ANIMAL SHELTER WEBSITE

SHIRISH SHETTY -2103164
MAITHALI SHINDE -2103166
ADARSH SHUKLA -2103167

EXPERIMENT NO 6

AIM: Identify scenarios & develop use case diagram for the project

THEORY:

A Use Case Diagram (UCD) is a graphical representation used in software engineering to illustrate the interactions between various actors (users or external systems) and the functionalities or use cases offered by a system or software application. UCDs provide a high-level view of the system's behavior from a user's perspective, highlighting the system's functionality and the roles played by different actors. They are instrumental in communicating system requirements and functionalities to developers, designers, and stakeholders involved in the software development process.

Similar to Data Flow Diagrams (DFD) and State-Activity Diagrams (SAD), Use Case Diagrams have their own set of components and rules:

Components of a Use Case Diagram:

1. Actor: Actors represent the different roles or entities that interact with the system. These could be users, external systems, or even other software applications. Actors are depicted as stick figures or named entities, and they are placed outside the system boundary. Actors are essential for defining who interacts with the system and what their roles entail.

2. Use Case: Use cases represent the specific functionalities or actions that the system provides to its actors. These functionalities describe what the system does and how it responds to different user interactions. Use cases are depicted as ovals or ellipses within the system boundary. Each use case is labeled with a descriptive name.

3. Relationships: Arrows or lines connecting actors and use cases represent the relationships or interactions between actors and the use cases they participate in. These relationships indicate which actors are involved in or initiate specific use cases. Relationships can be categorized into associations, generalizations (inheritance), or dependencies.

4. System Boundary: The system boundary is a rectangle that encloses all the actors and use cases. It defines the scope of the system being modeled in the diagram.

Rules for Creating a Use Case Diagram:

1. Clear Actor Roles: Ensure that the roles of actors are clearly defined and named so that their interactions with the system are easily understood.

2. Descriptive Use Case Names: Name each use case descriptively to convey its purpose or functionality. Use clear and concise language to make it easy for stakeholders to comprehend the system's capabilities.

3. Proper Relationships: Use arrows or lines to connect actors with the use cases they interact with. Choose the appropriate relationship type (association, generalization, or dependency) to accurately represent the nature of the interaction.

4. Keep it Simple: While a Use Case Diagram can represent a complex system, it's essential to maintain simplicity for clarity. Avoid overloading the diagram with too many actors or use cases.

5. Consistency: Ensure consistency in the representation of actors, use cases, and relationships throughout the diagram. This consistency aids in maintaining the diagram's readability.

Symbols Used in a Use Case Diagram:

- Stick Figure or Named Entity: Represents an actor, portraying the roles that interact with the system.

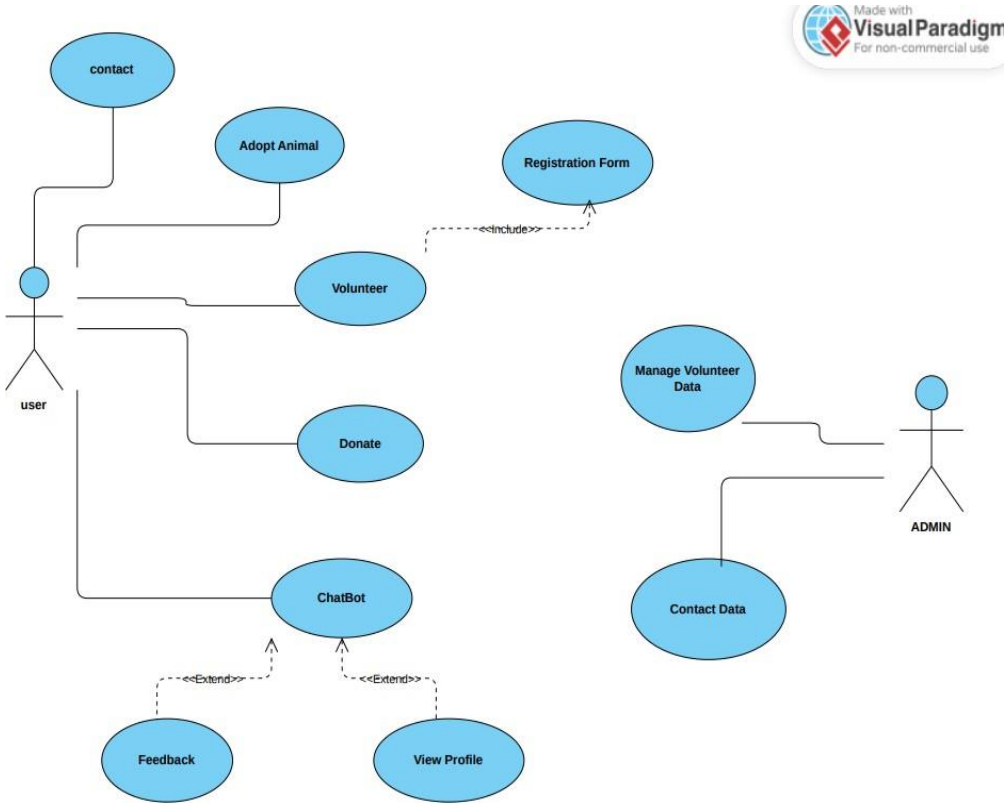
- Oval or Ellipse: Represents a use case, depicting a specific system functionality or action.

- Arrows or Lines: Depict relationships between actors and use cases, indicating the interactions and roles involved.

- Rectangle: Encloses all actors and use cases, defining the system boundary.

In summary, a Use Case Diagram is a valuable tool in software engineering for visualizing the interactions between actors and the functionalities provided by a software system. It helps stakeholders understand how different users or external entities interact with the system and what functionalities the system offers to meet their needs.

OUTPUT:


contact
Adopt Animal
Registration Form
<<include>>
Volunteer
user
Donate
Manage Volunteer Data
ADMIN
Contact Data
ChatBot
<<Extend>>
<<Extend>>
Feedback
View Profile