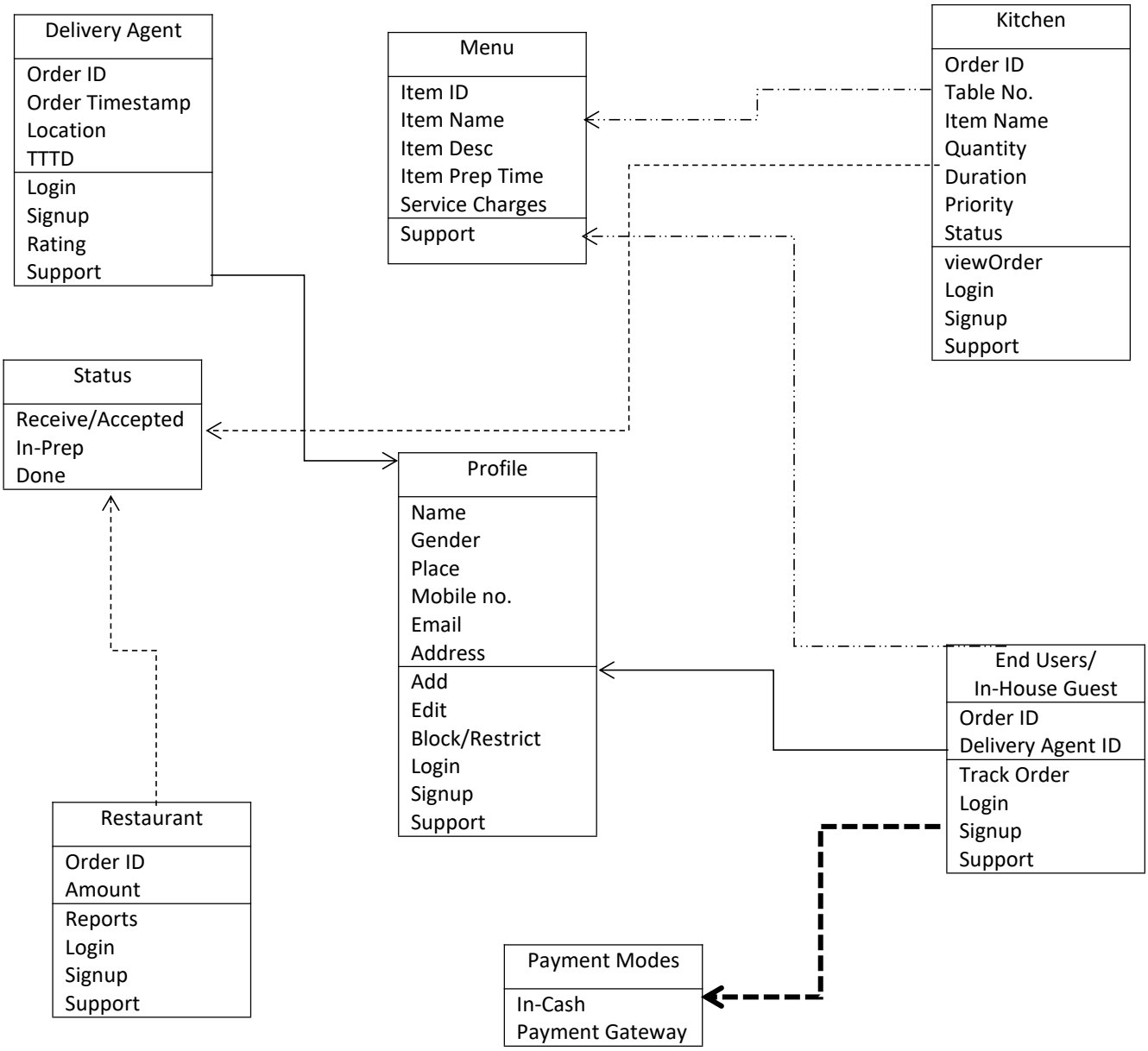
CLASS DIAGRAM



Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

The purpose of the class diagram can be summarized as −

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

Data Flow Diagram

