Experiment 10

Aim: To create a class diagram for the project Object Detection Solutions

Requirements:

Hardware:

- Intel 7th Gen CPU
- Graphics Card
- Storage – Hard Disk

Software:

- OS: Ubuntu, Windows, MacOS
- Umbrello UML Modeler

Theory:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

Purpose of Class Diagrams

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

A UML class diagram is made up of:

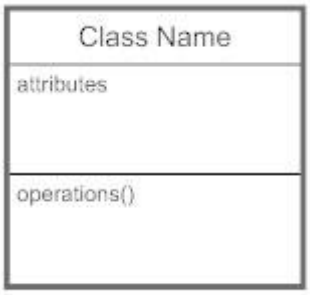
- A set of classes and
- A set of relationships between classes

Basic Class Diagram Symbols and Notations

- **Classes:** Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.
  Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition (left-aligned, not bolded, and lowercase), and write operations into the third.

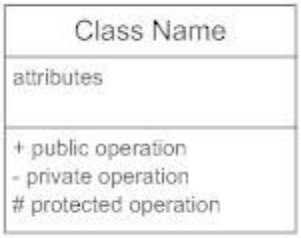
Class

- **Active Classes:** Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.



Active class

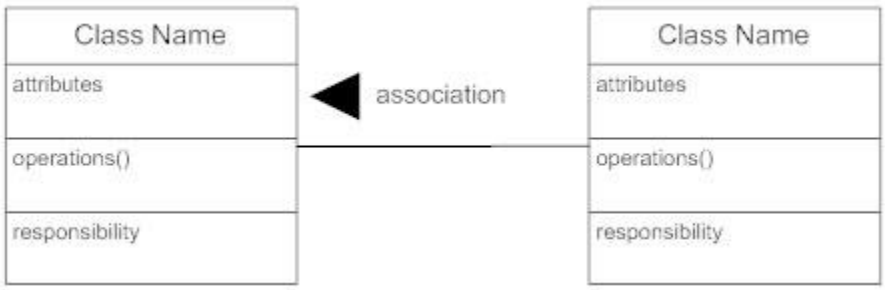
- **Visibility:** Use visibility markers to signify who can access the information contained within a class. Private visibility, denoted with a - sign, hides information from anything outside the class partition. Public visibility, denoted with a + sign, allows all other classes to view the marked information. Protected visibility, denoted with a # sign, allows child classes to access information they inherited from a parent class.



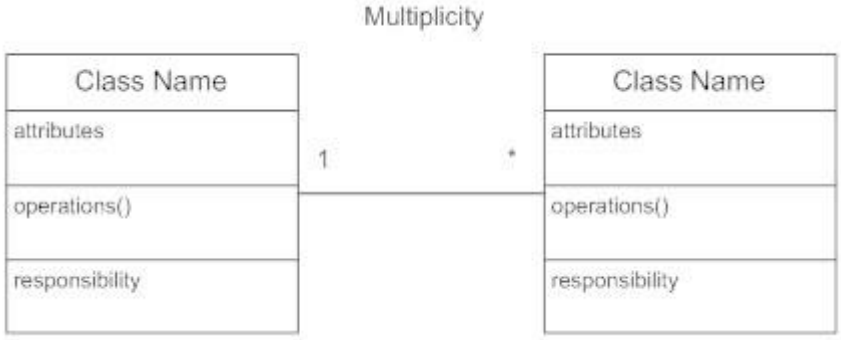
Visibility

Marker	Visibility
+	public
-	private
#	protected
~	package

- **Associations:** Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes see each other.

