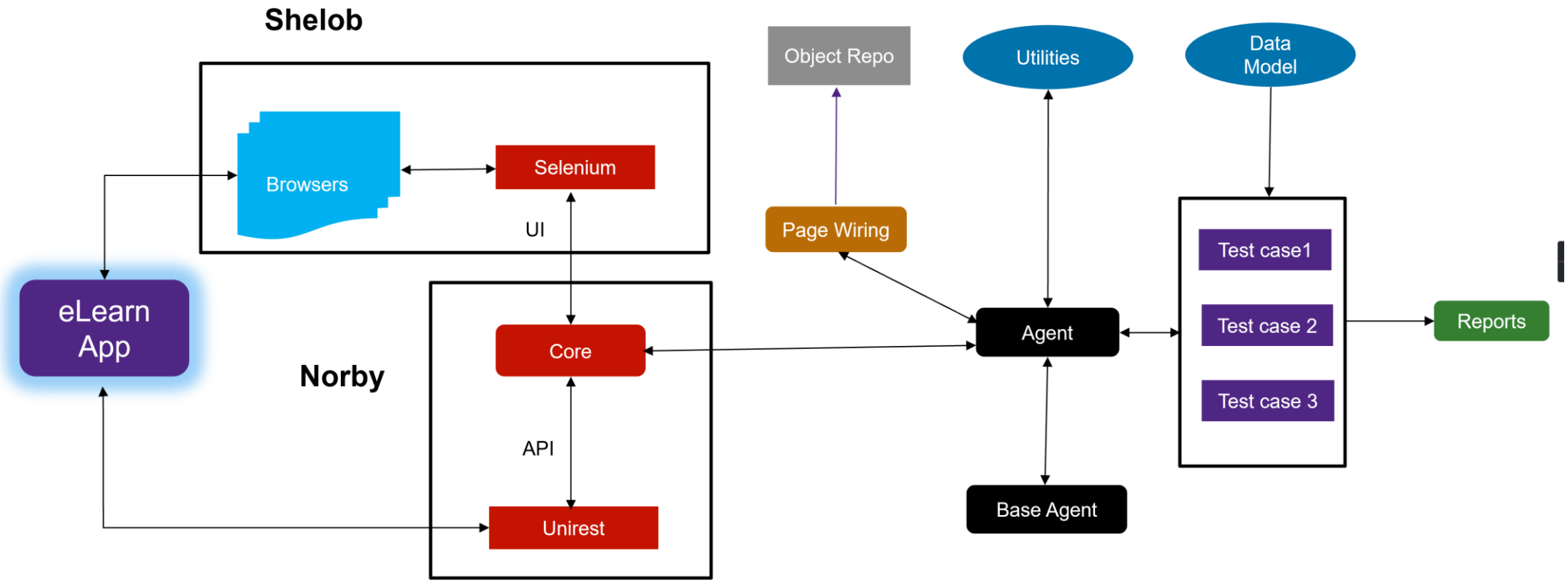
A testing framework is a set of guidelines or rules used for creating and designing test cases. A framework is comprised of a combination of practices and tools that are designed to help QA professionals test more efficiently.

These guidelines could include coding standards, test-data handling methods, object repositories, processes for storing test results, or information on how to access external resources.

BENEFITS OF A TEST AUTOMATION FRAMEWORK

- Improved test efficiency
- Minimal manual intervention
- Maximum test coverage
- Reusability of code

