NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Software Design and Architecture (SE220)

Lab Instructor: Sandia Kumari

sandia.kumari@nu.edu.pk

Lab Session # 02

Objectives:

- **Introduction to User Stories**
- **Domain Modeling**
- **Steps to create domain model**
- **Scenario for modeling**
- **Association Relationship**
- **Generation of domain model from source code**
- **Exercise**

User Stories:

A user story is a software development tool. Its purpose is to generate understanding of a software feature from the end user's perspective.
A user story describes functionality that will be valuable to either a user or purchaser of a system or software. In one sentence, a user story will describe user type, what that user wants, and why.

How to format a user story:

User stories are usually written on index cards or even sticky notes, meaning that they are short and sweet. The general structure of a user story is

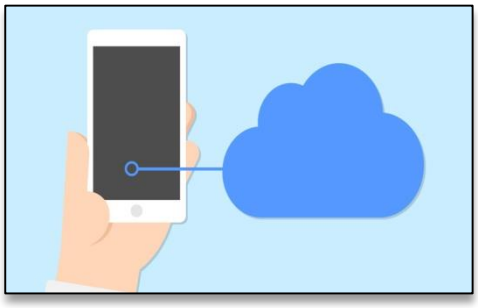
'As an [end user role],

I want [ability or feature of the product]

because [the benefit/function of the feature].'

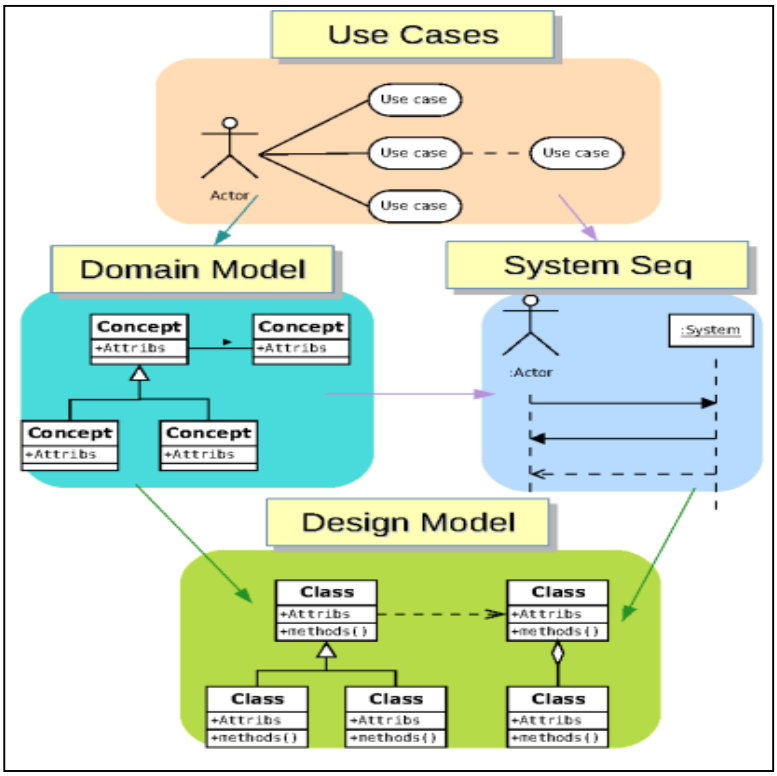
Example:

As a **mobile app user,**
I want to **save all my data to the cloud**
So I can **access from another device**.



User Story Example

Design Road:



- Use case is the start point of the design
- Domain model describes domain (context) of the model
- Domain model is followed by System Sequence diagram
- After that OO model is created

Domain Model:

- The domain model is created during object-oriented analysis to decompose the domain into concepts or objects in the real world.
- Domain Model illustrates meaningful conceptual classes in a problem

domain. It is a representation of real-world concepts, not software components.

- It is NOT a set of diagrams describing software classes, or software objects and their responsibilities.
- It is the basis for the design of the software
- In UML, the Domain Model is illustrated with a set of class diagrams without methods
- Why: Domain modeling helps us to identify the relevant concepts and ideas of a domain
- When: Domain modeling is done during object-oriented analysis.

