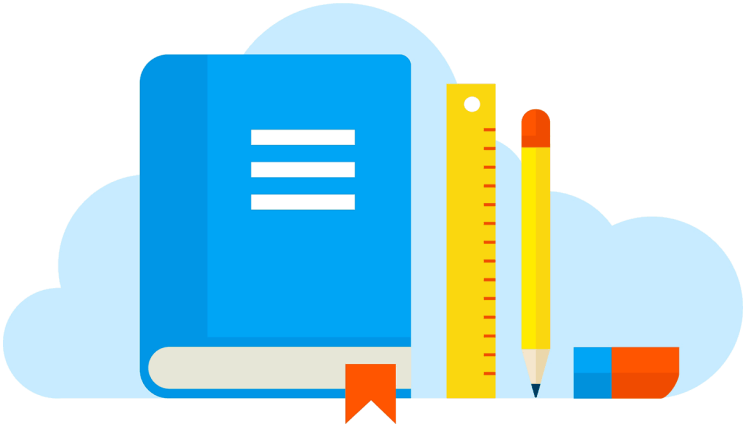

Lec Notes

Date: 06/10/2024

Software architectures & design pattern

- What is architecture – problem - **scale**,
- Recurring style -
- Layers , rules in included in architecture / **(code organize)**
- Software qlty mesher **internal qlty , extranle qlty**
- **exnternal qlty - example - user interface (client)**
- **Internal - code qlty (developer)**
- **Code qlty ekt thama design pattern use krnne**
- **Mvc issues - model eke data access / qrys mix wel thiyenawa**
- **Code repeat wela thiyenawa**
- **New architttecther**
- **Extended arqtec (layer arqtect)**
- **Presentation layer -** user kenek intract wena user interface eka (fxml files genath daganna wa ui) **/**

- **business layer -**ape project eke business logik daganna ek **/**
- **persistence layer -** ape application eke tiyena database eke qry part eka data layer eka database conection eka adala part eka data save krla tiyagnna eka **/**
- **data layer -** software eke database eka aqr krna data store karala thiyagann eka

Rule

- **Loose coupling - ape project wala hama tightly coupling hamak ma ain karala dana eka**

tight coupling mean - mokk hri class ekk behaviros class ekk inrstence ekk directly access krnn innwnn ethana tight coupling ekk kiyala

Ex - `package lk.ijse;`

```
public class boy {


   public void chatwithgirl(){
       girl girl = new girl();


               girl.chat();
   }
}
package lk.ijse;


public class girl {
   public void chat() {
```

```
        System.out.println("chatting");

    }

}
```

- **Dependency injection -** mokkhari class thawa class ekk ekka depend wela iddi e depend wela inna dependency eka ee adala class ekata apply krna eka
- A class —----------
- B class —----------

Dependency apply krgnna akara 4 thiye

- **Property injection** -
- **constructor injection -**
- **Setter method true injection**
- **Interface true injection -**

Meka use krnne eka eka awasthawaladi use krna data eka pawichi krnna welawa anuwa thama godak welawt use wenne

- **High cohesion -** eka single unit ekaka adala dewal ee unit ekema thiyan inna eka

- **Less boilerplate codes -** complex code /duplicate code

Persistence layer = **dAO layer kiyala represent wenne implement krddi**

Date: 06/20/2024

design pattern

- **Software ekk build krddi ena repetitive coe problem ekata laba dena elegant solution ekk thama design pattern ekk kiyanne**
- Singleton - eka istence ekk hadala eka reuse karana eka

Factory design pattern (New design pattern)

- **Object creation logic ekka hide karana eka thama factory design pattern kiyanne**

Facade design pattern

- **Code complexity eka hide krnna / super dao eki crude dao ekei thama tiynne**

Strategy design pattern

- **ape code eke run time behavior ekk**