

Assignment A4

Team name: PowerPuff Rangers

Team members:

1. Rosheen Naeem
2. Mehdi Karmouche
3. Alejandro Enriquez

Introduction of our metamodel:

Our metamodel is a web application which consists of multiple pages. We can create as many instances of meta-model as needed, i.e., the number of websites. Each website will have a name, dark_mode as attributes. Each page of the website will have content to display, this content can be of different types. Static content, i.e., Form and dynamic content, i.e., DContent. The content class refers to the Entity class, which means we are getting the content from the database, and every entity is treated as a table of the database. Every entity has an EAttribute and EReference. Attribute will define the content we are getting from the database and EReference will refer to the database.

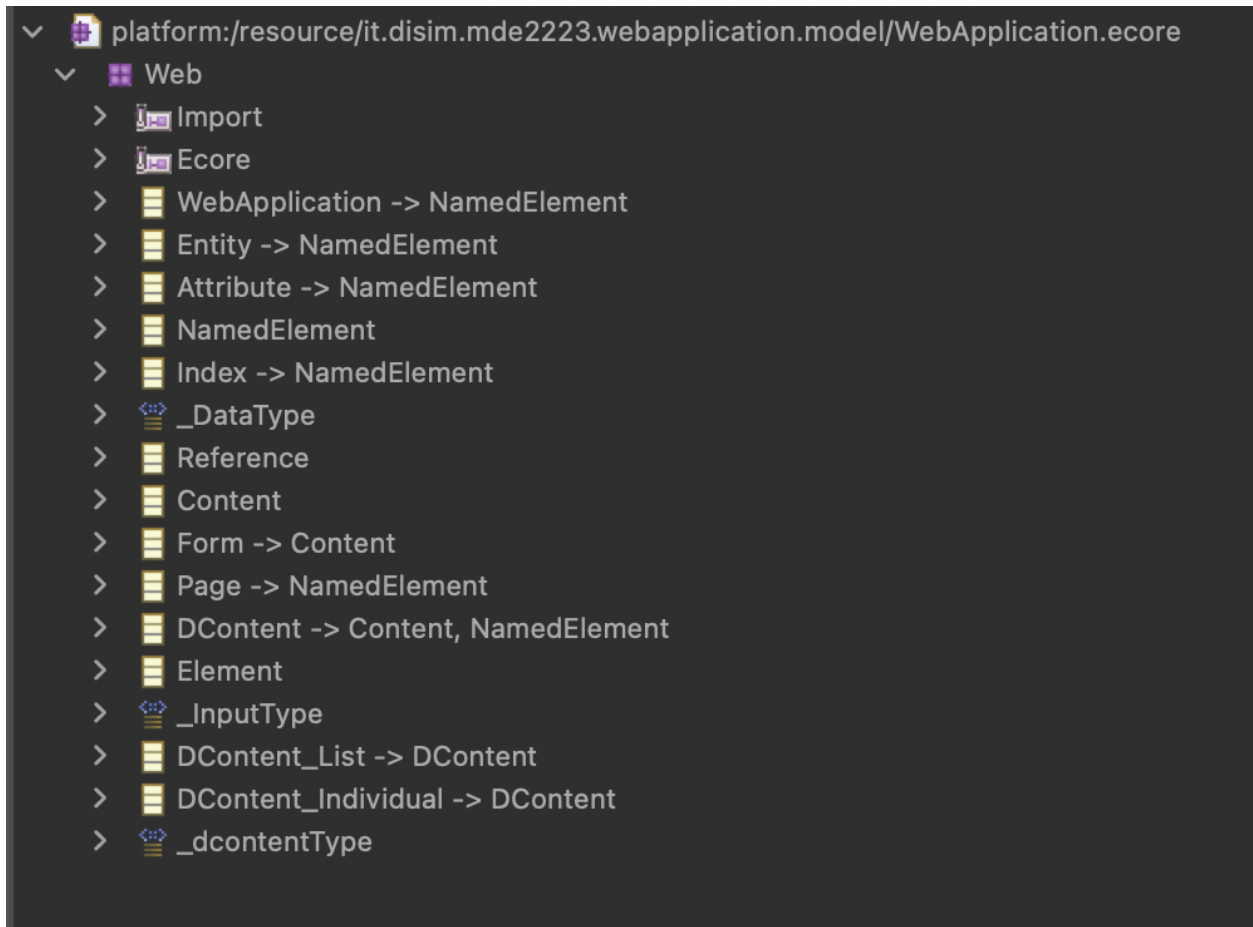
For the Content subclasses, Form consists of input elements of different types, like textbox or dropdown. We have represented forme elements by adding an enumeration '_InputType'. On the other hand, We have further divided the DContent into two types, 'DContent_List' and 'DContent_Individual'.