Domain Model covers:

- The conceptual or domain classes
- Attributes of the conceptual classes
- Associations between the conceptual classes

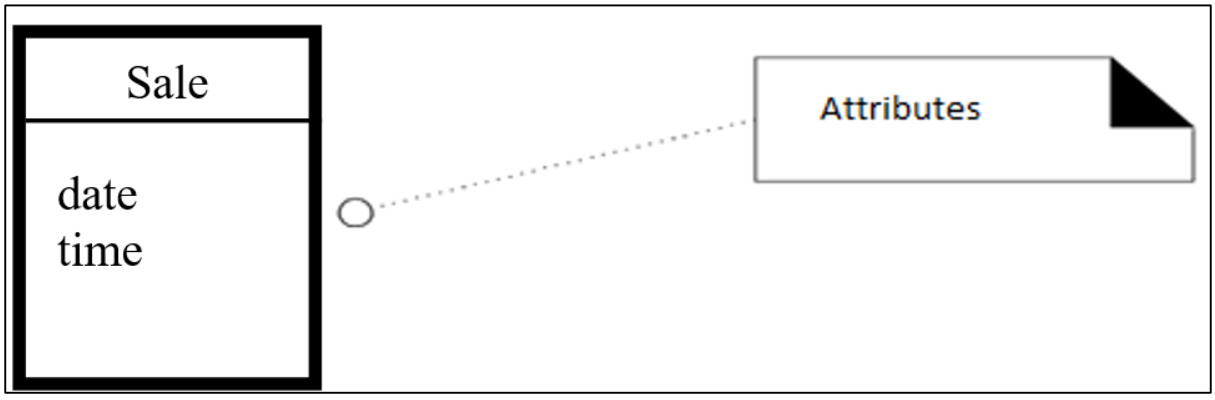
A conceptual model captures the important concepts and relationships in some domain. Concepts are represented by classes, while relationships are represented by associations.

Conceptual or Domain Class:

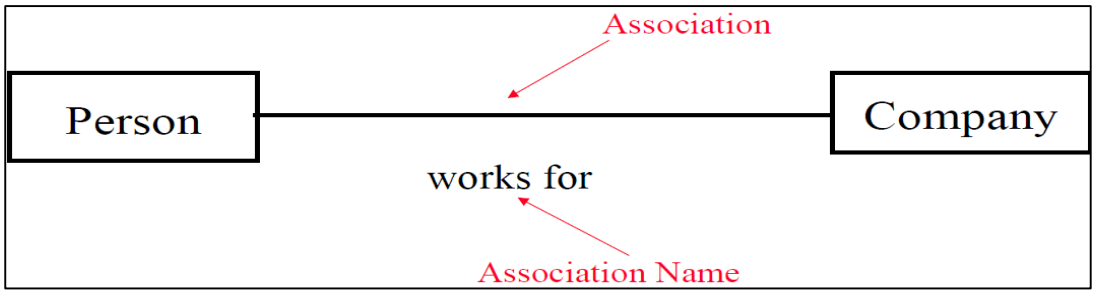
- Each domain class denotes a type of object
- Consider a use case description

Attributes:

Attributes refer to property that define the class.
For example: A class sale will have attributes date and time



Association Relationship:
A link between two classes
For example: A person works for a company



