Bilkent University
Spring 2022-2023

CS319

Object-Oriented Software Engineering

The Take Home Assignment

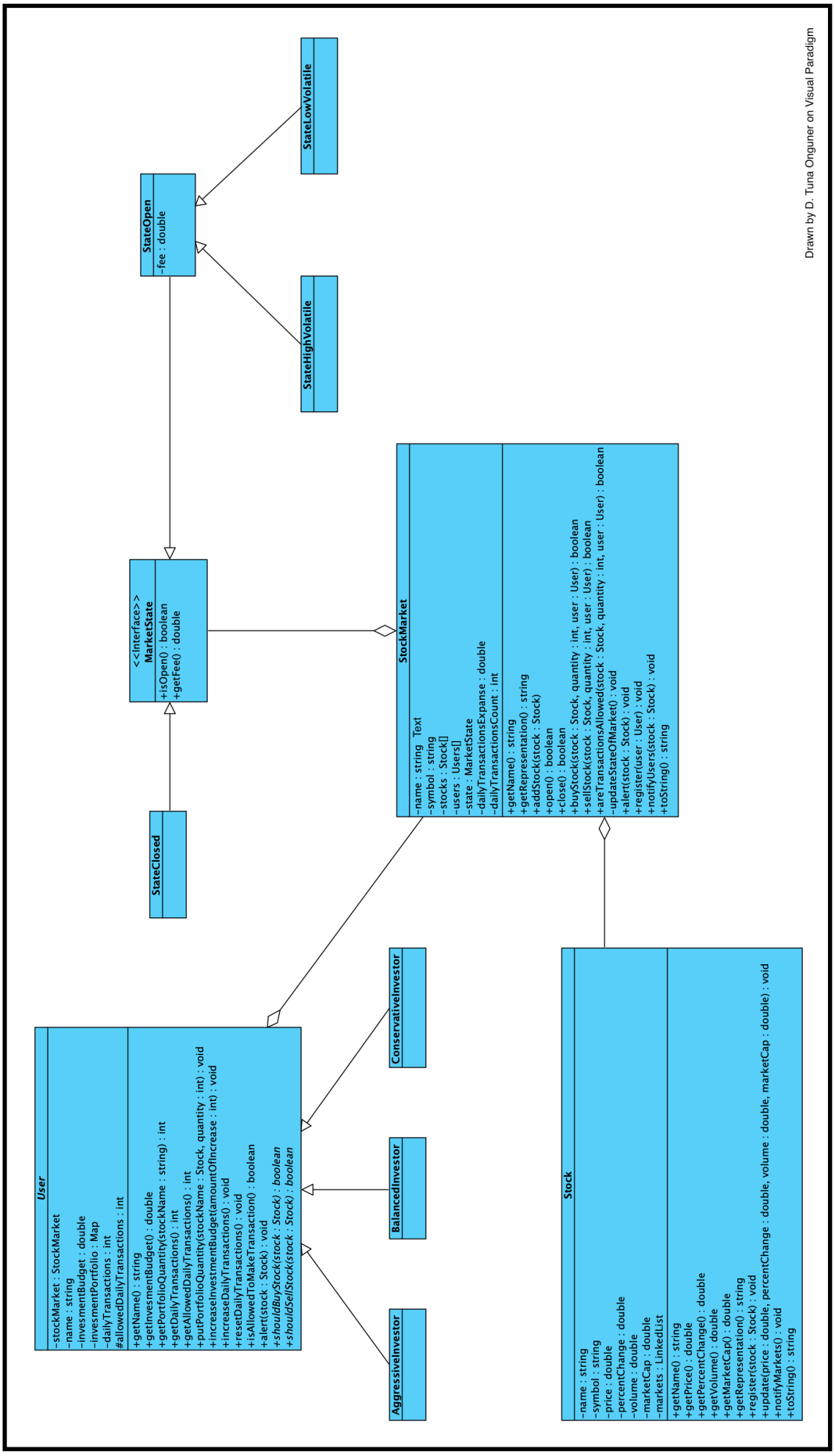
Instructor: Eray Tüzün

Deniz Tuna Onguner
22001788

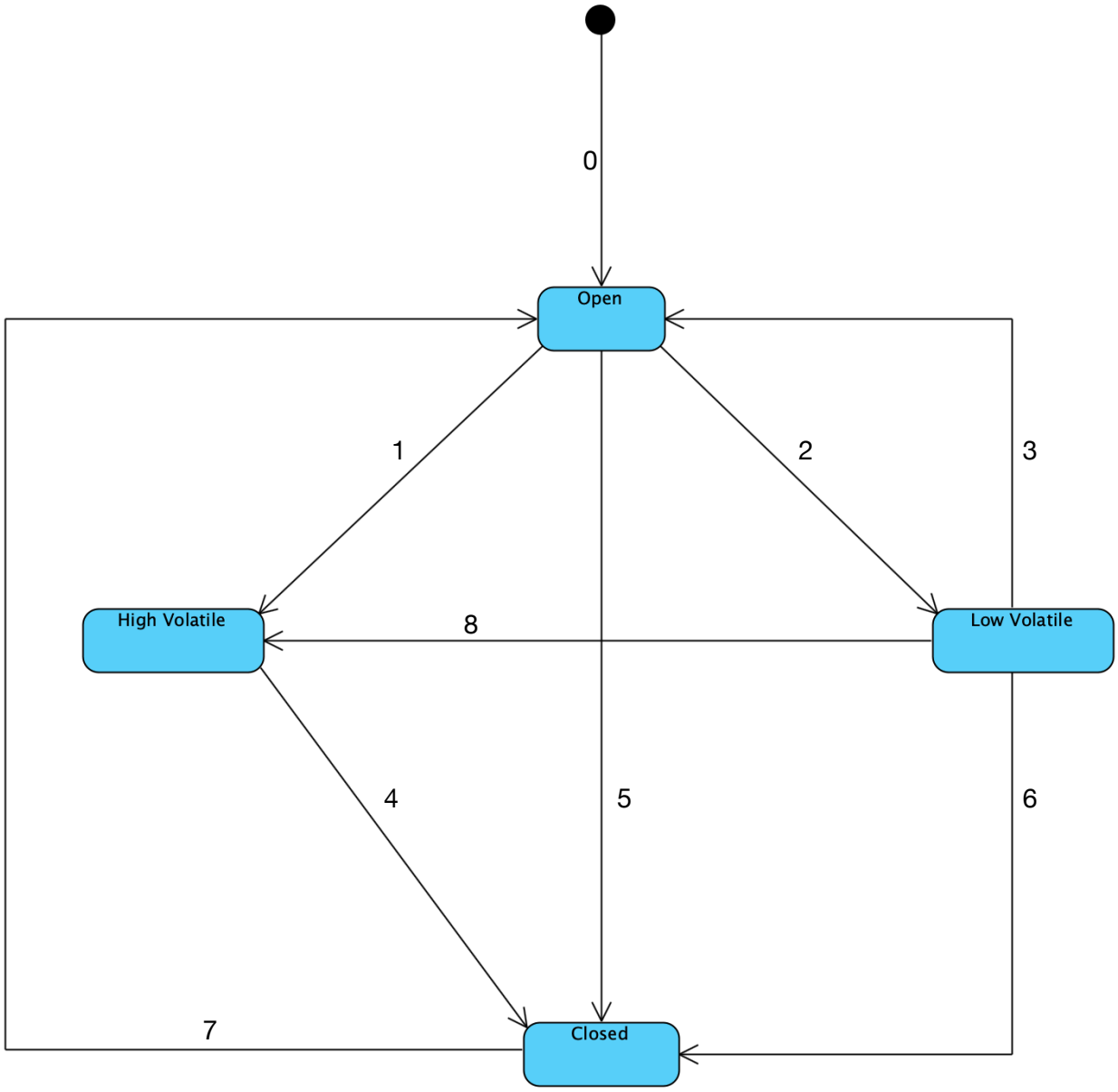
May 17, Wednesday

CONTENT

1. CLASS DIAGRAM



2. STATE DIAGRAM



0: Market is opening.
1: Market moves to the high volatile state.
2: Market moves to the low volatile state.
3: Market moves back to the open state from low volatile state.
4: Market is getting closed.
5: Market is getting closed.
6: Market is getting closed.
7: Market is getting opened again.
8: Market moves to the hight volatile state from low volatile state.

3. DESIGN PATTERNS

Used Design Patterns: Observer & State

3.1. Usage of State Design Pattern

The State design pattern is a behavioral design pattern that allows an object to alter its behavior when its internal state changes. It enables an object to appear as if it changes its class when its state changes, allowing for cleaner and more maintainable code. The pattern provides benefits such as better separation of concerns, improved extensibility, and easier maintenance. It allows adding new states and state-specific behavior without modifying existing code, promoting a flexible and modular design [1].

Since there are 4 states/cases as Open, Closed, High Volatile, and Low Volatile that Stock Market can be in, it is a good and effective choice to implement the program by applying State Design Pattern. So that, in the program, an interface MarketState is defined, then states Open and Closed implements this MarketState; and HighVolatile and LowVolatile extends from the Open.

3.2. Usage of Observer Design Pattern

The Observer design pattern is a behavioral design pattern that establishes a one-to-many dependency between objects. In this pattern, one object, known as the subject or observable, maintains a list of dependents, known as observers, and automatically notifies them of any state changes, ensuring that they stay up to date [2].

Observer design pattern is a good choice to prefer while implementing Stock, StockMarket, and Users. StockMarket will observer Stocks, and Users will observe StockMarket. When a Stock is updated, StockMarket will be alerted, and it will alert Users in itself. So that, unless a Stock is updated, there will not be additional overhead to keep tracking.

4. REFERENCES

[1] “State,” Refactoring.Guru, https://refactoring.guru/design-patterns/state (accessed May 17, 2023).

[2] “Observer,” Refactoring.Guru, https://refactoring.guru/design-patterns/observer (accessed May 17, 2023).