ELearn Test Automation



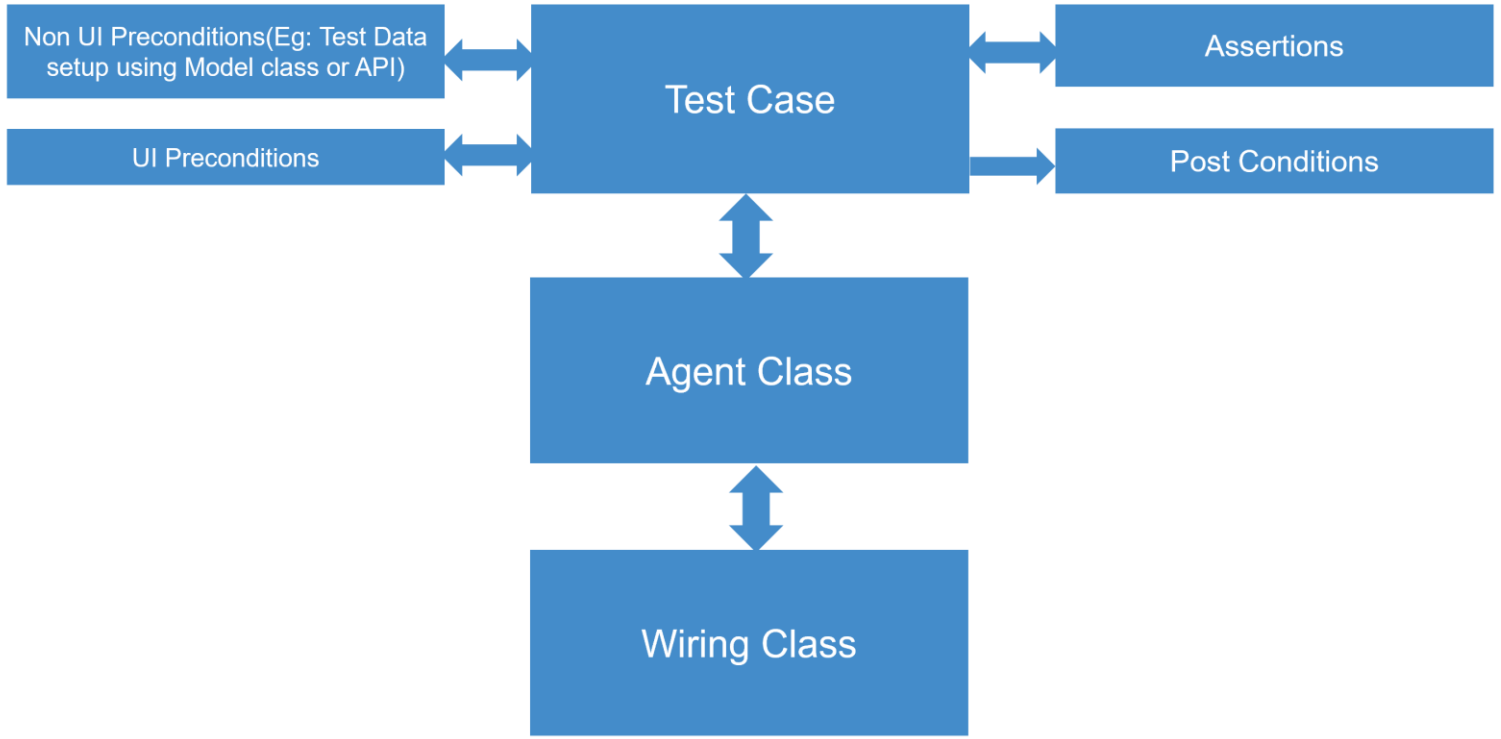
Technologies used…

- **Java - 1.8**
- **Selenium - 3.9.1**
- **Maven - 3.5**
- **Unirest - 1.4.9**
- **TestNG – 6.8**
- **Eclipse**
- **Test Rail**
- **STS**

Project Structure

- **agents** package contains methods that performs user actions on the page using wiring classes.
- **cases** package contains the test cases.
- **common** package contains application utility classes. i.e. emailUtility.
- **Infrastructure** package contains the learn utility classes.
- **wiring (UI)** package contains the classes to store the locators of the page.
- **wiring(API)** package contains the classes to store the APIs responses.
- **resources** package contains the files that are needed in the project like Courses, CVE/CSV file etc.

UI Test Workflow



HOW TO CREATE UI TESTS?

UI Wiring:

- •Capture the locators(ID, name, Xpath, etc.) from the web page
- •Different types of elements(Textbox, Dropdown etc.) are identified with their locator value and stored in a **Element collection** map
- •Agent methods access these elements to perform the UI actions on the page

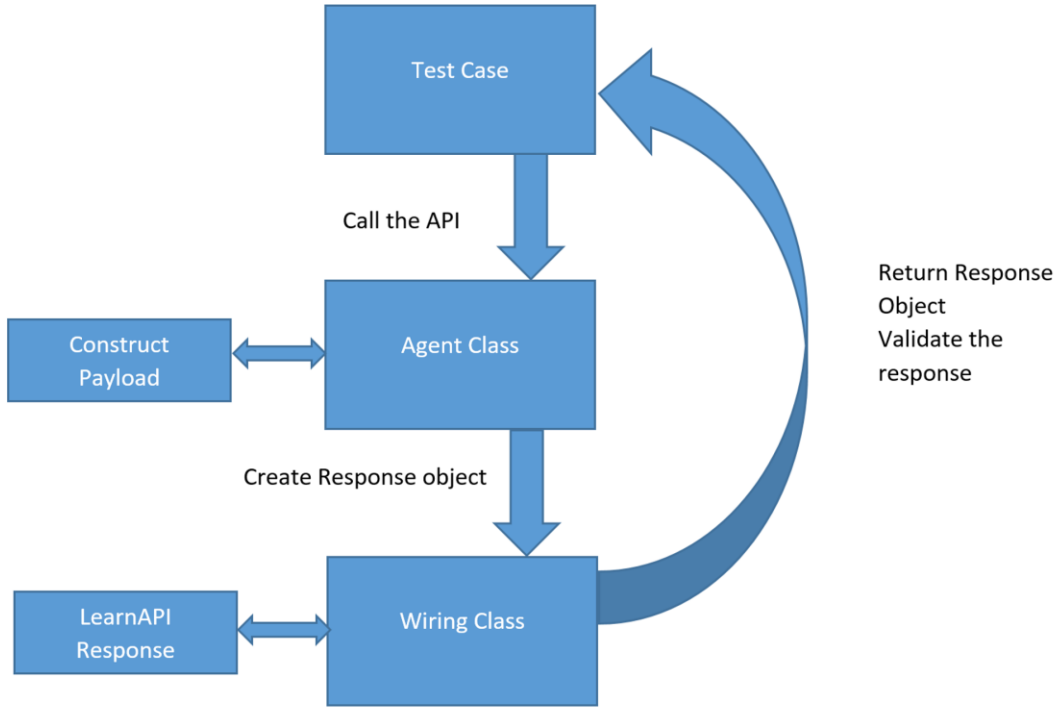
UI Agents:

