**Practical: Model Driven Engineering - Exercise 1**

# Introduction

The full name of UML is Unified Modelling Language. UML is usually used in the "requirement analysis" and "design" phases of software development, that is, before the "coding" phase. Its main function for product managers is to better sort out logic in demand analysis, and at the same time improve communication efficiency. UML mainly includes the thirteen types in the chart as shown in the table below (Table 1). Usually, the business concepts and other static structures are systematically sorted out and refined. We call it structural modelling. In order to systematically sort and refine business processes and other dynamic content, we call it behaviour modelling.

The core purpose of requirements analysis is to solve the problem of whether the software is useful. Software design is to solve the problem of how much software costs. Therefore, the primary task of requirements analysis is to ensure the value of software. As software systems become more and more complex, it is required that we must implement model planning from requirements to detailed design. This is a very important part. Subsequent coding, testing, and maintenance are only carried out according to the previous planning plan. A very detailed design of UML can almost express the entire system architecture and code architecture, which not only simplifies the development difficulty of complex systems, but also regulates the software development process of the system, making the system more controllable and reliable.

|  |  |  |
| --- | --- | --- |
| **Class** | **UML Diagram** | **Effect** |
| Structural | Class diagram | Used to conceptual model analysis. |
| Component diagram | Used to analyze the demands of IT infrastructure, software architecture, etc. Requires experience in infrastructure or software design. |
| Composite structure diagram |
| Deployment diagram |
| Object diagram | Developers will use |
| Package diagram | Used to organize class diagrams, etc., is rarely used in practice. |
| Behavioral | Activity diagram | Used to analyze the process |
| State diagram |
| Sequence diagram |
| Communication diagram | Another representation of sequence diagram is not commonly used. |
| Timing diagram | Represents the state of something changing with time, basically has not been used. |
| Interaction overview diagram | Express most of the demands of software systems |
| Use case diagram |

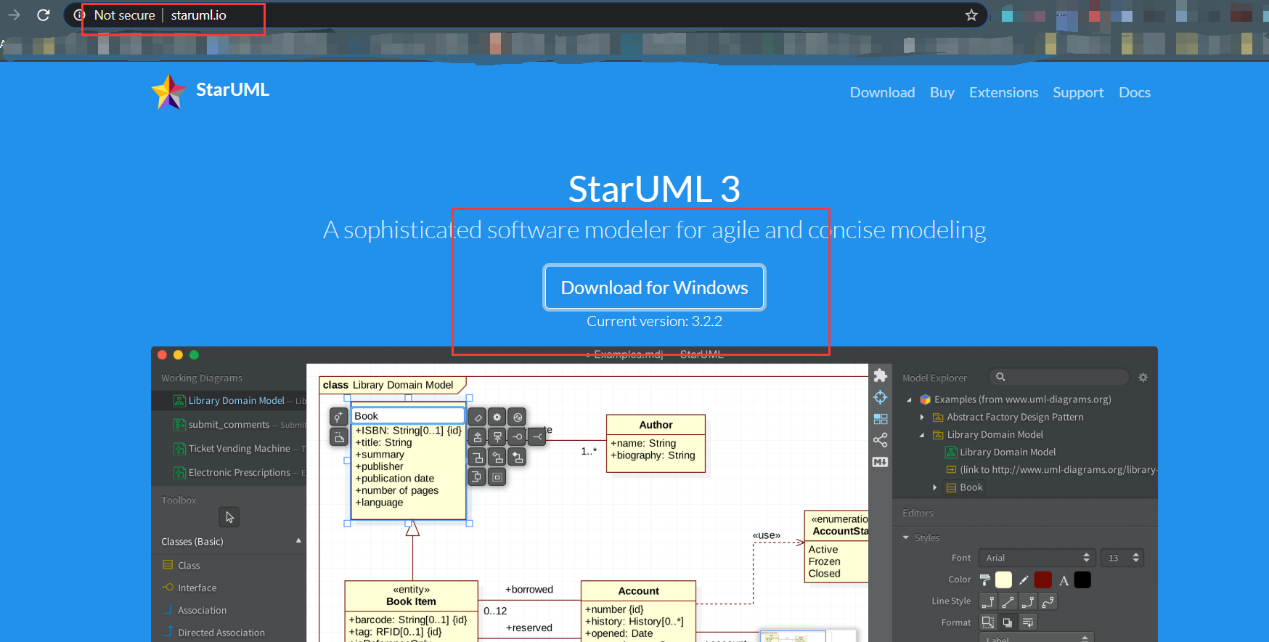
Table 1 Category of UML Diagrams

On the basis of understanding the above information, UML is a well-defined, easy to express, powerful and universally applicable modelling language; UML is not limited to supporting object-oriented analysis and design, but also supports the entire process of software development from requirements analysis.

