স্থায়ী সনদ প্রাপ্ত বিশ্ববিদ্যালয়

City University

...creating a culture of excellence

Department of CSE

LAB REPORT 02

Course Code : CSE-325

Course Title : System Anlysis and Design

Experiment Name : Class Diagram of Online Shopping System

Submitted To : Supta Richard Philip

City University , Bangladesh

Submitted By

ID : 171442528

Name : MD.Amran Hossen

Program : CSE 44th Batch (Eve)

Semester : 7th Semester (Spring)

Date Of assigned : 28-05-2019

What is Class Diagram?

The Unified Modeling Language (UML) can help you model systems in various ways. One of the more popular types in UML is the class diagram. Popular among software engineers to document software architecture, class diagrams are a type of structure diagram because they describe what must be present in the system being modeled. No matter your level of familiarity with UML or class diagrams, our UML software is designed to be simple and easy to use.

Learn UML Faster, Better and Easier

Are you looking for a Free UML tool for learning UML faster, easier and quicker? Visual Paradigm Community Edition is a UML software that supports all UML diagram types. It is an international award-winning UML modeler, and yet it is easy-to-use,

Purpose of Class Diagrams

1. Shows static structure of classifiers in a system
2. Diagram provides basic notation for other structure diagrams prescribed by UML
3. Helpful for developers and other team members too
4. Business Analysts can use class diagrams to model systems from business perspective

A UML class diagram is made up of:

- A set of classes and
- A set of relationships between classes
-

What is a Class ?

A description of a group of objects all with similar roles in the system, which consists of:

Structural features (attributes) define what objects of the class "know"

- Represent the state of an object of the class
- Are descriptions of the structural or static features of a class

Behavioral features (operations) define what objects of the class "can do"

- Define the way in which objects may interact
- Operations are descriptions of behavioral or dynamic features of a class

Class Notation

A class notation consists of three parts:

1. **Class Name**
   - The name of the class appears in the first partition.
2. **Class Attributes**
   - Attributes are shown in the second partition.
   - The attribute type is shown after the colon.
   - Attributes map onto member variables (data members) in code.
3. **Class Operations** (Methods)
   - Operations are shown in the third partition. They are services the class provides.
   - The return type of a method is shown after the colon at the end of the method signature.
   - The return type of method parameters are shown after the colon following the parameter name.
   - Operations map onto class methods in code

Benefits of class diagrams

Class diagrams offer a number of benefits for any organization. Use UML class diagrams to:

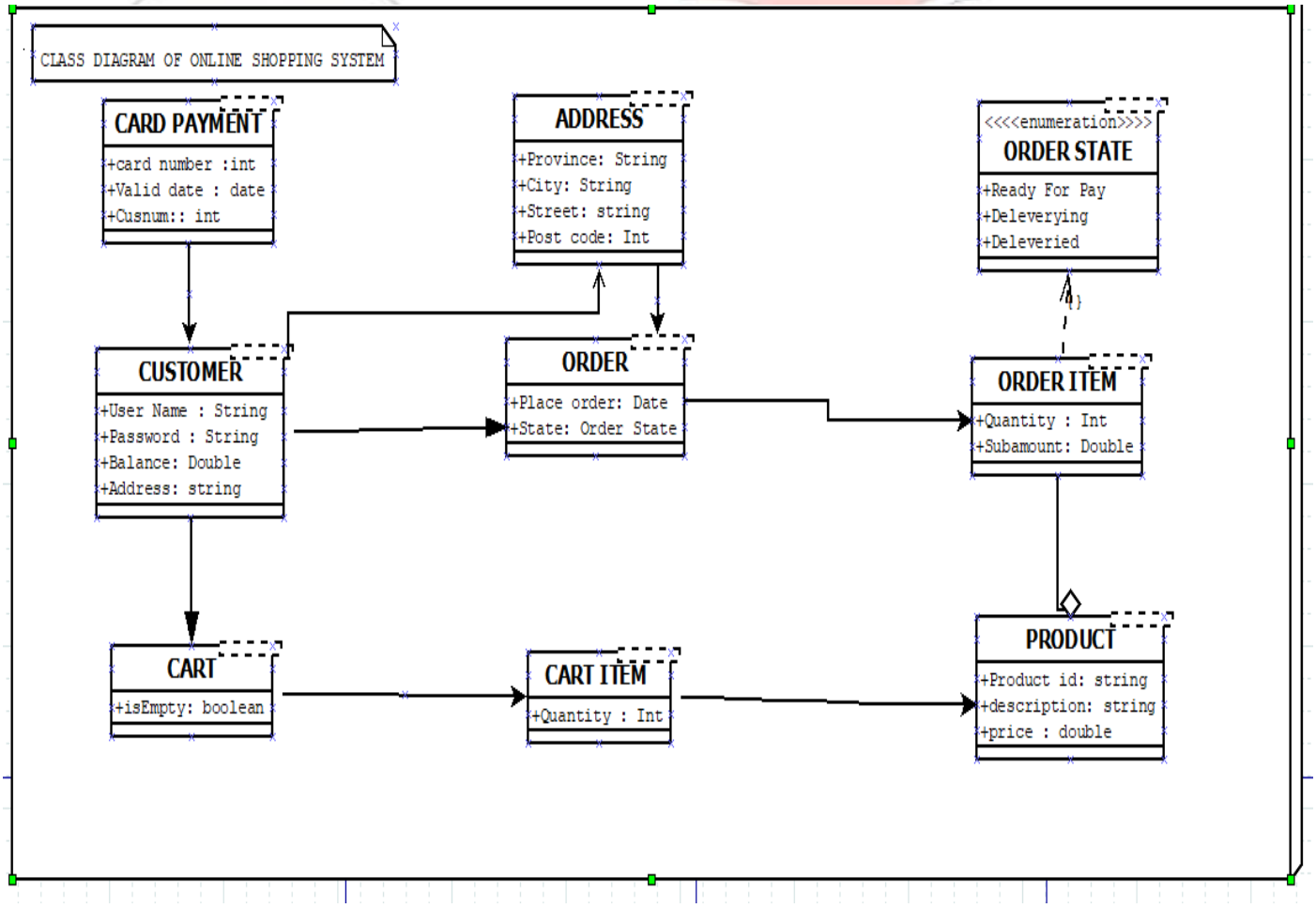
- Illustrate data models for information systems, no matter how simple or complex.
- Better understand the general overview of the schematics of an application.
- Visually express any specific needs of a system and disseminate that information throughout the business.
- Create detailed charts that highlight any specific code needed to be programmed and implemented to the described structure.
- Provide an implementation-independent description of types used in a system that are later passed between its components.

Basic components of a class diagram

The standard class diagram is composed of three sections:

- **Upper section:** Contains the name of the class. This section is always required, whether you are talking about the classifier or an object.
- **Middle section:** Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
- **Bottom section:** Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.
-