We are using EMF to create the model and OCL to create constraints.

TASK A4.1:

Metamodel:

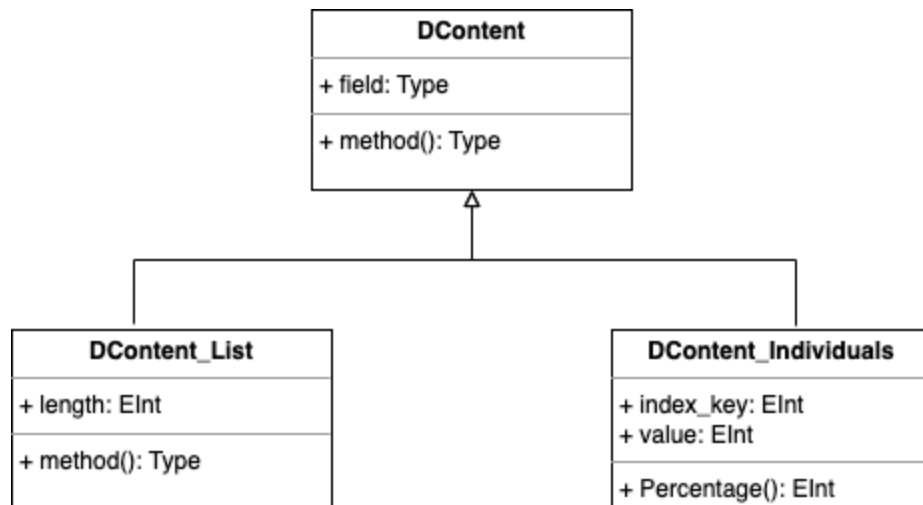
We created 13 metaclasses, which are connected to each other, to describe our domain.



Inheritance:

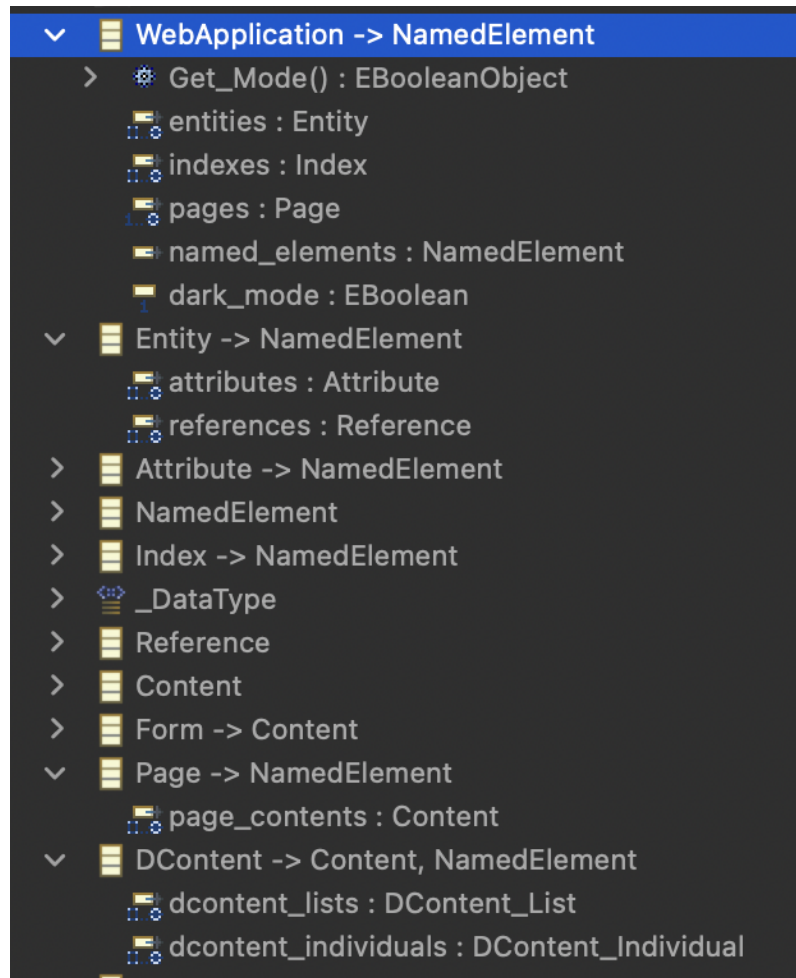
We added multiple inheritance concepts in our metamodel. For instance, the Content class is extended by two classes, Static Content, i.e., Form, and Dynamic Content, i.e., DContent.

