**Design Patterns**

Tutor Application

Kleition Bendo, Hannah Yelle, Calvin Mueller, Chase Pisone, Hunter McDaniels-Rossiter

**Table of Contents:**

**UML Classes:**

User

Student Profile

Tutor Profile

DB\_Mgr

Tutor Application

Tutor Finder

Calendar

Date

Time

Tutor Pair

Login Form  
 Search

Profile Manager

Bank System

Bank Account

Make Payment

**Models:**

Create User Profile

Search Tutor

Calendar & Compare Schedules

Log In

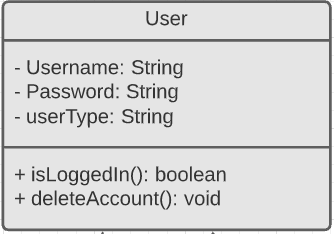
Students with similar tutors/subjects

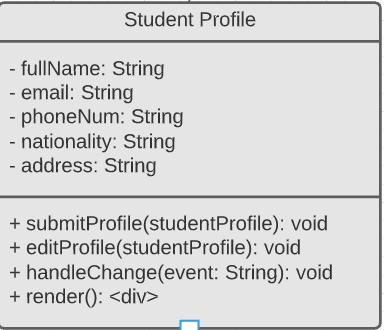
Direct Deposit Set up & Process User Payment

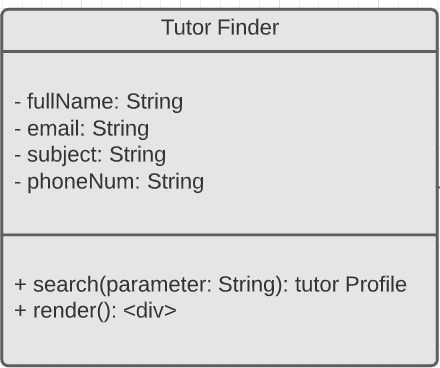
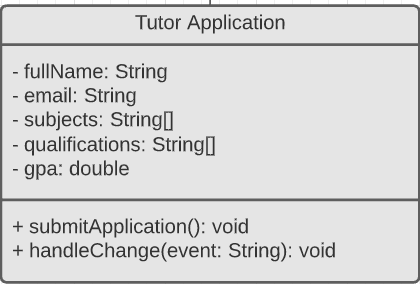
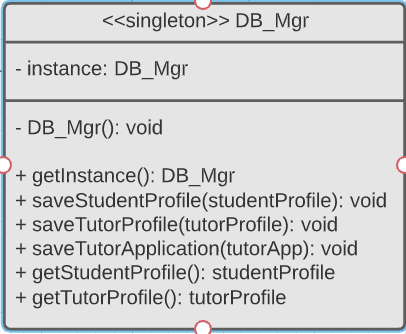
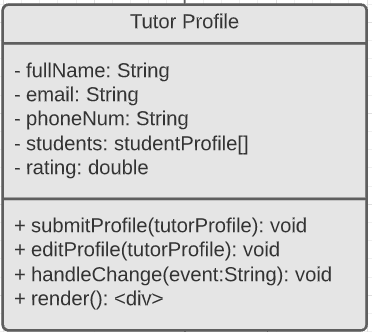
**Design Patterns:**

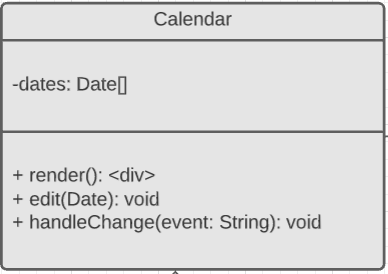
Singleton (DB\_Mgr)

