ارزیابی طراحی بر اساس اصولSOLID

(سامانه آموزش آنلاین شاد)

ارزیابی طراحی سیستم بر اساس اصولSOLID

در طراحی سامانه آموزش آنلاین «شاد»، تلاش شده است اصول طراحی شیءگرا به‌ویژه اصول SOLID به‌عنوان مبنای اصلی معماری نرم‌افزار رعایت گردد. رعایت این اصول موجب افزایش انعطاف‌پذیری، توسعه‌پذیری، قابلیت نگهداری و کاهش وابستگی بین ماژول‌ها در سیستم می‌شود. در ادامه، هر یک از اصول SOLID در چارچوب این سامانه بررسی و ارزیابی می‌گردد.

اصل(Single Responsibility Principle (SRP

اصل مسئولیت واحد

بر اساس این اصل، هر کلاس یا ماژول باید تنها یک مسئولیت مشخص داشته باشد و تغییرات در آن تنها به یک دلیل صورت گیرد.

در سامانه شاد:

سیستم به ماژول‌های مستقل تقسیم شده است:

ماژول مدیریت کاربران(User Management)
ماژول کلاس‌های آموزشی(Classroom)
ماژول تکالیف(Assignments)
ماژول آزمون‌ها(Exams)
ماژول پیام‌رسان(Chat)

ارزیابی:

این تفکیک باعث شده هر بخش مستقل توسعه یابد و تغییر در یک بخش (مثلاً آزمون‌ها) تأثیری بر سایر بخش‌ها نداشته باشد.
در نسخه‌های اولیه، برخی سرویس‌ها مانند «داشبورد دانش‌آموز» چند مسئولیت (نمایش + دریافت داده + مدیریت وضعیت) را همزمان داشتند که نیازمند تفکیک به سرویس‌های کوچک‌تر است.

Open/Closed Principle (OCP)

اصل باز/بسته
نرم‌افزار باید برای توسعه باز و برای تغییر بسته باشد.

در سامانه شاد:

سیستم آزمون‌ها و تکالیف به صورت ماژولار طراحی شده‌اند:

*امکان افزودن نوع سؤال جدید (تستی، تشریحی، آپلود فایل)
*بدون تغییر در ساختار اصلی سیستم

مثال:
در بخش آزمون، می‌توان انواع جدید سؤال را از طریق توسعه کلاس QuestionType اضافه کرد بدون تغییر در منطق اصلی آزمون.

ارزیابی:
ساختار سیستم تا حد زیادی با OCP سازگار است و قابلیت توسعه خوبی دارد.

Liskov Substitution Principle (LSP)

اصل جایگزینی لیسکوف

زیرکلاس‌ها باید بتوانند بدون ایجاد خطا جایگزین کلاس‌های پایه شوند.

در سامانه شاد:

ساختار کاربران به صورت ارث‌بری طراحی شده است:

User (کلاس پایه)

Student

Teacher

Admin

ارزیابی:
تمام زیرکلاس‌ها می‌توانند بدون ایجاد اختلال در سیستم، جایگزین کلاس پایه شوند و رفتار مورد انتظار را حفظ کنند.

نکته:
در برخی عملیات (مثل دسترسی به نمره‌ها)، نیاز به بررسی نقش کاربر به صورت runtime وجود دارد که در صورت طراحی ضعیف می‌تواند نقض LSP ایجاد کند.

Interface Segregation Principle (ISP)

اصل تفکیک رابطها

هیچ کاربری نباید مجبور به پیاده‌سازی متدهایی شود که به آن نیازی ندارد.

در سامانه شاد:

رابطها (Interfaces) به صورت جداگانه تعریف شده‌اند:

IAuthService (ورود و ثبت‌نام)

IClassService (مدیریت کلاس)

IExamService (آزمون)

IChatService (پیام‌رسان)

ارزیابی:
تفکیک رابطها باعث شده معلمان، دانش‌آموزان و مدیران فقط سرویس‌های مرتبط با نقش خود را استفاده کنند.

