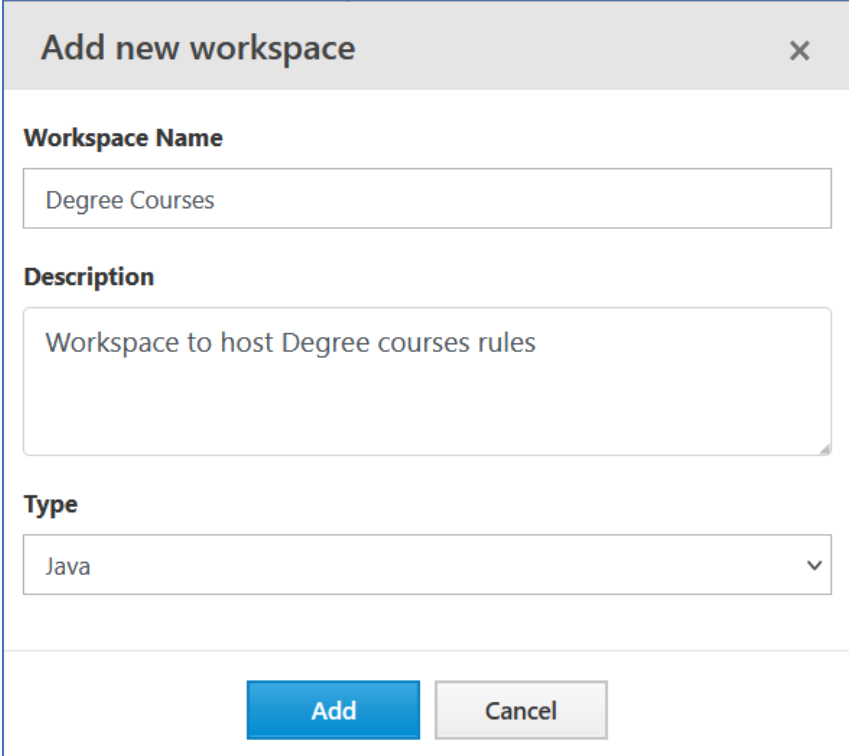


We will be using Rulesmatix LABS Rules Management System to create the rules for the DM Community's July 2025 Challenge. The Rulesmatix LABS Rules Management System is a web-based system that allows users to create object models, write rules and then generate the rules code in Java or Drools DRL syntax. In this document, we present step-by-step screen snapshots to show how workspace, technical and business projects, object model, rules in the ruleset and an entry point is created. We also show the Java and Drools Code generated by the system and a test Java code along with the output from Java and Drools code generated by the system. For more information and details, please reach out to us at [info@rulesmatix.com](mailto:info@rulesmatix.com) or [bsingh@rulesmatix.com](mailto:bsingh@rulesmatix.com)

To start with, we will need to create a workspace. A workspace has a name, description and either Java or Drools Type for the executable code format. We will be using Java for this challenge.