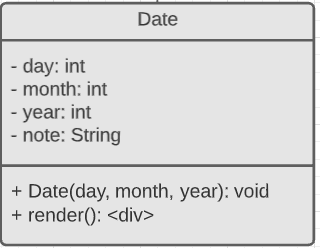
**UML Classes**

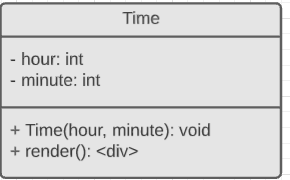
****

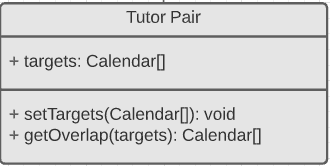
****

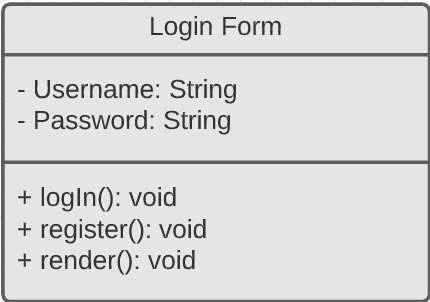
****

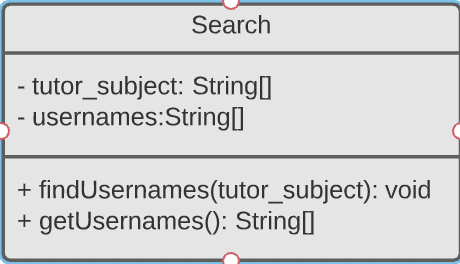
****

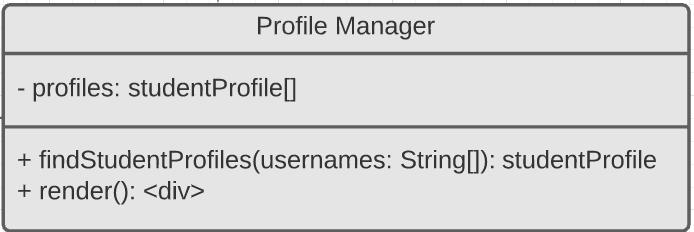
****

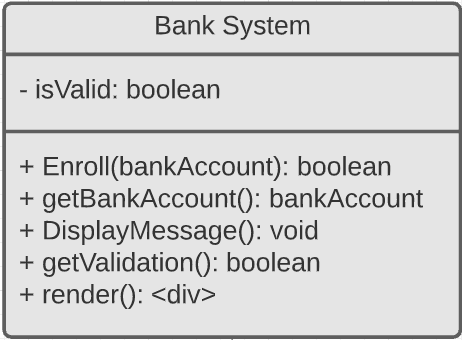
****

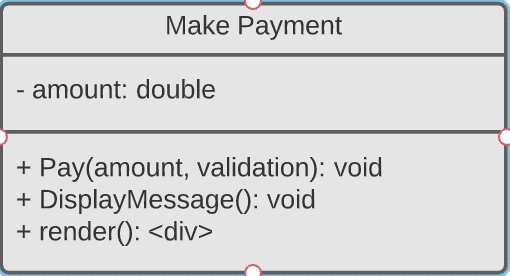
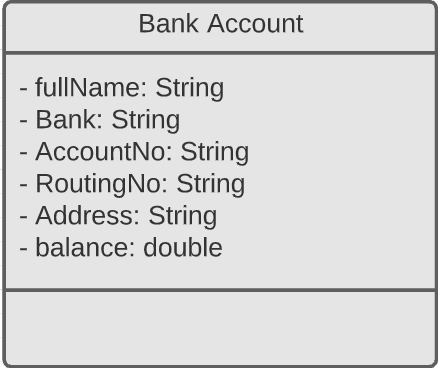
****