Dependency Inversion Principle (DIP)

اصل معکوس‌سازی وابستگی

ماژول‌های سطح بالا نباید به ماژول‌های سطح پایین وابسته باشند، بلکه هر دو باید به abstraction وابسته باشند.

در سامانه شاد:

*استفاده از API Service Layer
*ارتباط فرانت‌اند با بک‌اند از طریق interface (REST API)
*استفاده از dependency injection در سرویس‌ها

مثال:

Dashboard به جای اتصال مستقیم به دیتابیس، از طریق:
ClassService → API → Database

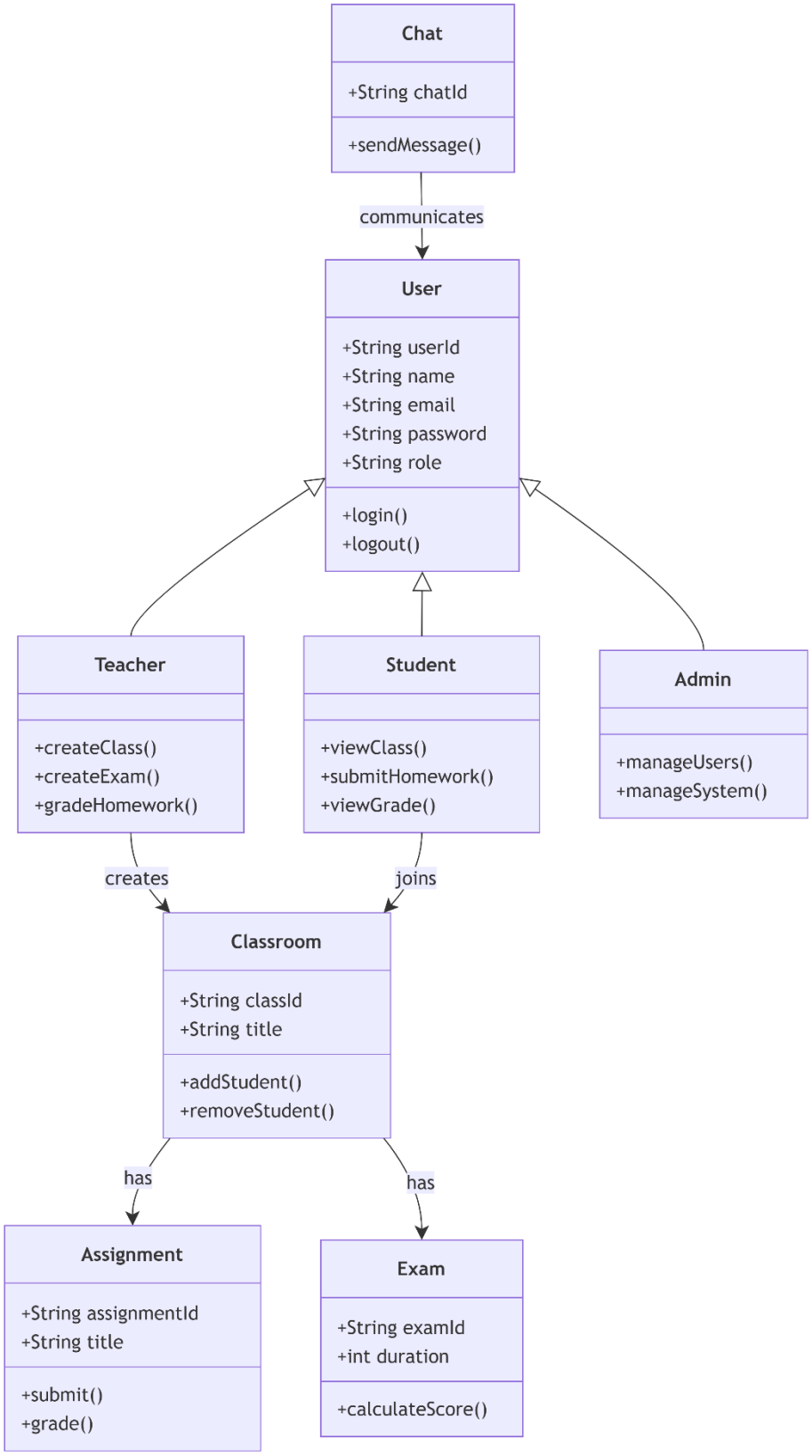
ارزیابی:
این ساختار باعث شده تغییر در دیتابیس یا بک‌اند تأثیری بر فرانت‌اند نداشته باشد.

