A Project Report on

CURA HEALTHCARE SERVICE

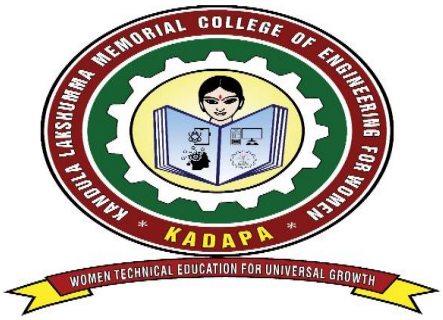
by

S. Nikhat Anjum

Under the Guidance of

MR.B. MAHESH NAYAK, MTech

Associate Professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K.L.M COLLEGE OF ENGINEERING FOR WOMEN

(APPROVED BY AICTE AND AFILIATED TO J.N.T.U.A., ANANTAPUR)
KADAPA-516003

ABSTRACT

In the ever-evolving landscape of healthcare, the demand for innovative and patient-centered solutions is paramount. Cura Healthcare Services emerges as a pioneering entity, poised to revolutionize the healthcare industry by offering a multifaceted approach to care delivery. This abstract provides an overview of Cure Healthcare Services, highlighting its core principles, services, and potential impact on the healthcare ecosystem.

Cura Health Services, a distinguished healthcare provider, understands the criticality of maintaining impeccable software systems to support their medical operations and deliver exceptional patient care.

The healthcare industry demands a stringent adherence to regulatory requirements and industry standards to safeguard patient data and ensure the utmost patient safety. To achieve these essential objectives, Cura Health Services has devised a robust software testing framework, encompassing various testing methodologies such as functional testing, performance testing, security testing, and usability testing.

The organization places significant emphasis on compliance and quality assurance, adhering rigorously to testing protocols to meet industry standards and regulatory obligations

A comprehensive approach is adopted, employing both manual and automated testing techniques, to achieve an all-encompassing test coverage. By incorporating automated tests into their Continuous Integration (CI) and Continuous Deployment (CD) pipeline, Cura Health Services ensures continuous testing, providing rapid feedback on application quality.

By employing an unwavering focus on software testing, Cura Health Services endeavors to deliver secure, reliable, and high-quality healthcare applications. This dedication allows them to provide exceptional medical services to their patients while maintaining full compliance with healthcare regulations and industry best practices. Ultimately, their robust software testing strategy underpins their commitment to excellence in healthcare delivery. Feedback from testing and end-users is thoughtfully incorporated to enhance software functionality and usability continuously. In this pursuit, software testing assumes paramount significance, serving as a fundamental pillar to ensure the reliability, security, and effectiveness of their healthcare applications.

CONTENTS

CHAPTER 1

DEFINE PROBLEM/ PROBLEM UNDERSTANDING

1.1 Specify the business problem:

The business problem faced by patients in the earlier days was the Limited Accessibility, High Cost, Long Waiting Times, Limited Treatment Options, etc. This led to long waiting times and uncertainty about the availability of doctors even after reaching the hospital. CURA Healthcare Service aims to overcome these issues by providing a prior appointment feature.

1.2 Business impact:

The lack of a prior appointment system had several negative impacts on both patients and healthcare providers. Some of the key business impacts were:

a) Inefficient Resource Management: Hospitals and clinics might have faced challenges in managing their resources effectively without a structured appointment system. Doctors' availability might not have been optimized.

b) Patient Dissatisfaction: Patients had to endure long waiting times, which led to dissatisfaction and frustration. This might have resulted in some patients seeking alternative healthcare options.

c) Revenue Loss: Without an appointment system, patients might have chosen not to visit the hospital due to the uncertainty of getting timely medical attention. This could lead to potential revenue loss for the healthcare service providers.

d) Limited Accessibility: Healthcare services were often concentrated in urban areas or major cities, making it difficult for individuals in rural or remote regions to access medical facilities. The lack of transportation options and infrastructure posed a significant barrier to healthcare access.

CHAPTER 2

TEST CASE PREPARATION

2.1 Analyze requirements:

The requirements for the CURA Healthcare Service are to implement a prior appointment system that allows patients to schedule appointments with doctors efficiently. The system needs to verify the login credentials of users before enabling them to book appointments. Additionally, it should have a functionality to check the availability of doctors based on different health conditions.

2.2 Create the Scenarios and Collect the Input Data:

Scenarios are situations that represent different use cases for the application. For the CURA Healthcare Service, some scenarios might include:

Scenario 1: Patient logs in with valid credentials and schedules an appointment with a cardiologist.
Scenario 2: Patient logs in with invalid credentials and receives an error message.
Scenario 3: Patient tries to book an appointment without providing all required information and receives a validation message.
Scenario 4: Patient checks the availability of cardiologists on a specific date and time.
Scenario 5: Patient tries to book an appointment with a non-existent doctor and receives an error message.

