مهندسی نرم‌افزار

نیم‌سال اول ۰۴-۱۴۰۳

دانشکده‌ی مهندسی کامپیوتر

استاد: دکتر مهران ریواده

کوئیز سوم

پاسخ‌دهنده: معین آعلی - ۴۰۱۱۰۵۵۶۱

فهرست مسائل

پاسخ مسئله‌ی ۱.

یک الگوی طراحی مثل Observer-Pattern می‌تواند در Sequence-Diagram به خوبی نمایش داده شود زیرا تعاملات بین Subjecter و Observers (مانند پیام‌رسانی و به‌روزرسانی) به ترتیب خاصی اتفاق می‌افتد. در اینجا، پیام‌هایی که به Observers ارسال می‌شود در Sequence-Diagram قابل مشاهده است، اما Class-Diagram فقط روابط ایستا و وابستگی‌ها را نمایش می‌دهد و ترتیب پیام‌ها را نشان نمی‌دهد.

پاسخ مسئله‌ی ۲.

الف

Abstraction

انتزاع به معنای تمرکز بر ویژگی‌های ضروری یک شیء یا مفهوم است، بدون توجه به جزئیات غیرضروری. این مفهوم کمک می‌کند تا پیچیدگی کاهش یابد و طراحی ساده‌تر شود. انتزاع مشخص می‌کند که یک سیستم یا مؤلفه باید چه کاری انجام دهد، نه اینکه چگونه آن را انجام می‌دهد. سیستم به چندین سطح انتزاع تقسیم می‌شود، از کلی‌ترین سطح تا سطوح دقیق‌تر. مثلا در کلاس Car می‌توانیم متدهای عمومی مانند drive و start و stop را داشته باشیم اما جزئیات پیاده‌سازی این متدها (مانند نحوه کار کردن موتور) برای کاربر مخفی باشد و او فقط با عملکرد کلی این متدها آشنا شود.

Information-Hiding

پنهان‌سازی اطلاعات به معنای محدود کردن دسترسی به جزئیات داخلی یک مؤلفه یا کلاس است تا کاربران یا بخش‌های دیگر سیستم تنها بتوانند از طریق رابط‌های مشخص و تعریف‌شده با آن تعامل داشته باشند. این مفهوم با استفاده از اصول Encapsulation در طراحی پیاده‌سازی می‌شود. پنهان‌سازی اطلاعات بر نحوه دسترسی و استفاده از داده‌ها و متدها تمرکز دارد. جزئیات داخلی مانند متغیرها و متدهای خصوصی از دید کاربر یا کلاس‌های دیگر مخفی می‌شوند. تغییر در جزئیات پیاده‌سازی تأثیری بر سایر بخش‌های سیستم ندارد، زیرا آن‌ها تنها با رابط Interface تعامل دارند.

ب

Abstraction و Information-Hiding با همکاری یکدیگر کیفیت نرم‌افزار را از طریق کاهش پیچیدگی، افزایش قابلیت نگهداری، گسترش‌پذیری و استفاده مجدد بهبود می‌بخشند. انتزاع با تمرکز بر چیستی سیستم، تنها مفاهیم اصلی و رابط‌های مورد نیاز را نمایش می‌دهد، در حالی که پنهان‌سازی اطلاعات با محدود کردن دسترسی به جزئیات داخلی، از وابستگی و اختلال در بخش‌های دیگر جلوگیری می‌کند. این همکاری امکان مدیریت تغییرات، تست آسان‌تر، و گسترش سیستم بدون تأثیر بر سایر مؤلفه‌ها را فراهم کرده و نرم‌افزاری انعطاف‌پذیر و پایدار ایجاد می‌کند.

پاسخ مسئله‌ی ۳.

هدف

SOLID بر مجموعه‌ای از اصول کلی طراحی متمرکز است که به توسعه‌دهندگان کمک می‌کند تا کدهایی انعطاف‌پذیر، قابل نگهداری و گسترش‌پذیر ایجاد کنند. این اصول پایه‌هایی برای طراحی خوب هستند و راهنماهای کلی ارائه می‌دهند.

GoF شامل راه‌حل‌های خاص برای مشکلات تکراری طراحی نرم‌افزار است. این الگوها مانند دستورالعمل‌هایی هستند که به کارگیری آن‌ها در شرایط مناسب به بهبود طراحی کمک می‌کند.

سطح انتزاع

SOLID در سطح اصول و مفاهیم کلی قرار دارد و می‌تواند به تمام بخش‌های طراحی نرم‌افزار اعمال شود.

GoF در سطح عملیاتی و خاص‌تر عمل می‌کند، زیرا راهکارهای دقیق برای مسائل مشخصی مانند مدیریت اشیا، ساختار کلاس‌ها، یا رفتارهاارائه می‌دهد.

نحوه استفاده

SOLID هنگام طراحی یک سیستم یا نوشتن کلاس‌ها و ماژول‌ها، به عنوان راهنمای طراحی کلی برای رعایت بهترین شیوه‌ها استفاده می‌شود.

GoF زمانی به کار می‌رود که مشکلی خاص در طراحی وجود داشته باشد و بخواهیم از الگوهای از پیش تعریف‌شده‌ای استفاده کنیم تا آن مشکل را حل کنیم.

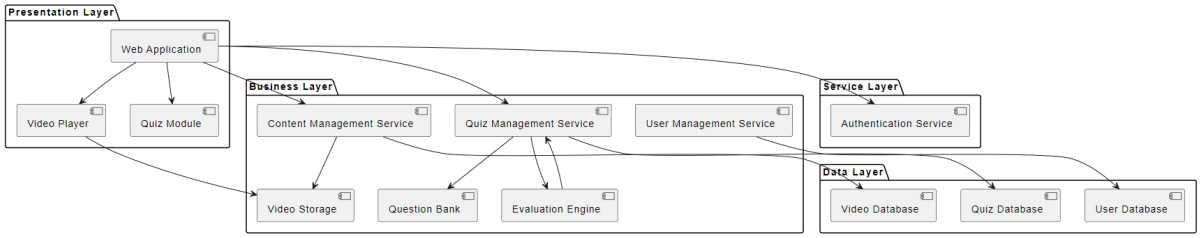
تمرکز

SOLID تمرکز بر کیفیت کد و معماری کلی است؛ مثلاً جداسازی مسئولیت‌ها یا کاهش وابستگی بین مؤلفه‌ها.

GoF بر ارائه ساختارها و الگوهای استاندارد برای حل مشکلات خاص طراحی مانند ایجاد اشیا Factory-Pattern ، مدیریت وابستگی‌ها Decorator-Pattern ، یا تعامل بین کلاس‌ها Observer-Pattern تمرکز دارد.

پاسخ مسئله‌ی ۴.

این سامانه می‌تواند با استفاده از معماری چندلایه طراحی شود. این معماری قابلیت مقیاس‌پذیری و نگهداری مناسب را فراهم می‌کند. لایه‌های پیشنهادی به شرح زیر هستند:



لایه Presentation مربوط به واسط کاربری است و تعامل کاربران با سیستم را مدیریت می‌کند.

لایه Business تمامی قوانین و منطق اصلی سیستم را اجرا می‌کند و به عنوان واسطه بین لایه نمایش و لایه سرویس عمل می‌کند.

لایه Service وظیفه ارائه API ها و مدیریت ارتباط بین لایه کسب‌وکار و لایه داده را دارد.

لایه Data وظیفه ذخیره و مدیریت داده‌ها را بر عهده دارد.