Tutorial 2

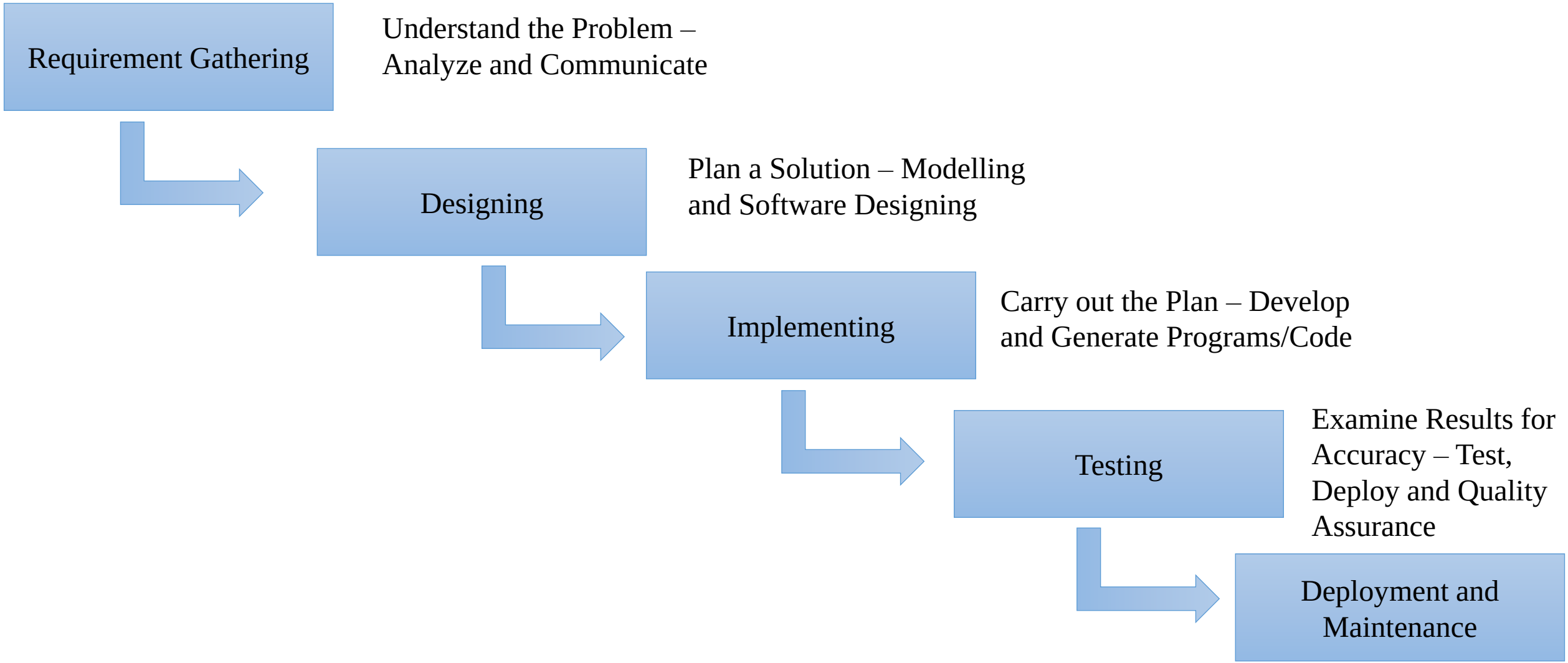
COMP 354- Fall 23

Apekshaba Gohil apekshabagohil.it@gmail.com

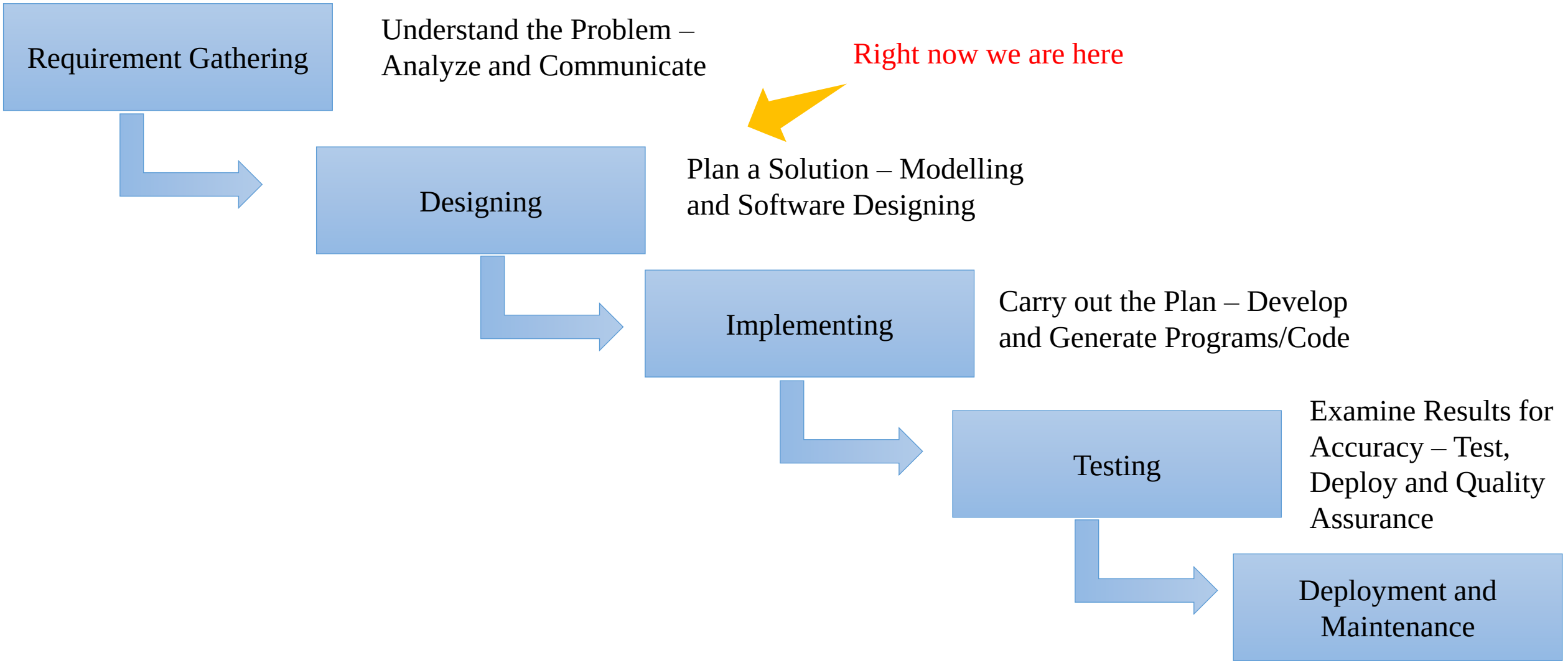
Anam A. Shaikh anams403@gmail.com

Marzieh Adeli marzieh.adeli@gmail.com

Software Engineering Process



Software Engineering Process



2. Software Design

- Software design is a process to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.

- It moves the concentration from problem domain to the solution domain. i.e it tries to specify **How to fulfill the requirements in SRS ?**

- Design concepts must be understood before the mechanics of design practice are applied. Which means we need to **plan** our system design before implementing it.

Fundamental Quality aspects of a Design

1. **The design should be a readable, understandable guide**

    1.1. For those who generate code – Developers
    1.2. For those who test code the software code – Testers
    1.3. For those who maintains the software – Support Team

2. **The design should provide a complete picture of the software**
    - 2.1. Addressing the data
    - 2.2. Functional Domains
    - 2.3. Behavioral domains from an implementation perspective.

3. **The design must contain all of the implicit requirements (Login the System) and the explicit requirements (e.g. Security Requirements → Authorization)**

Design Concepts

1. **Abstraction** – Hiding the unnecessary details from the users. i.e. depict only relevant details.
    1.1. Data Abstraction – Named collection of data describing the data objects.
    1.2. Procedural Abstraction – Named sequence of instructions with specific functions.