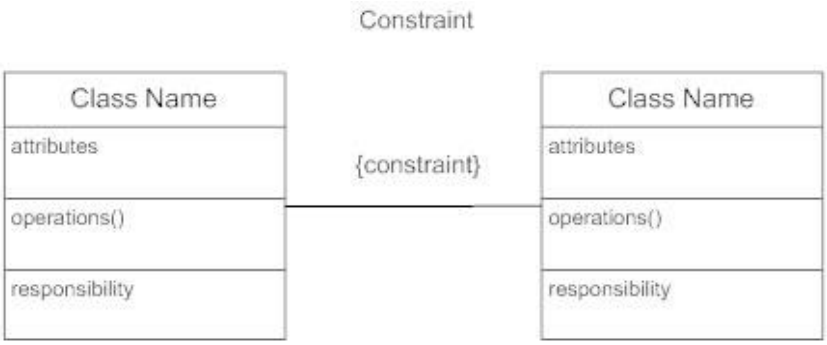


- **Multiplicity (Cardinality):** Place multiplicity notations near the ends of an association. These symbols indicate the number of instances of one class linked to one instance of the other class. For example, one company will have one or more employees, but each employee works for just one company.

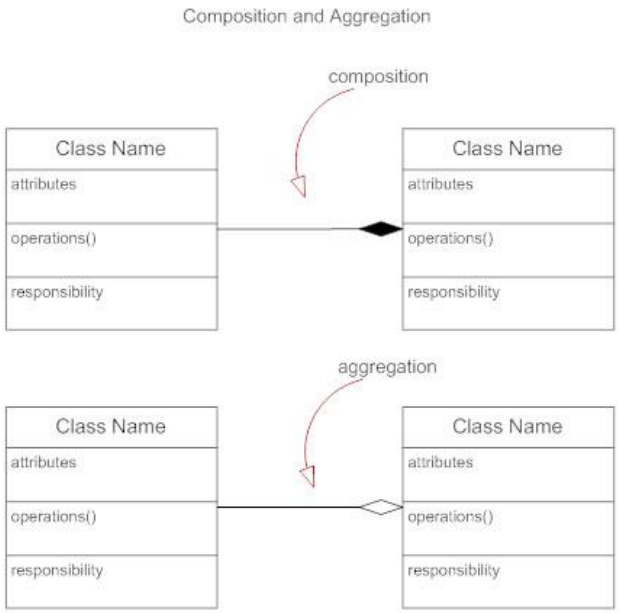


Indicator	Meaning
0..1	Zero or one
1	One only
0..*	0 or more
1..* *	1 or more
n	Only n (where n > 1)
0..n	Zero to n (where n >1)
1..n	One to n (where n > 1)

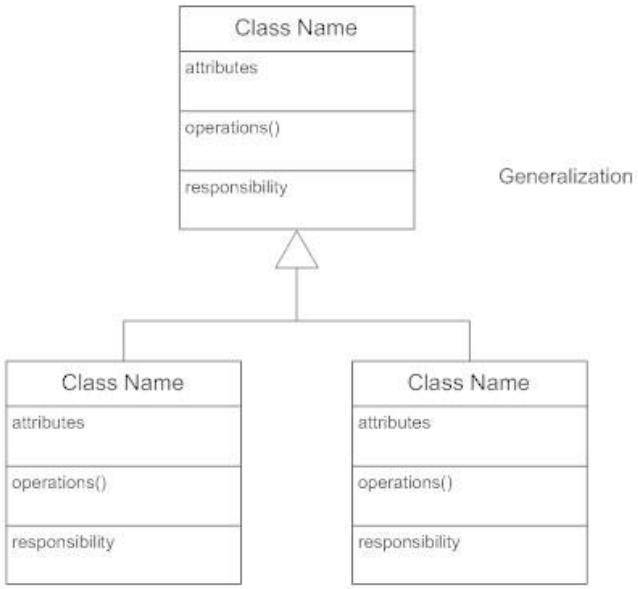
- **Constraint:** Place constraints inside curly braces {}.



- **Composition and Aggregation:** Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole, and Class B, its part. Illustrate composition with a filled diamond. Use a hollow diamond to represent a simple aggregation relationship, in which the "whole" class plays a more important role than the "part" class, but the two classes are not dependent on each other. The diamond ends in both composition and aggregation relationships point toward the "whole" class (i.e., the aggregation).