2.3 Preparation of Test Cases:

Test cases are detailed steps that testers follow to validate the functionality of the application. For example:

Test Case 1: Verify that a registered patient can log in successfully using valid credentials.
Test Case 2: Verify that a patient with invalid credentials cannot log in and receives an appropriate error message.
Test Case 3: Verify that all mandatory fields are validated when booking an appointment.

Test Case 4: Verify that attempting to book an appointment with a non-existent doctor result in an error message.

2.4 Test Data Preparation (in the form of Validation Table) as per baseline document:

A validation table contains sample input data and the expected outcomes.

For example:

Test Case	Input Data	Expected Outcome
TC-001	Username: John Doe, Password: ThisIsNotAPassword	Successful login
TC-002	Username: invalid user, Password: ThisIsNotAPassword	Error message: "Invalid username or password"
TC-003	Missing required fields	Validation messages for missing fields
TC-004	Username: John Doe, Password: ThisIsNotAPassword	Appointment Confirmed

These are just a few examples, and the actual number of test cases and validation data will depend on the complexity of the application and its requirements.

CHAPTER 3

SCRIPT /TEST CASE EXECUTION UNDER TEST SUITE AND TEST SUITE COLLECTION LEVEL

In Katalon Studio, you can organize your test cases into test suites and test suite collections to execute them efficiently. Here's a brief description of test suite and test suite collection execution:

3.1 Test Suite Execution:

Test Suite: A test suite is a collection of test cases that are intended to be used to test a software program to show that it has some specified set of behaviors based on a specific functionality or scenario of the application.

Execution: To execute a test suite in Katalon Studio, you can follow these steps:

1. Open Katalon Studio and load your project.
2. Go to the 'Test Suites' view by clicking on 'Test Suites' on the left-hand side.
3. Right-click on the desired test suite and choose 'Run' from the context menu.
4. Katalon Studio will run all the test cases within the test suite one by one, and you can see the test execution progress and results in the 'Console' view**.**

3.2 Test Suite Collection Execution:

Test Suite Collection: A test suite collection is a collection of multiple test suites. It allows you to execute multiple test suites in a sequence or parallel, making it useful for running end-to-end test scenarios or regression testing.

Execution: To execute a test suite collection in Katalon Studio, follow these steps:

1. Open Katalon Studio and load your project.
2. Go to the 'Test Suite Collection' view by clicking on 'Test Suite Collection' on the left-hand side.
3. Right-click on the desired test suite collection and choose 'Run' from the context menu.
4. Katalon Studio will start executing all the test suites included in the collection, either sequentially or in parallel, depending on your configuration. You can see the overall test execution progress and results in the 'Console' view.

CHAPTER 4

HANDLING AND VALIDATING BUTTONS

Handling and validating buttons in Katalon Studio for the Cura Healthcare Service project involves interacting with the buttons on the web application and ensuring their functionality. Here's a step-by-step guide to handling and validating buttons in Katalon Studio:

4.1 Identify the Buttons:

Use Katalon Studio's built-in Web Spy tool or the Object Spy to identify the buttons on the web application's user interface. The Object Spy allows you to inspect elements and capture their properties for further use in test scripts.

4.2 Interacting with Buttons:

To click a button, you can use the `click` keyword in Katalon Studio. For example:

```groovy
WebUI.click(findTestObject('path/to/your/button'))
```

4.3 Validating Button States or Text:

To verify the text on a button, you can use the `verifyElementText` keyword. For example:

```groovy
WebUI.verifyElementText(findTestObject('path/to/your/button'), 'Expected Button Text')
```

4.4 Validating Button Availability or Visibility:

To check if a button is visible on the web page, you can use the `verifyElementVisible` keyword. For example:

```groovy
WebUI.verifyElementVisible(findTestObject('path/to/your/button'), FailureHandling.CONTINUE_ON_FAILURE)
```

4.5 Button State Validation (Enabled/Disabled):

To validate if a button is enabled or disabled, you can use the `verifyElementEnabled` keyword. For example:

```groovy
WebUI.verifyElementEnabled(findTestObject('path/to/your/button'), FailureHandling.CONTINUE_ON_FAILURE)
```

CHAPTER 5

TESTLISTENER

In Katalon Studio, Test Listeners allow you to execute custom code before or after certain test events. These listeners help you enhance your test automation project's functionality and perform additional actions during test execution. To use Test Listeners in the Cura Healthcare Service project, follow these steps:

5.1 Create a Test Listener:

Right-click on the Test Case or Test Suite in the 'Test Explorer' panel.

