**CHAPTER ONE**

* 1. Requirement Analysis(Use Case)

A high-level use case diagram for a kindergarten management system is a visual representation that provides a simplified overview of the system's core functionalities (use cases) and the primary roles (actors) that interact with it. It focuses on the main goals that different users can achieve through the system, without delving into the detailed steps or internal processes. It essentially paints a broad picture of what the system does and who interacts with it.

1.2 Components Of High Level Use Case Diagram

1.**Actors:** These are entities outside the system that interact with it. They represent roles that users or other systems can play. In your diagram, the actors are represented by stick figures and labeled:

* **Admin:** Represents the administrator of the KG Management System.
* **Teacher:** Represents the teachers who use the system.
* **Parent:** Represents the parents of the students.
* **System:** Represents an external system that interacts with the KG Management System (e.g., for payment processing or sending notifications).

2.**Use Cases:** These represent the main functionalities or goals that the actors can achieve by interacting with the system. They are depicted as ovals and labeled with verb phrases indicating the action. Examples from your diagram include:

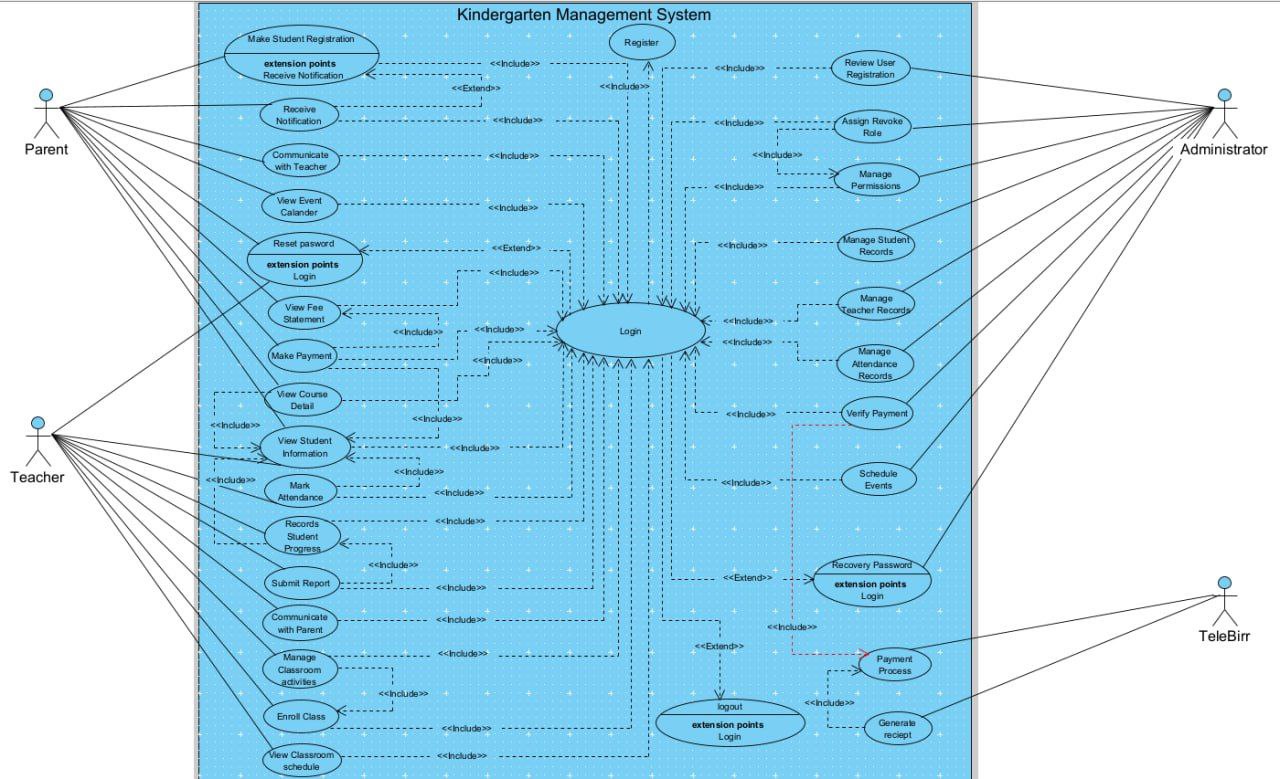
* Manage User Accounts
* Authenticate User
* Manage System Settings
* Manage Class Information
* Make Payments
* Process Payment
* Manage Classes
* Record Student Attendance
* Communicate with Parents
* Manage Learning Materials
* Send Notifications
* Communicate with Teacher
* Manage Child Profile
* View Progress Reports

3.**System Boundary:** This is the rectangle that encloses all the use cases. It visually defines the scope of the KG Management System, separating it from the external actors. The label "KG Management System" at the top of the rectangle clearly identifies what the system is.