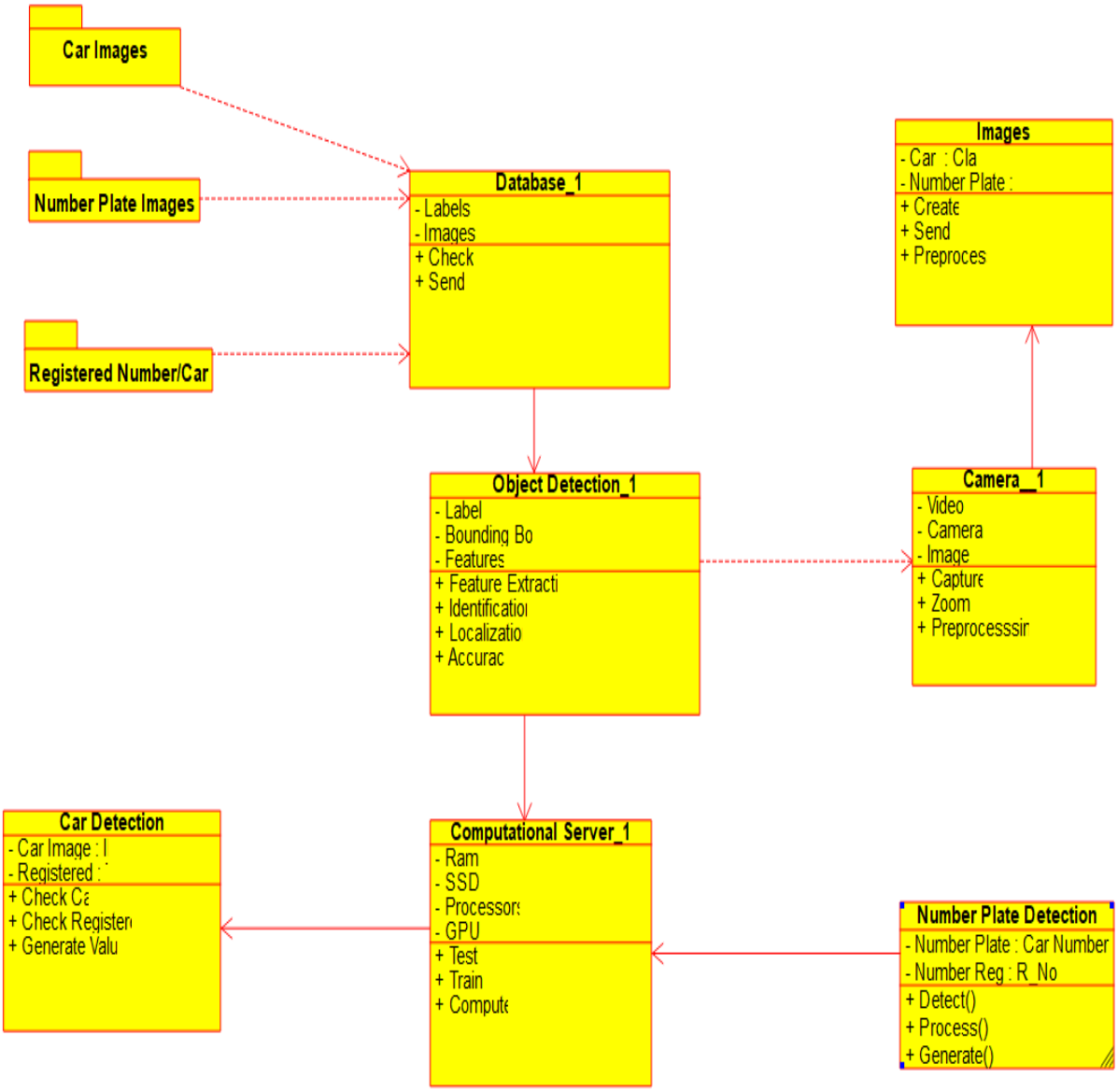


- **Generalization:** Generalization is another name for inheritance or an "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. For example, Honda is a type of car. So the class Honda would have a generalization relationship with the class car.



