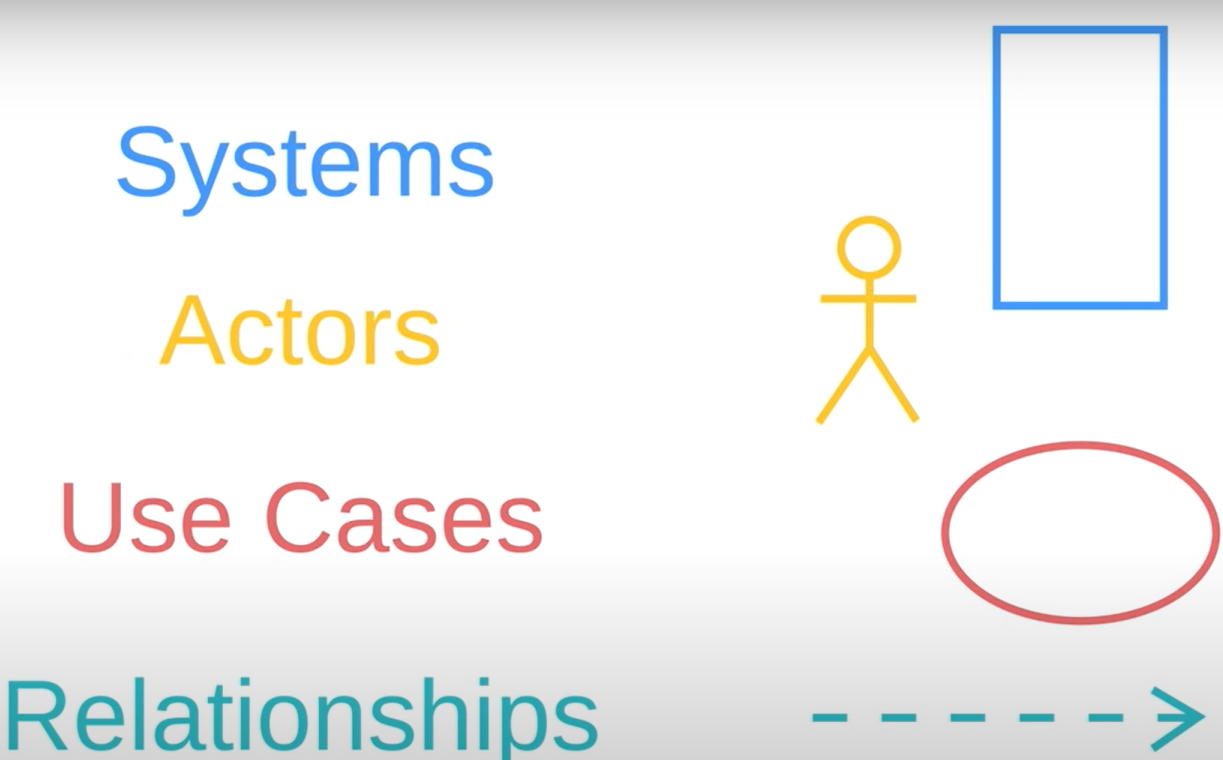
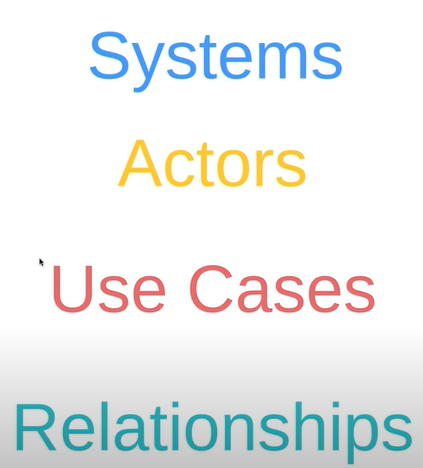
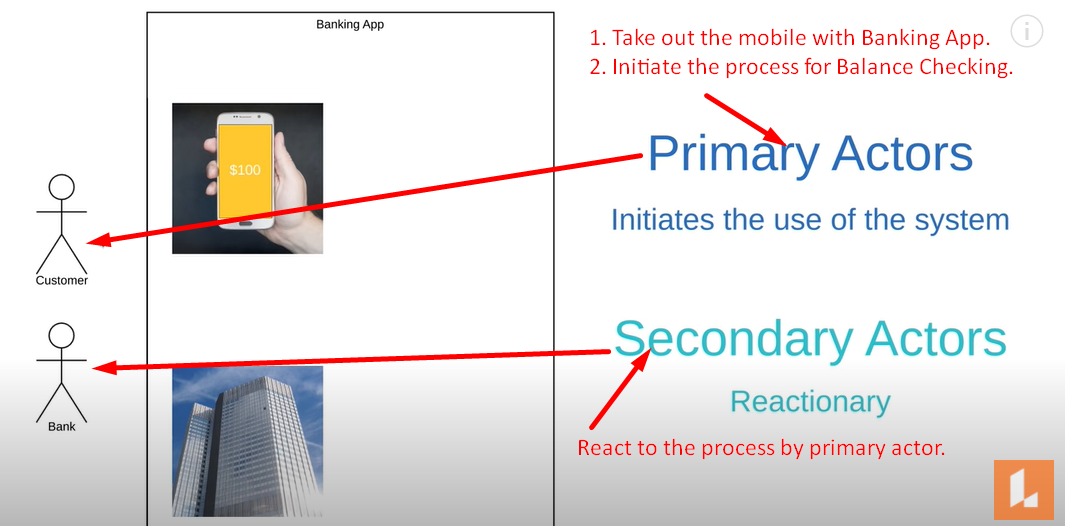