Select 'New' > 'Test Listener' from the context menu.

Choose the appropriate event (e.g., 'beforeTestCase', 'afterTestCase', 'beforeTestSuite', 'afterTestSuite', etc.) to handle.

Give a name to the Test Listener and click 'OK.'

5.2 Implement the Test Listener Code:

The Test Listener script will be opened for editing.

In this script, we can write custom code to perform actions before or after the selected event.

5.3 Examples of Test Listener Usage:

Before executing a Test Case, we can set up some preconditions, data setup, or login to the application:

```groovy
import com.kms.katalon.core.annotation.BeforeTestCase
import com.kms.katalon.core.annotation.BeforeTestSuite
```

5.4 Register Test Listener:

To activate Test Listener, we need to register it in Katalon Studio.

Go to 'Project' > 'Settings' > 'Execution' > 'Default' > 'Listener'

Add your Test Listener class to the list of custom listeners.

5.5. Save and Run:

Save Test Listener script and run your Test Case or Test Suite.

The Test Listener code will be executed before or after the specified test events.

By using Test Listeners in your Cura Healthcare Service project, we can add custom actions, log additional information, and improve the overall test execution process. Test Listeners provide a way to modularize your test scripts and make them more maintainable and reusable.

CHAPTER 6

BUILD DELIVERY

Integrating Katalon Studio with Git and Jenkins allows you to automate the build and delivery process for your Cura Healthcare Service project. Here's a step-by-step guide to set up the integration:

6.1 Initialize Git Repository:

Create a Git repository for your Cura Healthcare Service project. If you're using a version control platform like GitHub, Bitbucket, or GitLab, create a new repository there and follow the instructions to initialize the repository locally on your machine.

6.2 Link Katalon Studio Project with Git:

Open your Cura Healthcare Service project in Katalon Studio.

Go to 'Team' > 'Git' > 'Initialize' to link the project to the Git repository you created in step1.

Commit and push your project files to the remote Git repository.

6.3 Install Jenkins:

Download and install Jenkins on the server or machine where you want to run the automated build and delivery process.

Follow the instructions for your specific operating system:
https://www.jenkins.io/download/

6.4 Configure Jenkins for Katalon Integration:

Install the "Katalon Plugin" in Jenkins.

In Jenkins, go to 'Manage Jenkins' > 'Manage Plugins' > 'Available' tab.

Search for "Katalon" in the search box and install the "Katalon Plugin."

6.5 Create Jenkins Job:

Create a new Jenkins job for your Cura Healthcare Service project.

Click on 'New Item' in Jenkins.

Enter a name for the job, choose the "Freestyle project" option, and click 'OK.'

In the job configuration, under the 'Source Code Management' section, select Git and provide the URL of your Git repository.

Specify the branch to be built (e.g., master/main) or use the default value.

Save the job configuration.

6.6 Configure Jenkins Build Steps for Katalon:

In the Jenkins job configuration, add a new build step to execute Katalon Studio.

Choose the "Execute Katalon Studio Tests" option.

Provide the Katalon Studio executable path.

Set the path to your Katalon Studio project.

Choose the appropriate test suite to execute (or run individual test cases if needed).

Configure other options like browser, report generation, etc., as per your project's requirements.

6.7 Save and Run the Jenkins Job:

Save the Jenkins job configuration.

Click on 'Build Now' to trigger the build process.

Jenkins will pull the latest code from the Git repository, execute the Katalon Studio tests, and generate test reports.

6.8 Configure Post-Build Actions:

In Jenkins, you can set up post-build actions, such as sending email notifications, publishing test reports, or triggering other Jenkins jobs based on the build result**.**

6.9 Schedule Builds (Optional):

If needed, you can schedule periodic builds in Jenkins to automate the testing process at specific intervals.

With this integration, Jenkins will automatically fetch the latest code from the Git repository, trigger Katalon Studio to execute the tests, and generate test reports. This setup enables continuous integration and delivery for your Cura Healthcare Service project, ensuring that automated tests are executed whenever there are code changes, providing faster feedback on the application's health and quality.

CHAPTER 7

INTEGRATING KATALON TO GIT AND JENKINS

Integrating Katalon Studio with Git and Jenkins allows you to manage version control, automate test execution, and facilitate continuous integration and delivery for your Cura Healthcare Service project.

With this integration, Jenkins will automatically fetch the latest code from the Git repository, trigger Katalon Studio to execute the tests, and generate test reports. This setup enables continuous integration and delivery for your Cura Healthcare Service project, ensuring that automated tests are executed whenever there are code changes, providing faster feedback on the application's health and quality.