Class Diagram Diagrams for the Given Projects: Object Detection Solutions

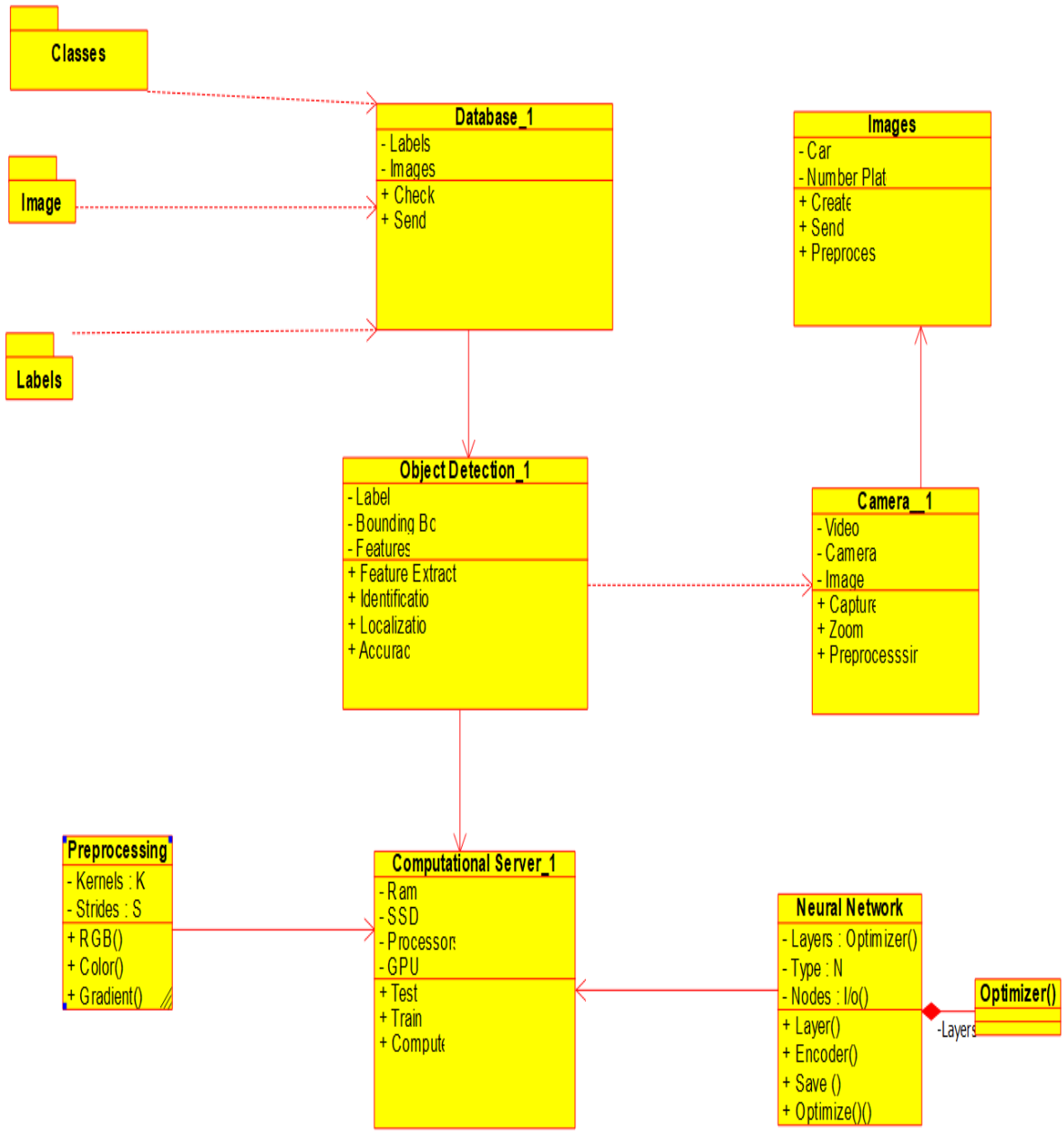
Class Diagram 1:



Car & Number Plate Detection

Class Diagram for Car & Number 1plate Detection, it takes the images of the car and then recognizes it if it is registered or not, then notifies accordingly.

Class Diagram 2:



Object Detection
Methodology

Class Diagram for Object Detection system, It identifies the object, checks the database for the Label and then gives the user the output. This tells about all the Methodologies in Object Detection.

Conclusion:

The Class Diagram for the project Object Detection Solution has been made.