Similarly, DContent is further an ESuper Type for DContent_List and DContent_Individual as shown in the diagram below.



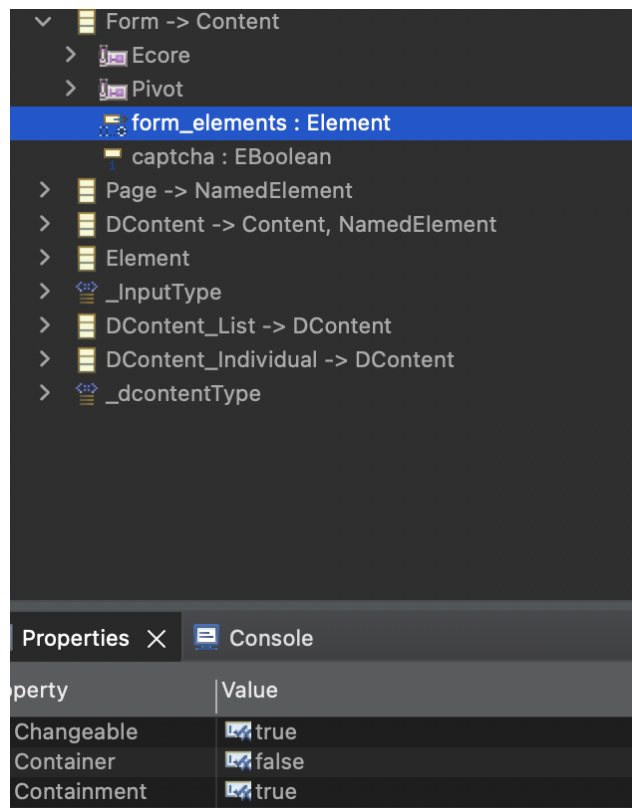
Relations:

Almost every metaclass consists of EReferences, in our metamodel as shown in the image below. Our metamodel is a web application which consists of multiple pages. Every page has content which is referring to the database entities.



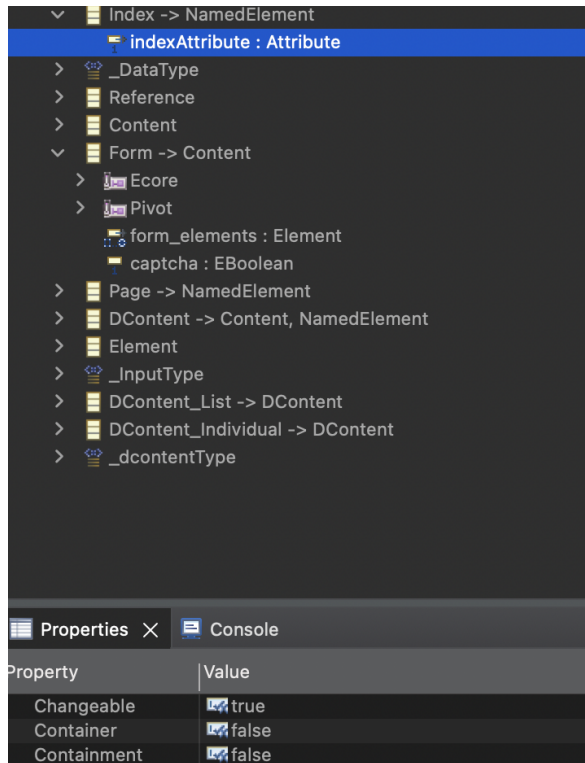
Containment:

Most of the relations in our metamodel have containment and are used to create references to other classes in instance models. For example, the Form class has EReference to Element, for which containment is true, and in instance models, we are creating elements in the model.



