We will be using Rulesmatix LABS Rules Management System to create the rules for the DM Communitys 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 snap shots 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 or 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.

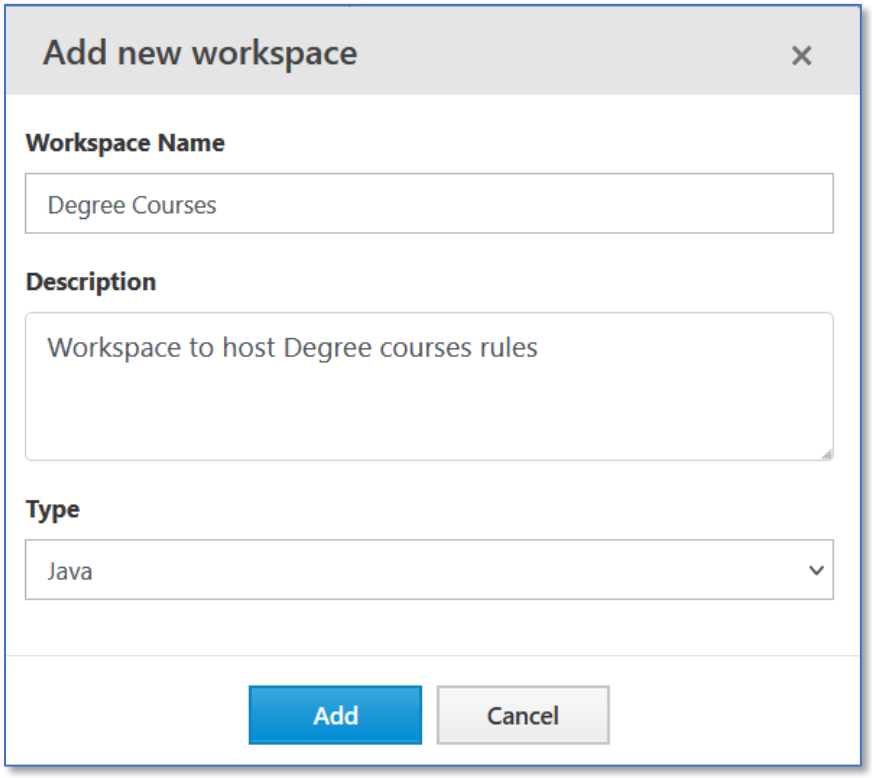