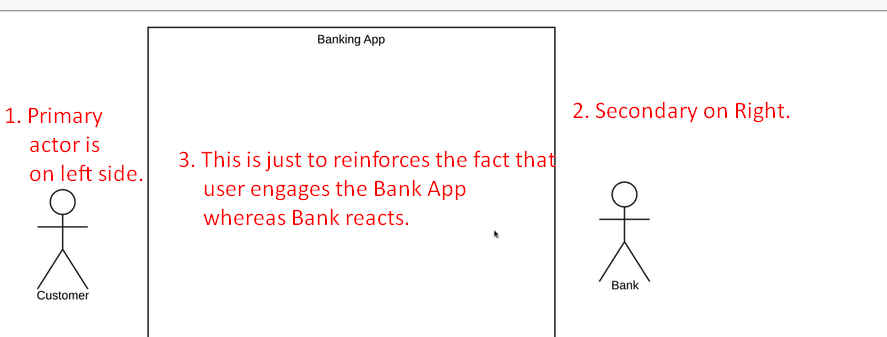
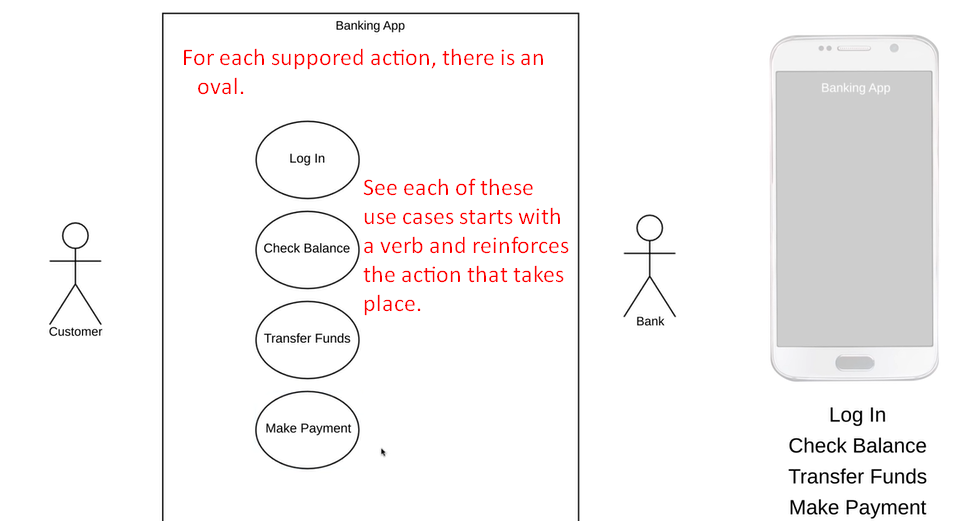
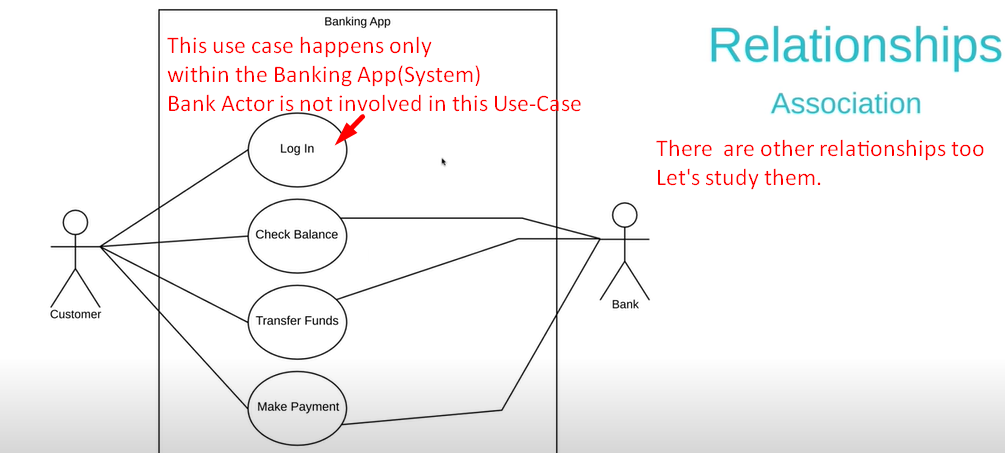