**Add new workspace** [X]

**Workspace Name**  
Degree Courses

**Description**  
Workspace to host Degree courses rules

**Type**  
Java [v]

**Add** **Cancel**

Figure 1: Add new Workspace

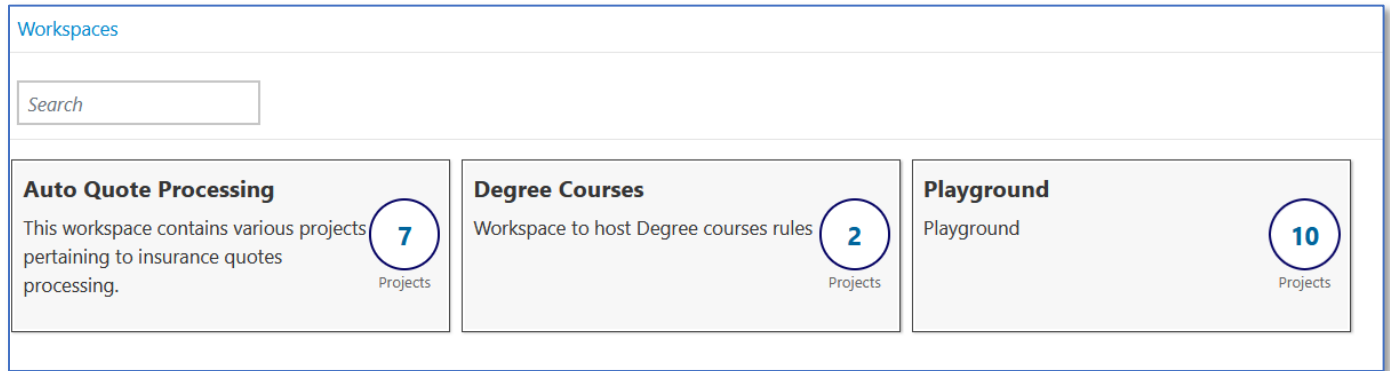


Figure 2: All Workspaces

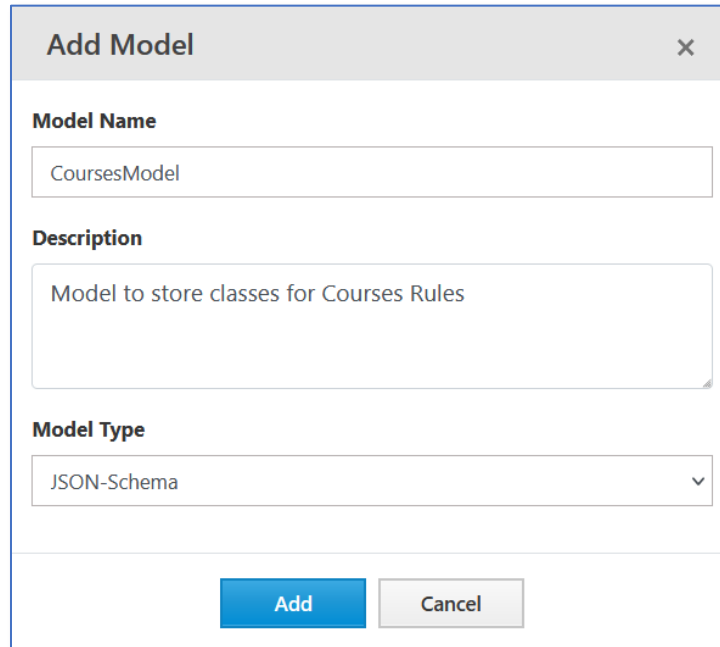
The 'Add project' dialog box contains the following fields:

- Project Name**: Technical Project
- Description**: Technical Project to store the Object Model for the rules
- Project Type**: Technical Project (dropdown menu)
- Project Package Name**: com.rm.dm.courses

At the bottom are 'Add' and 'Cancel' buttons.

Figure 3: Add a new Technical Project for Models and Classes

Next, we create the Model and Classes. We will be creating a “CourseModel” with two classes: Course class to represent the “degree code” and “course code” properties. In the “Request” class, we will create a list of courses, which will be the input and allowed and not allowed courses list which will be populated by the rules based on the challenge rules