Figure 1: Add new Workspace

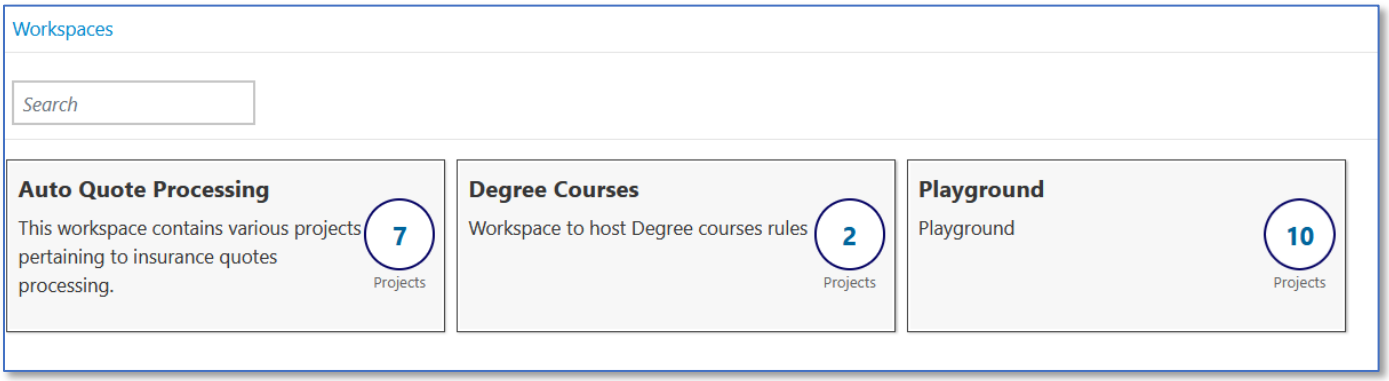

Figure 2: All Workspaces

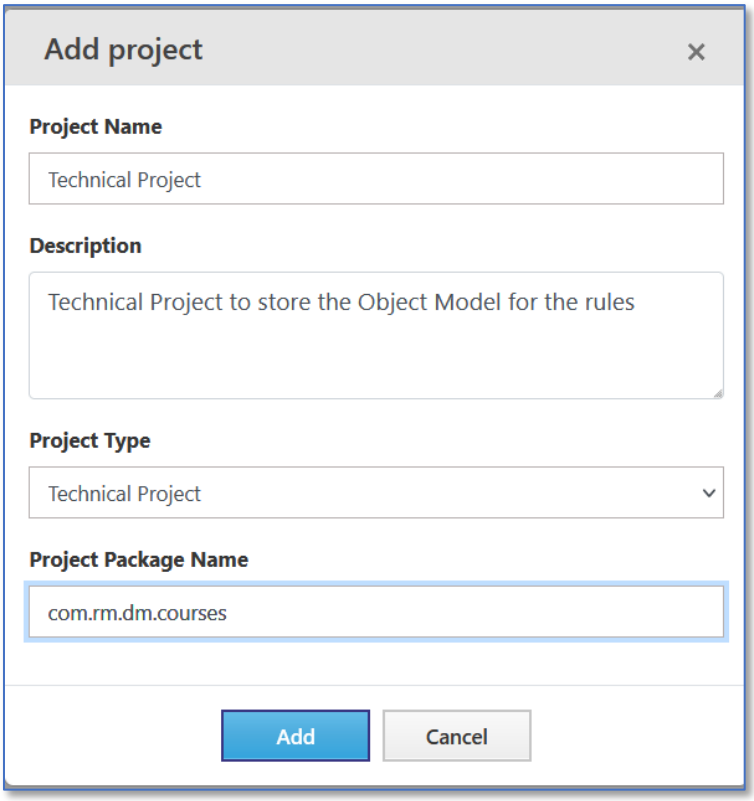

Figure 3: Add a new Technical Project for Models and Classes

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

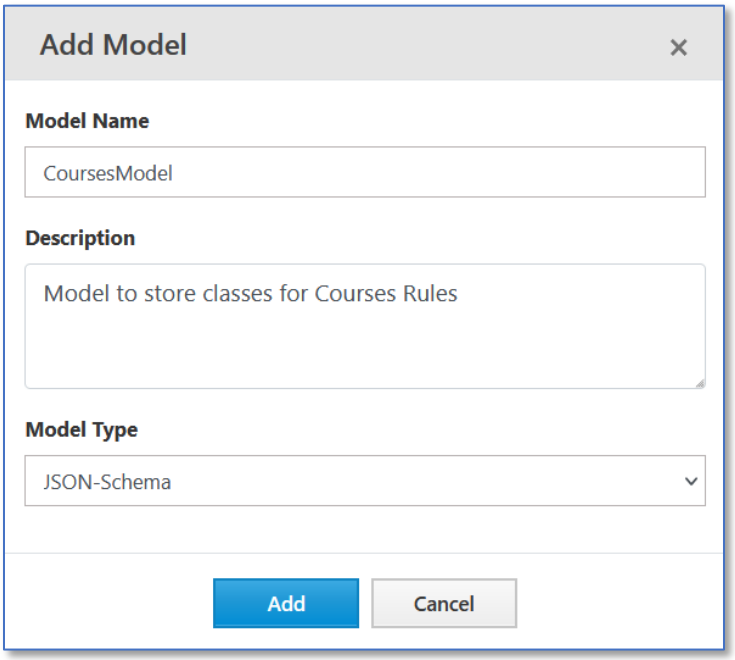

Figure 4: Add Courses Model