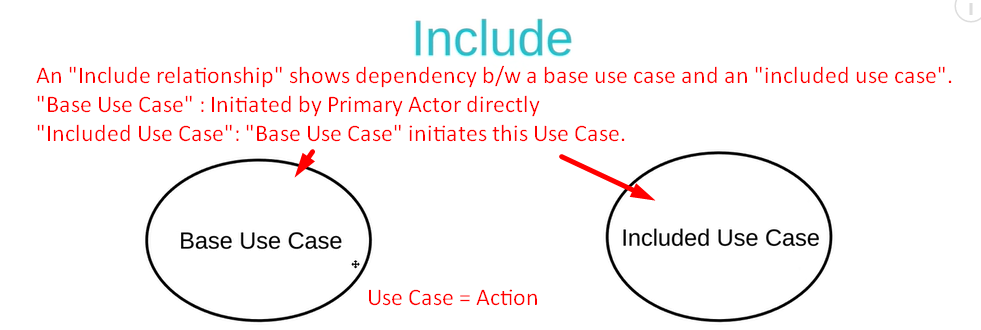
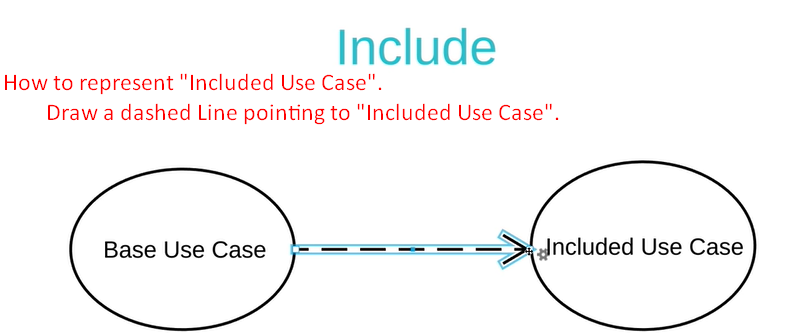
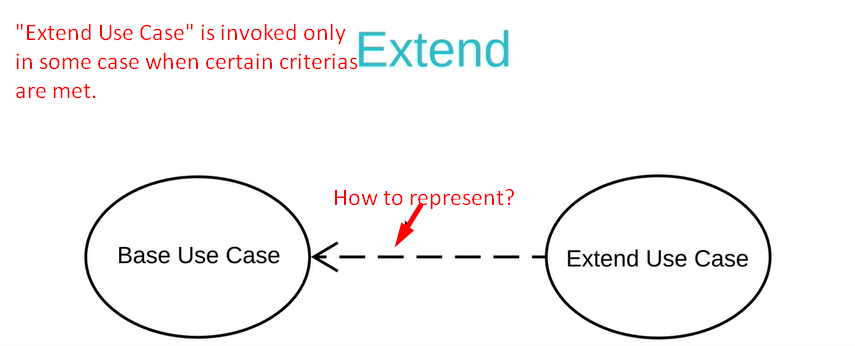