2. **Modularization -** Modularization is a technique to divide a software system into multiple discrete and independent modules, which are expected to be capable of carrying out task(s) independently.

Designers tend to design modules such that they can be executed and/or compiled separately and independently.

3. **Concurrency -** Implemented by splitting the software into multiple independent units of execution, like modules and executing them in parallel for better performance.

e.g. The spell check feature in word processor is a module of software, which runs along side the word processor itself.

Design Concepts

4. Coupling - Coupling is a measure that defines the level of inter-dependability among modules of a program.

- Coupling demonstrates what levels the modules interact with each other.
- Less coupling = Better Module.

5. Cohesion - Cohesion is a measure that defines the degree of intra-dependability within elements of a module.

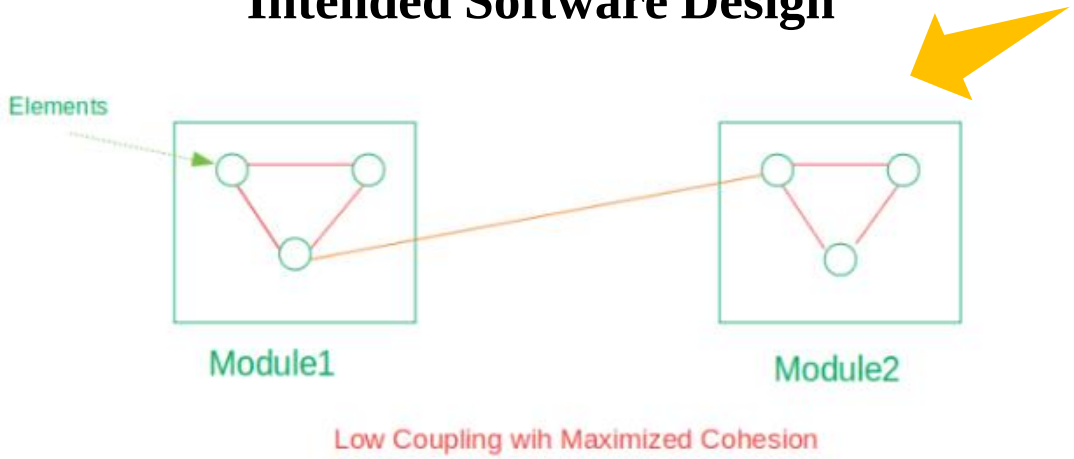
- The greater the cohesion, the better is the program design.

- Some important kinds of Cohesion

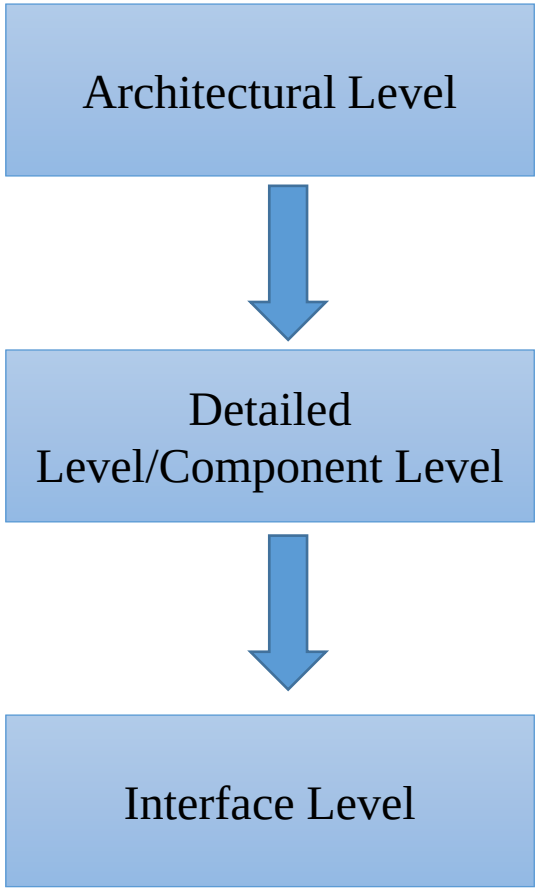
- **Sequential cohesion -** When elements of module are grouped because the output of one element serves as input to another and so on, it is called sequential cohesion.
- **Functional cohesion -** It is considered to be the highest degree of cohesion, and it is highly expected. Elements of module in functional cohesion are grouped because they all contribute to a single well-defined function. It can also be reused.
- **Communicational cohesion -** When elements of module are grouped together, which are executed sequentially and work on same data (information), it is called communicational cohesion.