STEPS TO CREATE A DOMAIN MODEL

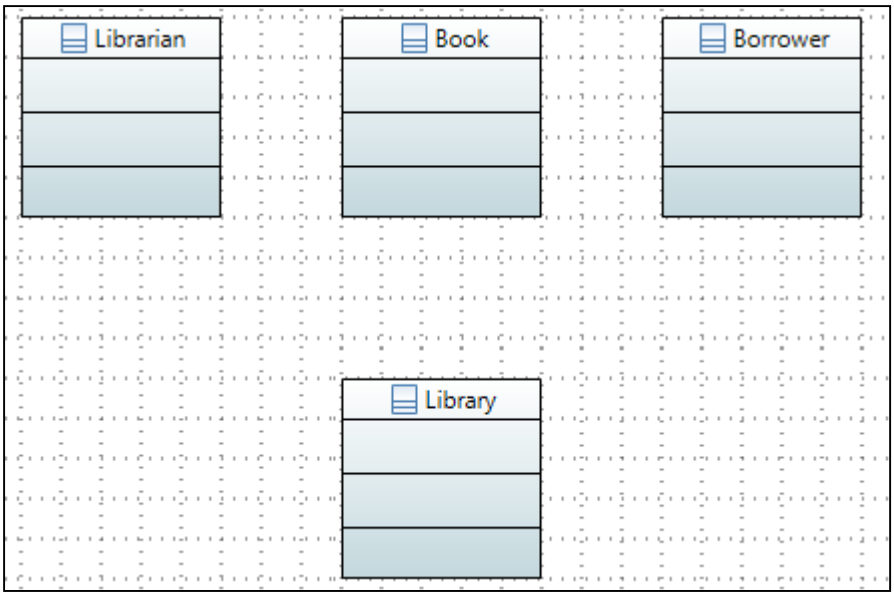
1. Create User Stories
2. Identify candidate conceptual classes
3. Draw them in a UML domain model
4. Add associations necessary to record the relationships that must be retained
5. Add attributes necessary for information to be preserved
6. Use existing names for things, the vocabulary of the domain

Example (Library Scenario)

- A library has one or many librarians
- A borrower borrows zero or many books from the library
- No or many books stock by zero or many librarians
- Zero or many borrowers visit no or more times a library

Identifying Conceptual Classes

The conceptual classes of the above scenario may be the following



Adding Associations

The association relationship is further described by the multiplicity concept.

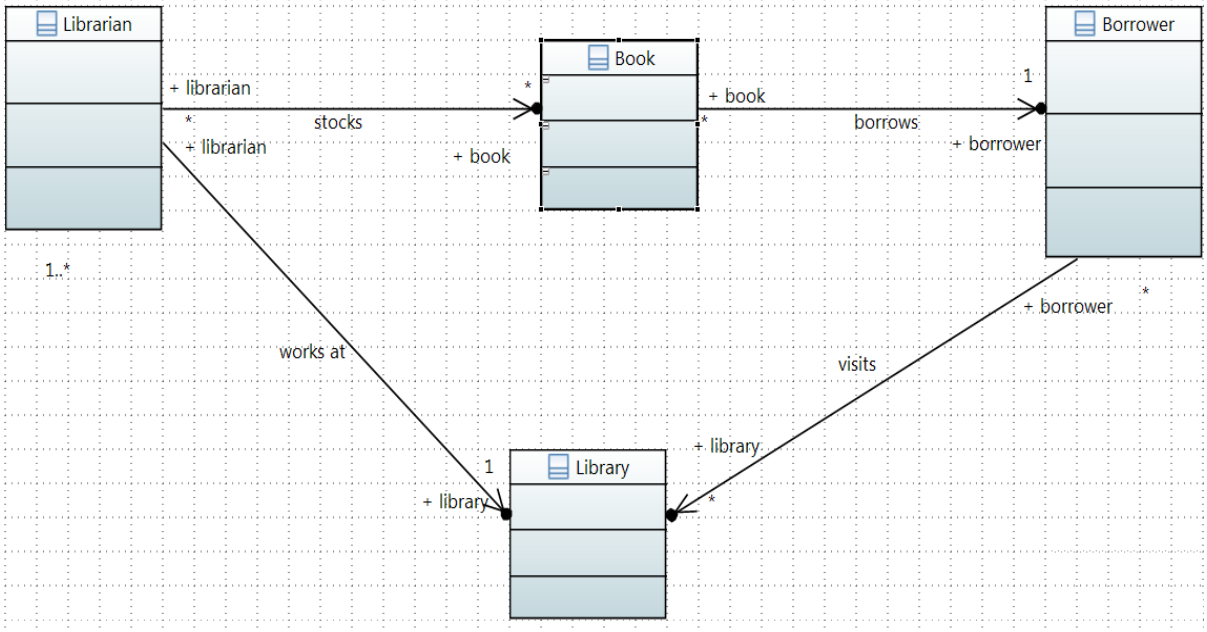
Relationship	Indicator
Exactly one	1
Zero or more (unlimited)	0…*
One or more	1…*
Zero or one only	0..1
Specified range	2..4
Multiple, disjoint ranges	2, 4..6, 8