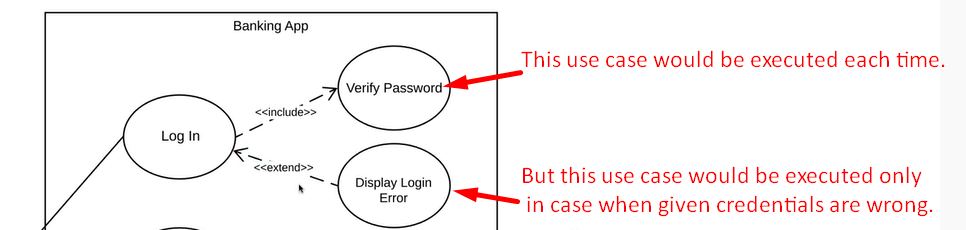
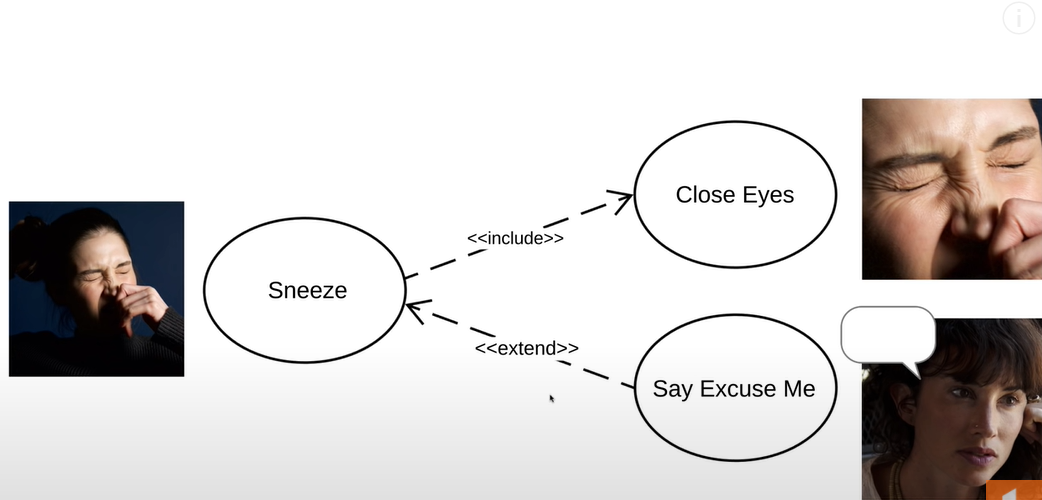
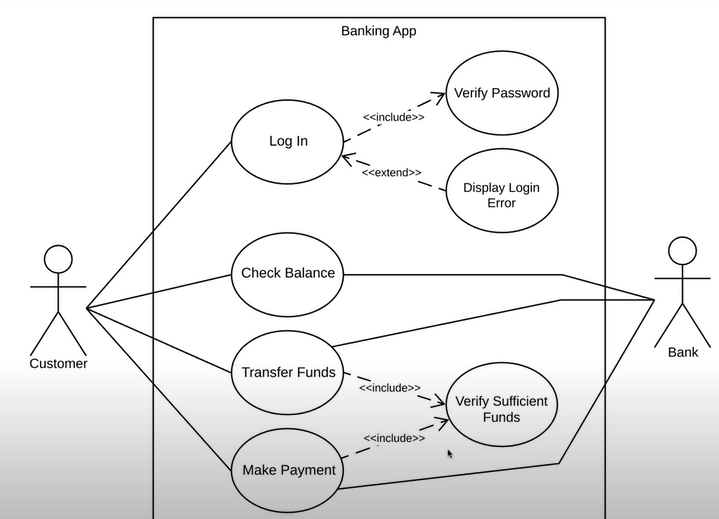
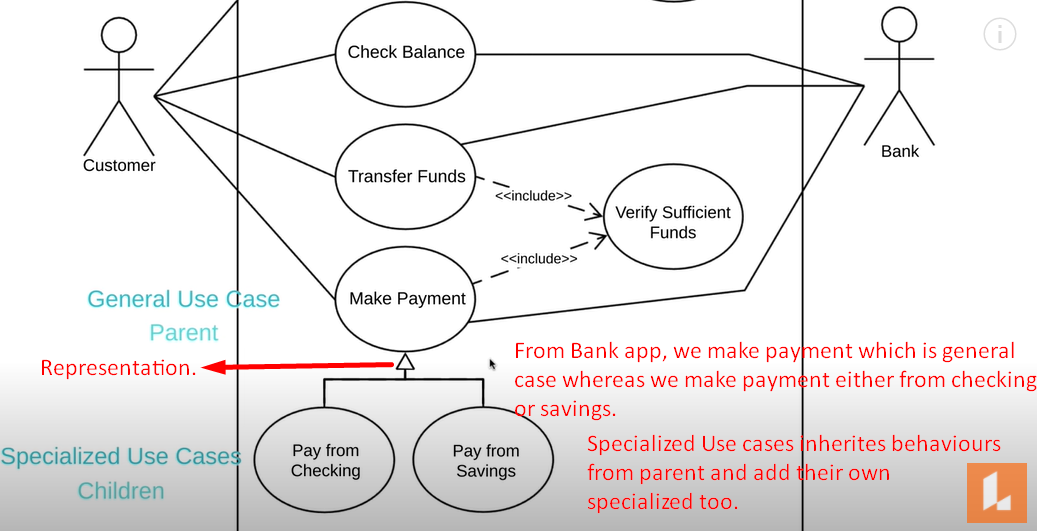