****

****

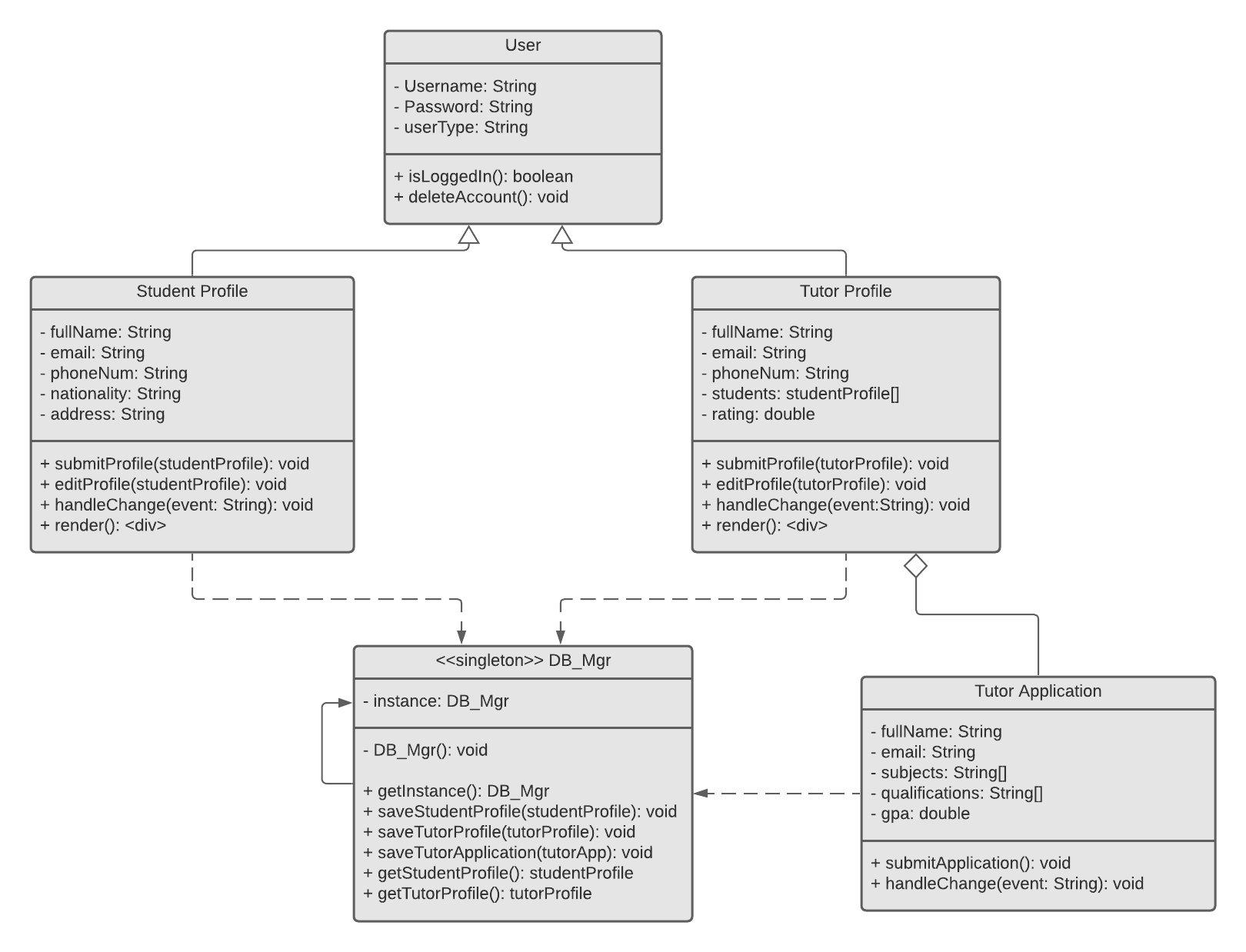
****

****