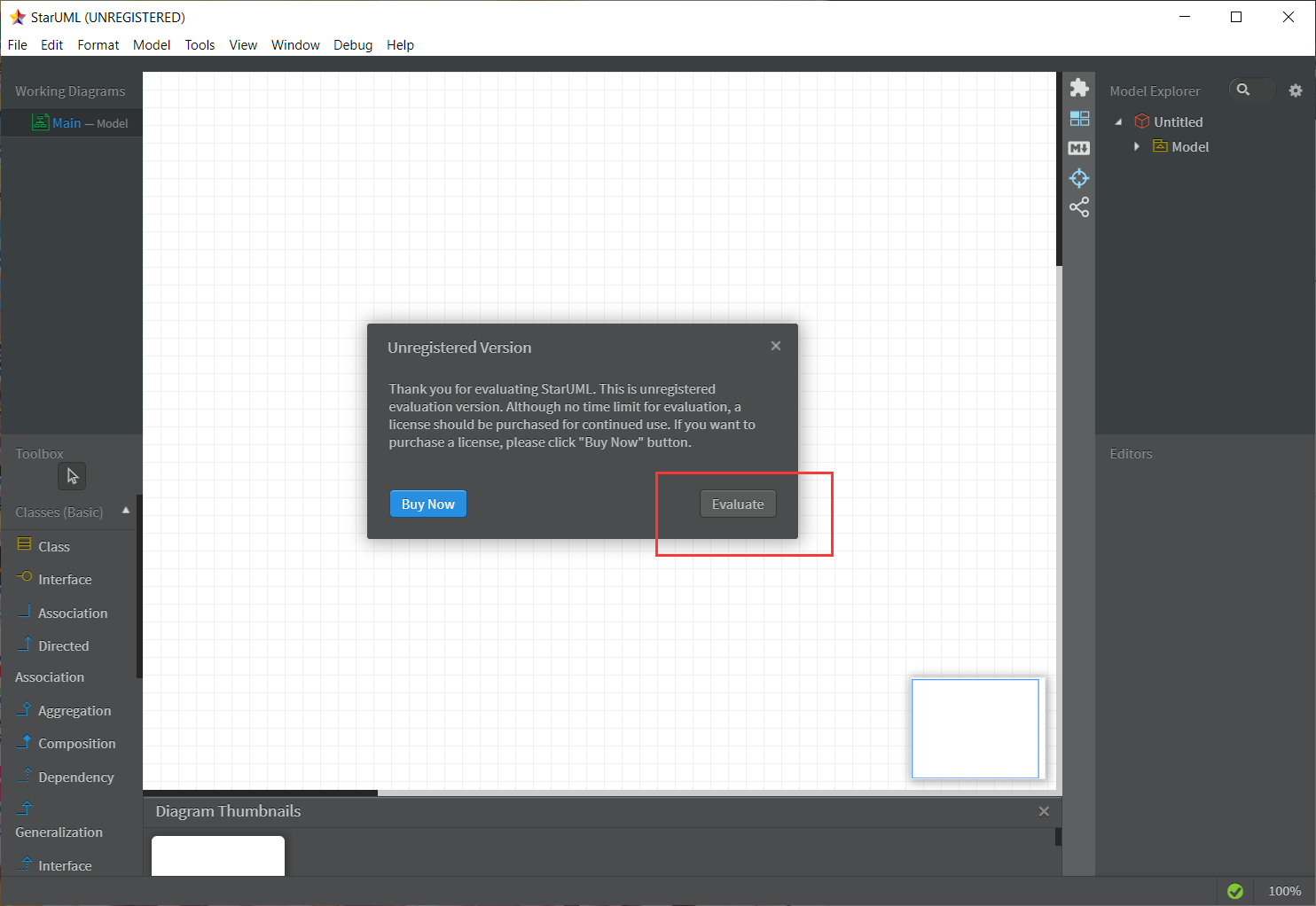
Here we will introduce how to make a class diagram with StarUML, and how to automatically build the relevant code through the code generator.

# StarUML installation

* First, visit and download the latest version of StarUML from its website: <http://staruml.io/>