.JPG FILE

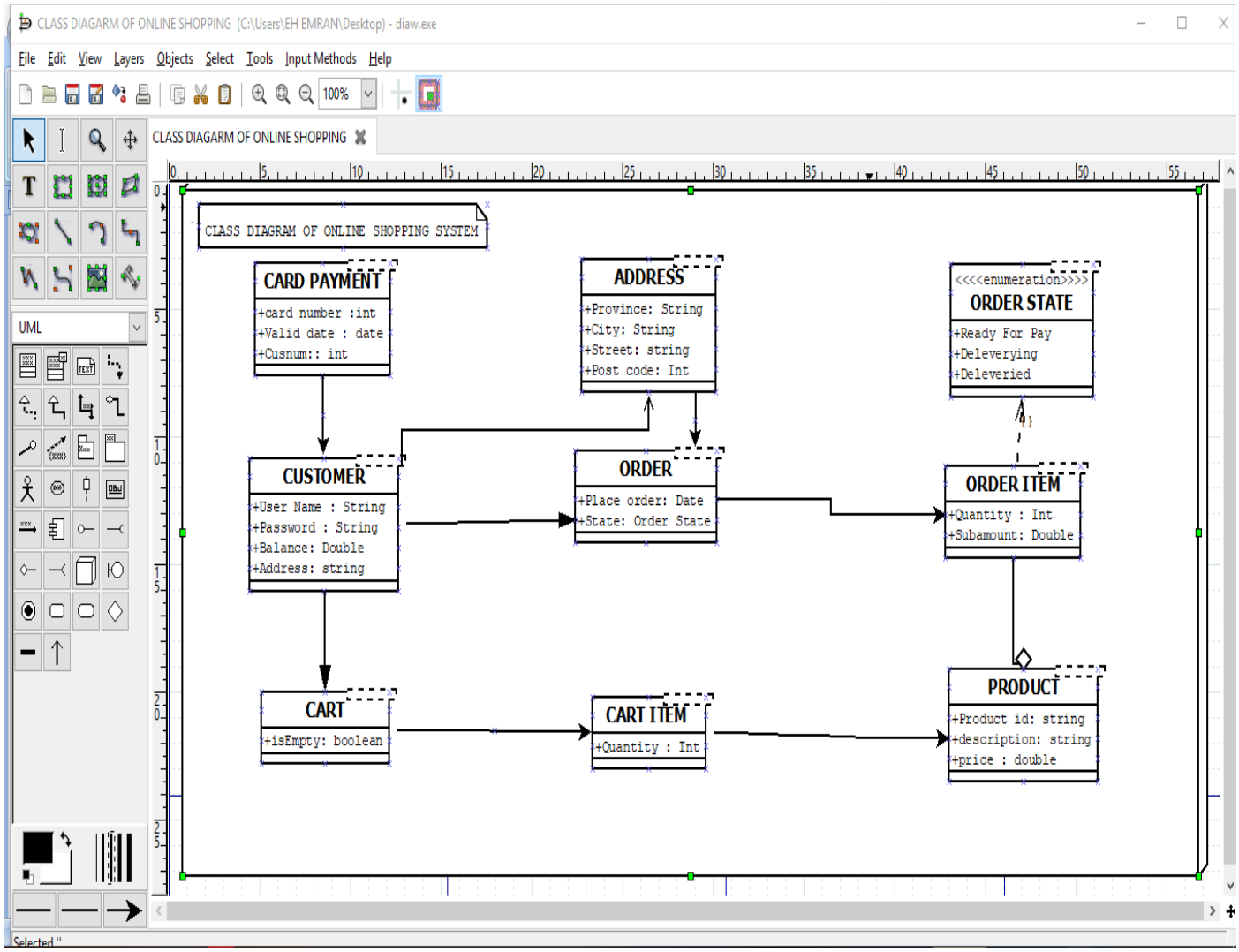


How to make a class diagram

In Lucidchart, creating a class diagram from scratch is surprisingly simple. Just follow these steps:

1. Open a blank document or start with a template.
2. Enable the UML shape library. On the left side of the Lucidchart editor, click "Shapes." Once you're in the Shape Library Manager, check "UML" and click "Save."
3. From the libraries you just added, select the shape you want and drag it from the toolbox to the canvas.
4. Model the process flow by drawing lines between shapes while adding text.

Working Screen Shot :



Additional class diagram components

Depending on the context, classes in a class diagram can represent the main objects, interactions in the application, or classes to be programmed. To answer the question "What is a class diagram in UML?" you should first understand its basic makeup.

- **Classes:** A template for creating objects and implementing behavior in a system. In UML, a class represents an object or a set of objects that share a common structure and behavior. They're represented by a rectangle that includes rows of the class name, its attributes, and its operations. When you draw a class in a class diagram, you're only required to fill out the top row—the others are optional if you'd like to provide more detail.
    - **Name:** The first row in a class shape.
    - **Attributes:** The second row in a class shape. Each attribute of the class is displayed on a separate line.
    - **Methods:** The third row in a class shape. Also known as operations, methods are displayed in list format with each operation on its own line.
- **Signals**: Symbols that represent one-way, asynchronous communications between active objects.
- **Data types:** Classifiers that define data values. Data types can model both primitive types and enumerations.
- **Packages:** Shapes designed to organize related classifiers in a diagram. They are symbolized with a large tabbed rectangle shape.
- **Interfaces:** A collection of operation signatures and/or attribute definitions that define a cohesive set of behaviors. Interfaces are similar to classes, except that a class can have an instance of its type, and an interface must have at least one class to implement it.
- **Enumerations:** Representations of user-defined data types. An enumeration includes groups of identifiers that represent values of the enumeration.
- **Objects:** Instances of a class or classes. Objects can be added to a class diagram to represent either concrete or prototypical instances.
- **Artifacts:** Model elements that represent the concrete entities in a software system, such as documents, databases, executable files, software components, etc.