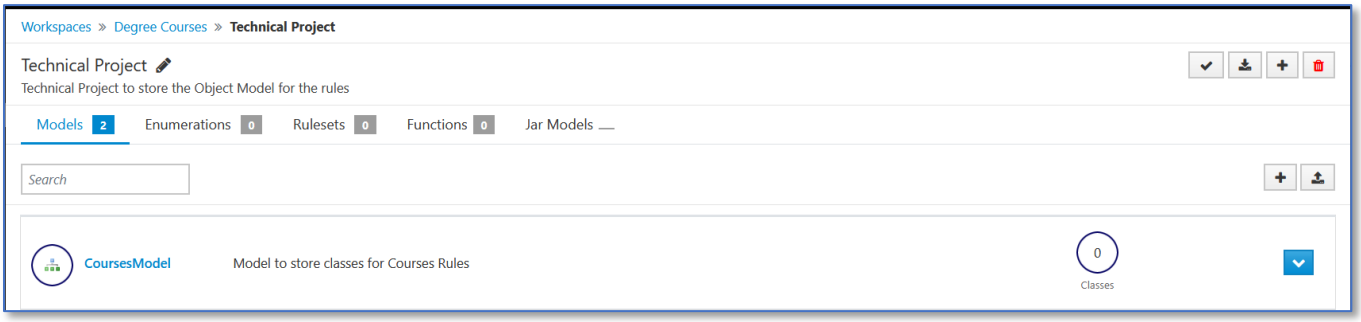

Figure 5: Models in the Project

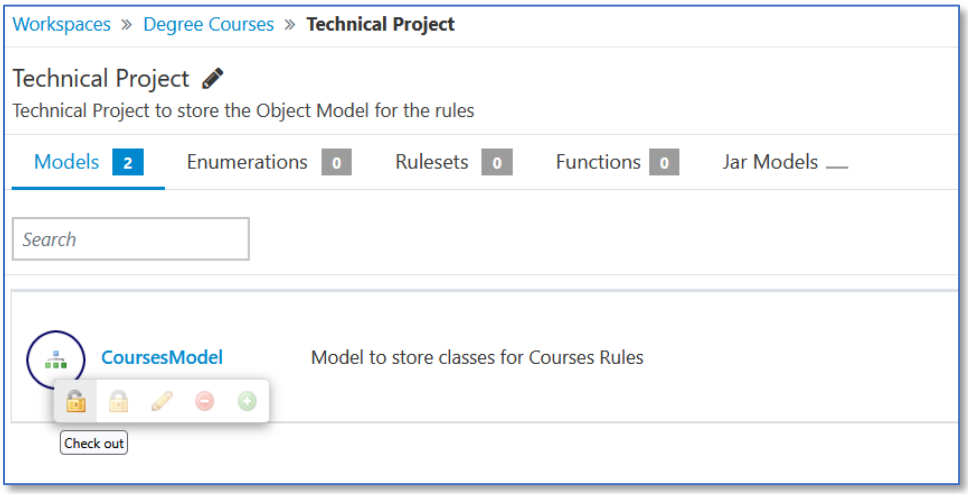

Figure 6: Check out the Model to add Classes

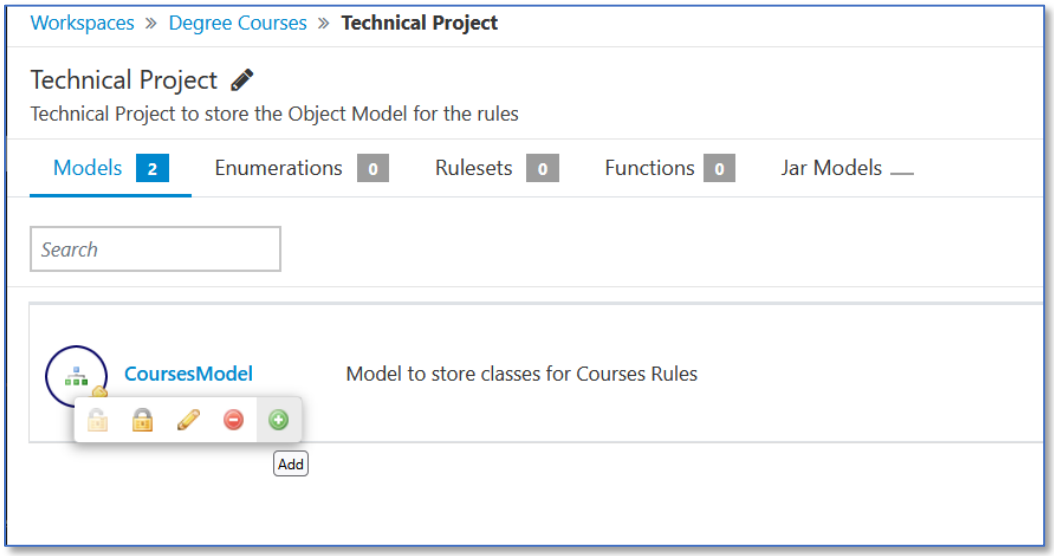

Figure 7: Add Class to the Model

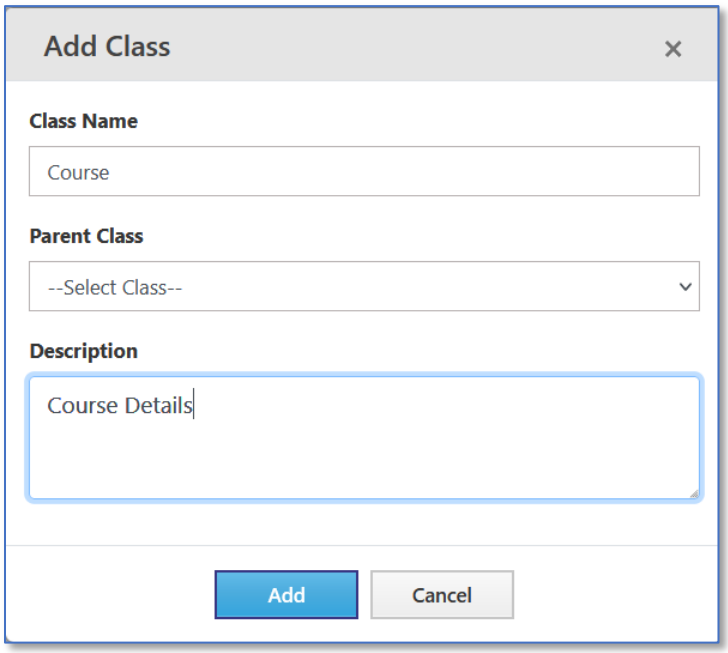

Figure 8: Courses Class

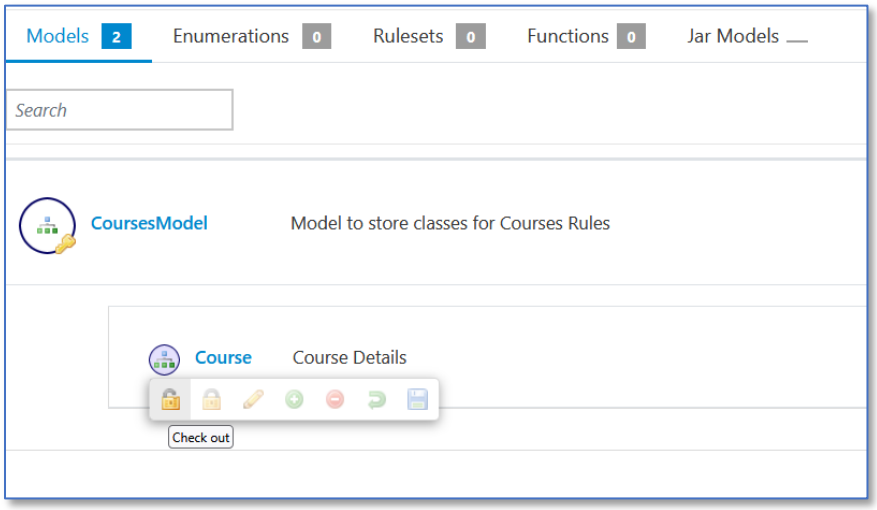