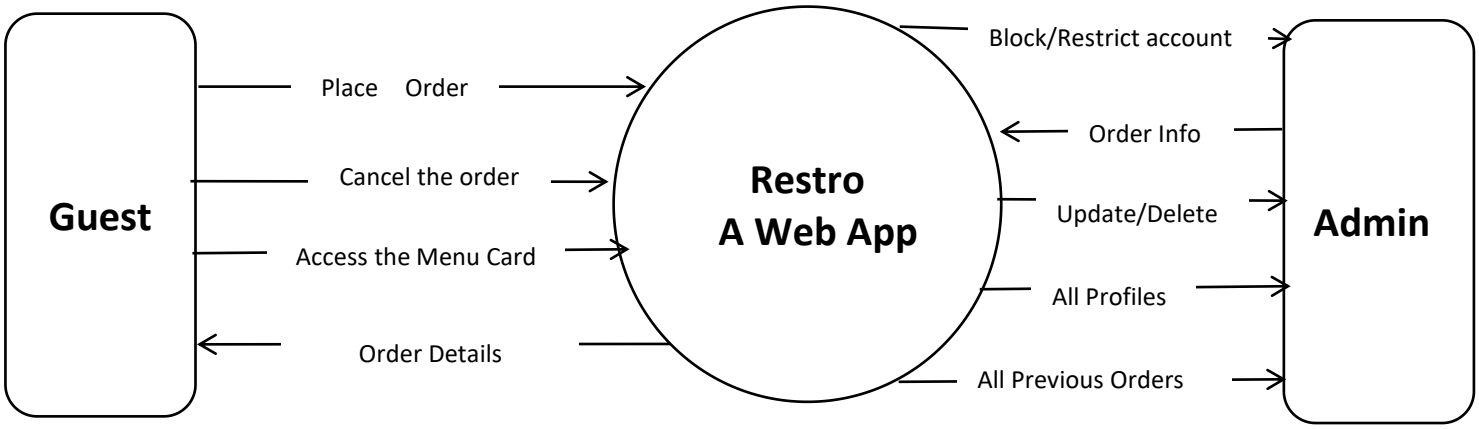
- A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

- DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

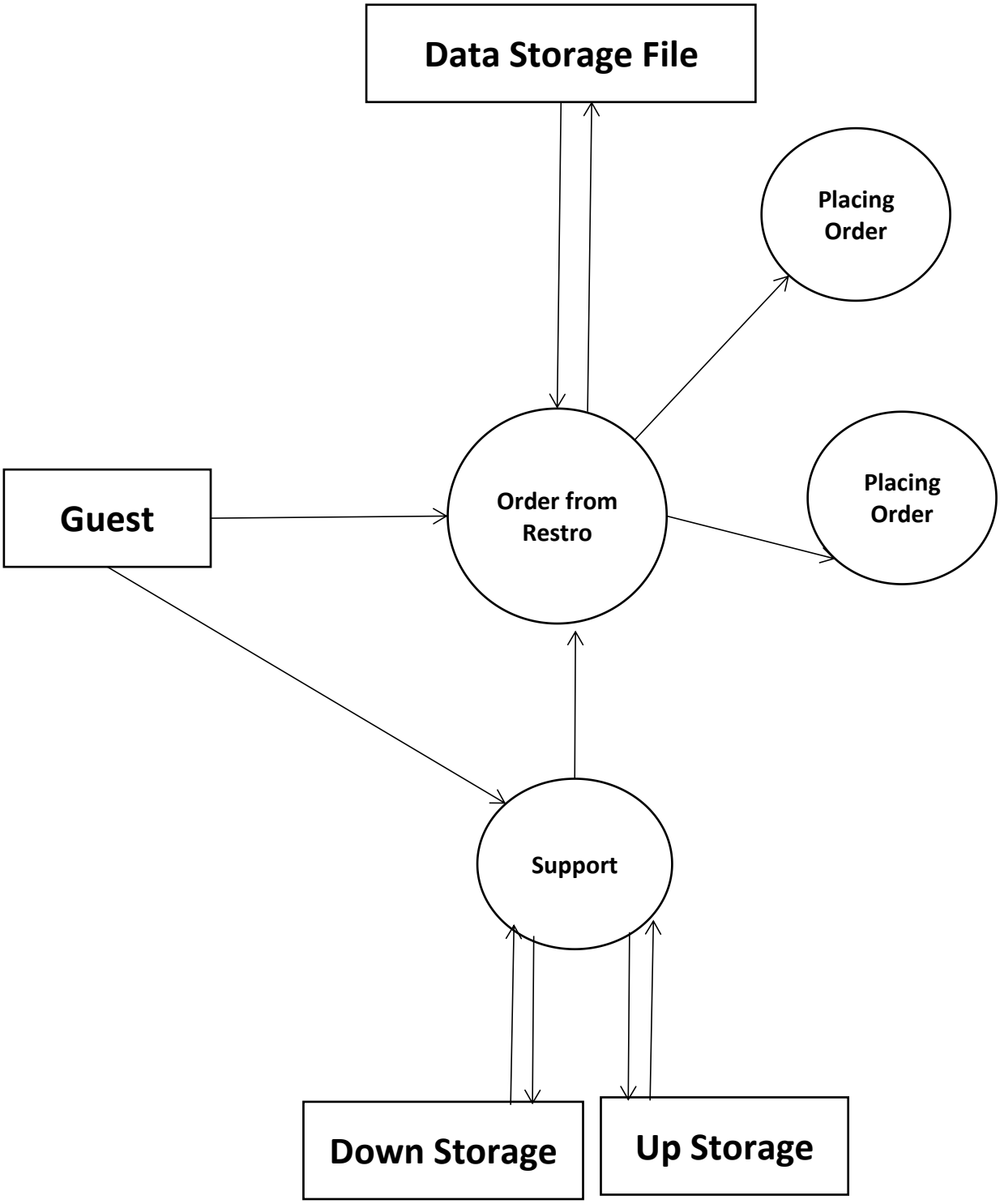
- DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its sub processes.