7.1 Git Integration:

Set up a Git repository to manage your Katalon Studio project files. You can use popular Git hosting services like GitHub, GitLab, or Bitbucket.

Add your Katalon project files to the Git repository and commit them.

Create branches for different features or bug fixes, allowing your team to work collaboratively and manage changes efficiently.

Make use of pull requests and code reviews to ensure the quality of the test scripts before merging them into the main branch.

7.2 Jenkins Integration:

Install Jenkins on a server or machine and set it up to run automated tests.

Install the Katalon Studio plugin for Jenkins to enable Jenkins to execute Katalon tests seamlessly.

Set up a Jenkins job that triggers the Katalon test suite to run automatically whenever there are changes pushed to the Git repository.

Configure the Jenkins job to generate test reports and notifications about test results.

7.3 CI/CD Pipeline:

Integrate the Git and Jenkins setup into your Continuous Integration (CI) and Continuous Deployment (CD) pipeline.

Automate the process of building and deploying your application along with running Katalon tests.

This way, you can ensure that tests are automatically executed whenever there are changes to the application code, allowing you to catch issues early and ensure continuous quality improvement.

7.4 Benefits of this Integration:

Version control: Git allows you to manage changes in your Katalon Studio project files effectively, making collaboration easier and helping you track changes over time.

Automation: Jenkins automates the execution of Katalon tests, ensuring that tests are run consistently and regularly without manual intervention.

Continuous Testing: By integrating Katalon with CI/CD pipeline, you can implement continuous testing practices, ensuring that tests are executed automatically at every stage of the development lifecycle.

Faster Feedback: Automated tests running in Jenkins provide faster feedback on the quality of code changes, allowing your team to identify and fix issues quickly.

CHAPTER 8

CROSS BROWSER TESTING USING TESTCLOUD

Here's how you can achieve cross-browser testing for the Cura Healthcare Service project in Katalon Studio:

8.1 Test Case Design:

Develop your test cases in Katalon Studio using the WebUI testing approach. Ensure your test cases are modular, reusable, and independent of specific browsers.

8.2 Configure Browsers in Katalon Studio:

In Katalon Studio, go to 'Project' > 'Settings' > 'Desired Capabilities' to configure multiple browsers for testing (e.g., Chrome, Firefox, Edge, etc.).

8.3 Parameterize Browser Selection:

Modify your test cases to use variables for the browser selection. You can use Test Data or Global Variables to control the browser to be used during test execution.

8.4 Run Test Cases with Different Browsers:

Run your test cases individually or in test suites, and Katalon Studio will execute them on the specified browsers. You can use the 'Run' button in the Katalon Studio toolbar or use command-line execution.

8.5 Review Test Results:

After the test execution is complete, review the test results and reports in Katalon Studio. The reports will show the results for each test case on each browser.

8.6 Cross-Browser Testing Best Practices:

Ensure your web application is responsive and compatible with various browsers to minimize any discrepancies during cross-browser testing.

Use CSS and XPath selectors that are browser-independent to avoid issues with element identification.

Utilize data-driven testing to perform the same test with different data sets on multiple browsers.

Please note that while Katalon Studio supports cross-browser testing for desktop browsers, it does not provide native support for mobile testing. For mobile testing, you may need to consider using Appium or other mobile testing solutions.

CHAPTER 9

GENERATING AND ANALYZING REPORT AND SENDING REPORT THROUGH EMAIL

In Katalon Studio, you can generate and analyze reports for your Cura Healthcare Service project using built-in features and plugins. Additionally, you can send the generated reports through email to stakeholders. Here's how you can achieve this:

9.1 Generate Test Reports:

Katalon Studio automatically generates test reports after test execution. By default, the reports are saved in the 'Reports' folder of your project.

After running your test suite or test cases, go to the 'Reports' folder to find the HTML report ('index.html') and other related files.

9.2 Analyze Test Reports:

Open the HTML report ('index.html') in your web browser to analyze the test results.

The report provides detailed information about test case status, execution time, error messages, and more.

Use the report to identify test failures, errors, and overall test execution statistics.

9.3 Configure Email Notification:

Katalon Studio allows you to configure email notification settings to send the test reports via email after test execution.

Go to 'Project' > 'Settings' > 'Email' to set up the email configuration.

9.4 Send Test Reports via Email:

After configuring the email settings, use the 'Email' keyword in your test cases or test listeners to send the generated report via email.

You can use the `EmailUtil` class in Katalon Studio to send the email.

By following these steps, you can generate, analyze, and send test reports for your Cura Healthcare Service project in Katalon Studio. This will provide valuable insights into test results and help stakeholders stay informed about the health and quality of the application under test.