Figure 9: Check out Class to add Fields

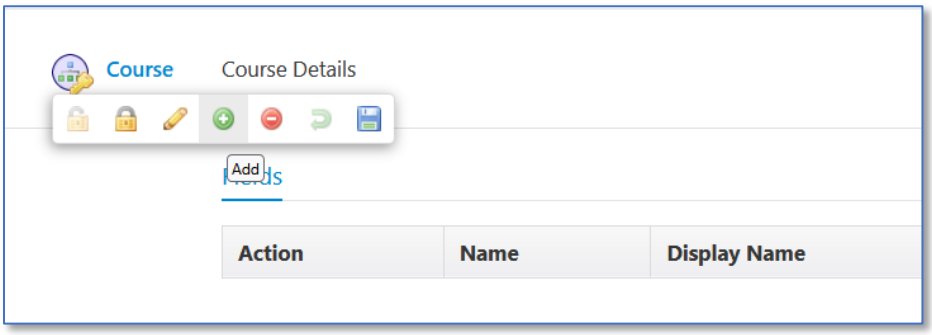

Figure 10: Add Field

Course Course Details 3 Fields

Fields

Action	Name	Display Name	Type	List?	Enumeration List
	courseCd	course code	System.String	☐	
	degreeCd	degree code	System.String	☐	
	nameTxt	name	System.String	☐	