- DFD Level 2 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail about the system's functioning.

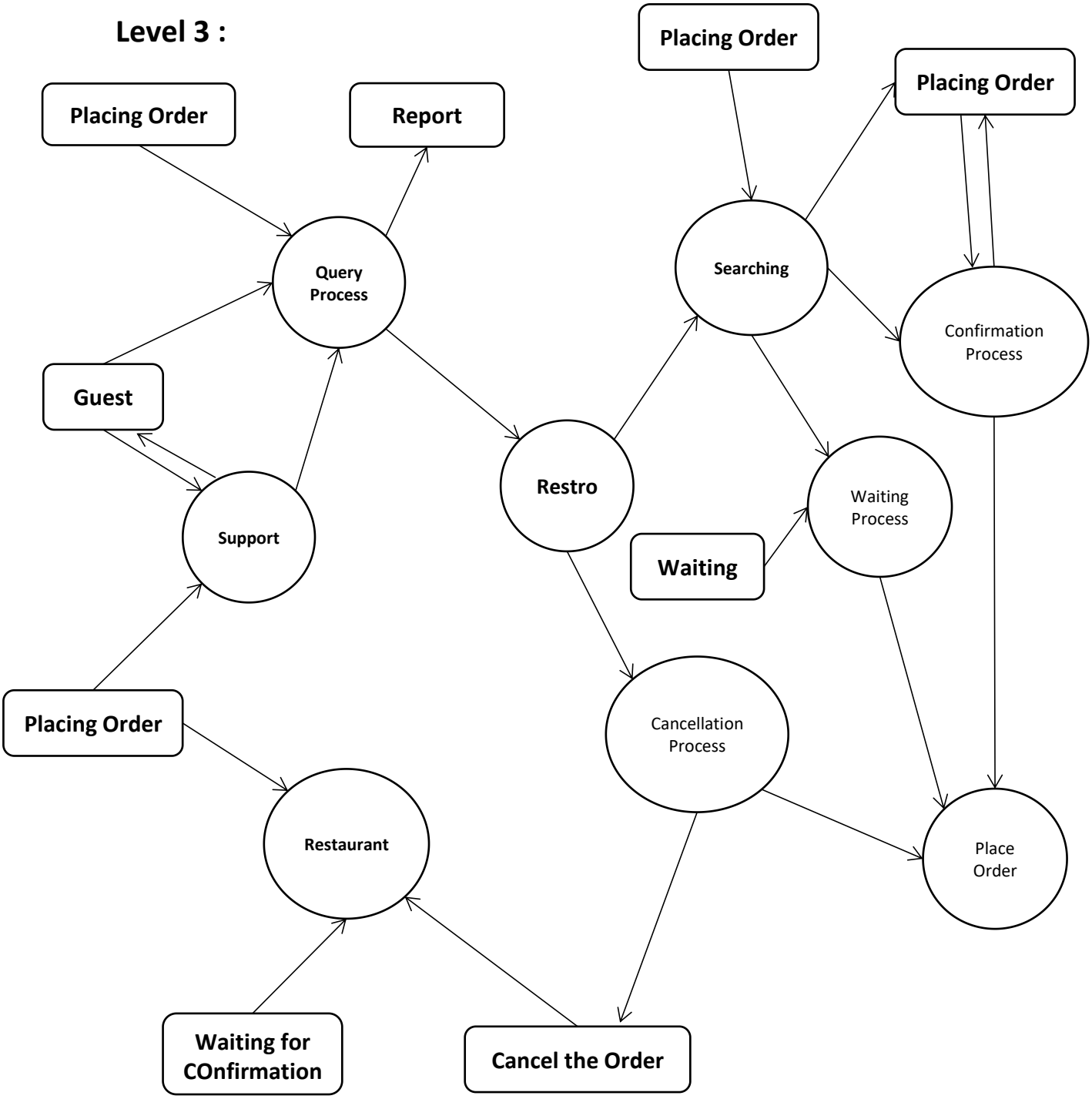
LEVEL 0 :


