CS319 D5

Pool

T1-havuz

Abdurrahman Bilal Kar 22003569

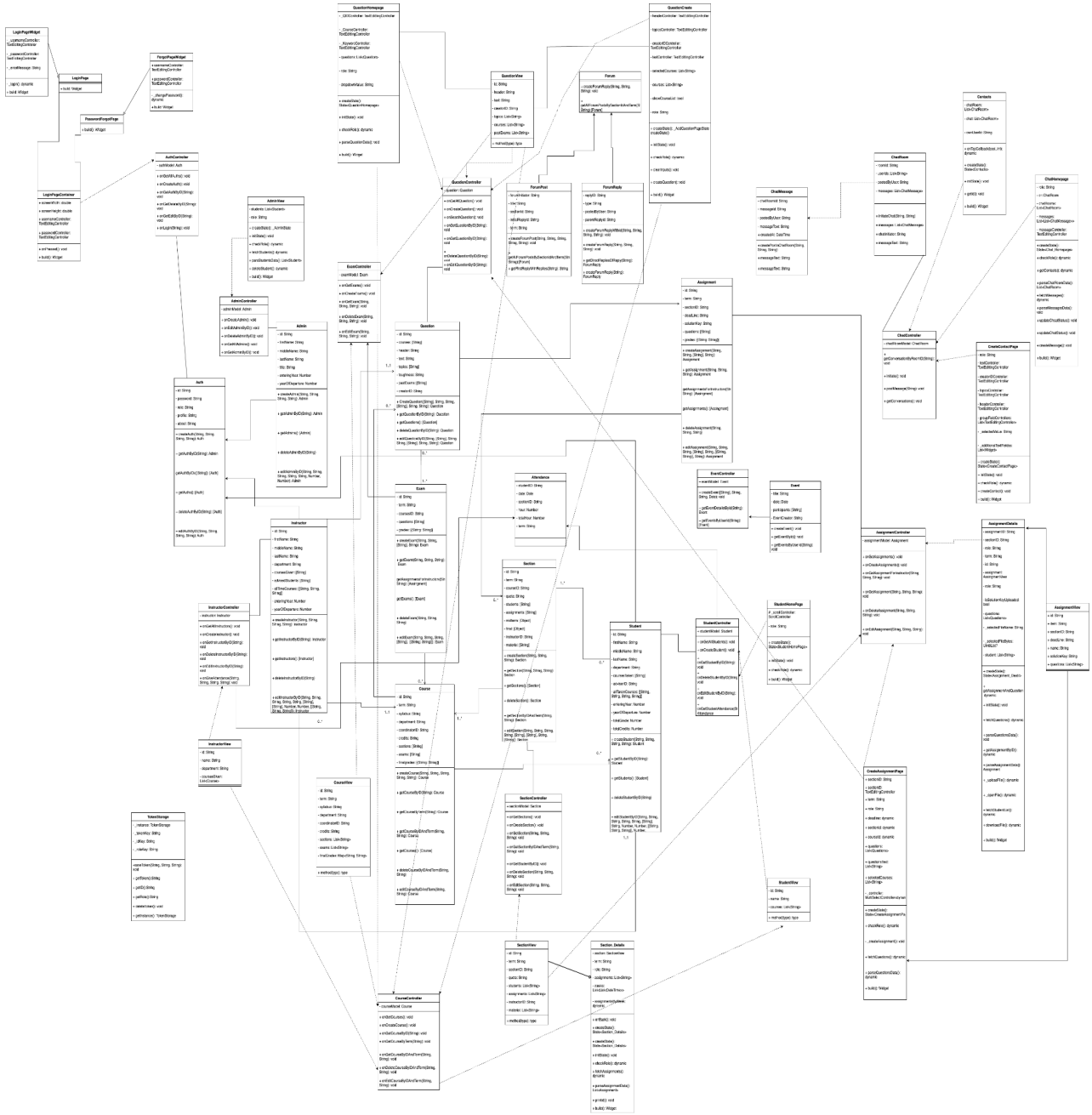
Ahmet Memduh Tutuş 22003153

Berkay Ayçiçek 22003111

Fırat Utku Gül 22003105

Arda Kırcı 22002031

1- Updated Class Diagram



High resolution format can be found in the Docs (the same directory with D5) as class.png

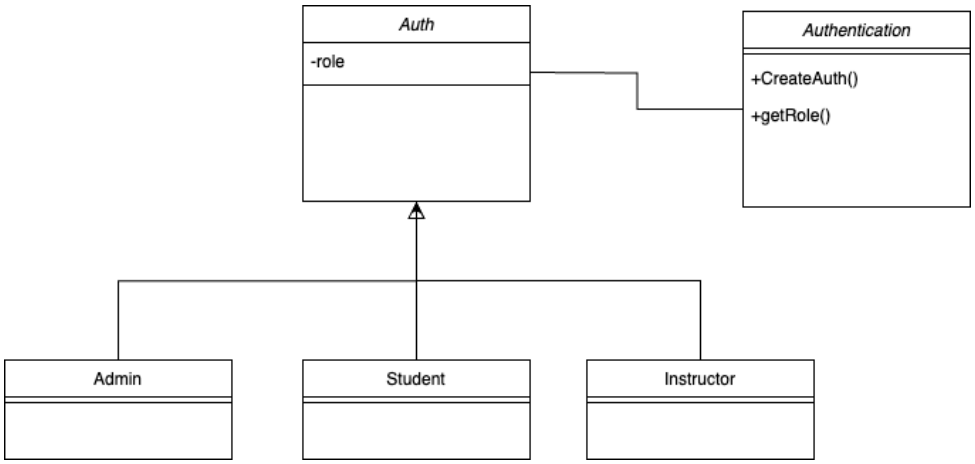
2- Design patterns

a-) **Factory**

In order to fulfill the functionalities in our project, the Factory Design Pattern is applied. Using this pattern, interfaces are used for generating instances of classes without specifying their concrete class names. It defines a method that its subclasses must implement to create objects. In our authentication system, Authentication acts as a factory to create instances of different types of authentication (Admin, Student, Instructor), based on the roles provided. The Factory design pattern within our authentication system allows for easy creation of authentication instances while hiding the way these objects are created from client code. This means that when we use this kind of design pattern we can easily reuse code and separate concerns by keeping the object creation process centralized.

Structure:

Detailed info can be seen from the updated class diagram.



b-) **Singleton**

A single instance of the TokenStorage class is retrieved by implementing the singleton pattern using a static method called getInstance(). Implementing Singleton Pattern makes sure that there's one instance of TokenStorage throughout the application. In our project, Static _instance is an instance variable. The Singleton class instance is globally accessible such that every part of the program can refer to it. Singletons are often used when we need centralized management for resources or configurations like database connections, thread pools, caching mechanisms, logging systems etc. However our token logic uses Singletons permitting users to utilize features while moving through pages.

Structure:

Detailed info can be seen from the updated class diagram.