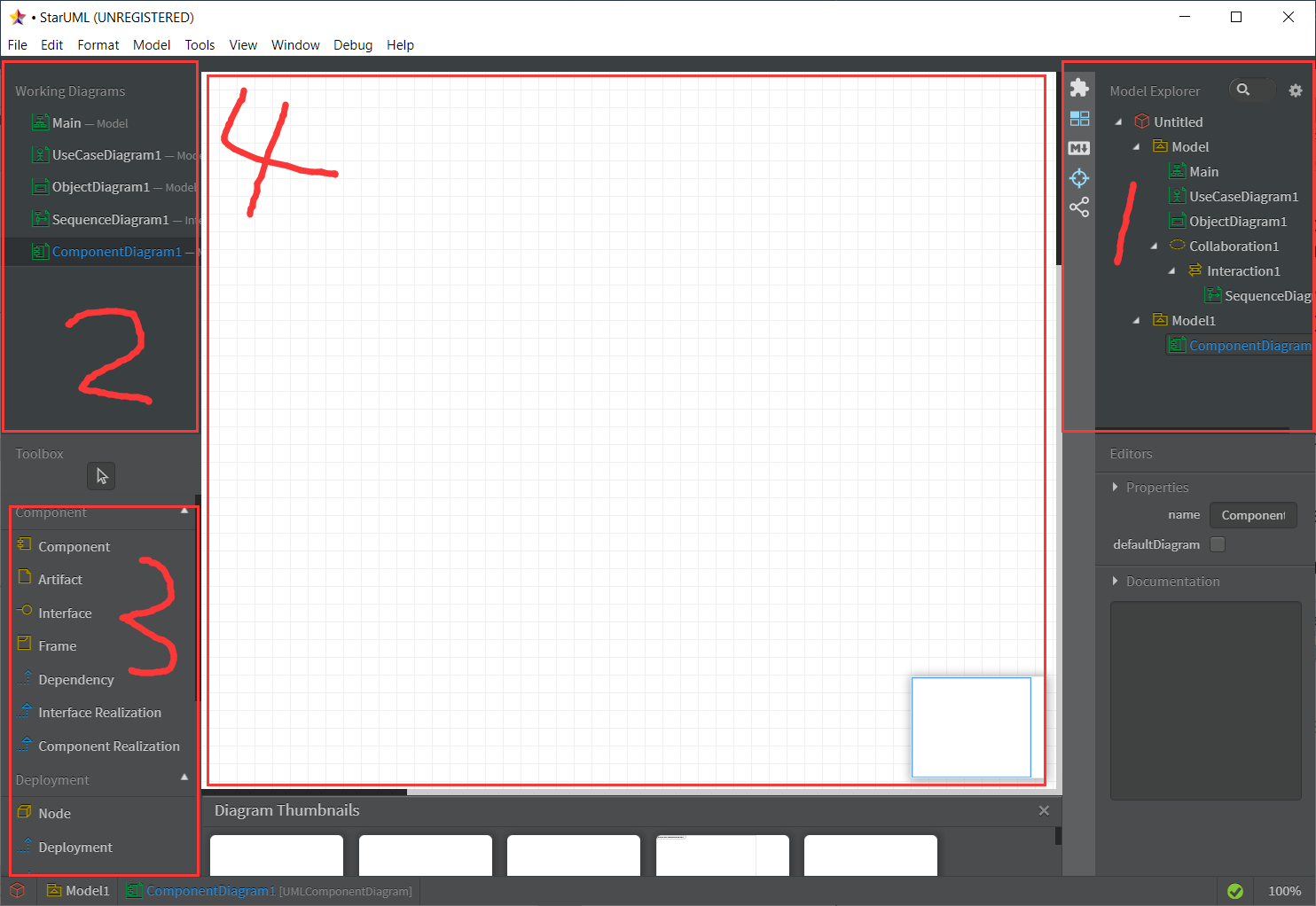
* Keep pressing next until the software is installed.
* StarUML is a free open source software, so we can directly use the unregistered version, which does not affect the use.



# 3. Designing a Class Diagram in StarUML.

## 3.1 Start drawing a class diagram

In StarUML Fill🡪New.



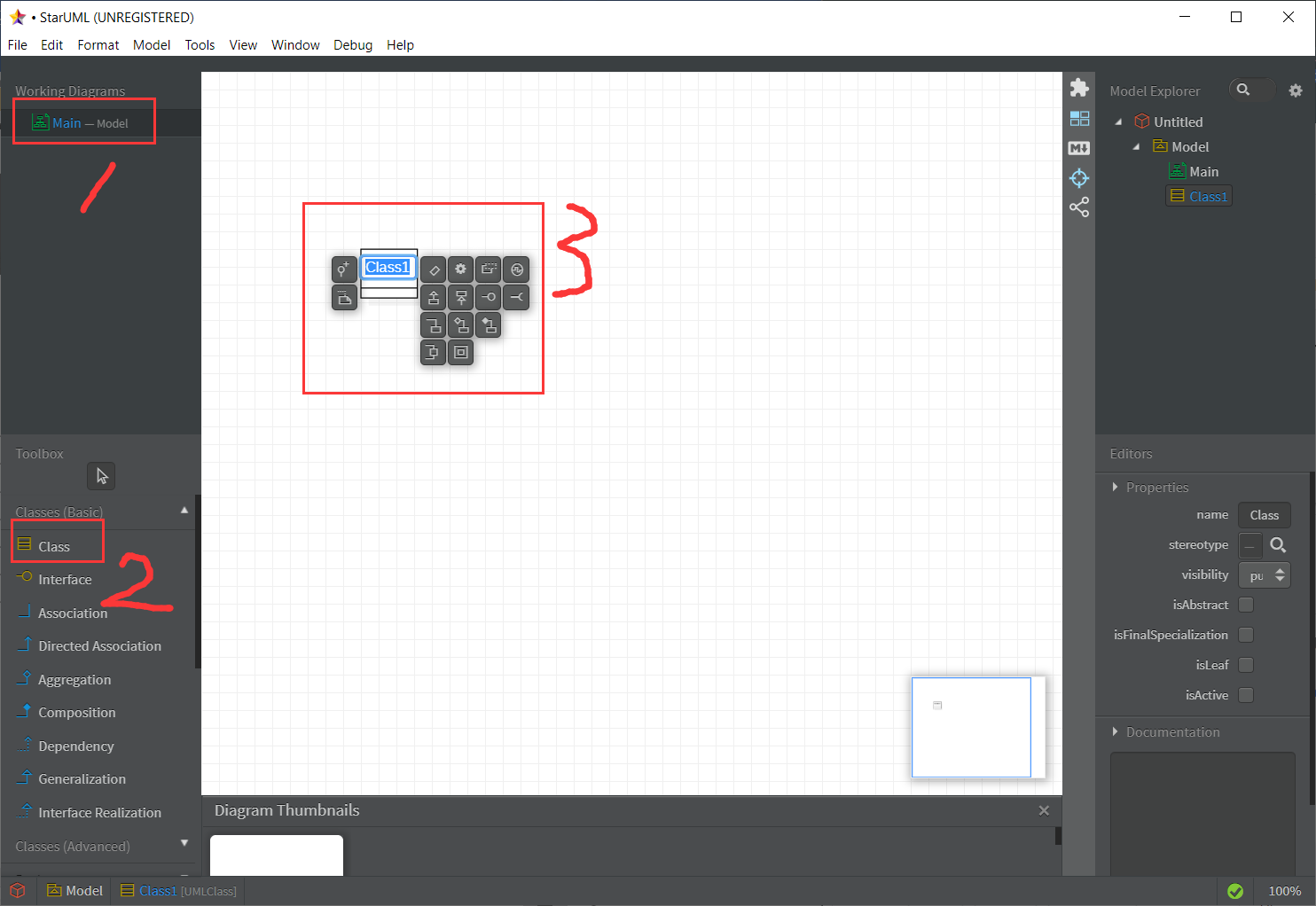
Area 1 -- List of entire projects (Right click to add various diagrams)