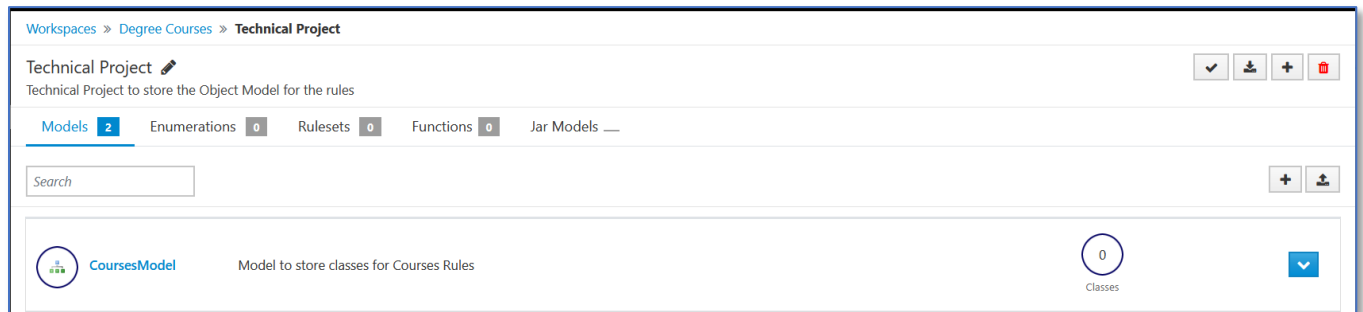


The 'Add Model' dialog box is shown with the following fields:

- Model Name:** CoursesModel
- Description:** Model to store classes for Courses Rules
- Model Type:** JSON-Schema

Buttons at the bottom: Add, Cancel

Figure 4: Add Courses Model



The 'Models in the Project' view shows the 'Technical Project' with the following details:

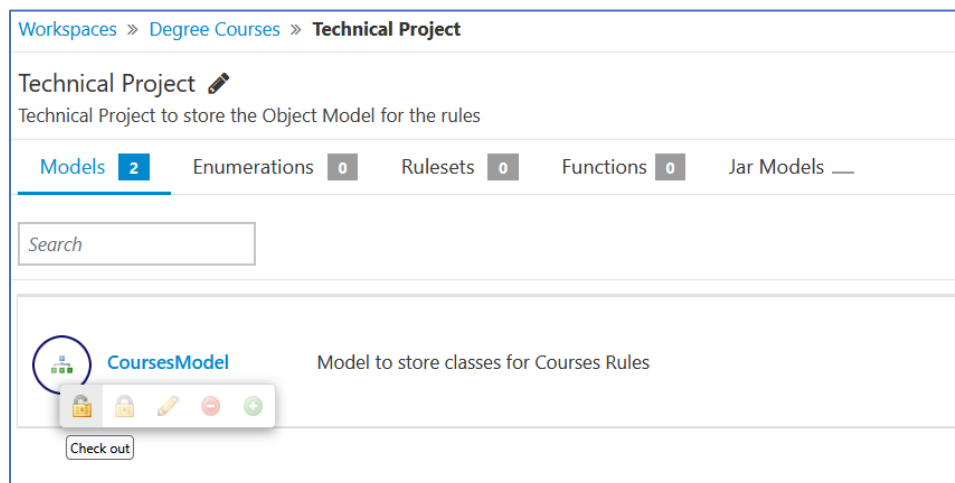
- Workspaces:** Degree Courses » Technical Project
- Technical Project:** Technical Project to store the Object Model for the rules
- Models:** 2 (highlighted)
- Enumerations:** 0
- Rulesets:** 0
- Functions:** 0
- Jar Models:** —

Search bar: Search

Model list:

- CoursesModel** (Model to store classes for Courses Rules) (0 Classes)

Figure 5: Models in the Project



The 'Check out the Model to add Classes' view shows the 'Technical Project' with the following details:

- Workspaces:** Degree Courses » Technical Project
- Technical Project:** Technical Project to store the Object Model for the rules
- Models:** 2 (highlighted)
- Enumerations:** 0
- Rulesets:** 0
- Functions:** 0
- Jar Models:** —

Search bar: Search

Model list:

- CoursesModel** (Model to store classes for Courses Rules)