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	☑	
	coursesList	courses list	CoursesModel.Course	☑	
	notAllowedCoursesList	not allowed courses list	CoursesModel.Course	☑	

Figure 11: Course and Request class fields

Add project ×

Project Name

Course Rules

Description

Rules Project to store rulesets and rules for the courses challenge

Project Type

Business User's Project

Project Package Name

com.rm.dm.course

Add Cancel

Figure 12: Add Business Project for the Rules

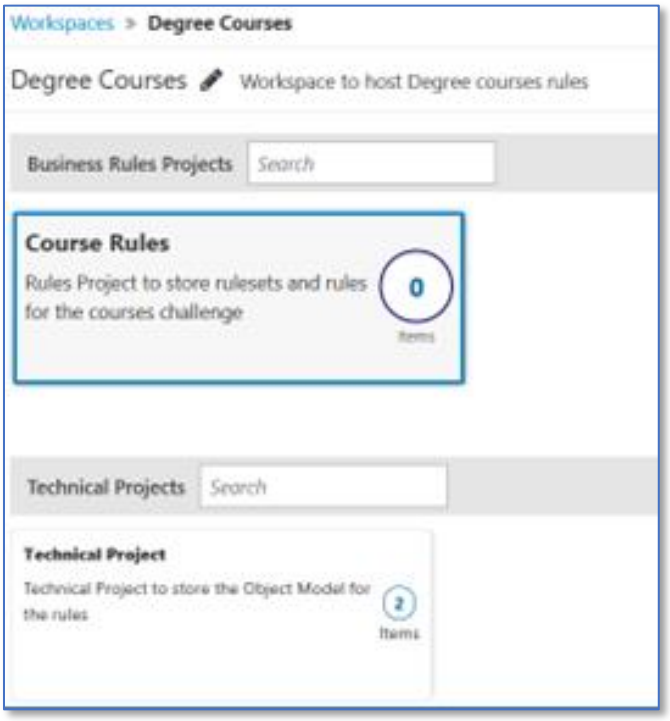

Figure 13: Projects in the Workspace

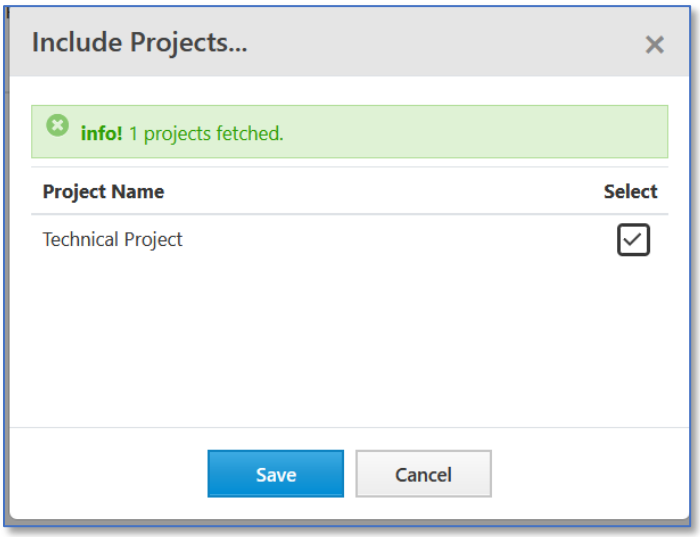

Figure 14: Include Technical Project to the Business Project

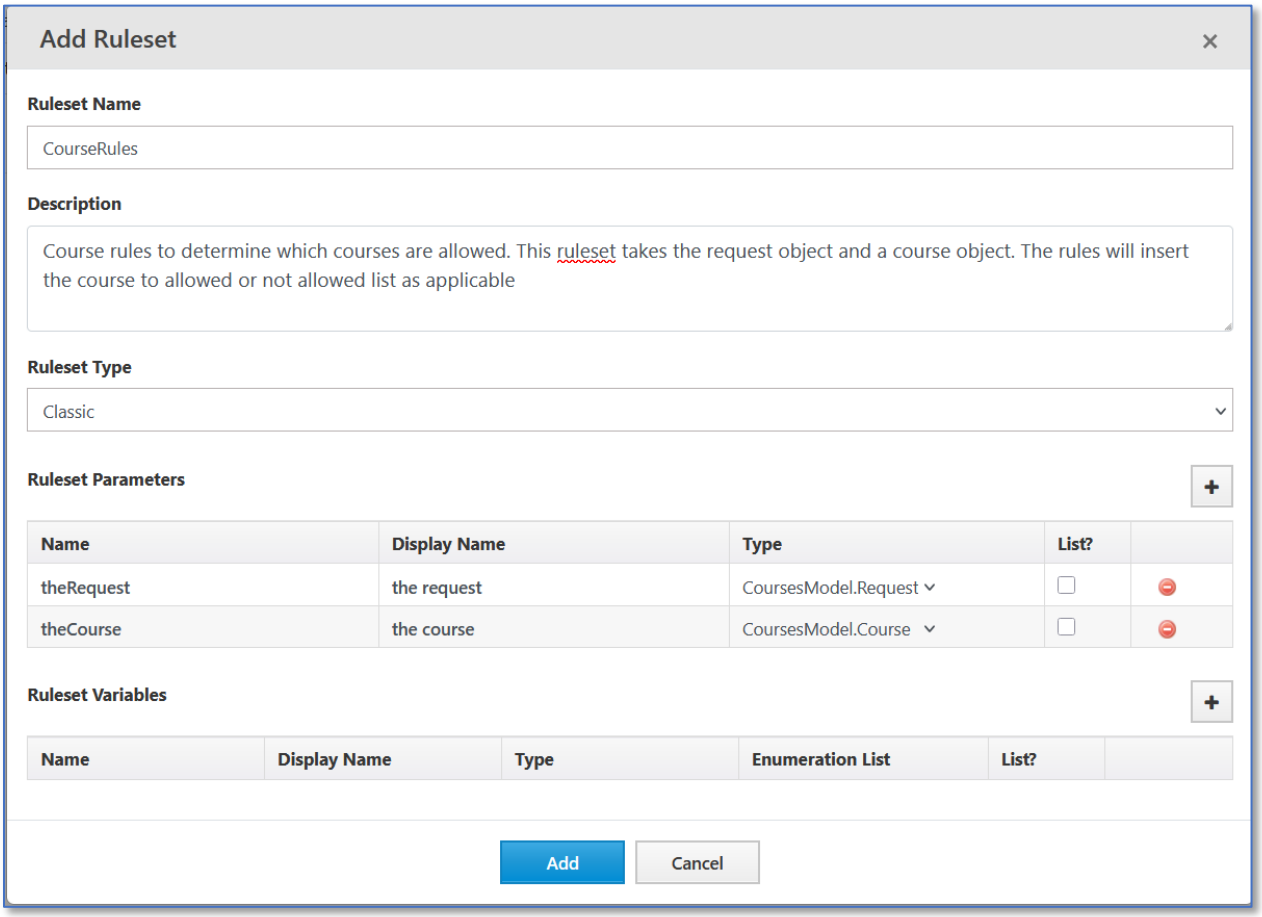

Figure 15: Add CourseRules Ruleset with request and course as input

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 MA Single
- any of the following are true
  - the course 's course code matches MA52.
  - the course 's course code matches MA62.

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 CS.*
- any of the following are true
  - the course 's course code matches CS.*288.*
  - the course 's course code matches CS.*289.*

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	☑	☐	☐
	Rule 2	CS Course Rule	☑	☐	☐

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	☐

Function Variables

Name	Display Name	Type	List?	Enumeration List	Initial Value
theCourse	the course	CoursesModel.Course	☐		☐