Area 2 -- List of diagrams

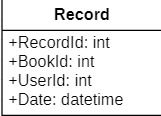
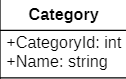
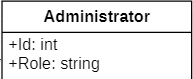
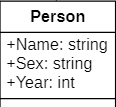
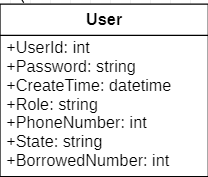
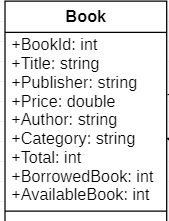
Area 3 -- The various components that make up the diagram

Area 4 -- Main work area

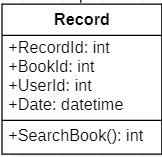
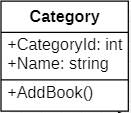
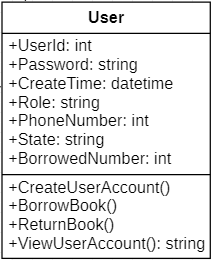
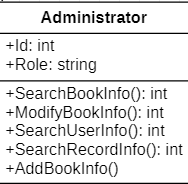
## 3.2. Design the Class Diagram



* Click Class diagram to enter the class diagram editing interface.
* Click to select ‘Class’
* Click on the workspace to create six classes and named:

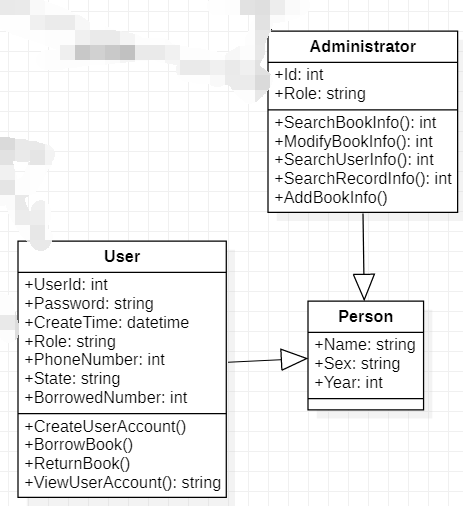


* Double-click Category, Administrator, User and Record class to add operation.

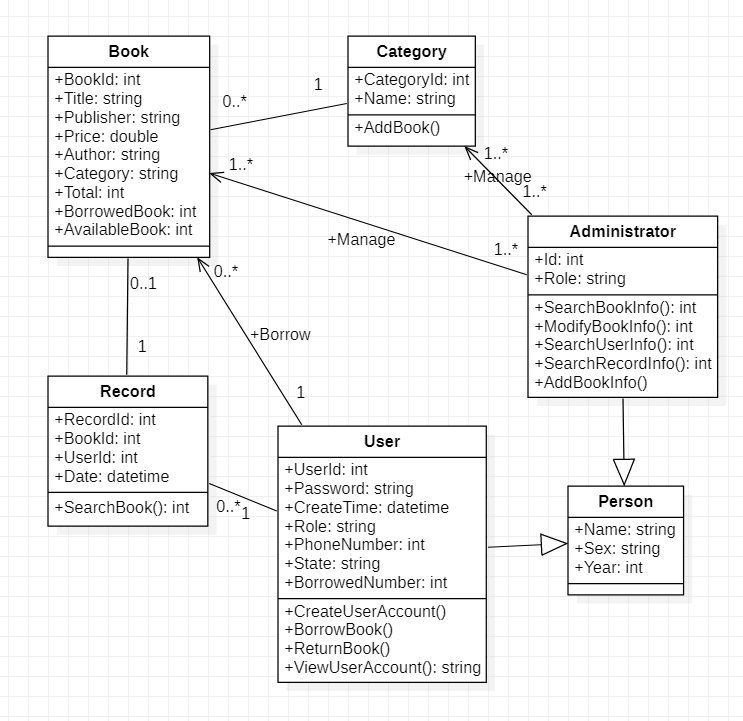


* Administrator and User Class are Subclass of Person Class





* There are many different relationships between classes and classes. Although this will not be reflected in the code of class, it still needs to be reflected in the UML diagram.