VISIBILITY

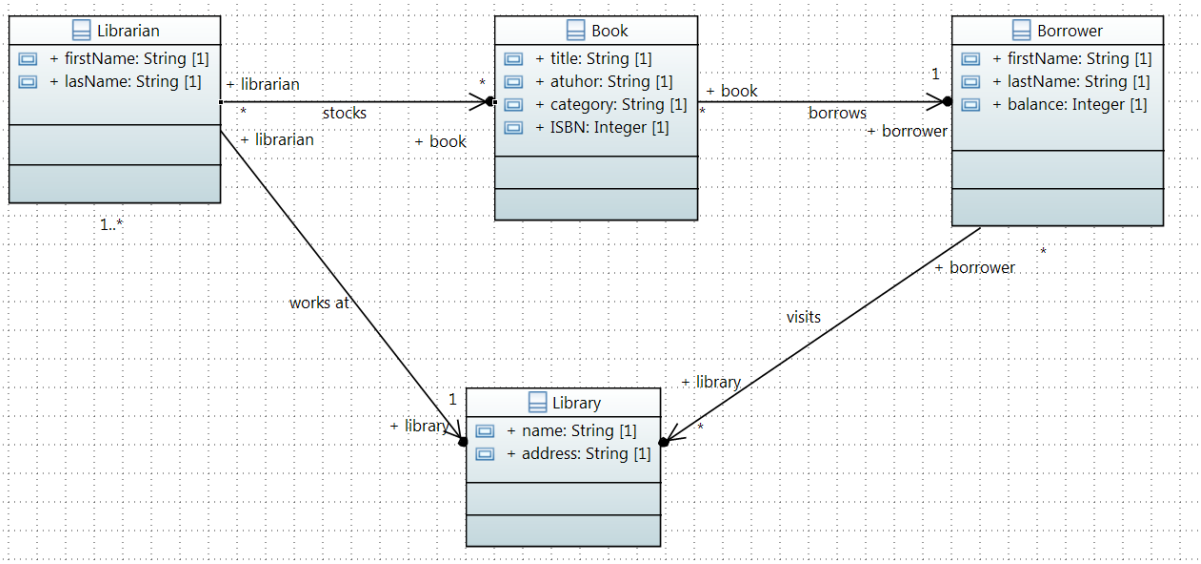
Use visibility markers to signify who can access the information contained within a class.

- Public +
- Private -
- Protected#

Following the above scenario, association with multiplicity is added



Adding Attributes



ASSOCIATION RELATIONSHIP

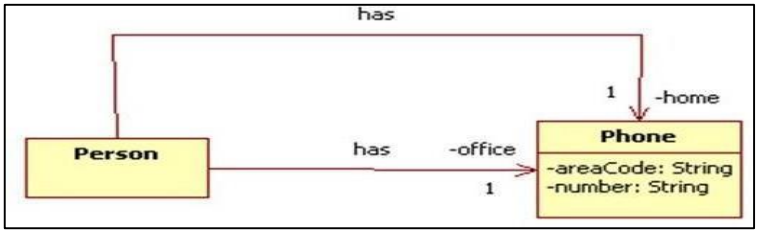
An association can be defined as a relationship between classes. An association relation is established, when two classes are connected to each other in any way. A class can be associated to itself too.

Examples of Associations:

1. **Association with 1 multiplicity:**

Assume some application needs to associate a home and business phone to each person in a database.

Example: A person has two phone numbers, one for office and one for home, both are private. This relationship can be represented by following association and code:

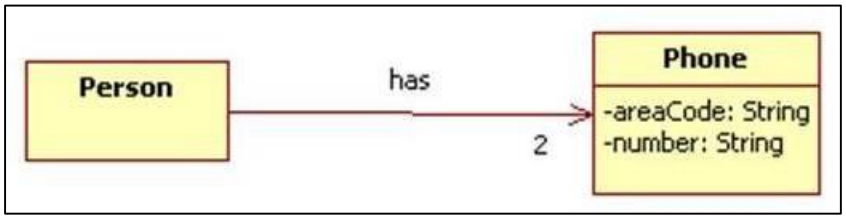


Code:

```
class Person {
private Phone home;
private Phone office;
// etc.
}
```