Check out button: Check out

Figure 6: Check out the Model to add Classes

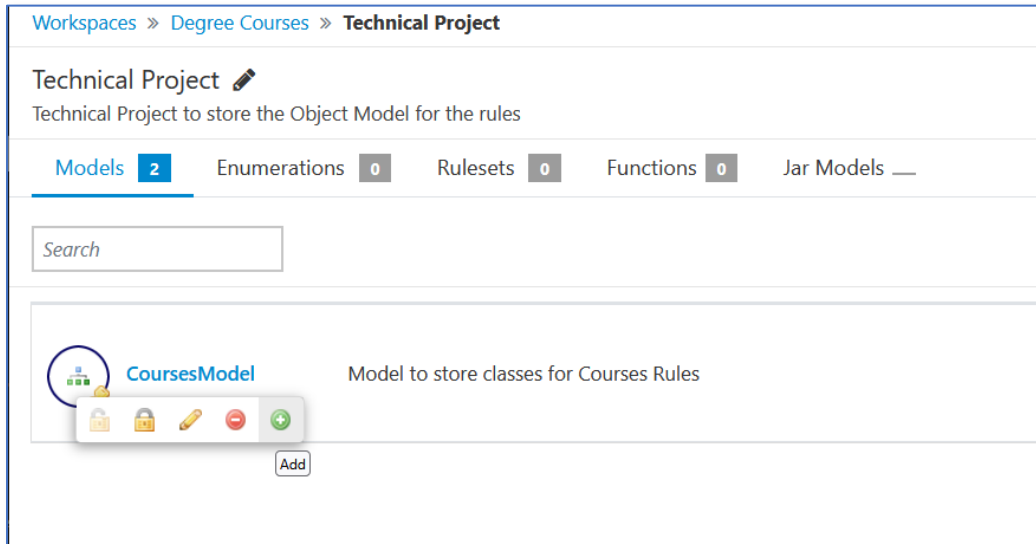


Figure 7: Add Class to the Model

The 'Add Class' dialog box is shown. It has a title bar with a close button. The form contains three fields: 'Class Name' with the value 'Course', 'Parent Class' with a dropdown menu showing '--Select Class--', and 'Description' with the value 'Course Details'. At the bottom, there are 'Add' and 'Cancel' buttons.