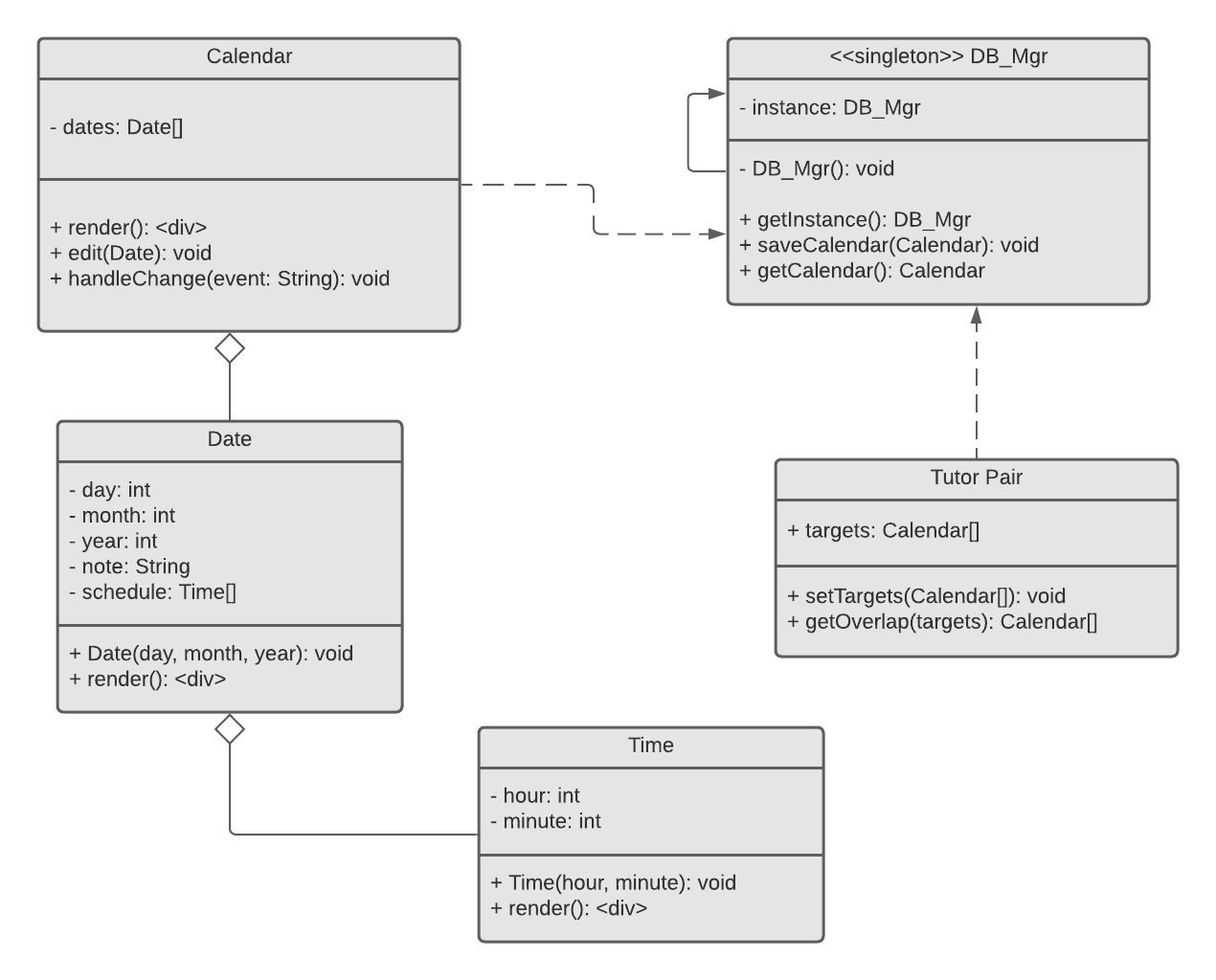
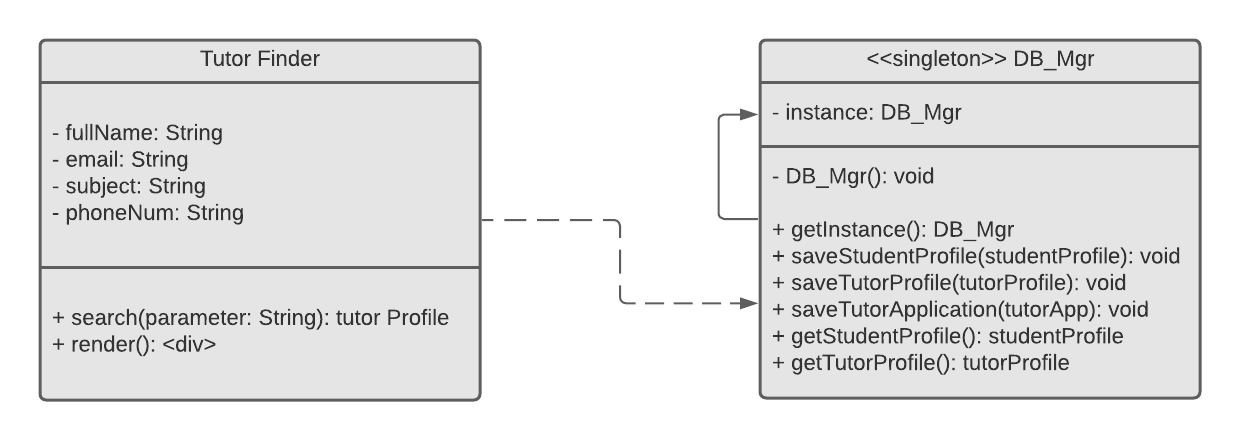
****