Coupling vs Cohesion

Intended Software Design



Software Design Levels



Software Architecture

- The architectural design is the highest abstract version of the system. It identifies the software as a system with many components interacting with each other.

- At this level, the designers get the idea of proposed solution domain.

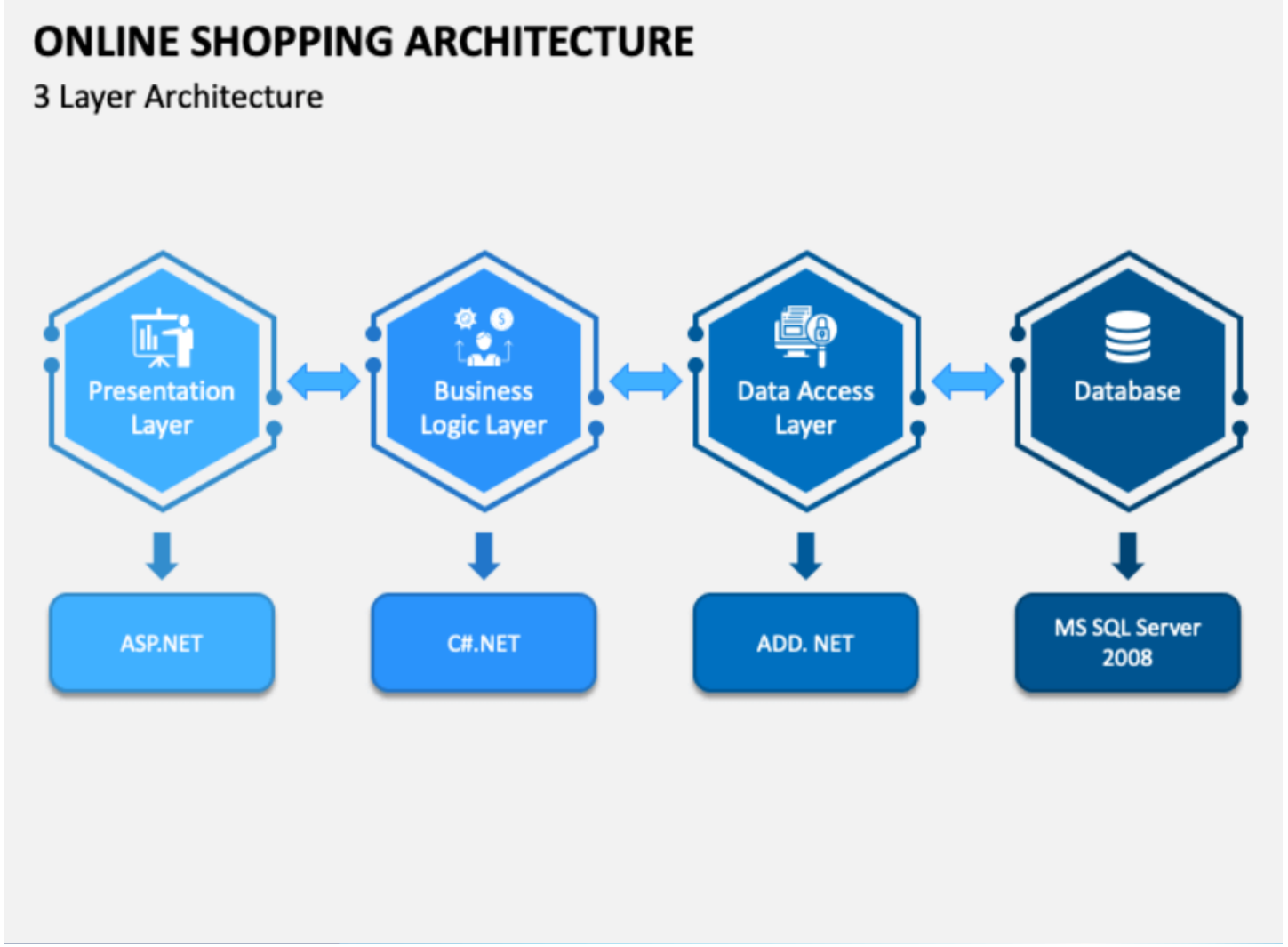
- Diagrams- Class Diagrams, Analysis packages and CRC Models.