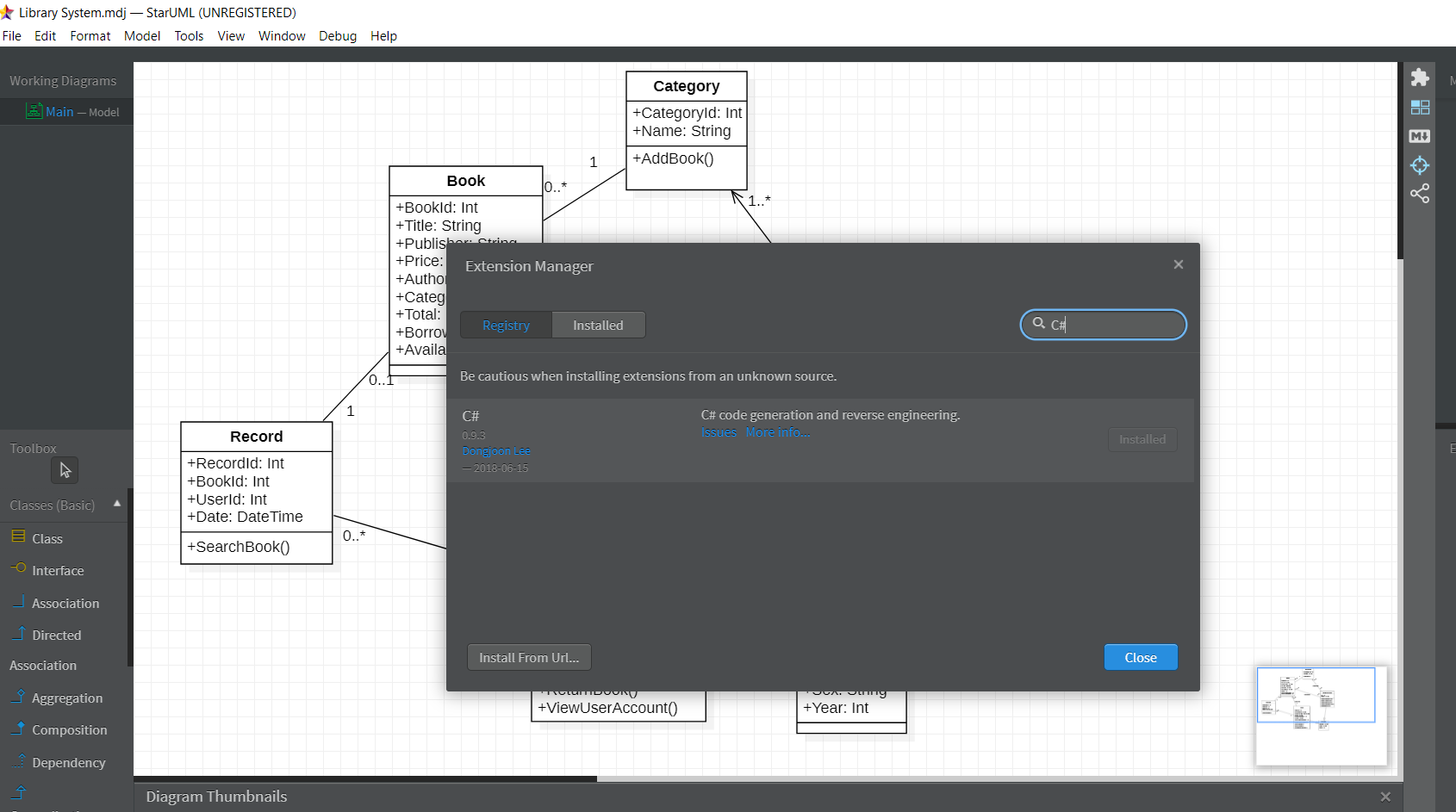


**3. Generating C# Code from the Class Diagram**

Now that the class diagram has been designed, we can begin to prepare it to automatically generate relevant code parts.