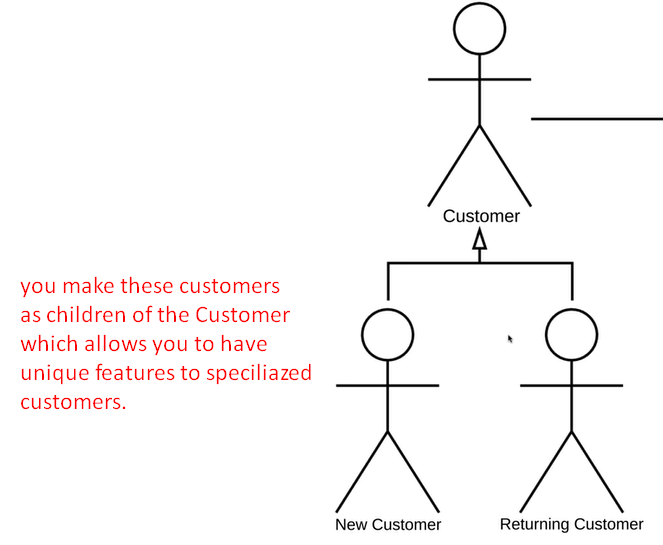
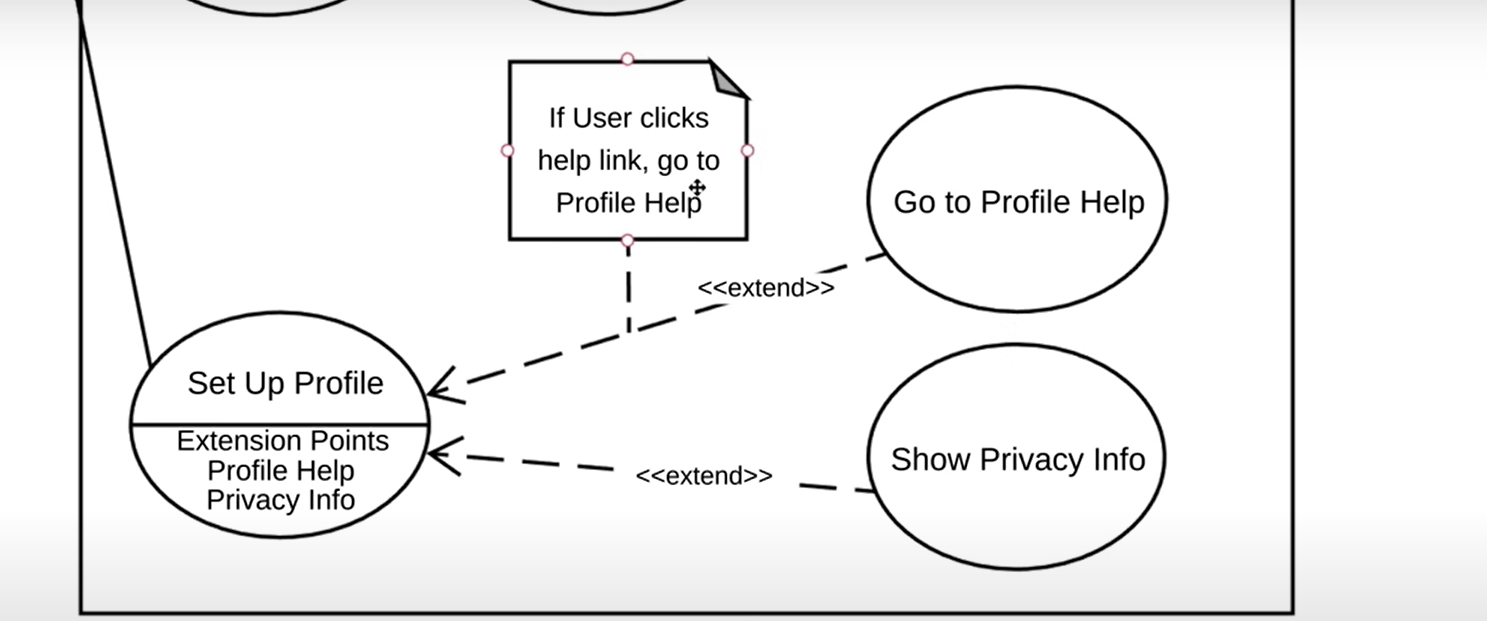
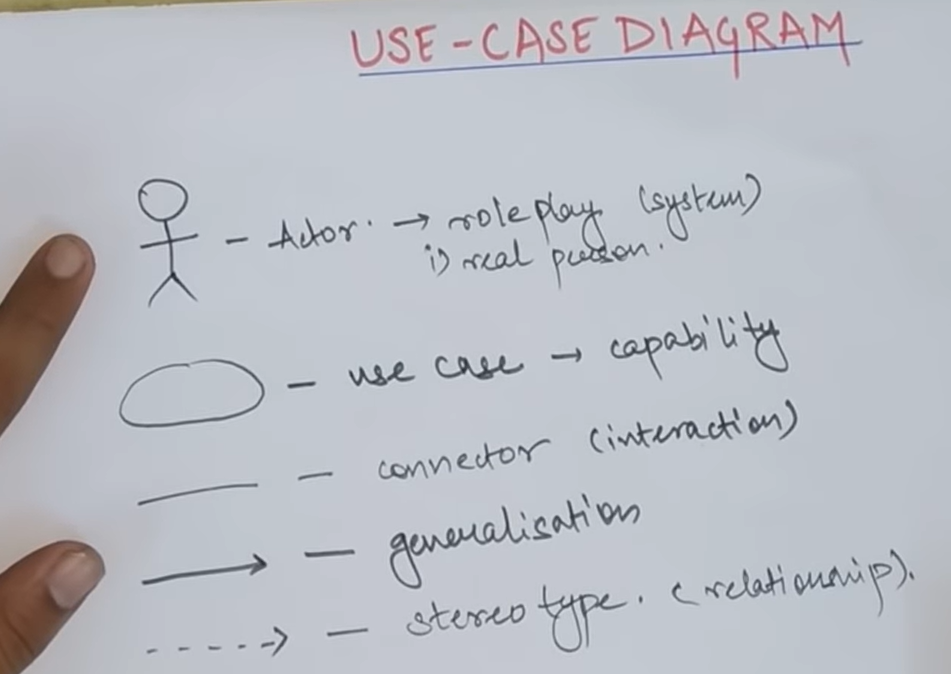
1. <https://www.youtube.com/watch?v=zid-MVo7M-E>
2. We will start with high-level over then the followings:
   1. 