- •We have to set the page context according to the respective Agent in the Agent Constructor.
- •Agents contain the convenience methods to perform actions on the web page using the identified elements from the Wiring class
- •Agent methods return the next Agent once the method executes successfully.

Test Cases:

- •Test Cases needs to extend the Base Classes where necessary pre conditions and post conditions gets executed
- •Model Classes are used to manage the Test Data
- •Test Cases call the Agent methods to perform the Test steps
- •Test Cases contain the assertions which compares the actual and expected results.

API Test Workflow



How to create API test cases?

- Create methods in the **Agent** Class to call required API's.
  - Each API has one or more methods.
- Create methods to fetch required value from the API response in the **wiring** class.
- Create **test case** class and call methods of Agent class and wiring class based on test scenarios.
  - Perform the assertions after calling agent and wiring class methods.
  - Log the steps.

Agent Class - API

- Construct the end point URL.
- Generate data class helps in generating payload with pre-defined/run time values.

```java
/**
 * Adds organization using OrganizationModel
 *
 * @param authCookie
 * @param organizationModel
 * @return PostOrganizationsResponse
 */
public PostOrganizationsResponse addOrganization(String authCookie, OrganizationModel organizationModel) {
    try {
        JSONObject bodyData = GenerateData.getAddOrganizationData(organizationModel);
        babble("Generated data for add organization is: " + bodyData);
        RestApi api = new RestApi().setBaseUrl(LearnTestProperties.getLearnApiAdminUrl())
                .setEndpoint("/v1.0/organizations").setMethod(RestApi.HttpMethod.POST)
                .addHeaderField("cookie", authCookie)
                .setBody(bodyData).logResponse(true);
        return new PostOrganizationsResponse(api.postAsJson());
    } catch (JSONException e) {
        throw new RestApiException("Failed to add organization: " + e.getMessage(), e);
    }
}
```

Wiring class - API

- Store API response and retrieve the required value from the response.

```java
/**
 * Get the first name from the response
 * @return
 */
public String getFirstname() {
    try {
        return jsonroot.getJSONObject(ResponseTags.DATA.tag).
                getString(ResponseTags.FIRSTNAME.tag).toString();
    } catch (JSONException e) {
        throw new AssertionError("Failed to get first name from get-user response: "
    + e.getMessage());
    }
}
```


```
{
    "data": {
        "id": "c3cb4d19-098d-424d-943e-5dec87c0c70c",
        "firstName": "root",
        "lastName": "root",
        "email": "root@codiscope.com",
        "title": "Root user",
        "city": null,
        "country": null,
        "active": true,
        "locked": false,
        "createdOn": "2018-03-06T10:13:33.256Z",
        "updatedOn": null,
        "authUserId": "-",
        "roles": [
            {
                "id": "3359229c-d37d-42ab-9c11-26570bf0ffc1",
                "name": "SuperAdmin",
                "cohortId": null,
                "organizationId": null,
                "createdOn": "2018-03-06T10:13:33.191555+00:00"
            }
```

Test Class - API

- Call Pre and post condition methods from base class.
- Call agent and wiring class methods based on test scenarios.

```
/**
 * verify client admin can access complete information of user belongs to his org but not other
 *
 * @author pkuma
 * Copyright © 2018, Synopsys, Inc. All rights reserved worldwide.
 */
public class GetBasicInfoAsCA extends LearnerApiTestBase {

        @Test(groups = { "API-Regression" })
        public void testGetBasicInfoAsCASuccess() {

            testCase("LEARN-3840,C451373", "verify client admin can access complete information o
            LoginResponse loginResponse = authAgent.login();
            String cookie = loginResponse.getAuthToken();
            authAgent.getAdminAgent().assertResponseStatusCode(loginResponse, 200);
            getLog().babble("Login as root user sucessfully");

            // Add ORG
            OrganizationModel organizationModel = new OrganizationModel();
            organizationModel.populateFields();
            PostOrganizationsResponse postOrganizationsResponse = authAgent.getAdminAgent().addOr
            authAgent.getAdminAgent().assertResponseStatusCode(postOrganizationsResponse, 200);
            String organizationID = postOrganizationsResponse.getOrganizationId();
            getLog().babble("Organization has been created with ID as " + organizationID);
```

THANK YOU

D.Prudhvi Teja