1. **Association with more than 1 multiplicity:**

Taking the example described above, we can also model the scenario as follows:

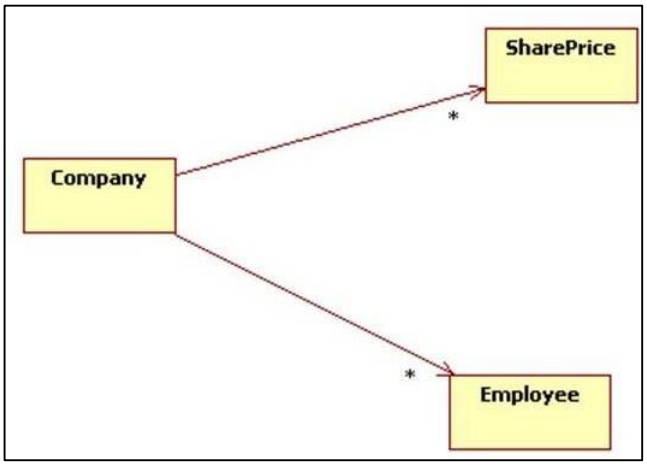


This can be represented in code as follows:

```
class Person {
      private Phone[] phones = new
      Phone[2];
      // etc.
}
```

1. **Association with many (*) multiplicity:**

Take the example that a company can have many employees and many share prices.



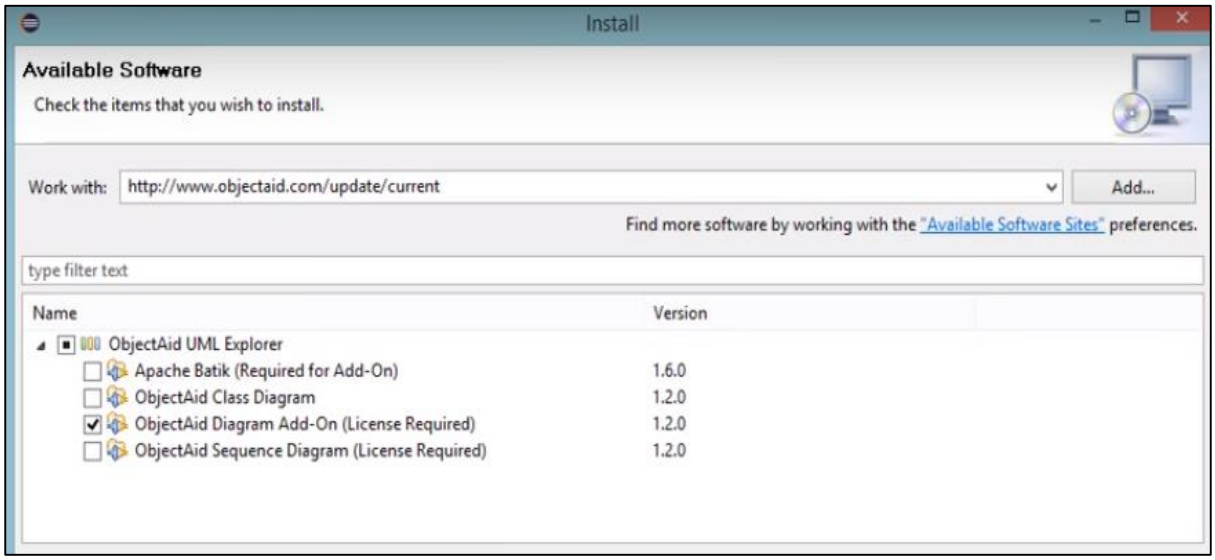
Code:

```
class Company {
private List<Employee> employees;
private List<SharePrice> sharePrices;
// etc.
```

Domain Model from the Java code using ObjectAid UML Explorer

Install ObjectAid plugin using the following link:

Name: ObjectAid UML Explorer
URL: http://www.objectaid.com/update/current



After Installation, create a java project with the following classes.