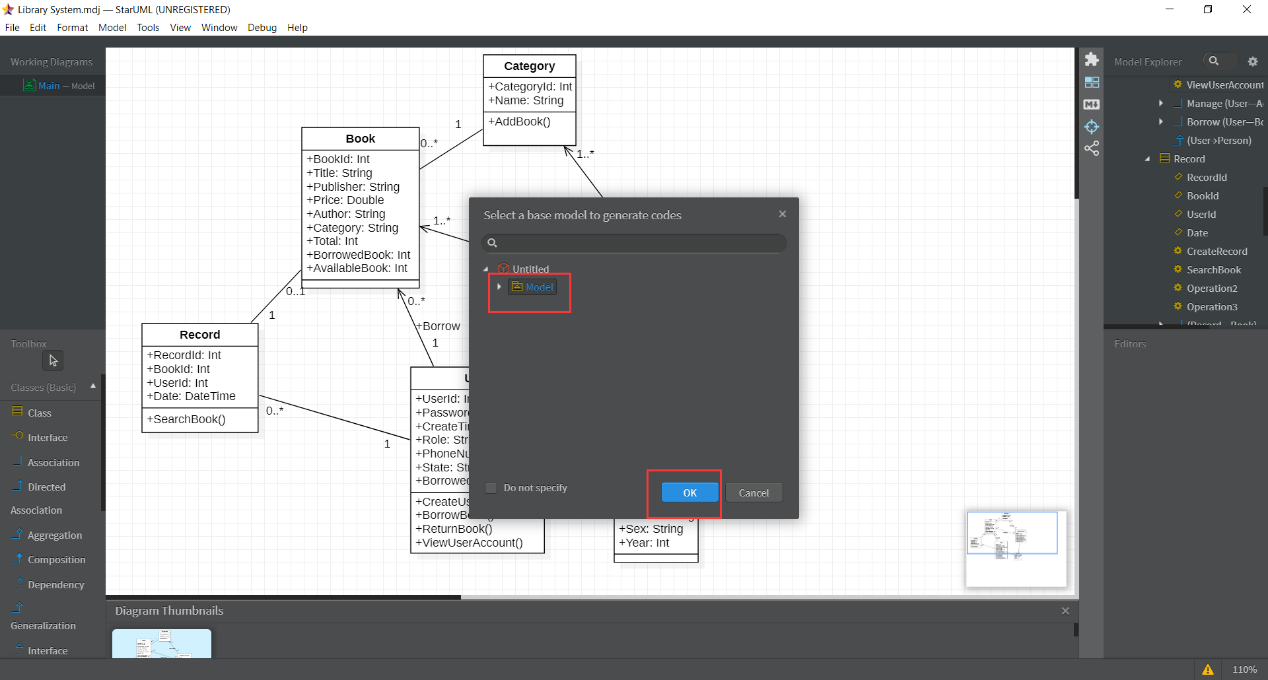
## 3.1 Install the code generator component

* Click Tools>Extension Manager
* Input C# into the search textbox and select the C# generator and install it.