Figure 8: Courses Class

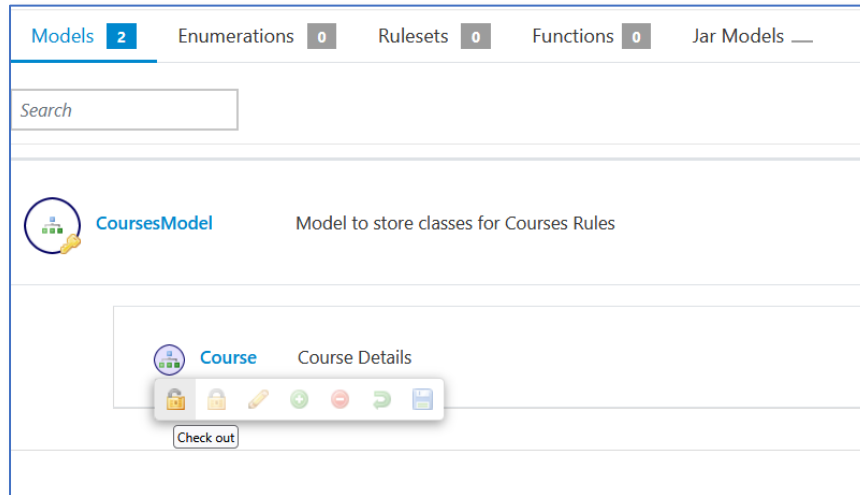


Figure 9: Check out Class to add Fields

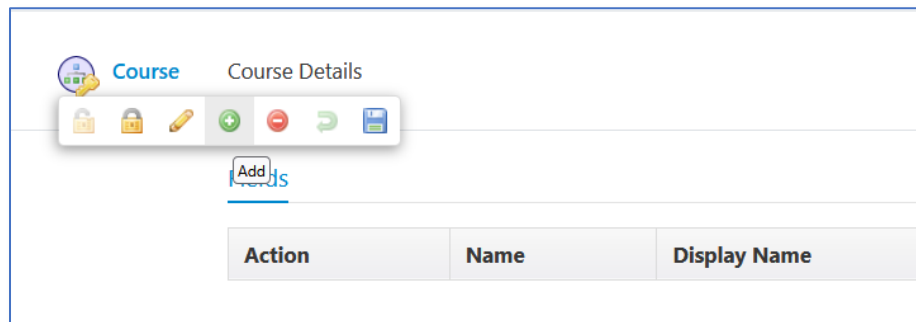



Figure 10: Add Field



**Course**

Course Details

3 Fields

Fields

Action	Name	Display Name	Type	List?	Enumeration List
	courseCd	course code	System.String ▾	<input type="checkbox"/>	▾
	degreeCd	degree code	System.String ▾	<input type="checkbox"/>	▾
	nameTxt	name	System.String ▾	<input type="checkbox"/>	▾


**Request**

Request Class with the list of courses to evaluate

4 Fields

Fields

Action	Name	Display Name	Type	List?	Enumeration List
	allowedCoursesList	allowed courses list	CoursesModel.Course ▾	<input checked="" type="checkbox"/>	▾
	coursesList	courses list	CoursesModel.Course ▾	<input checked="" type="checkbox"/>	▾
	notAllowedCoursesList	not allowed courses list	CoursesModel.Course ▾	<input checked="" type="checkbox"/>	▾

Figure 11: Course and Request class fields

**Add project**
×

**Project Name**

**Description**

**Project Type**  
 ▾

**Project Package Name**

Figure 12: Add Business Project for the Rules

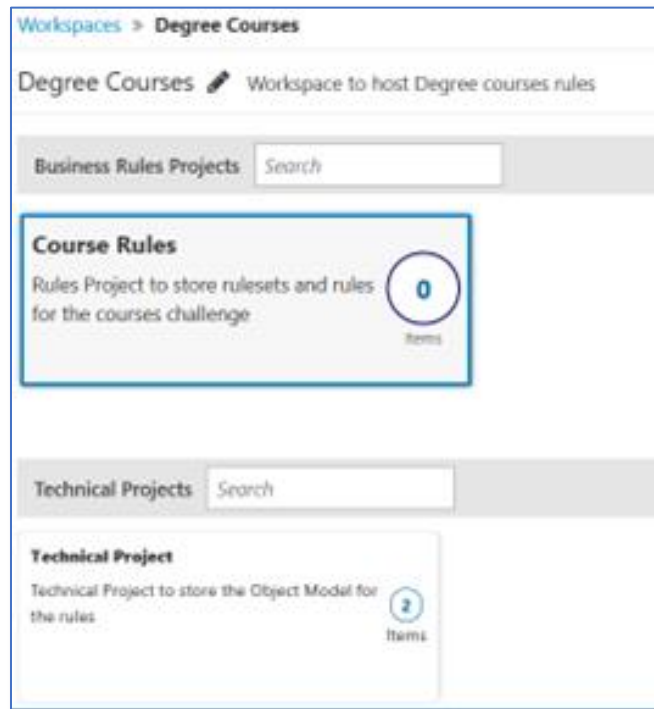


Figure 13: Projects in the Workspace

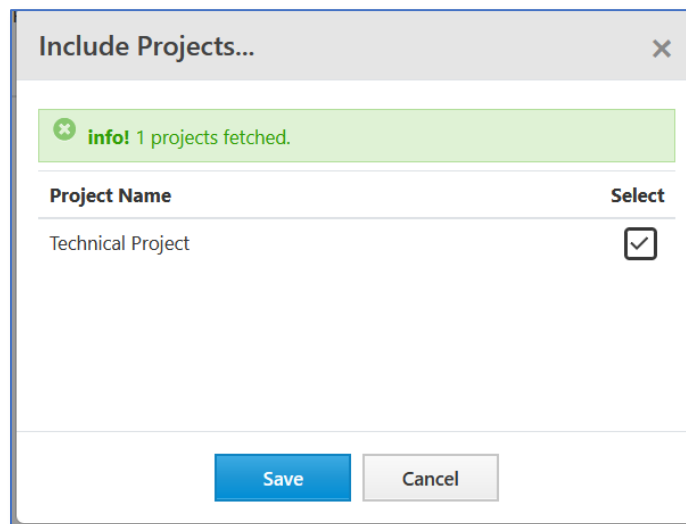


Figure 14: Include Technical Project to the Business Project

Add Ruleset

Rule Name

CourseRules

Description

Course rules to determine which courses are allowed. This ruleset takes the request object and a course object. The rules will insert the course to allowed or not allowed list as applicable