Save | Cancel

Figure 19: Add Entry Point Function

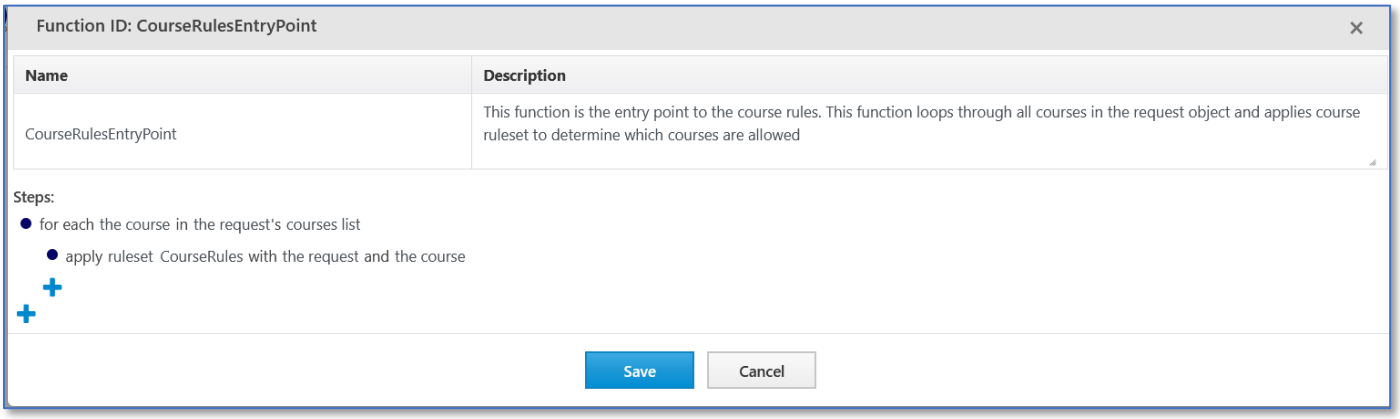

Figure 20: Loop in the Function

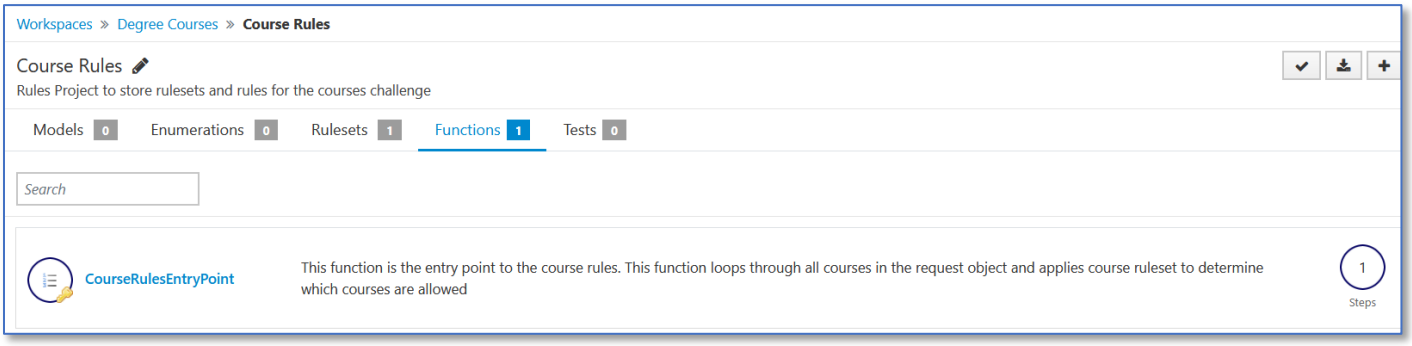

Figure 21: Entry point functions in the project

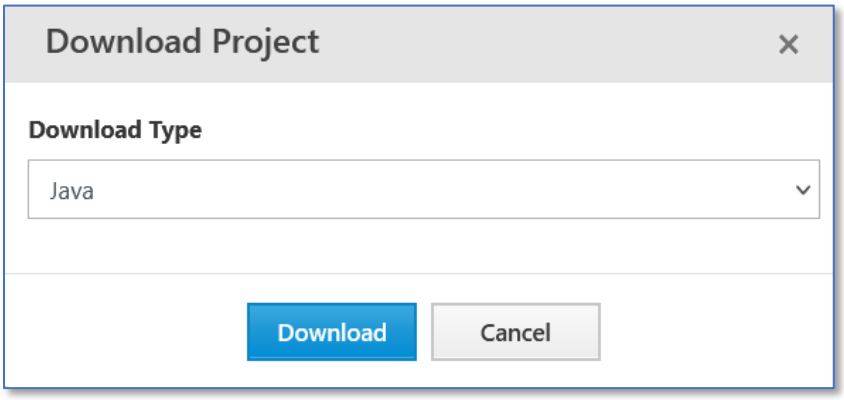

Figure 22: Download the project as a Java Project

Test.java | CourseRules.java

```java
public class CourseRules extends com.rm.rmt.engine.BaseRuleset {

  private com.rm.dm.course.models.CoursesModel.Request theRequest =
      new com.rm.dm.course.models.CoursesModel.Request();
  private com.rm.dm.course.models.CoursesModel.Course theCourse =
      new com.rm.dm.course.models.CoursesModel.Course();

  public CourseRules(
      com.rm.dm.course.models.CoursesModel.Request theRequest,
      com.rm.dm.course.models.CoursesModel.Course theCourse) {

    this.theRequest = theRequest;
    this.theCourse = theCourse;
  }

  public void execute() throws Exception {
    if (!rule_54693()) return;
    if (!rule_54694()) return;
  }

  // Rule 1-MA Single Course Rule
  private static final String RULE_NAME_54693 = "CourseRules.Rule 1";

  private boolean rule_54693() throws Exception {
    boolean retValue = true;

    if (compare(theCourse.getDegreeCd(), EQ, CONSTANT_0)
        && (compare(theCourse.getCourseCd(), MATCHES, CONSTANT_1)
            || compare(theCourse.getCourseCd(), MATCHES, CONSTANT_2))) {
      if (theRequest.getAllowedCoursesList() == null) {
        theRequest.setAllowedCoursesList(
            new java.util.ArrayList<com.rm.dm.course.models.CoursesModel.Course>());
      }
      theRequest.getAllowedCoursesList().add(theCourse);
      theCourse = new com.rm.dm.course.models.CoursesModel.Course();

      addRulesFiredList(RULE_NAME_54693);
    }
    return retValue;
  }

  // Rule 2-CS Course Rule
  private static final String RULE_NAME_54694 = "CourseRules.Rule 2";

  private boolean rule_54694() throws Exception {
    boolean retValue = true;

    if (compare(theCourse.getDegreeCd(), NOT_MATCHES, CONSTANT_3)
        && (compare(theCourse.getCourseCd(), MATCHES, CONSTANT_4)
            || compare(theCourse.getCourseCd(), MATCHES, CONSTANT_5))) {
      if (theRequest.getNotAllowedCoursesList() == null) {
        theRequest.setNotAllowedCoursesList(
            new java.util.ArrayList<com.rm.dm.course.models.CoursesModel.Course>());