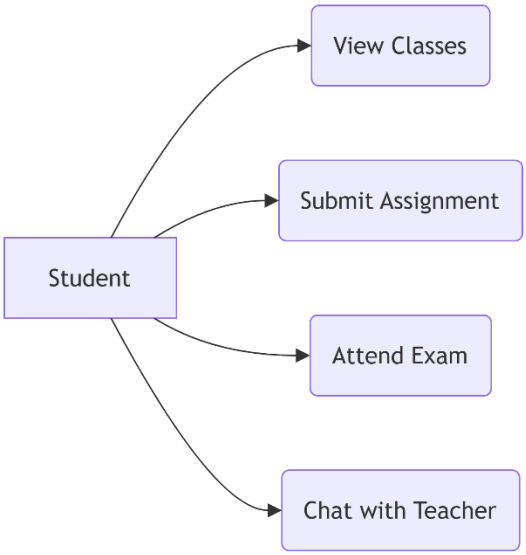
جمع‌بندی کلی ارزیابی SOLID

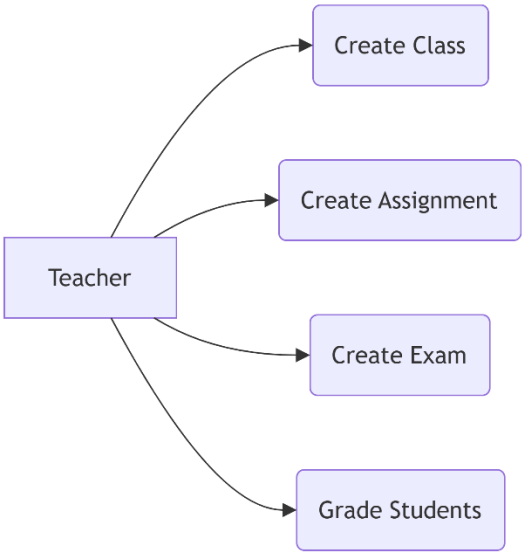
در مجموع، طراحی سامانه آموزش آنلاین «شاد» تا حد زیادی منطبق بر اصول SOLID می‌باشد و این موضوع باعث شده است:

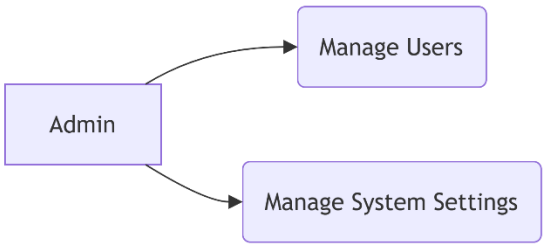
- افزایش مقیاس‌پذیری سیستم
- کاهش وابستگی بین ماژول‌ها
- افزایش قابلیت تست‌پذیری
- بهبود نگهداری و توسعه آینده