Student.java
Course.java
Main.java

A student can take 0 or many courses

Course.java

```
public class Course {

	private String name="";

	public Course() {
		}
public Course (String name) {
	this.name= name;
	}
public String getName() {
	return name;

}
public void setName(String
name) {
this.name= name;

}
}
```

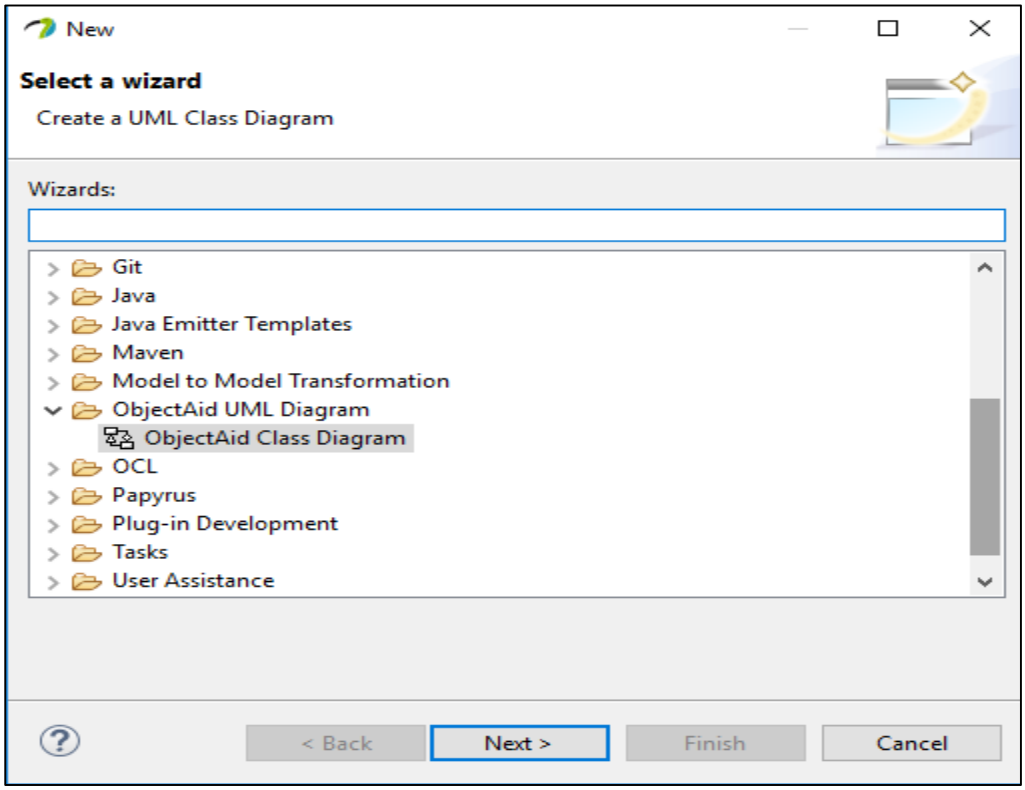
Student.java

```
public class Student {
private Course[] courses = new Course[6];
private int numberOfCourses= 0;
private String name = "";


public String getName() {
	return name;
}
public void setName(String name){
	this.name= name;

}
public Course [] getCourses() {
	return courses;

}
public void setCourses(Course course) {
	courses[numberOfCourses]= course;
	numberOfCourses++;}

public int getNumberOfCourses() {
	return numberOfCourses;
}
	public void setNumberOfCourses() {

	this.numberOfCourses=numberOfCourses;
	}
}
```

