**In semester test: given a use case, and we must list a model, why we would use it. Also listing advantages and disadvantages to a model.**  
5 Steps when building an App

1. Communication
2. Planning
3. Modelling
4. Construction
5. Deployment

**Generic Process Models**

These are different models you can follow to implement/develop software.

**Linear Process Flow:** Do once, you can’t move back.

Communication - > Planning -> … -> Deployment

**Evolution Process Flow:** At Deployment, it can go back to communication or finish. Wastes time. If there are any errors in your eg Planning, you can only get feedback when you are finished with the project.

Communication - > Planning -> … -> Deployment -> Communication/finish->…

**Iterative Process Flow:** At each step, can go back to previous steps. Room for feedback at each step.

**Parallel Process Flow:** Doing multiple steps at the same time.

**Prescriptive Models**   
These are strategies that apply to certain models.

**Waterfall Model**: You can only go back from the deployment.

This is a very old model. Sequential approach. Works well only if you understand your requirements really well. It is difficult for clients to state all their requirements the first time. Takes long = customers must have lots of patience. Leads to blocking as previous step must be completed before moving onto the next.

This model is good for very robust systems such as medical (critical life systems). Ie You don’t have the luxury to have your client test the system as it can’t result in death. They are also very good for budget systems.

**Prototyping**

Communication -> Quick Planning -> Quick Modelling -> Prototype Construction -> Deployment & Feedback -> Communication

**Incremental Model**

**Spiral Model**

Each Section starts with less time allocated to it, and as everything’s repeated (Deployment back to communication), there is more and more time allocated to it (like a circular spiral, from the outer edge to the center)

This repeats until the project is finished.

**Kanban**: The key thing to success here, is prioritizing tasks that are key features, and leaving things with a high cost associated with it for later if its not very important.