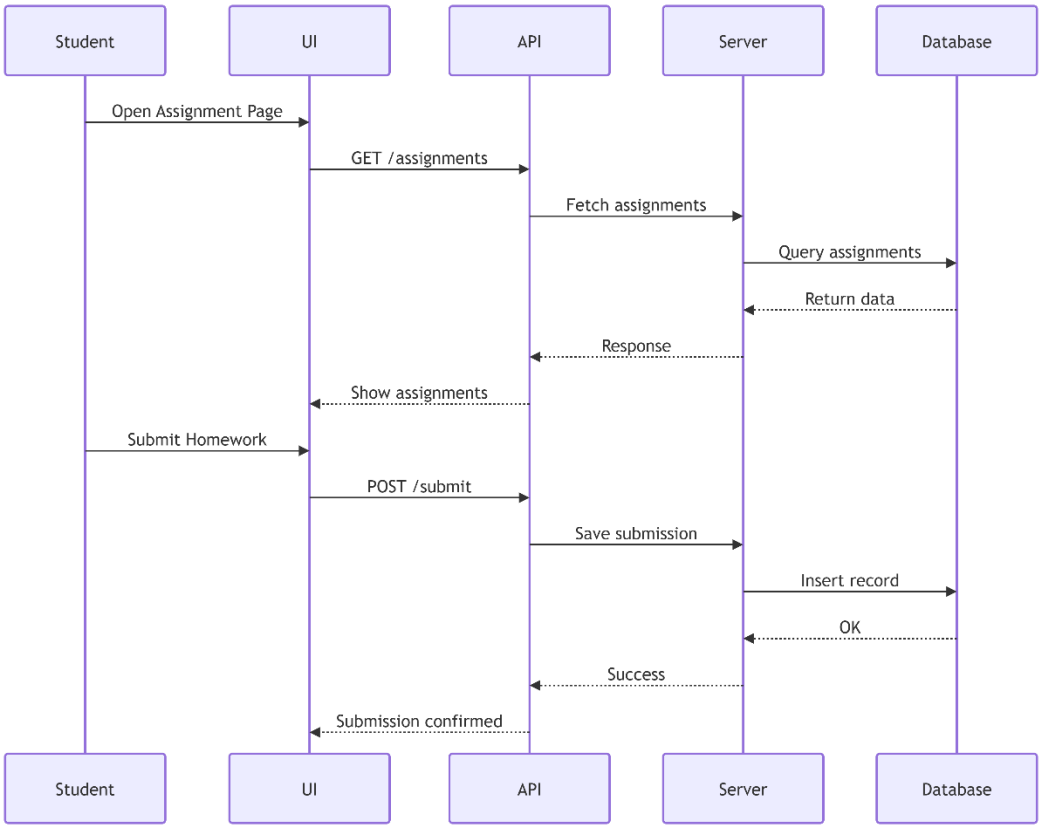
با این حال، برخی بخش‌ها مانند داشبوردهای ترکیبی و منطق‌های چندمسئولیتی نیازمند بازطراحی و Refactoring هستند تا انطباق کامل با اصول SOLID حاصل گردد.