Non-Containment:

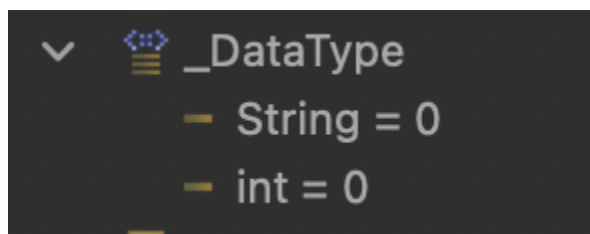
For a few relations, containment is false, for example, in the Index class, the EReference 'indexAttribute' has non-containment as shown in the image below.



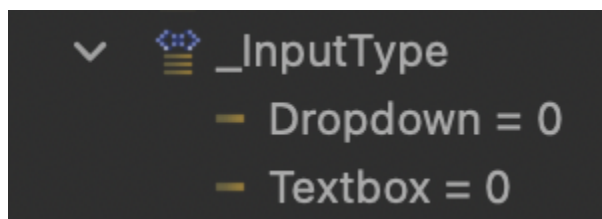
Enumeration Types:

Our metamodel consists of three enumeration types.

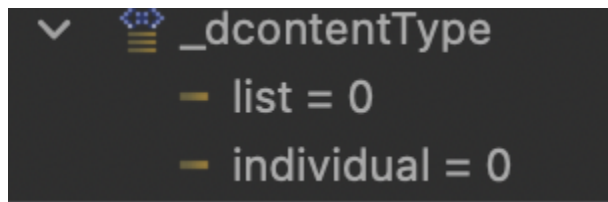
1. `_DataType`



2. `_InputType`



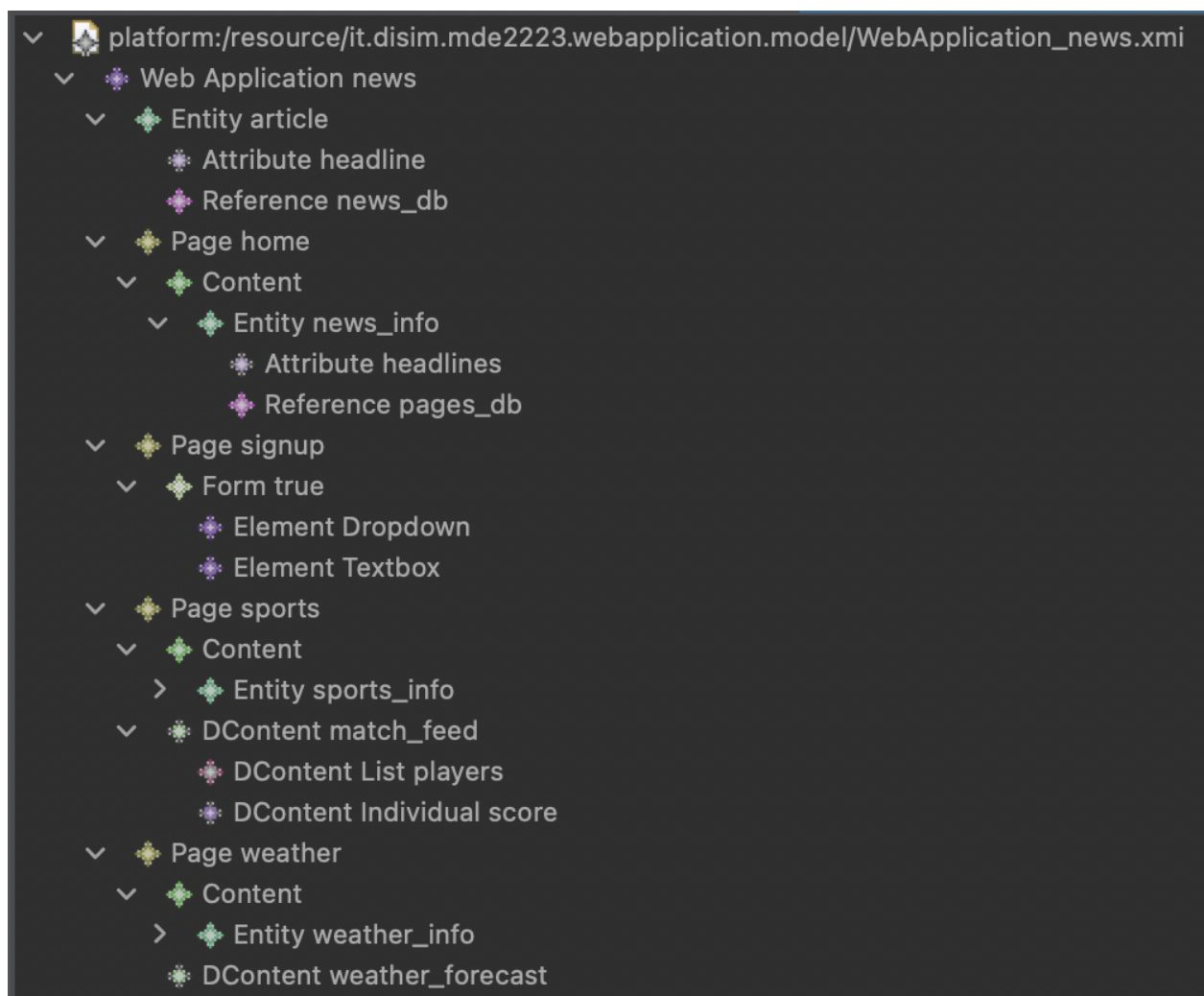
3. `_dcontentType`



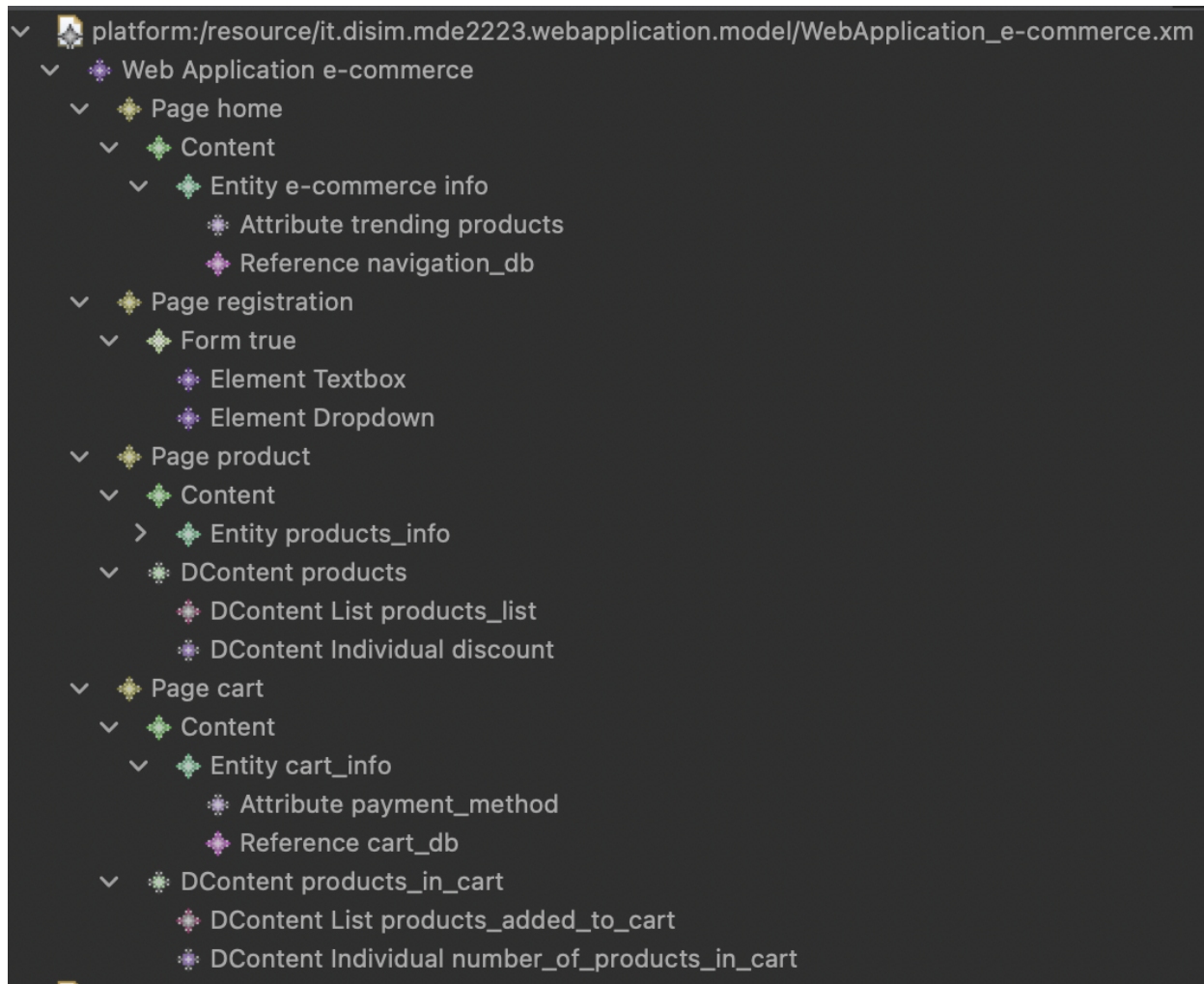
Task A4.2

We created two concrete instances of our metamodel, and each is