SOEN 6441 Winter 2023 section WW: Assignment 1

Due: as indicated on Moodle

You can work in pairs; or in a group of 3; no credit will be given for bigger groups or working alone.

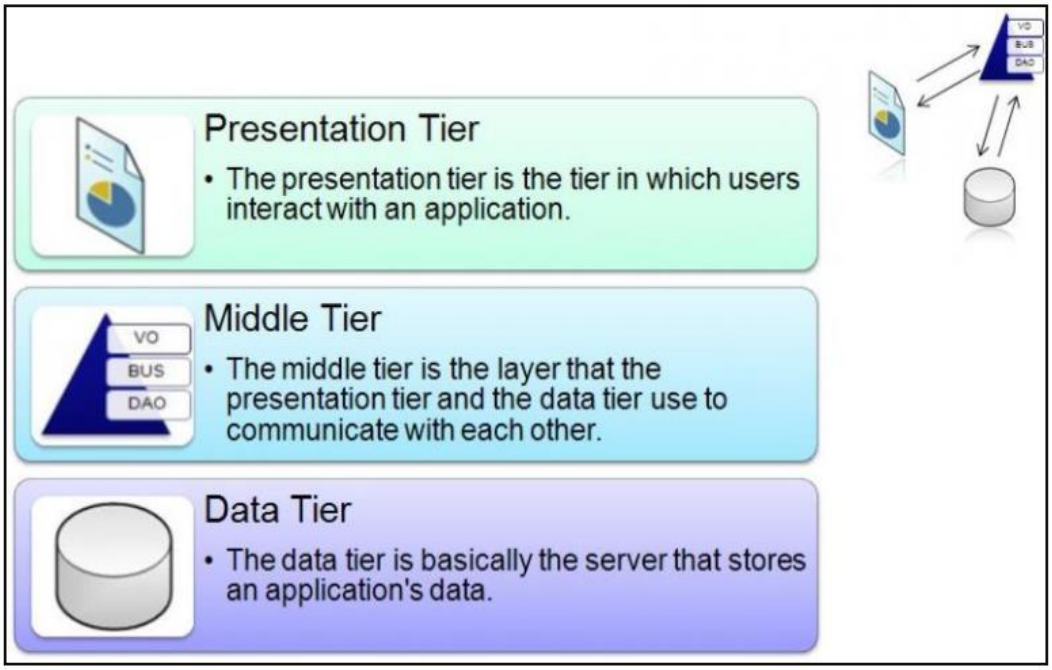
Objectives:

- Unit testing
- 3-tier architecture/ 3-layer design
- Design patterns
- Mock objects

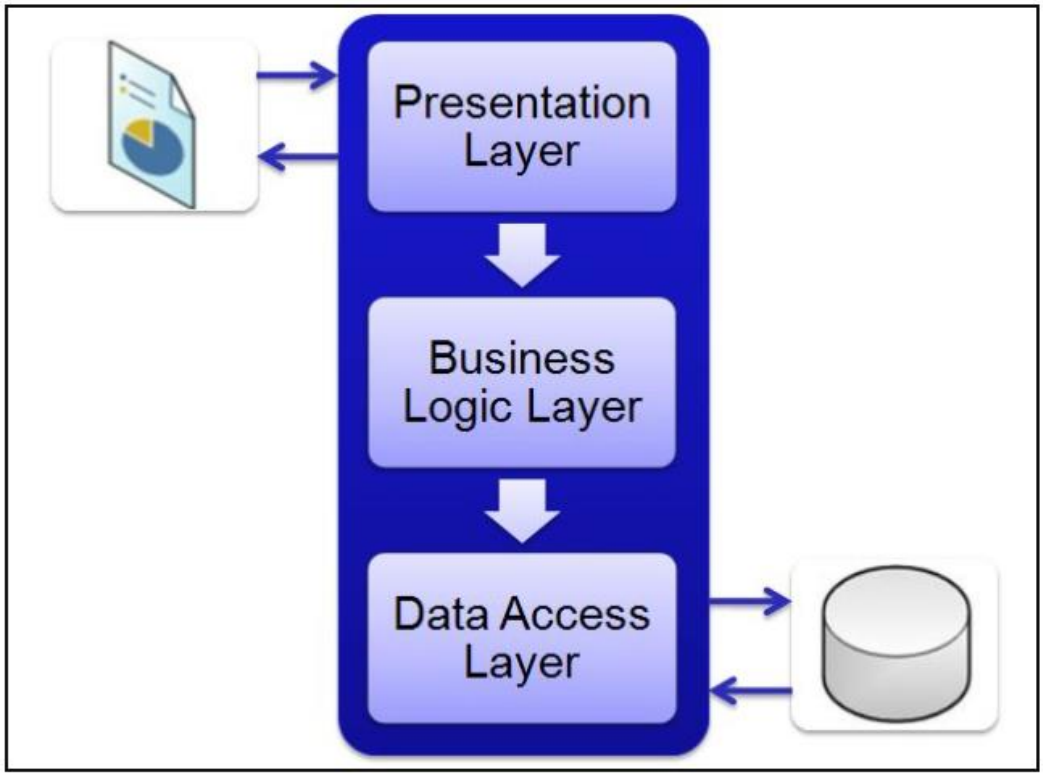
3-tier Architecture

In this project, your design must follow the 3-layer architecture. The 3-layer architecture consists of the following:

1. Presentation layer: that will be your JavaFX user interface
2. Application layer: that is the business logic. Sometimes, it is referred to as the logic layer.
3. The data layer: that is the database layer. In this project we will use a mock object to simulate the DB.