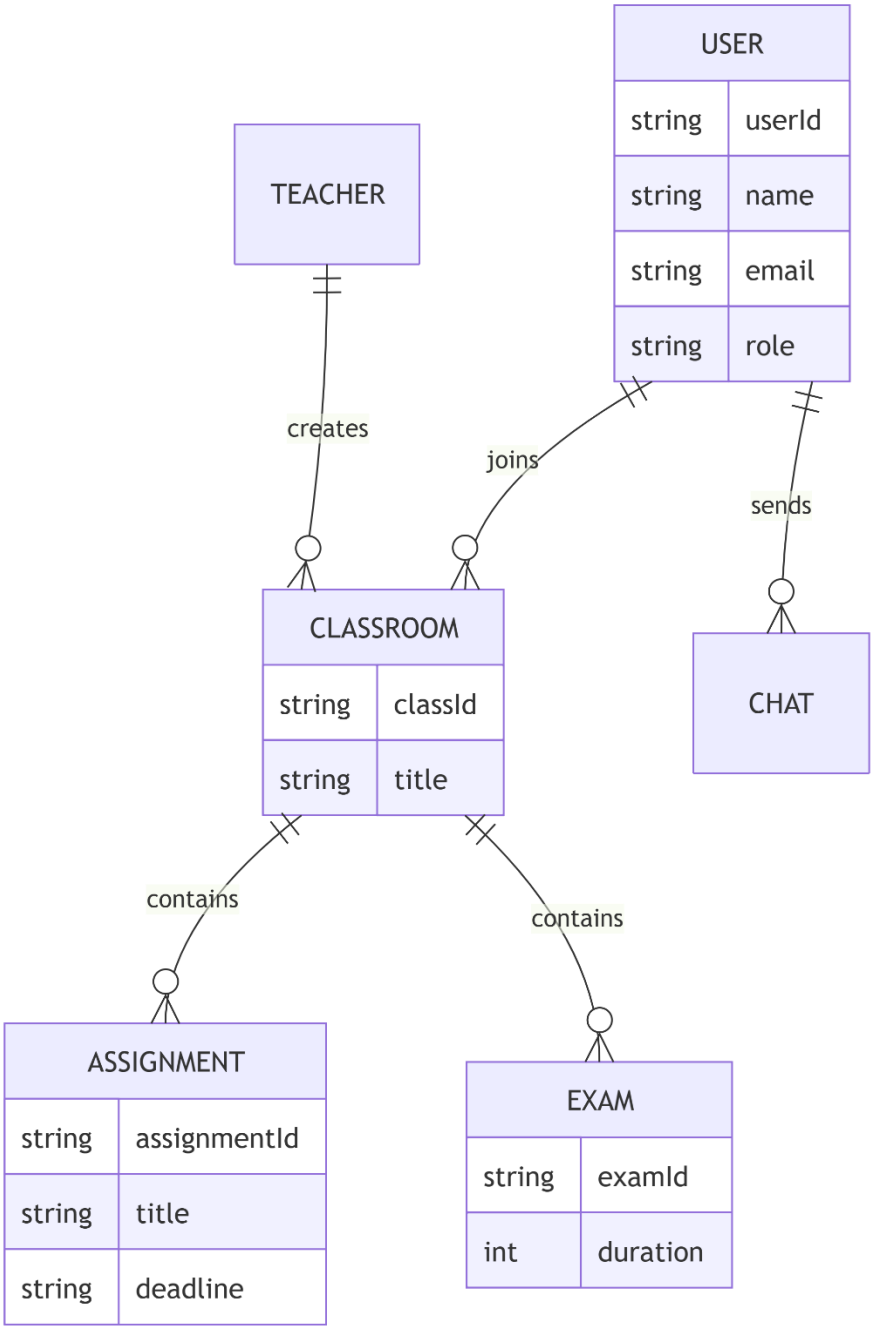

Chat
+String chatId
+sendMessage()
communicates
User
+String userId
+String name
+String email
+String password
+String role
+login()
+logout()
Teacher
+createClass()
+createExam()
+gradeHomework()
Student
+viewClass()
+submitHomework()
+viewGrade()
Admin
+manageUsers()
+manageSystem()
creates
joins
Classroom
+String classId
+String title
+addStudent()
+removeStudent()
has
has
Assignment
+String assignmentId
+String title
+submit()
+grade()
Exam
+String examId
+int duration
+calculateScore()


Student
View Classes
Submit Assignment
Attend Exam
Chat with Teacher


Teacher
Create Class
Create Assignment
Create Exam
Grade Students


Admin
Manage Users
Manage System Settings


Student
UI
API
Server
Database
Open Assignment Page
GET /assignments
Fetch assignments
Query assignments
Return data
Response
Show assignments
Submit Homework
POST /submit
Save submission
Insert record
OK
Success
Submission confirmed
Student
UI
API
Server
Database


USER
string
userId
string
name
string
email
string
role
TEACHER
creates
joins
sends
CLASSROOM
string
classId
string
title
CHAT
contains
contains
ASSIGNMENT
string
assignmentId
string
title
string
deadline
EXAM
string
examId
int
duration