covering every metaclass of the model, References, EAttributes, constraints, operations, and derives fields.

1. Model instance News Website



2. Model Instance E-Comeerce website



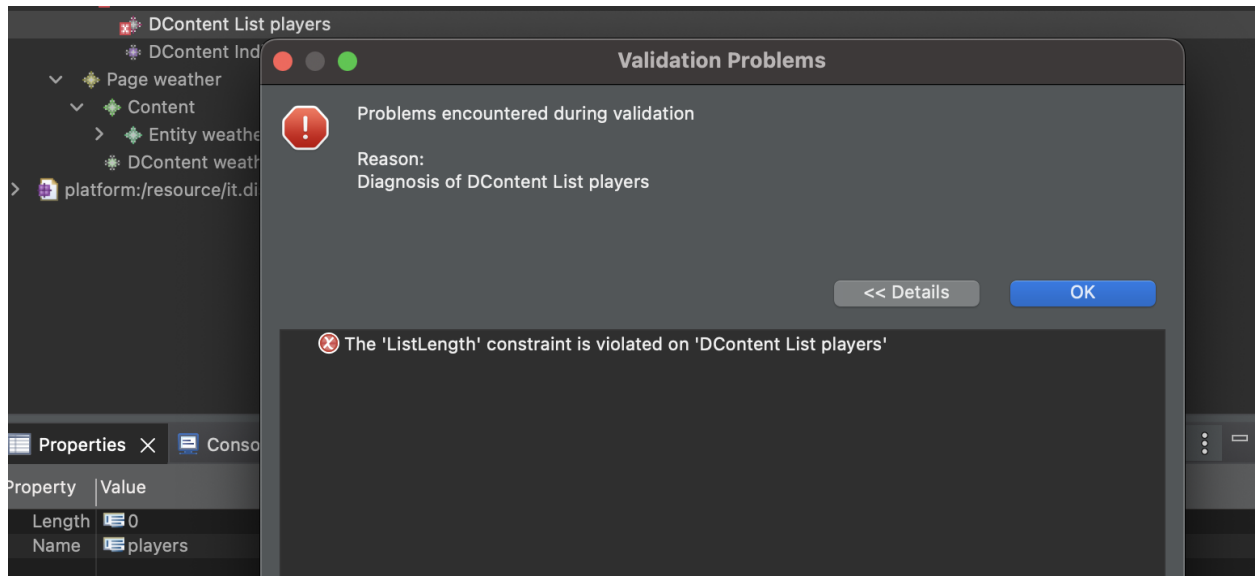
TASK A4.3

Constraints:

We added 3 constraints on different metaclasses.