Rule Type

Classic

Rule Parameters

Name	Display Name	Type	List?	
theRequest	the request	CoursesModel.Request ▾	<input type="checkbox"/>	⊖
theCourse	the course	CoursesModel.Course ▾	<input type="checkbox"/>	⊖

Rule Variables

Name	Display Name	Type	Enumeration List	List?	
------	--------------	------	------------------	-------	--

Add

Cancel

Figure 15: Add CourseRules Ruleset with request and course as input



Rule ID: 54693

Name	Description
Rule 1	MA Single Course Rule

When all of the following are true

- the course 's degree code is equal to
- any of the following are true
  - the course 's course code matches
  - the course 's course code matches

+  
+  
then do the following

- Add the course to the request's allowed courses list

+  
Save Cancel

Figure 16: Rule 1 from the Challenge

Rule ID: 54694

Name	Description
Rule 2	CS Course Rule

When all of the following are true

- the course 's degree code does not match
- any of the following are true
  - the course 's course code matches
  - the course 's course code matches

+  
+  
then do the following

- Add the course to the request's not allowed courses list

+  
Save Cancel

Figure 17: Rule 2 from the Challenge


Workspaces » Degree Courses » Course Rules

### Course Rules

Rules Project to store rulesets and rules for the courses challenge

Models **0** Enumerations **0** **Rulesets 1** Functions **1** Tests **0**

Search


**CourseRules**

Course rules to determine which courses are allowed. This ruleset takes the request object and a course object. The rules will insert the course to allowed or not allowed list as applicable

2 Rules



Action	Name	Description	Active	Effective Date	Expiration
	Rule 1	MA Single Course Rule	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Rule 2	CS Course Rule	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 18: Rules in the Ruleset

### Edit Function ✕


**Function Name**

CourseRulesEntryPoint


**Description**

This function is the entry point to the course rules. This function loops through all courses in the request object and applies course ruleset to determine which courses are allowed

**Function Parameters** +

Name	Display Name	Type	List?	
theRequest	the request	CoursesModel.Request ▾	<input type="checkbox"/>	

**Function Variables** +

Name	Display Name	Type	List?	Enumeration List	Initial Value	
theCourse	the course	CoursesModel.Course ▾	<input type="checkbox"/>	▾	<input type="checkbox"/>	

**Save** **Cancel**

Figure 19: Add Entry Point Function

Function ID: CourseRulesEntryPoint

Name	Description
CourseRulesEntryPoint	This function is the entry point to the course rules. This function loops through all courses in the request object and applies course ruleset to determine which courses are allowed

Steps:

- for each the course in the request's courses list
  - apply ruleset CourseRules with the request and the course

+  
+

Save Cancel

Figure 20: Loop in the Function

Workspaces » Degree Courses » Course Rules

Course Rules

Rules Project to store rulesets and rules for the courses challenge

Models 0 Enumerations 0 Rulesets 1 Functions 1 Tests 0

Search

CourseRulesEntryPoint This function is the entry point to the course rules. This function loops through all courses in the request object and applies course ruleset to determine which courses are allowed

