**Lab Exercise 9- Using Regex Operations in Drools**

**Objective:**

To demonstrate how to use regular expressions in Drools rules to match patterns and validate data.

**Step 1: Set Up the Project**

* Ensure you have a Maven project set up with Drools dependencies. You can use your existing project setup or create a new one.

**Step 2: Define Java Model Classes**

**Create a User Class:**

In the src/main/java/com/example/model directory, create a User.java file with the following content:

package com.example.model;

public class User {

private String username;

private String email;

public User(String username, String email) {

this.username = username;

this.email = email;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

@Override

public String toString() {

return "User{username='" + username + "', email='" + email + "'}";

}

}

**Step 3: Create DRL Rules with Regex Operations**

**Create the Rule File:**

In the src/main/resources directory, create a file named userRules.drl with the following content:

package com.example.rules

import com.example.model.User

rule "Validate Username"

when

$user : User(username matches "[a-zA-Z0-9\_]{5,15}")

then

System.out.println($user.getUsername() + " has a valid username.");

end

rule "Validate Email"

when

$user : User(email matches "^[\\w.%+-]+@[\\w.-]+\\.[a-zA-Z]{2,}$")

then

System.out.println($user.getEmail() + " has a valid email address.");

end

rule "Invalid Username"

when

$user : User(username not matches "[a-zA-Z0-9\_]{5,15}")

then

System.out.println($user.getUsername() + " has an invalid username.");

end

rule "Invalid Email"

when

$user : User(email not matches "^[\\w.%+-]+@[\\w.-]+\\.[a-zA-Z]{2,}$")

then

System.out.println($user.getEmail() + " has an invalid email address.");

end

**Explanation of the Rules:**

* Validate Username: Uses regex to validate that the username contains only alphanumeric characters and underscores, and is between 5 to 15 characters long.
* Validate Email: Uses regex to validate that the email address is in a standard format.
* Invalid Username: Uses regex to detect invalid usernames.
* Invalid Email: Uses regex to detect invalid email addresses.

**Step 4: Update the Main Application**

**Modify the DroolsTest Class:**

Update the DroolsTest.java file to demonstrate regex operations:

package com.example;

import com.example.model.User;

import org.kie.api.KieServices;

import org.kie.api.runtime.KieContainer;

import org.kie.api.runtime.KieSession;

public class DroolsTest {

public static void main(String[] args) {

// Load the knowledge base

KieServices ks = KieServices.Factory.get();

KieContainer kContainer = ks.getKieClasspathContainer();

KieSession kSession = kContainer.newKieSession("ksession-rules");

// Create sample users

User user1 = new User("alice123", "alice@example.com");

User user2 = new User("bob!", "bob@example");

User user3 = new User("charlie", "charlie@domain");

// Insert users into the session

kSession.insert(user1);

kSession.insert(user2);

kSession.insert(user3);

// Fire all rules

kSession.fireAllRules();

// Dispose the session

kSession.dispose();

}

}

**Step 5: Run the Application**

**Compile and Run:**

* Compile and run the DroolsTest class.

**Expected Output:**

The console should display output similar to the following:

alice123 has a valid username.

alice@example.com has a valid email address.

bob! has an invalid username.

bob@example has an invalid email address.

charlie has an invalid username.

charlie@domain has an invalid email address.