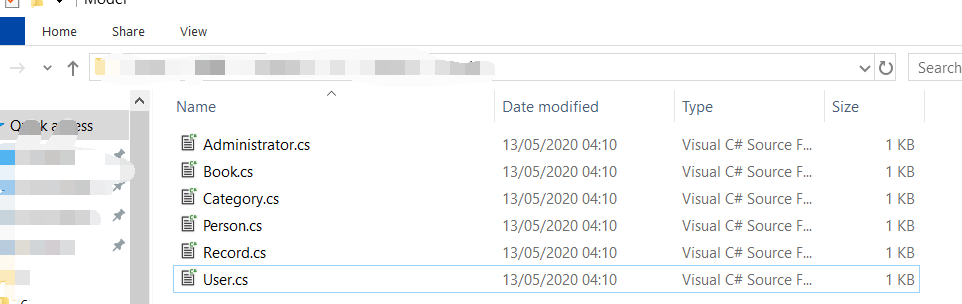


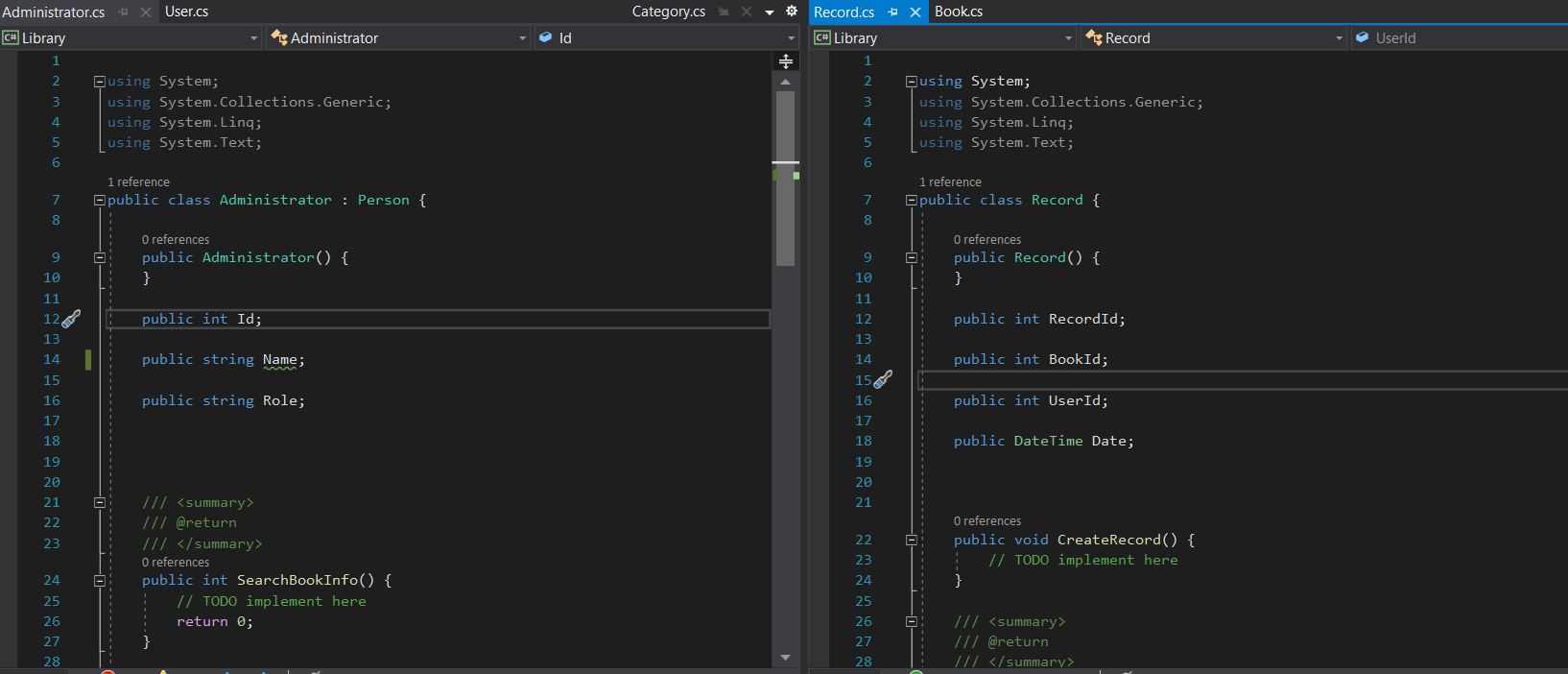
## 3.2 Generate C# code

## Restart StarUML to refresh the whole tool. Click Tools-->C#-->Generate Code



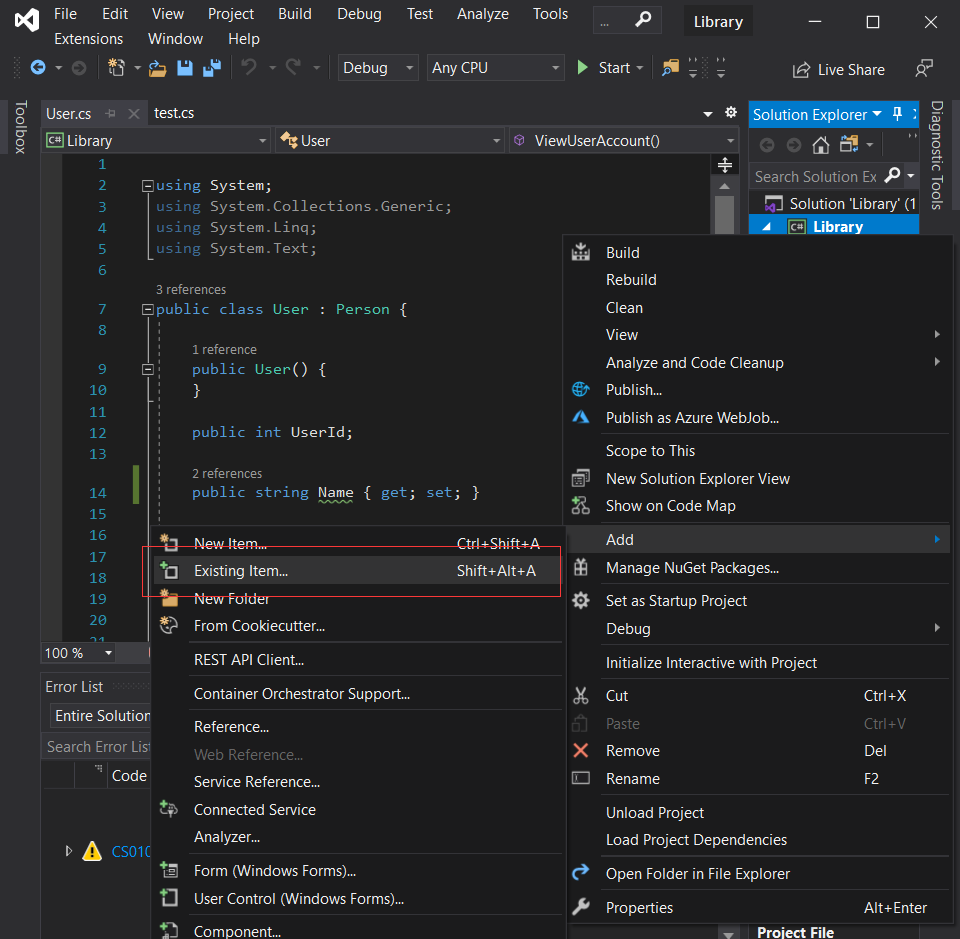
Then we can find the related class files that have been generated in your save path.