1 Steps

Figure 21: Entry point functions in the project

Download Project

Download Type

Java

Download Cancel

Figure 22: Download the project as a Java Project

```

Test.java CourseRules.java
2
3 public class CourseRules extends com.rm.rmt.engine.BaseRuleset {
4
5 private com.rm.dm.course.models.CoursesModel.Request theRequest =
6     new com.rm.dm.course.models.CoursesModel.Request();
7 private com.rm.dm.course.models.CoursesModel.Course theCourse =
8     new com.rm.dm.course.models.CoursesModel.Course();
9
10 public CourseRules(
11     com.rm.dm.course.models.CoursesModel.Request theRequest,
12     com.rm.dm.course.models.CoursesModel.Course theCourse) {
13
14     this.theRequest = theRequest;
15     this.theCourse = theCourse;
16 }
17
18 public void execute() throws Exception {
19     if (!rule_54693()) return;
20     if (!rule_54694()) return;
21 }
22
23 // Rule 1-MA Single Course Rule
24 private static final String RULE_NAME_54693 = "CourseRules.Rule 1";
25
26 private boolean rule_54693() throws Exception {
27     boolean retValue = true;
28
29     if (compare(theCourse.getDegreeCd(), EQ, CONSTANT_0)
30         && (compare(theCourse.getCourseCd(), MATCHES, CONSTANT_1)
31             || compare(theCourse.getCourseCd(), MATCHES, CONSTANT_2))) {
32         if (theRequest.getAllowedCoursesList() == null) {
33             theRequest.setAllowedCoursesList(
34                 new java.util.ArrayList<com.rm.dm.course.models.CoursesModel.Course>());
35         }
36         theRequest.getAllowedCoursesList().add(theCourse);
37         theCourse = new com.rm.dm.course.models.CoursesModel.Course();
38
39         addRulesFiredList(RULE_NAME_54693);
40     }
41     return retValue;
42 }
43
44 // Rule 2-CS Course Rule
45 private static final String RULE_NAME_54694 = "CourseRules.Rule 2";
46
47 private boolean rule_54694() throws Exception {
48     boolean retValue = true;
49
50     if (compare(theCourse.getDegreeCd(), NOT_MATCHES, CONSTANT_3)
51         && (compare(theCourse.getCourseCd(), MATCHES, CONSTANT_4)
52             || compare(theCourse.getCourseCd(), MATCHES, CONSTANT_5))) {
53         if (theRequest.getNotAllowedCoursesList() == null) {
54             theRequest.setNotAllowedCoursesList(
55                 new java.util.ArrayList<com.rm.dm.course.models.CoursesModel.Course>());
56         }
57         theRequest.getNotAllowedCoursesList().add(theCourse);
58         theCourse = new com.rm.dm.course.models.CoursesModel.Course();
59
60         addRulesFiredList(RULE_NAME_54694);
61     }
62     return retValue;
63 }
64
65 public static final String CONSTANT_0 = "MA Single";
66 public static final String CONSTANT_1 = "MA52.";
67 public static final String CONSTANT_2 = "MA62.";
68 public static final String CONSTANT_3 = "CS.*";
69 public static final String CONSTANT_4 = "CS.*288.*";
70 public static final String CONSTANT_5 = "CS.*289.*";
71 }

```

Figure 23: Executable Java Code with the Two Rules

```

private Request buildRequest() {
    Request req = new Request();

    Course c1 = new Course();
    c1.setDegreeCd("MA Single");
    c1.setCourseCd("MA521");
    req.getCoursesList().add(c1);

    Course c2 = new Course();
    c2.setDegreeCd("MA Single");
    c2.setCourseCd("MA522");
    req.getCoursesList().add(c2);

    Course c3 = new Course();
    c3.setDegreeCd("MA Single");
    c3.setCourseCd("MA621");
    req.getCoursesList().add(c3);

    Course c4 = new Course();
    c4.setDegreeCd("MA Single");
    c4.setCourseCd("MA622");
    req.getCoursesList().add(c4);

    Course c5 = new Course();
    c5.setDegreeCd("MA Single");
    c5.setCourseCd("CS12881");
    req.getCoursesList().add(c5);

    Course c6 = new Course();
    c6.setDegreeCd("MA Single");
    c6.setCourseCd("CS12891");
    req.getCoursesList().add(c6);

    return req;
}

private void execute() throws Exception {
    CourseRulesEntryPoint crep = new CourseRulesEntryPoint();
    Request req = buildRequest();
    crep.initialize(new Object[] { req});

    Date startTime = new Date();
    crep.execute();
    Date endTime = new Date();
    System.out.println("Execution Time:" + (endTime.getTime()-startTime.getTime()));

    System.out.println("\nRules Fired:");
    for (String rule:crep.getRulesFiredList()) {
        System.out.println("    Rule: " + rule);
    }

    System.out.println("\nAllowed Courses:");
    for (Course allowedCourse:req.getAllowedCoursesList()) {
        System.out.println("    Degree: " + allowedCourse.getDegreeCd() + " Course: " + allowedCourse.getCourseCd());
    }
    System.out.println("\nNot Allowed Courses:");
    for (Course notAllowedCourse:req.getNotAllowedCoursesList()) {
        System.out.println("    Degree: " + notAllowedCourse.getDegreeCd() + " Course: " + notAllowedCourse.getCourseCd());
    }
}