Guest
Place Order
Cancel the order
Access the Menu Card
Order Details
Restro
A Web App
Block/Restrict account
Order Info
Update/Delete
All Profiles
All Previous Orders
Admin

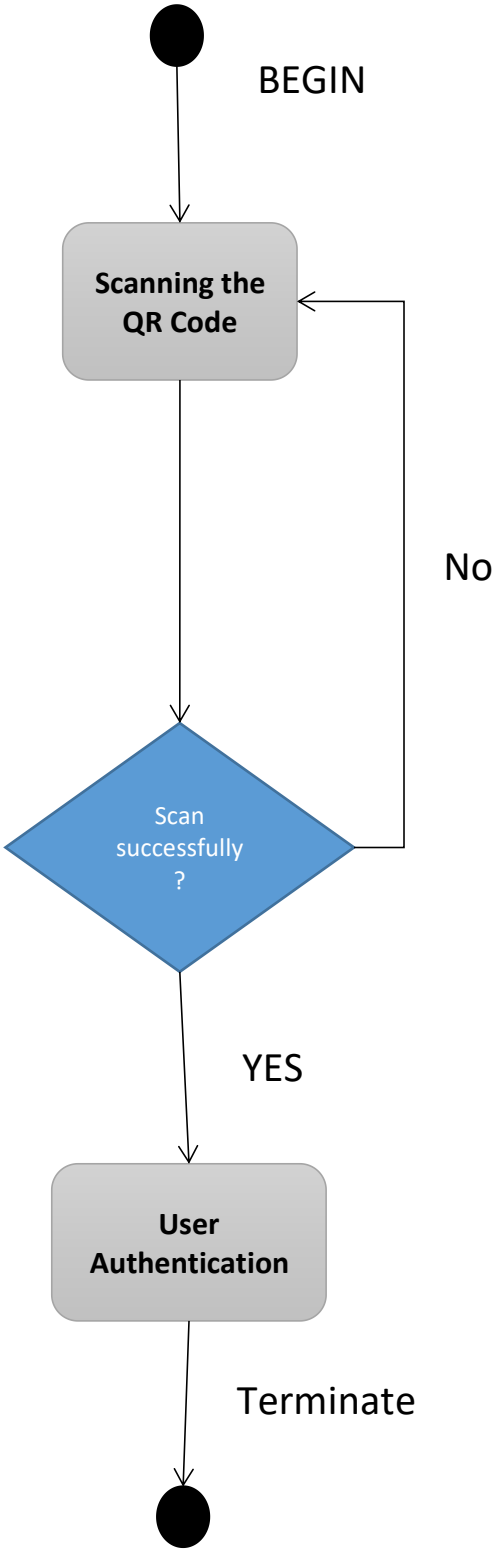
LEVEL 1 :


Data Storage File
Placing Order
Guest
Order from Restro
Placing Order
Support
Down Storage
Up Storage

Level 3 :


Placing Order
Report
Query Process
Guest
Support
Placing Order
Restaurant
Waiting for COnfirmation
Placing Order
Searching
Placing Order
Confirmation Process
Restro
Waiting Process
Waiting
Cancellation Process
Place Order
Cancel the Order