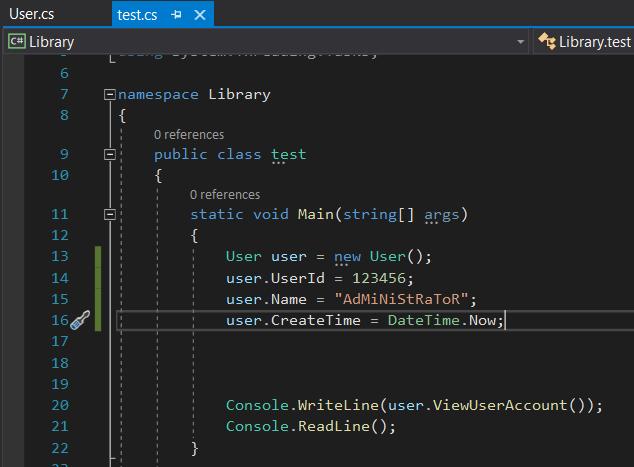


# 4. Testing the Model Classes

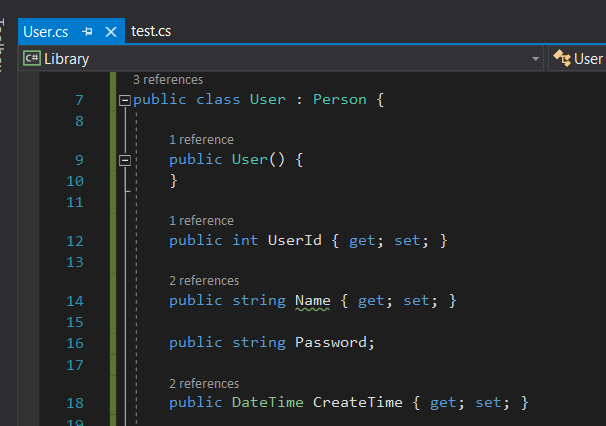
* Create a Console app in VS 2019 Called Library;
* Add those classes into this project;

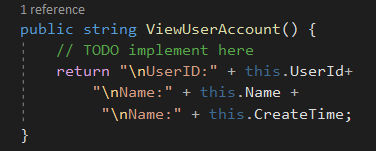


* Create a new class called test.cs;

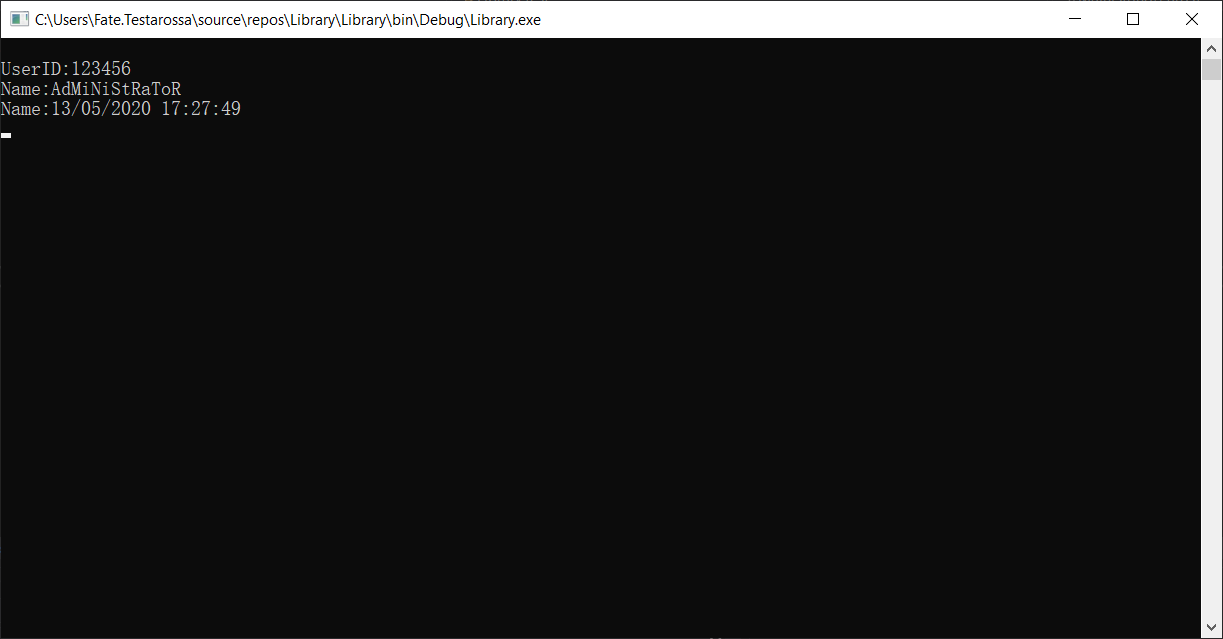


* Add some code in User.cs





* Run the project to check if the code can be used as usual.



You have successfully driven your model (class diagrams) to C# classes, which can be used in creating a console application.

# Further Work

StarUML can integrate with other code generators which create code in other programming languages, such as Java, Python.

Visual Paradigm is another example of MDSE tool, see <https://www.visual-paradigm.com/>. It is a [UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language) [CASE](https://en.wikipedia.org/wiki/Computer-aided_software_engineering) Tool supporting UML 2, [SysML](https://en.wikipedia.org/wiki/Systems_Modeling_Language" \o "Systems Modeling Language) and [Business Process Modeling Notation (BPMN)](https://en.wikipedia.org/wiki/Business_Process_Modeling_Notation) from the [Object Management Group (OMG)](https://en.wikipedia.org/wiki/Object_Management_Group). In addition to modeling support, it provides report generation and code engineering capabilities including [code generation](https://en.wikipedia.org/wiki/Automatic_programming). It can [reverse engineer](https://en.wikipedia.org/wiki/Reverse_engineer) diagrams from code, and provide [round-trip engineering](https://en.wikipedia.org/wiki/Round-trip_engineering) for various [programming languages](https://en.wikipedia.org/wiki/Programming_language).

Visual Paradigm has a free online version (Express level). It has a free Community version, which is free but the functions included are very limited. It also has a free version (fuller but still quite limited) for 30-days trial use. See <https://www.visual-paradigm.com/download/>.

Have a look to expand your knowledge if you have time.