4.**Relationships:** These illustrate how actors interact with use cases and how use cases relate to each other. The types of relationships shown in your diagram are:

* ·**Association:** Represented by a solid line connecting an actor to a use case. It indicates that the actor participates in or initiates that use case. For example, the Admin is associated with "Manage User Accounts" and "Manage System Settings."
* **Include (**<<include>>**):** Represented by a dashed arrow with an open arrowhead pointing from a base use case to an included use case, labeled with <<include>>. It signifies that the base use case explicitly incorporates the behavior of the included use case. For example, "Manage User Accounts"<<include>>"Authenticate User," meaning that authenticating a user is a necessary step within managing user accounts. Similarly, "make payments"<<include>>"Process Payment."
* **Extend (**<<extend>>**):** Represented by a dashed arrow with an open arrowhead pointing from an extending use case to a base use case, labeled with <<extend>>. It indicates that the extending use case provides additional or optional behavior to the base use case at specific extension points. For example, "Send Notifications"<<extend>>"Communicate with Parents," suggesting that sending notifications is an optional extension of the communication process. Similarly, "View Progress Reports"<<extend>>"Manage Child Profile" with an extension point "upload child info," and "Communicate with Teacher" and "Communicate with Parents" are extended by the possibility of sending attachments.

1.3 Example Of High Level Use Case Diagram

****

**Figure 1.1 High Level Use Case Diagram For Kindergarten Management**

**1.4 Overall Description of the KG Management System Use Case Diagram**

**Actors:**

* **Parent:** Interacts with the system for tasks like student registration, receiving notifications, communicating with teachers, viewing the event calendar and fee statements, making payments, viewing course details, and managing their login (resetting password).
* **Teacher:** Interacts with the system for login, viewing student information, recording student progress, submitting reports, communicating with parents, managing classroom activities, enrolling students in classes, viewing the classroom schedule, and marking attendance.
* **Administrator:** Has comprehensive access for managing the system, including registering new users, reviewing user registrations, assigning and revoking roles, managing permissions, student records, teacher records, and attendance records. They also handle financial aspects like verifying payments and generating reports, as well as scheduling events and managing their login (including password recovery and logout).
* **TeleBirr:** Represents an external payment gateway system that interacts with the Kindergarten Management System for processing payments and generating receipts.

**Key Use Cases:**

The diagram illustrates a wide range of functionalities, broadly categorized as:

* **User Management:** Registration, Login, Logout, Reset Password, Recover Password, Review User Registration, Assign/Revoke Role, Manage Permissions.
* **Student Management:** Make Student Registration, View Student Information, Records Student Progress, Enroll Class, View Classroom Schedule, Manage Student Records, Mark Attendance, Manage Attendance Records.
* **Communication:** Receive Notification, Communicate with Teacher, Communicate with Parent, View Event Calendar.
* **Financial Management:** View Fee Statement, Make Payment, Verify Payment, Process Payment, Generate Receipt.
* **Course/Class Management:** View Course Detail, Manage Classroom Activities.
* **Reporting:** Submit Report, Manage Teacher Records, Schedule Events.

**Relationships:**

* **Associations:** Solid lines connect actors to the use cases they participate in.
* <<include>>**:** Dashed arrows labeled <<include>> indicate that a base use case incorporates the behavior of another use case. For example, "Make Student Registration" includes "Receive Notification."
* <<extend>>**:** Dashed arrows labeled <<extend>> indicate that an extending use case adds optional behavior to a base use case at a specific extension point. For example, "Reset Password,""Recover Password," and "Logout" extend the "Login" use case at the "extension points Login."

1.5 Tools And Steps To Draw High Level Use Case

**Tool:**

* **Visual Paradigm:**Robust UML tool with good features for various UML diagrams.

**Steps to Draw This High-Level Use Case Diagram:**

**Phase 1: Setting Up the Diagram**

1. **Open Visual Paradigm:** Launch your Visual Paradigm application.
2. **Open Your Project:** Open the project where you want to create this diagram. If you don't have a project open, create a new one by going to **File > New Project**.
3. **Navigate to Diagram Creation:** In your project, navigate to the location where you want to store the diagram (e.g., a specific package). You can do this in the **Project Browser** on the left.
4. **Create New Use Case Diagram:**
   1. **Right-Click:** Right-click on the target package in the Project Browser or in the Diagram Editor if you have another diagram open.
   2. **Select "New Diagram":** In the context menu, select **New Diagram**.
   3. **Choose "UML":** In the "New Diagram" window, select **UML**.
   4. **Select "Use Case Diagram":** From the list of UML diagrams, choose **Use Case Diagram**.
   5. **Name the Diagram:** In the "Create Diagram" dialog, enter a descriptive name for your diagram (e.g., "Kindergarten Management System Use Case Diagram").
   6. **Click "OK":** Click the **OK** button to create the diagram. A blank Use Case Diagram canvas will open.