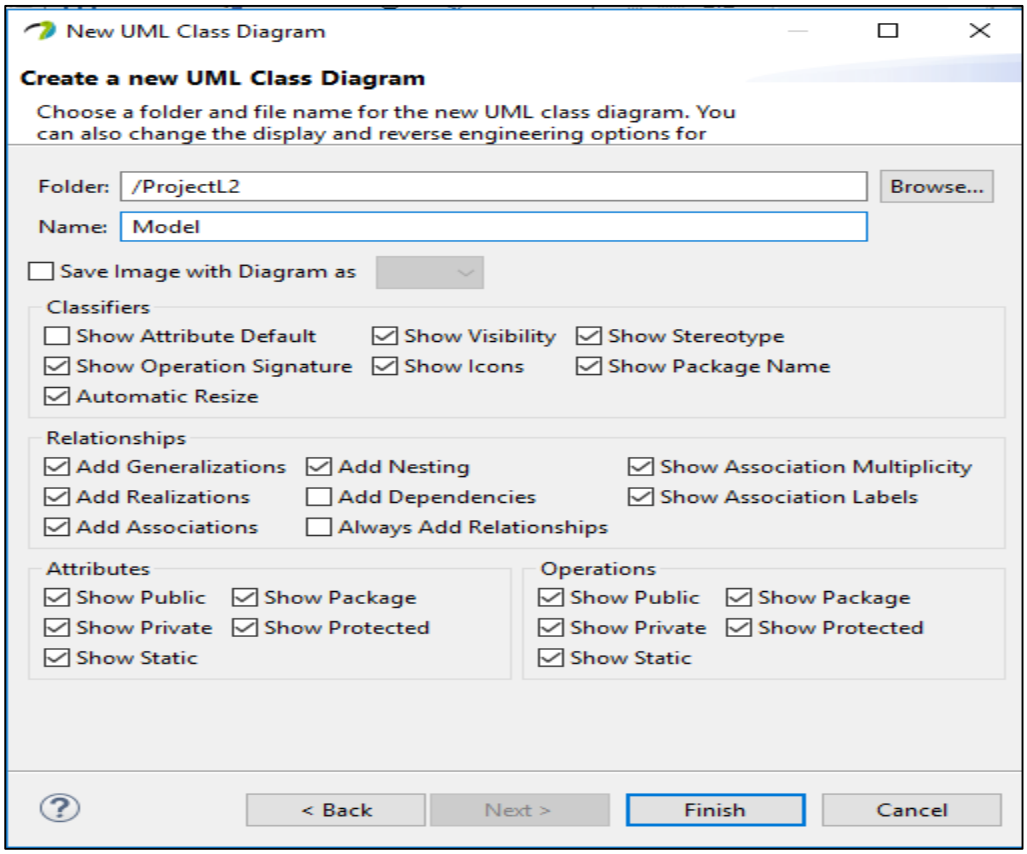
Main.java

```
public class Main {

	public static void main(String[]
args) {


Student john = new Student();
john.setName("john");
Course java = new Course ("Java");

Course c = new Course ("CLanguage");
john.setCourses(java);
john.setCourses(c);
System.out.println("Number of
courses"+john.getNumberOfCourses());
	}
}
```

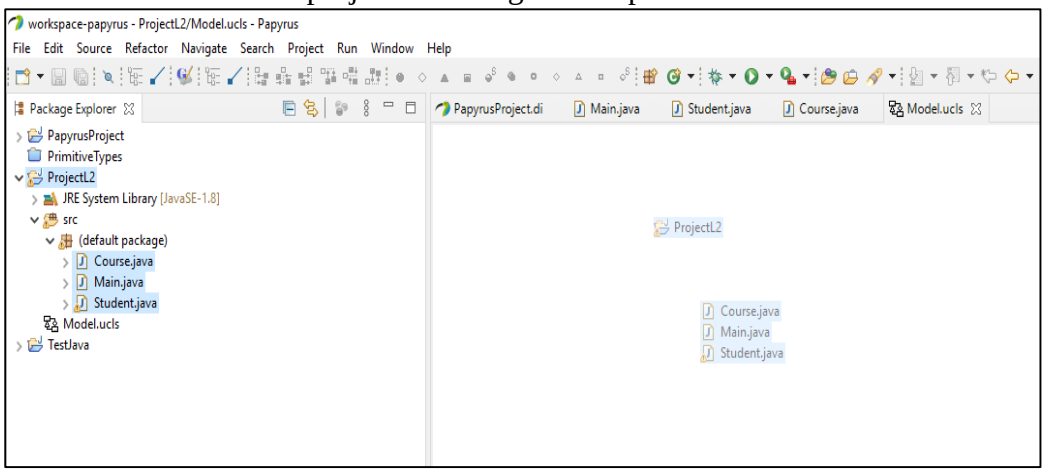
Right click on project>New>Other>ObjectAid Class Diagram



Select ObjectAid Class Diagram



Select all the classes of project then drag and drop



Domain model will be created from the java code