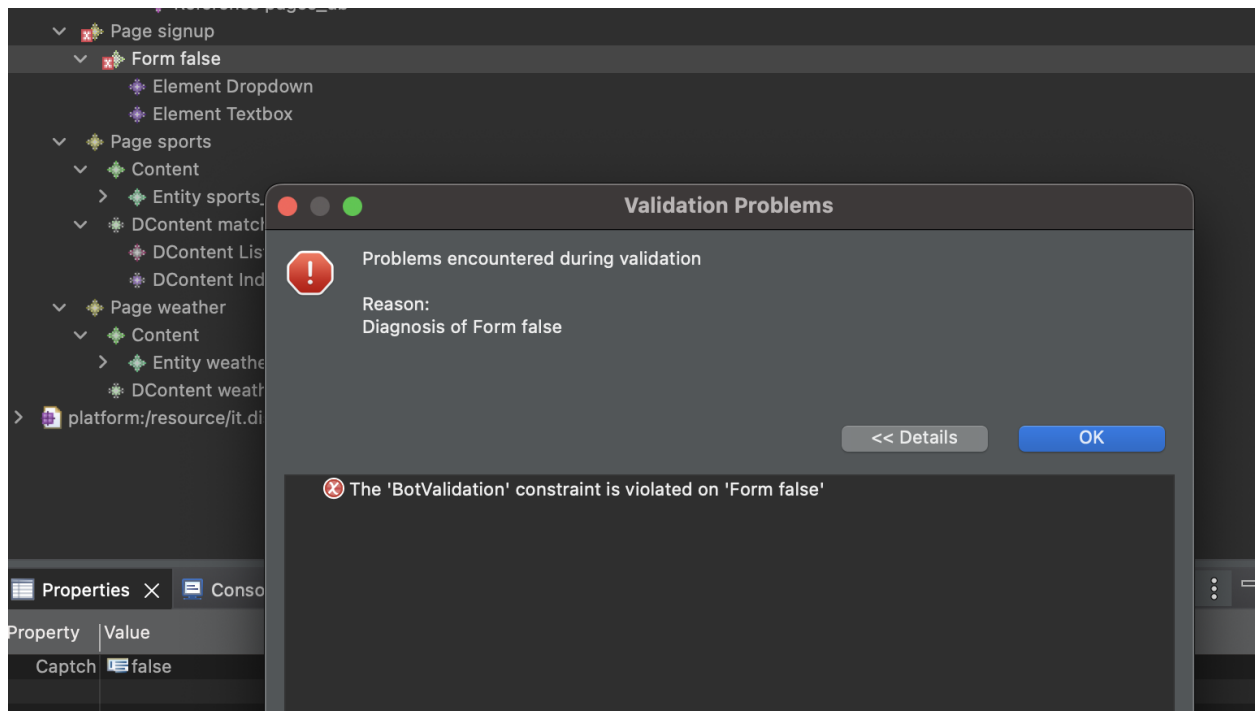
ListLength() on **DContent_list**:

The constraint checks the length of DContent and throws an error when it is not greater than 0. It is applied to the attribute length of the DContent_List. The length of the list can not be zero or less than zero.



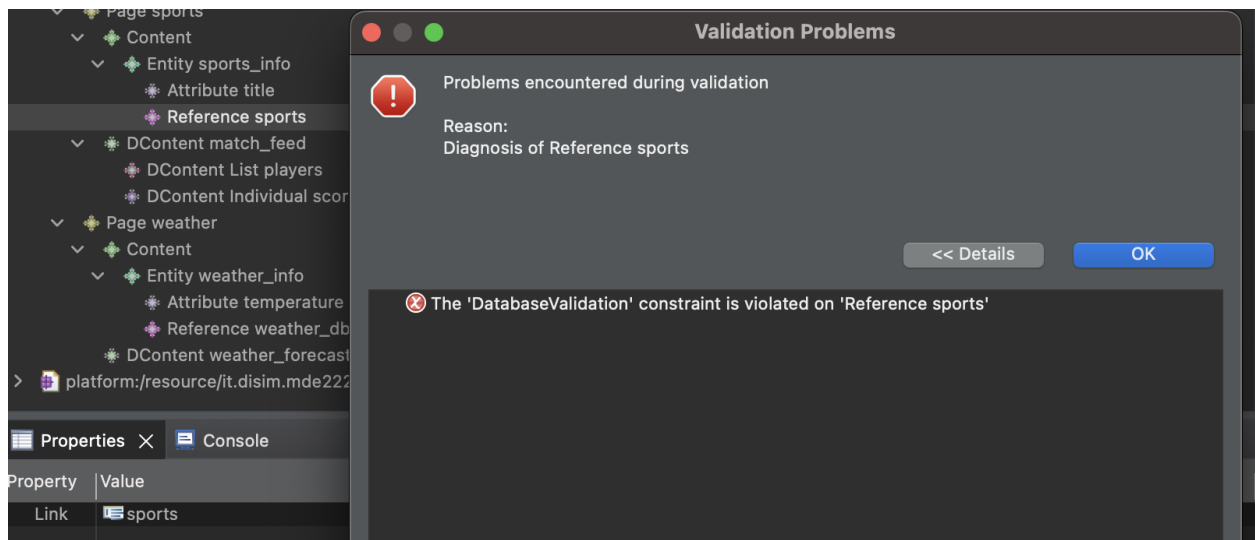
BotValidation() on Form:

The Form class has an attribute captcha. For submitting the form, the captcha should always be checked. The constraints make sure that the value of the captcha is always true, otherwise, it throws an error.