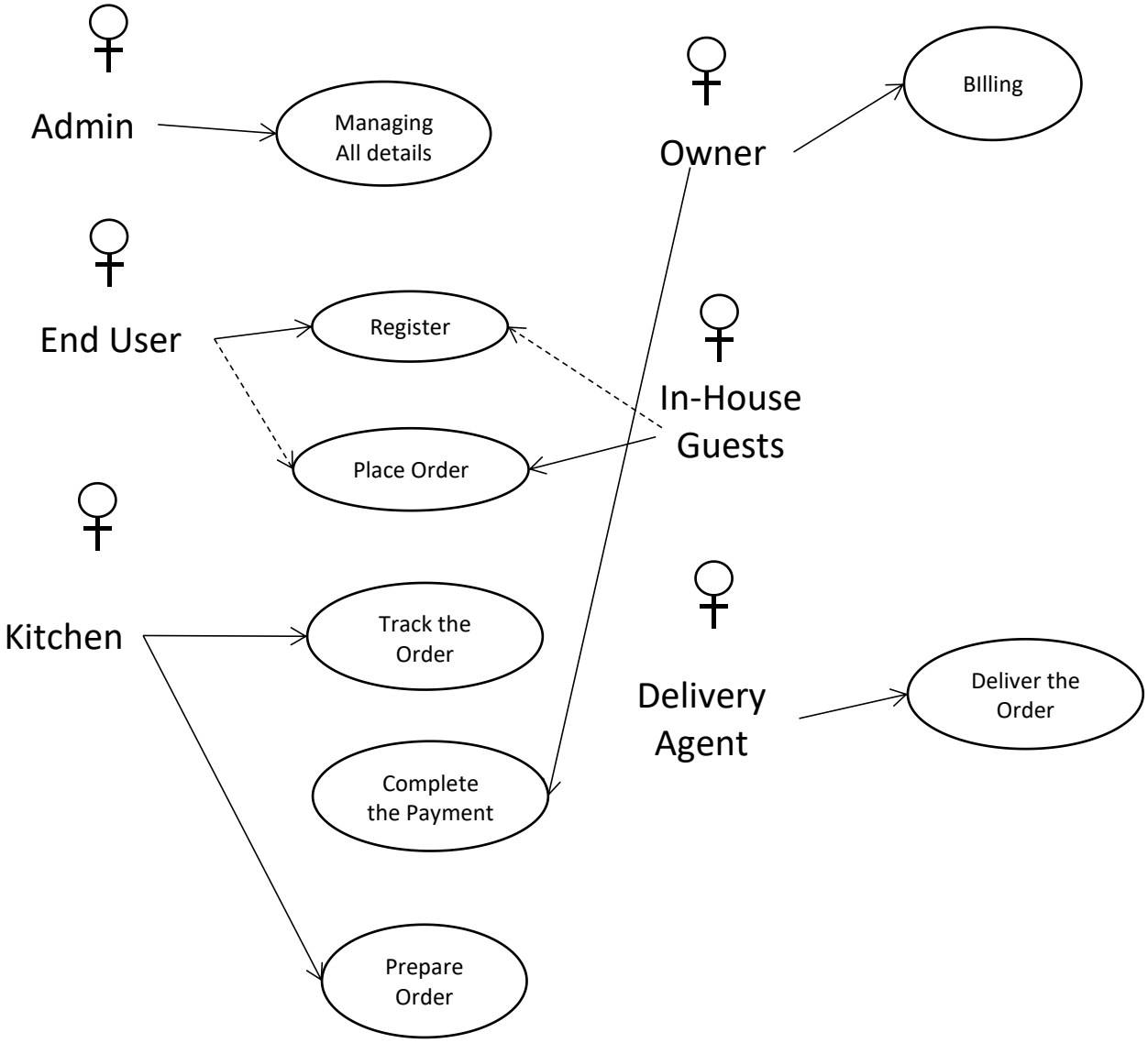
State Diagram



A state diagram is the graphical representation of a state machine and one of the 14 UML diagram types for software and systems. State diagrams show a behavioral model consisting of states, state transitions and actions. UML state diagrams are based on the concept of state diagrams by David Harel. State diagrams depict the permitted states and transitions as well as the events that effect these transitions.

State diagrams are commonly used in the area of embedded systems. State diagrams help to visualize the entire life cycle of objects and thus help to provide a better understanding of state-based systems. An example of such a state-based system is a cash machine: Upon activation either the state *ready* or the state *malfunction* could be reached. As soon as the debit card is inserted it is verified. Depending on the result of the verification the pin number is requested or the process is aborted. Other possible states are account query or availability check etc.

Use Case Diagram



In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

UML use case diagrams are ideal for:

- Representing the goals of system-user interactions
- Defining and organizing functional requirements in a system
- Specifying the context and requirements of a system
- Modeling the basic flow of events in a use case