Software Architecture

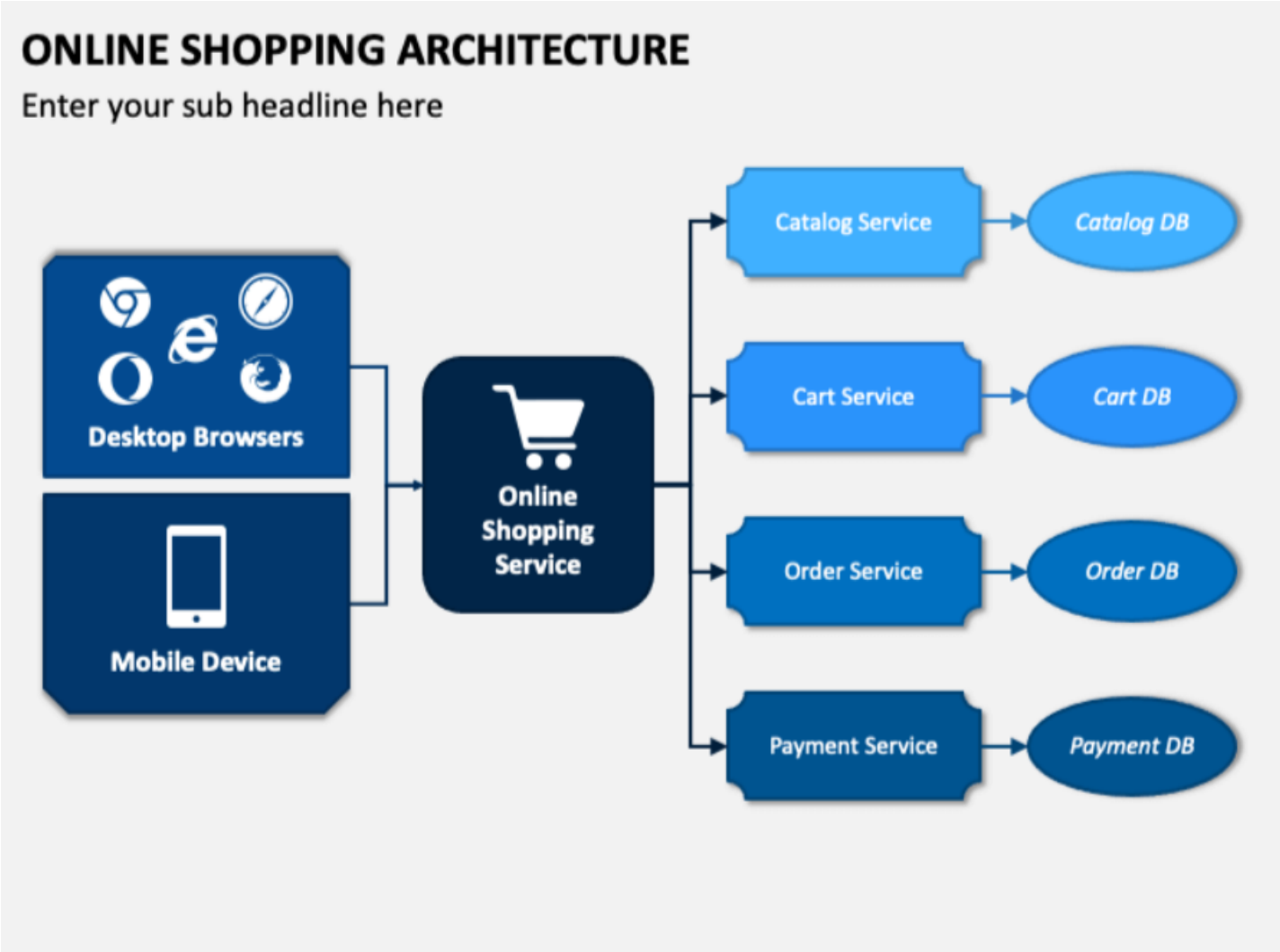
- Types of Software Architecture :