DatabaseValidation() on Reference:

In our metamodel, the Reference class is referring to the database. This constraint checks that reference strong should include 'db' at the end of the string. It is using .substring() builtin operation of OCL.



Operations:

Percentage() on DContent_Individual:

This operation is added to the attribute 'value' of the DContent_Individual class. It returns the percentage or rate depending on the value.



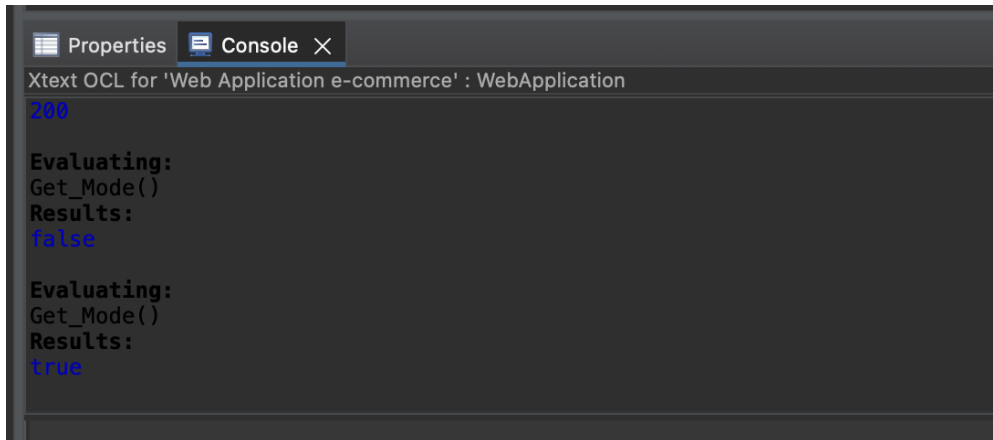
```
Properties Console X
Xtext OCL for 'Web Application news::Page sports::DContent match_feed::D
200

Evaluating:
Percentage()
Results:
200

Evaluating:
Percentage()
Results:
200
```

Get_Mode() on WebApplication():

The Get_mode() operation of this class returns the value of the attribute 'dark_mode'.



```
Properties Console X
Xtext OCL for 'Web Application e-commerce' : WebApplication
200

Evaluating:
Get_Mode()
Results:
false

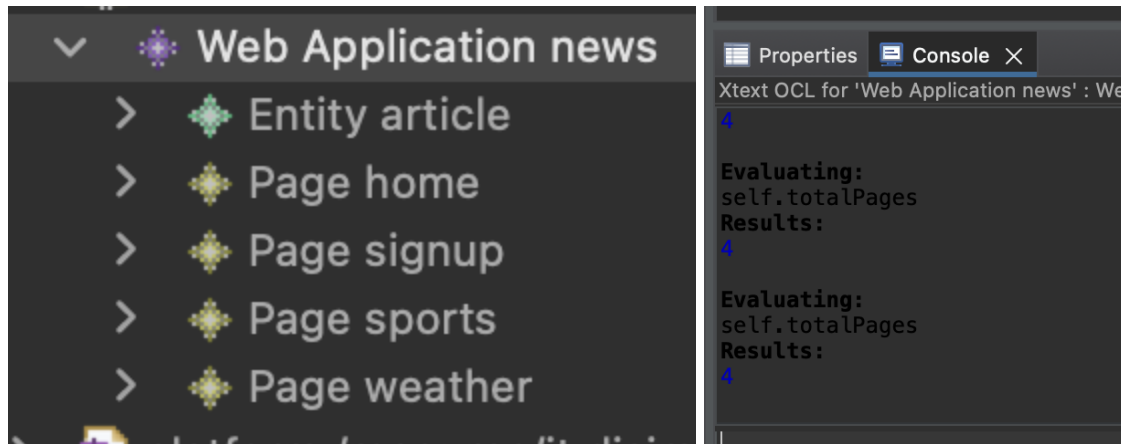
Evaluating:
Get_Mode()
Results:
true
```

Derived Fields:

We added two derived fields.

totalPages on WebApplication:

The 'totalPages' EAttribute of class WebApplication returns the total number of pages on the website.



totalEntites on WebApplication:

The 'totalEntites' Eattribute shows the number of contents.