**Phase 2: Adding Actors**

1. **Select the Actor Tool:** On the Diagram Toolbar (usually on the left side of the application window), find the **Actor** symbol. It typically looks like a stick figure icon. Click on this icon to select the Actor tool.
2. **Place the "Parent" Actor:**
   1. Click on the diagram canvas where you want to place the "Parent" actor (typically on the left side).
   2. A rectangle with a placeholder name will appear. Immediately type Parent and press **Enter**.
3. **Place the "Teacher" Actor:**
   1. Ensure the Actor tool is still selected.
   2. Click on the canvas below or near the "Parent" actor.
   3. Type Teacher and press **Enter**.
4. **Place the "Administrator" Actor:**
   1. Ensure the Actor tool is still selected.
   2. Click on the canvas on the right side of where you plan to place the use cases.
   3. Type Administrator and press **Enter**.
5. **Place the "TeleBirr" Actor:**
   1. Ensure the Actor tool is still selected.
   2. Click on the canvas below or near the "Administrator" actor.
   3. Type TeleBirr and press **Enter**.

**Phase 3: Adding Use Cases**

1. **Select the Use Case Tool:** On the Diagram Toolbar, find the **Use Case** symbol. It typically looks like an oval icon. Click on this icon to select the Use Case tool.
2. **Place and Name Use Cases (one by one):** For each use case in the uploaded diagram, perform the following:
   1. Click on the diagram canvas where you want to place the use case.
   2. A blank oval will appear. Immediately type the name of the use case exactly as it appears in the diagram and press **Enter**.
3. **Arrange Use Cases:** Arrange the use case ovals in a layout similar to the uploaded diagram, generally grouping related use cases together and placing the "Login" use case centrally as it has many relationships.

**Phase 4: Drawing Associations**

1. **Select the Association Connector:** On the Diagram Toolbar, find the **Association** symbol. It's usually a solid line icon. Click on it.
2. **Connect Actors to Use Cases:** For each interaction shown in the diagram:
   1. Click on an actor.
   2. Drag the mouse to the corresponding use case and click on it. A solid line will be drawn connecting them.
   3. Draw associations between:
      1. **Parent** and: Make Student Registration, Receive Notification, Communicate with Teacher, View Event Calendar, Reset Password, Login, View Fee Statement, Make Payment, View Course Detail.
      2. **Teacher** and: Login, View Student Information, Records Student Progress, Submit Report, Communicate with Parent, Manage Classroom Activities, Enroll Class, View Classroom Schedule, Mark Attendance.
      3. **Administrator** and: Register, Review User Registration, Assign/Revoke Role, Manage Permissions, Manage Student Records, Manage Teacher Records, Manage Attendance Records, Verify Payment, Schedule Events, Recover Password, Login, Logout.
      4. **TeleBirr** and: Process Payment, Generate Receipt.

**Phase 5: Drawing Include Relationships**

1. **Select the Include Connector:** On the Diagram Toolbar, find the **Include** symbol. It's usually a dashed line with an open arrowhead and the label <<include>>. Click on it.
2. **Draw Include Relationships:** For each include relationship:
   1. Click on the base use case (the one that includes).
   2. Drag the mouse to the included use case and click on it. A dashed arrow with <<include>> will be drawn, pointing to the included use case.

**Phase 6: Drawing Extend Relationships and Adding Extension Points**

1. **Select the Extend Connector:** On the Diagram Toolbar, find the **Extend** symbol. It's usually a dashed line with an open arrowhead and the label <<extend>>. Click on it.
2. **Draw Extend Relationships:** For each extend relationship:
   1. Click on the extending use case.
   2. Drag the mouse to the base use case (Login) and click on it. A dashed arrow with <<extend>> will be drawn, pointing to the base use case.
   3. Draw the following extend relationships:
      1. Reset Password**&lt;&lt;extend>>**Login
      2. Recover Password**&lt;&lt;extend>>**Login
      3. Logout**&lt;&lt;extend>>**Login
3. **Add Extension Points to the "Login" Use Case:**
   1. **Select "Login":** Click on the Login use case oval to select it.
   2. **Open Specification View:** In Visual Paradigm, the properties of the selected element are usually displayed in a "Specification" view, often at the bottom right of the application. If you don't see it, go to **View > Specification**.
   3. **Navigate to "Extension Points":** In the Specification view, find the tab or section labeled **Extension Points**.
   4. **Add New Extension Point:** Click the **Add** button (usually a plus sign).
   5. **Name the Extension Point:** In the newly added row, type extension points Login in the "Name" column and press **Enter**. You only need to add this single extension point, as all three extend relationships target the same conceptual extension point of the "Login" functionality.

**Phase 7: Adding the System Boundary (Optional but Recommended)**

1. **Select the System Boundary Tool:** On the Diagram Toolbar, find the **System Boundary** symbol. It's usually a rectangle icon. Click on it.
2. **Draw the Boundary:** Click and drag on the diagram canvas to draw a rectangle that encloses all the use case ovals of the Kindergarten Management System.
3. **Label the Boundary:** Select the drawn rectangle. Type Kindergarten Management System and press **Enter**. The label will appear inside or near the boundary.

**Phase 8: Final Arrangement and Refinement**

1. **Adjust Layout:** Click and drag the actors and use cases to arrange them in a visually clear and organized manner, similar to the provided image.
2. **Adjust Connectors:** If necessary, click and drag the segments or endpoints of the association, include, and extend connectors to make them look neat and easy to follow.
3. **Review and Verify:** Carefully review your diagram to ensure all actors, use cases, relationships, and labels are correctly placed and match the uploaded image.