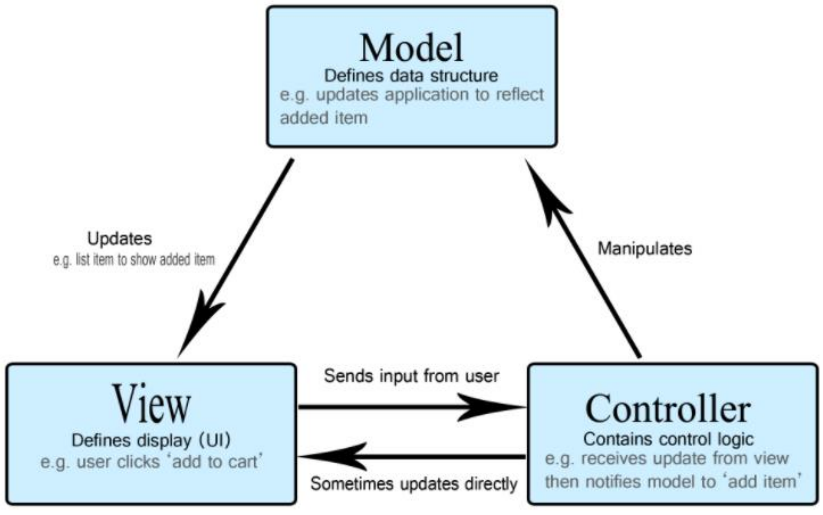


The 3-layer design



MVC architecture

In one of our classes, we discussed MVC where we separated the Model (the business logic) from the View (the user interface). In this project, we must also implement the Controller. The role of the controller is to facilitate the communication between the Model and the View.



Problem definition:

You are given the task to develop a piece of software to manage the rental units of a real estate company. The company owns apartment buildings, condos and houses. All of them are for residential occupancy only. An apartment building has one civic address and many apartments inside it. Each apartment is identified by an apartment number. Each apartment has a number of bedrooms, a number of bathrooms, and square footage. The condo is a privately owned apartment in a high end building. A condo has a street number in addition to the unit number. A house has a street number and no unit number. The other normal fields of address apply to all of them (street name, city, postal code, etc..).

When a potential tenant is interested in a specific rental unit and that rental unit is occupied, the system allows the tenant to register in the system indicating which unit he or she is interested in. Once the company knows that this unit will become available, an e-mail is sent to all potential tenants interested in that unit.

You need to create a centralized accounting system to be used by the company's administration and also the existing tenants. The accounting system should keep track of who has paid the rent and who has not.

A lease is the contract between the tenant and the company. It has the tenant info, the info of the unit rented to the tenant, lease start date, lease end date, rent amount.

A forum is created on Moodle to link you to the customer you are developing this app for. You can communicate with the customer using this forum. Any extra information added to the forum by the customer is considered as new requirements/clarification to this document. You can not communicate with the customer using any other means.

Requirements:

- Must use appropriate design pattern whenever appropriate.
- A full design in the form of a UML diagram must be submitted and it should be detailed enough to show your use of design patterns.
- Create proper inheritance hierarchies.
- Use polymorphism whenever appropriate.
- At this phase the model should be fully implemented.
- No GUI user interface is implemented at this phase. The output should be on the command prompt.
- Implement unit tests (minimum 10).
- The system displays the following menu:
  1. Add a property
  2. Add a tenant
  3. Rent a unit
  4. Display properties
  5. Display tenants
  6. Display rented units
  7. Display vacant units

8. Display all leases
9. Exit

Submission:

You must submit:

- A proper **detailed UML class diagram** in **pdf format** that shows
  * names and access modifiers of all attributes,
  * names, parameters, and access modifiers of all methods including constructors
  * proper relations between classes and interfaces
  * the design patterns you used
- **Sample runs** of the program that show all the functionalities required in **pdf format**
- **The code** itself

You will give a demo of your assignment to the TA. All group members must be present during the demo. No credit will be given without a demo. Any student should be able to answer questions about any part of the project.

Grading Rubric

UML diagram	20 marks	Proper notation / overall design
Proper use of inheritance	10 marks	
Proper use of polymorphism	10 marks	
Proper use of design patterns	35 marks	
Unit tests	15 mark	
Overall code quality	10 marks	Indentation/ comments / method names / variable names / class names

0 credit will be given if:

1- No UML is submitted. It is a must to submit the detailed UML class diagram OR
2- The code does not compile OR
3- No demo

One submission per group please. In the Moodle submission comment, please indicate the names and the ID's of the team members worked on the project. -2 if this information is not given accurately.