      }
      theRequest.getNotAllowedCoursesList().add(theCourse);
      theCourse = new com.rm.dm.course.models.CoursesModel.Course();

      addRulesFiredList(RULE_NAME_54694);
    }
    return retValue;
  }

  public static final String CONSTANT_0 = "MA Single";
  public static final String CONSTANT_1 = "MA52.";
  public static final String CONSTANT_2 = "MA62.";
  public static final String CONSTANT_3 = "CS.*";
  public static final String CONSTANT_4 = "CS.*288.*";
  public static final String CONSTANT_5 = "CS.*289.*";
}
```

Figure 23: Executable Java Code with the Two Rules

```java
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

```
package com.rm.dm.course.drools;
dialect  "mvel"

rule "54693-Rule 1"
agenda-group "CourseRules"


salience -0
when
        theRequest:com.rm.dm.course.models.CoursesModel.Request()
        theCourse:com.rm.dm.course.models.CoursesModel.Course()


        theRuleset: com.rm.rmt.engine.BaseRuleset(
            (theCourse.getDegreeCd() == (com.rm.dm.course.constants.Constants.CONSTANT_0)) &&
            ((theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_1)) ||
                    (theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_2))
                )
        )
then

        theRequest.getAllowedCoursesList().add(theCourse);
        theCourse = new com.rm.dm.course.models.CoursesModel.Course();
        theRuleset.addRulesFiredList("CourseRules.Rule 1");
end

rule "54694-Rule 2"
agenda-group "CourseRules"


salience -1
when
        theRequest:com.rm.dm.course.models.CoursesModel.Request()
        theCourse:com.rm.dm.course.models.CoursesModel.Course()


        theRuleset: com.rm.rmt.engine.BaseRuleset(
            (theCourse.getDegreeCd() not matches (com.rm.dm.course.constants.Constants.CONSTANT_3)) &&
            ((theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_4)) ||
                    (theCourse.getCourseCd() matches (com.rm.dm.course.constants.Constants.CONSTANT_5))
                )
        )
then

        theRequest.getNotAllowedCoursesList().add(theCourse);
        theCourse = new com.rm.dm.course.models.CoursesModel.Course();
        theRuleset.addRulesFiredList("CourseRules.Rule 2");
end


```

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