3. **Application**:
   1. Maybe you have some new idea in your head and you want to explain it to someone else.
   2. Maybe your idea is for a new app and every time you talk about it people don’t really understand how they interact with the app or what it would do?
   3. **Solution**: Use-Case Diagram.
4. Here is the simple description about UML Use-Case Diagram.
   1. It shows a system or app.
   2. Then it shows the people, organizations or other systems which interact with it.
   3. then it shows the basic flow of what the system/app does.
5. We’re going to break down Use-Case Diagram into 4 parts.
   1. 
6. **System**:
   1. Anything you’re developing.
      1. Website
      2. Software Component
      3. Business Process.
      4. App
      5. etc. (Any # of other things
   2. System is represented with a rectangle.
   3. Put the name at the top inside the rectangle.  
      As making app for Banking Domain so name is “Banking App”.
   4. Anything within this rectangle happens within the “Banking App”.  
      Anything outside this rectangle doesn’t happen in the “Banking App”.
7. **Actor**:
   1. Represented by Stick Figure.
   2. An actor is someone or something that uses our system to achieve a goal.
   3. That actor can be a person, an organization, another system or external device.
   4. Now who is going to use our banking app?
      1. The most obvious actor is the customer who is going to download and use the app.
      2. The Bank is going to provide information that feeds into our Banking App, like transactions and Balance.
   5. Some couple of things to keep in mind when dealing with Actors.
      1. These actors are external objects.
      2. So need to be placed outside of our system.
   6. Second, Actors need to be thought of as types or categories.
   7. Two kinds of actors.



* 1. 

1. **Use Case**:
   1. This is where you start to describe what our system does.
   2. A Use-Case is represented by Oval.  
      It represents an action that accomplishes some sort of task within the system.
   3. These ovals are placed within the rectangle (representing system) because they are actions that occur within the Banking App (System).
   4. So, what Banking app (system) is going to do (use-cases).   
      We will keep it very simple.
      1. Lets the user login.
      2. Lets the user check balance.
      3. Lets the user transfer funds.
      4. Lets the user make payment.
   5. So we will have actions (represented by ovals) for each of the above actions.
   6. 
2. **Relationship**:
   1. The final element in Use-Case Diagram is Relationship.
   2. 
   3. 
   4.   
      Every time “Base Use Case” is invoked, the “Included Use Case” is also invoked.  
      In other word, without “Included Use Case”, “Base Use Case” can’t be completed.
   5. 
   6. “**Extend Use Case**”
   7.   
      
   8. 
   9. 
3. **Generalization**:
   1. Last relationship.
   2. Also known as inheritance.
   3. 
   4. You can have generalization with an Actor.
   5. You might want to distinguish b/w a new customer and returning customer.
   6. 
   7. **Use case with Extension Points**:
      1. The name of the use case is above the line and then there are extension points below the line.
      2. Extension points are just “**Detailed version of extend relationship”**.
      3. This use case shows us that a customer can set up his profile in our banking app.
      4. Then the extension points show us that when a customer is setting up their profile, they will have the options to navigate to a couple of different screens.  
         If a customer is confused, they can go to the profile help.  
         If they want a detail about their private info, they can go to “Privacy Info” page.  
         Those extension points branch off to extended use cases; “Go to Profile Help” and “Show Privacy Info”.  
         We can even add a **note** to show what sort of conditions would lead to these extension points.  
         
4. 
5. 