**Models**

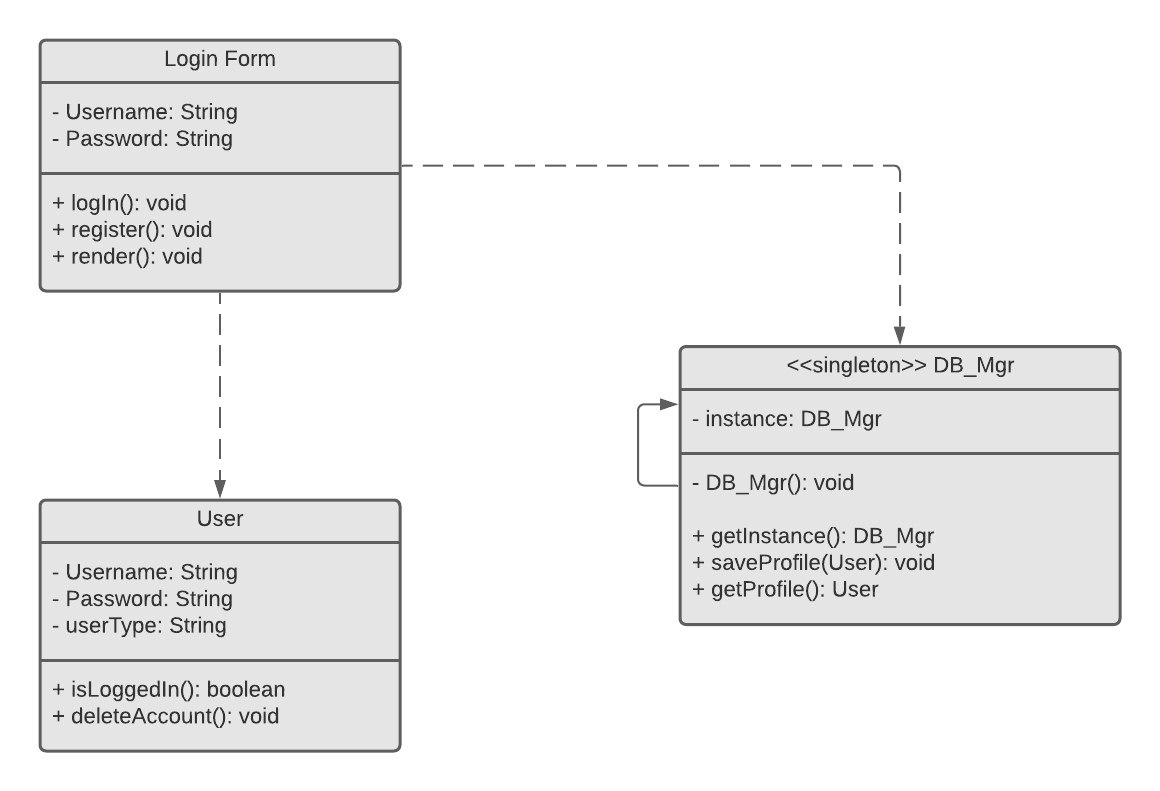
Create User Profile



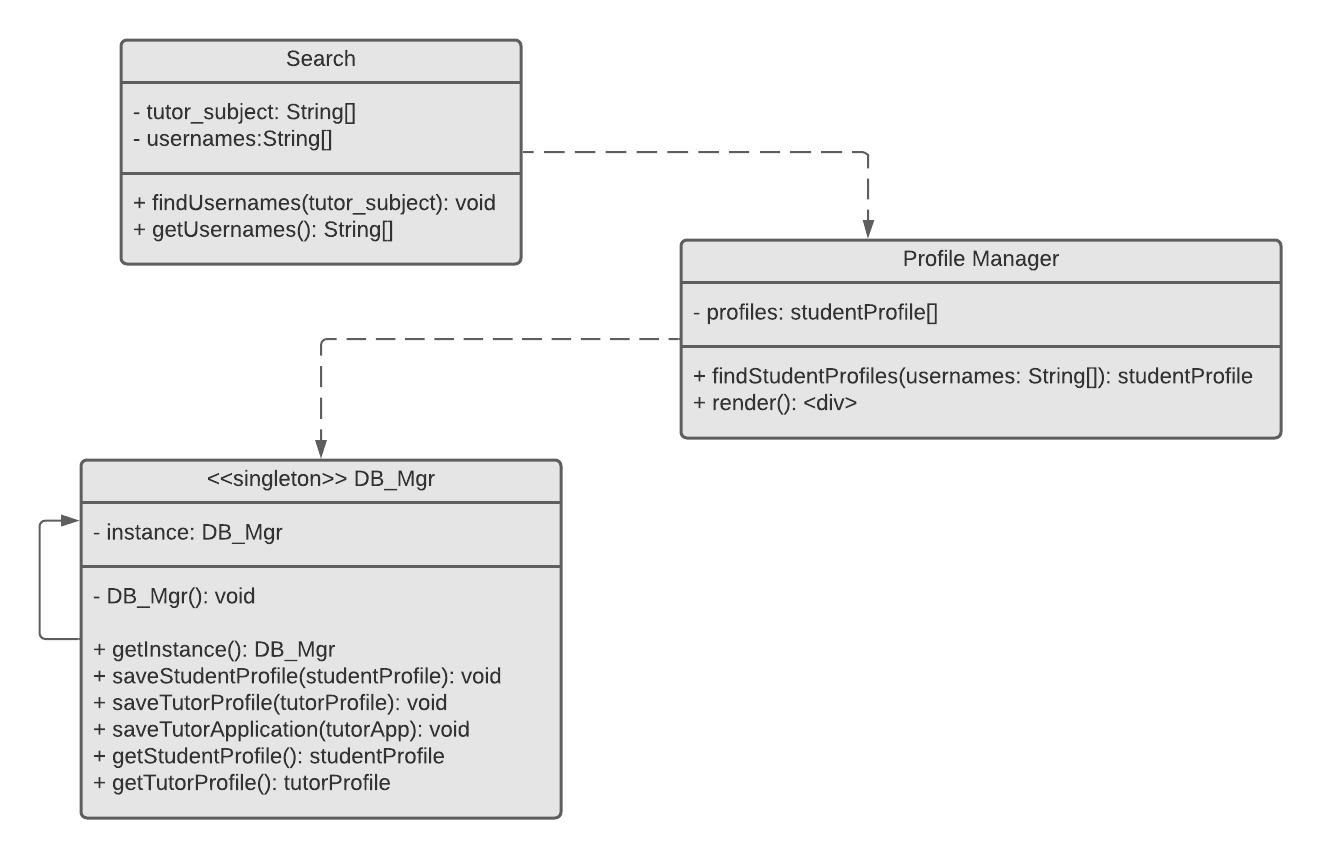
Search Tutor

Calendar & Compare Schedules

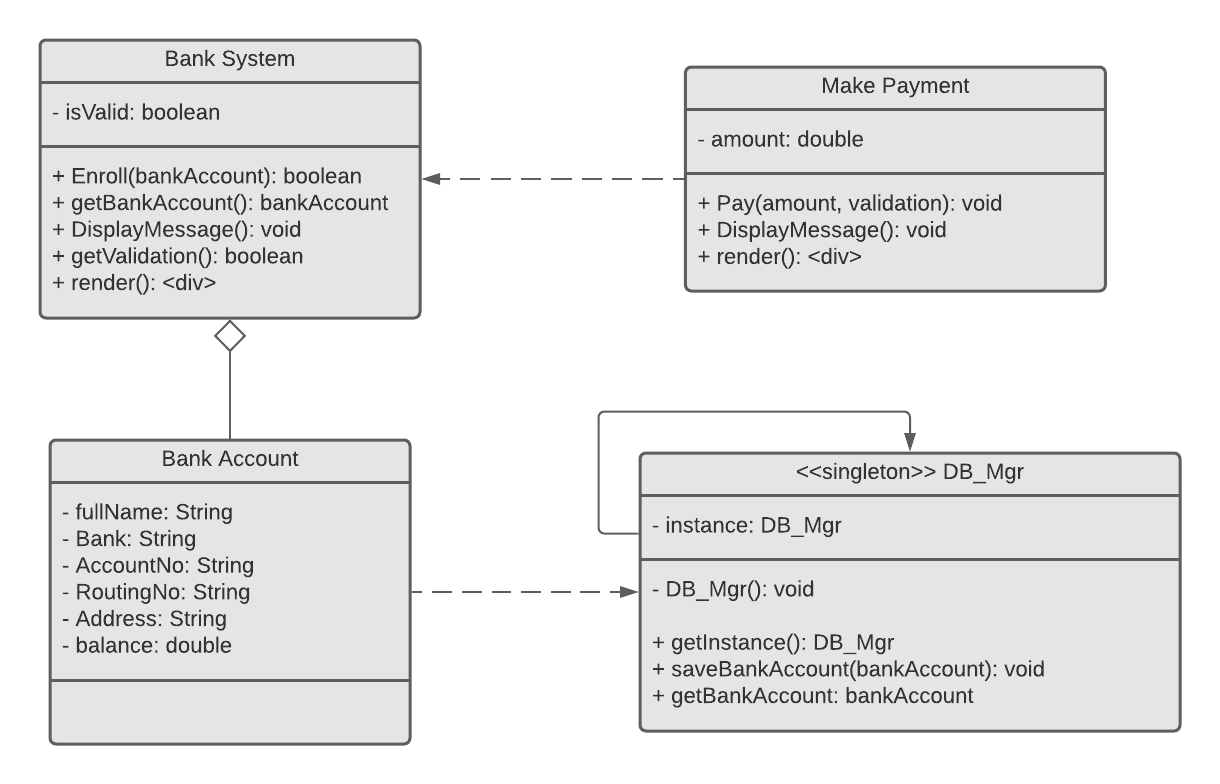
Log In



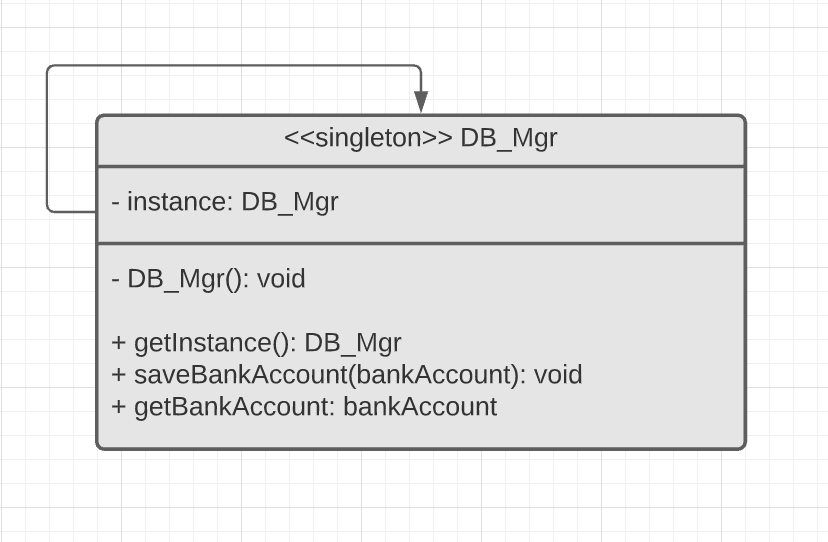
Students with similar tutor/subjects



Direct Deposit Set up & Process User Payment



**Design Patterns**

****

* A singleton design pattern is used when we want to ensure that exactly one object is needed to coordinate actions across the system.

In the case of a Database Manager class, it is useful to restrict the instantiation of a class to one single instance, because there is no need for multiple objects of the class to be declared at any point in time. All the classes that are associated to the DB\_Mgr class will only need one instance to coordinate actions across the Tutor App. To be able to implement a singleton pattern, one could declare all constructors of the class to be private, and provide a public static method that returns a reference to the instance.