1. Data Centered
2. Data Flow
3. Object Oriented
4. Layered Architecture

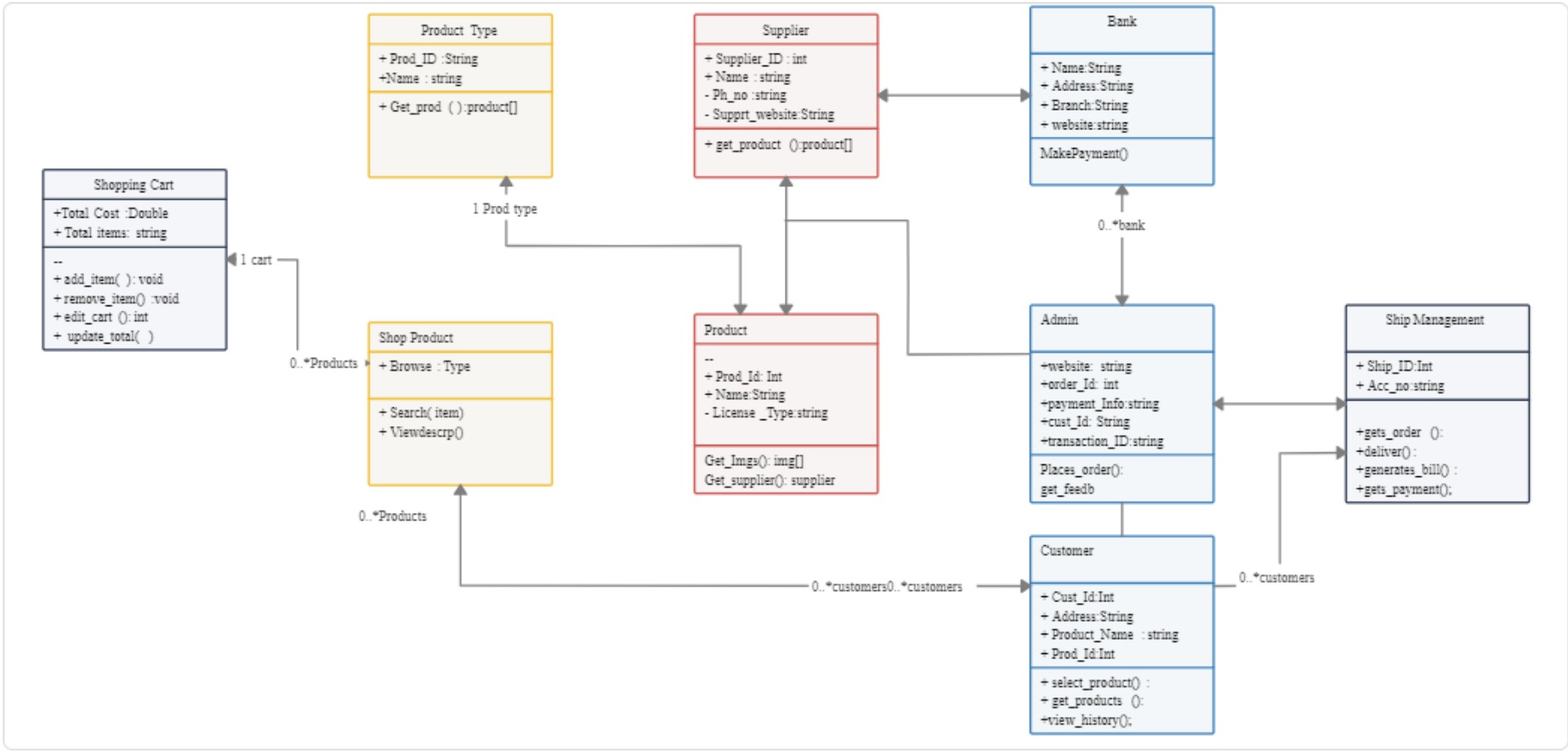
Online Shopping Layered Architecture



Online Shopping Data Flow Architecture



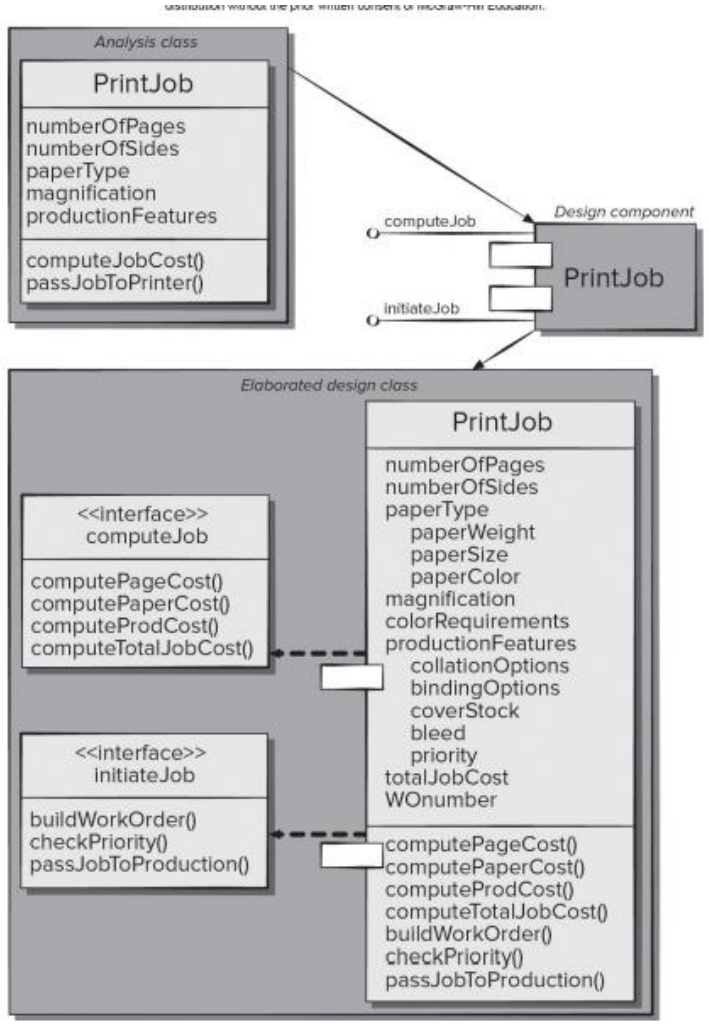
Online Shopping System Architecture using Class Diagrams



Component Level Design

- It is more detailed towards modules and their implementations.
- It defines logical structure of each module and their interfaces to communicate with other modules.
- A component contains processing logic, internal data structures that are required to implement the processing logic, and an interface that enables the component to be invoked and data to be passed to it.
- E.g. Collaborative Classes are Component Level Design.
- Diagrams – Class Diagrams, State Machine Diagram, Activity Diagrams.

Component Level Design



Interface Level Design

- User interface (UI) design is **the process designers use to build interfaces in software or computerized devices, focusing on looks or style**.

- UI design refers to graphical user interfaces and other forms—e.g., voice-controlled interfaces.