```

Figure 24: Test Code

```
Warm Up Execution Time:19
Execution Time:8

Rules Fired:
  Rule: CourseRulesEntryPoint.for each Course in theRequest.CoursesList
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 2
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 2

Allowed Courses:
  Degree: MA Single Course: MA521
  Degree: MA Single Course: MA522
  Degree: MA Single Course: MA621
  Degree: MA Single Course: MA622

Not Allowed Courses:
  Degree: MA Single Course: CS12881
  Degree: MA Single Course: CS12891
```

Figure 25: Test Code Output

```

1 package com.rm.dm.course.drools;
2 dialect "mvel"
3
4 rule "54693-Rule 1"
5 agenda-group "CourseRules"
6
7
8 salience -0
9 when
10     theRequest:com.rm.dm.course.models.CoursesModel.Request()
11     theCourse:com.rm.dm.course.models.CoursesModel.Course()
12
13
14     theRuleset: com.rm.rmt.engine.BaseRuleset(
15         (theCourse.getDegreeCd() == (com.rm.dm.course.constants.Constants.CONSTANT_0)) &&
16         ((theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_1)) ||
17         (theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_2))
18         )
19     )
20 then
21
22     theRequest.getAllowedCoursesList().add(theCourse);
23     theCourse = new com.rm.dm.course.models.CoursesModel.Course();
24     theRuleset.addRulesFiredList("CourseRules.Rule 1");
25 end
26
27 rule "54694-Rule 2"
28 agenda-group "CourseRules"
29
30
31 salience -1
32 when
33     theRequest:com.rm.dm.course.models.CoursesModel.Request()
34     theCourse:com.rm.dm.course.models.CoursesModel.Course()
35
36
37     theRuleset: com.rm.rmt.engine.BaseRuleset(
38         (theCourse.getDegreeCd() not matches (com.rm.dm.course.constants.Constants.CONSTANT_3)) &&
39         ((theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_4)) ||
40         (theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_5))
41         )
42     )
43 then
44
45     theRequest.getNotAllowedCoursesList().add(theCourse);
46     theCourse = new com.rm.dm.course.models.CoursesModel.Course();
47     theRuleset.addRulesFiredList("CourseRules.Rule 2");
48 end
49
50

```

Figure 26: The Drools DRL Code

```
Warm Up Execution Time:4110
Execution Time:16

Rules Fired:
  Rule: CourseRulesEntryPoint.for each Course in theRequest.CoursesList
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 1
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 2
  Rule: CourseRulesEntryPoint.apply ruleset CourseRules
  Rule: CourseRules.Rule 2

Allowed Courses:
  Degree: MA Single Course: MA521
  Degree: MA Single Course: MA522
  Degree: MA Single Course: MA621
  Degree: MA Single Course: MA622

Not Allowed Courses:
  Degree: MA Single Course: CS12881
  Degree: MA Single Course: CS12891
```

Figure 27: Drools run output