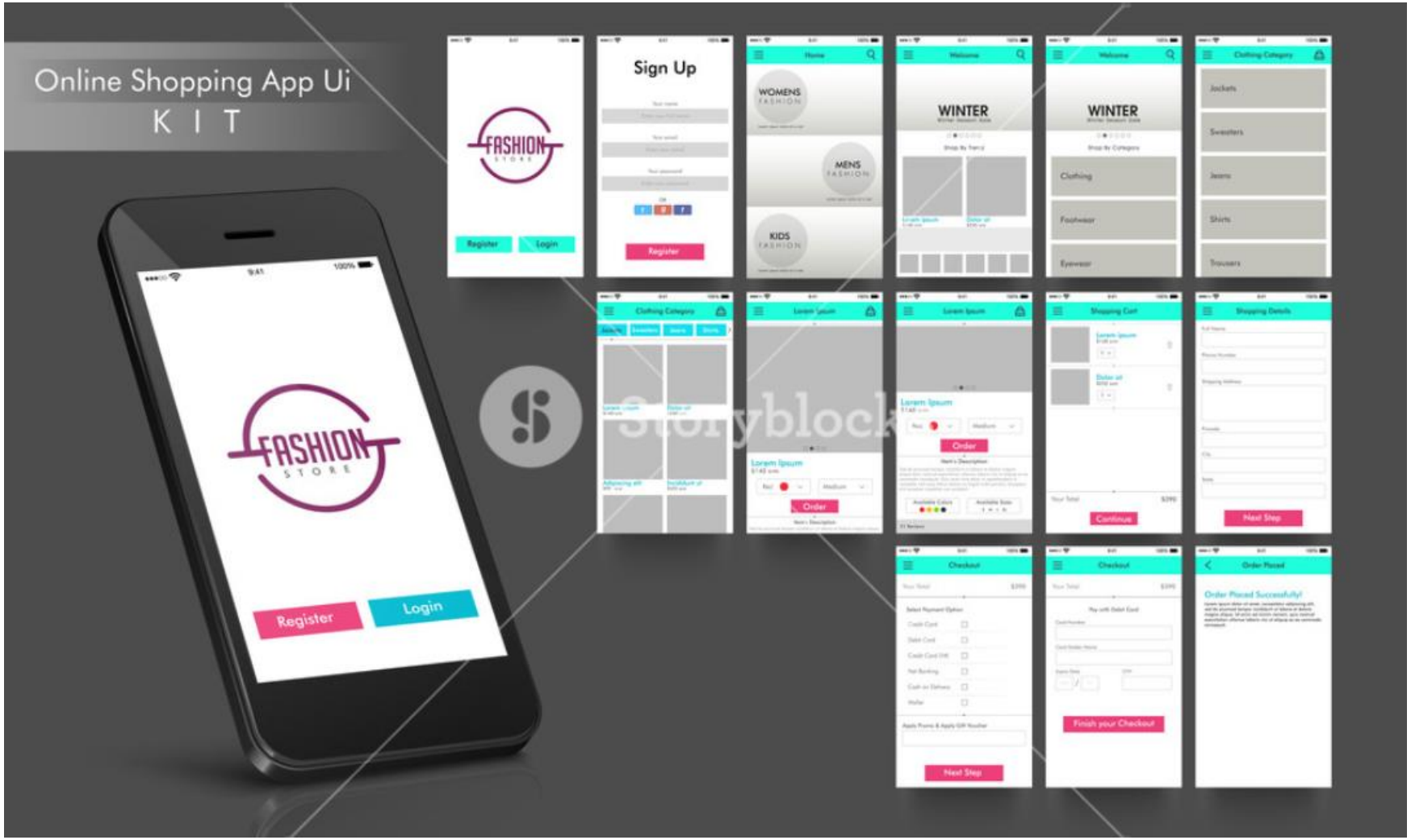
- Designers aim to create interfaces which users find easy to use and pleasurable.

Interface Level Design

- Types of Interface Level Design :

1. Information Design – can be represented in a hierarchical flow diagram
2. Interaction Design – UML Diagrams
3. Visual Design – Wireframes
4. User Experience Design – Actual UI/UX representation of the Software.

Interface Level Design



References

- https://programmerbay.com/difference-between-cohesion-and-coupling/
- https://www.techopedia.com/definition/3843/module
- https://www.sketchbubble.com/en/presentation-online-shopping-architecture.html
- https://www.interaction-design.org/literature/topics/ui-design
- https://www.storyblocks.com/images/stock/online-shopping-user-interface-layout-with-different-creative-screens-for-smartphone-bkmo1oupaxj1gu8hy4