به نام خدا

مهندسی نرم‌افزار

پاسخنامه تمرین سوم

استاد ناظر:

دکتر مهران ریواده

تیم طراحی:

همراز عرفاتی

مهدی عباس‌تبار

رویا قوامی

سارا آذرنوش

دانشگاه صنعتی شریف

زمستان ۱۴۰۳

سوال اول

- پاسخ:

۱) الگوی قطع‌کننده مدار (Circuit Breaker Pattern) در این سناریو مناسب است. این الگو با جلوگیری از ارسال درخواست‌های جدید به سرویس "مدیریت موجودی" که دچار مشکل شده، از بروز مشکلات بیشتر جلوگیری می‌کند.
هنگامی که درخواست‌ها به سرویس "مدیریت موجودی" بارها شکست می‌خورند یا زمان پاسخگویی طولانی می‌شود، مدار شکن فعال شده و مسیر ارتباط با این سرویس را قطع می‌کند. به جای ارسال درخواست به سرویس مختل‌شده، یک پاسخ جایگزین بازمی‌گرداند (مانند پیامی مبنی بر "عدم امکان بررسی موجودی در حال حاضر"). در عین حال، مدار شکن به صورت دوره‌ای سلامت سرویس را بررسی می‌کند. اگر مشکل حل شود، ارتباط به حالت عادی بازمی‌گردد.

مزایا:

- جلوگیری از هدر رفت منابع و انتظار طولانی درخواست‌ها.
- حفظ عملکرد کلی سیستم با ارائه پاسخ‌های جایگزین.
- جلوگیری از گسترش خرابی به سایر بخش‌های سیستم (مشکلات آبشاری).

۲) در این سناریو، می‌توان پیام‌هایی مانند:

  - "موجودی کالا در حال حاضر قابل بررسی نیست، اما سفارش شما ثبت شده است و وضعیت موجودی به‌زودی بررسی خواهد شد."
  - "سفارش شما به‌صورت موقت تأیید شده است. در صورت عدم موجودی کالا، مبلغ پرداختی به شما بازگردانده خواهد شد."
    ارائه داد.

این پیام‌ها باید شفاف و اطمینان‌بخش باشند تا کاربر احساس کند سیستم همچنان تحت کنترل است.

۳) برای بازگرداندن ارتباط، مدار شکن می‌تواند از حالت **نیمه‌باز (Half-Open)** استفاده کند:

- **پروتکل تست تدریجی**: تعداد محدودی از درخواست‌ها به سرویس ارسال شود. اگر پاسخ‌ها موفقیت‌آمیز باشند، مدار به حالت عادی بازمی‌گردد.

- **ابزارهای نظارتی**: از ابزارهایی مانند *Prometheus* یا *Grafana* برای پایش سلامت سرویس معیوب و هشداردهی استفاده شود.
- **بازبینی معیارها**: پس از رفع مشکل، معیارهای مدار شکن بازبینی و تنظیم مجدد شوند تا از فعال شدن بی‌مورد در آینده جلوگیری شود.

این اقدامات می‌توانند به بهینه‌سازی عملکرد سیستم در مواجهه با مشکلات کمک کنند و تجربه کاربری را در شرایط بحرانی بهبود بخشند.

سوال دوم

1. **ارزیابی کیفیت عملکرد (Performance Quality):**

کیفیت عملکرد به توانایی نرم‌افزار در ارائه نتایج مطلوب و پایداری در شرایط مختلف اشاره دارد. برای ارزیابی این نوع کیفیت می‌توان از معیارهایی مانند زمان پاسخ‌دهی، استفاده از منابع (مانند حافظه و پردازنده) و تحمل بار سیستم استفاده کرد.
تیم می‌تواند از تست‌های بارگذاری (Load Testing) و تست‌های تنش (Stress Testing) استفاده کند تا عملکرد اپلیکیشن تحت شرایط مختلف بار بررسی شود.

   - *مثال:* اندازه‌گیری زمان پاسخ‌دهی به درخواست‌های کاربران در شرایط عادی و زمان‌هایی که تعداد درخواست‌ها به اوج می‌رسد.
   - ابزارهای مورد استفاده: JMeter یا LoadRunner.

2. **ارزیابی کیفیت تطبیق (Conformance Quality):**

کیفیت تطبیق به میزان پیروی نرم‌افزار از الزامات مشخص‌شده، استانداردها و قوانین می‌پردازد. این نوع کیفیت با بررسی اسناد الزامات، آزمون‌های رسمی و تحلیل تطابق با استانداردهای صنعت اندازه‌گیری می‌شود.
تیم می‌تواند استانداردهای امنیتی مانند ISO/IEC 27001 و الزامات قانونی مرتبط (مانند GDPR برای حفاظت از داده‌ها) را به‌عنوان مبنای ارزیابی قرار دهد. آزمون‌های امنیتی (Security Testing) و تحلیل تطابق (Compliance Analysis) برای این کار مناسب هستند.

   - *مثال:* اطمینان از رمزنگاری داده‌های حساس کاربران.

3. **تفاوت و تأثیر بر تجربه کاربر:**
   - کیفیت عملکرد مستقیماً بر تجربه کاربر تأثیر دارد. اگر زمان پاسخ‌دهی اپلیکیشن بالا باشد، کاربران ناراضی خواهند شد و ممکن است اعتماد خود را به خدمات بانکی از دست بدهند.
   - کیفیت تطبیق تضمین می‌کند که اپلیکیشن ایمن و قابل‌اعتماد است. نقض این کیفیت می‌تواند به افشای داده‌ها یا حتی جریمه‌های قانونی منجر شود، که به اعتبار شرکت آسیب جدی می‌زند.

سوال سوم

پاسخ:

1. **چارچوب Boehm برای مدیریت ریسک:**
   چارچوب Boehm فرآیند مدیریت ریسک را به چهار مرحله اصلی تقسیم می‌کند:
   - **شناسایی:** تشخیص ریسک‌های بالقوه‌ای که ممکن است پروژه را تحت تأثیر قرار دهند.
   - **تحلیل:** ارزیابی تأثیر و احتمال وقوع هر ریسک برای تعیین اهمیت آن.
   - **اولویت‌بندی:** دسته‌بندی ریسک‌ها بر اساس اهمیت و تخصیص منابع برای مدیریت مهم‌ترین آنها.
   - **کاهش:** توسعه و اجرای برنامه‌هایی برای کاهش احتمال وقوع ریسک یا تأثیر آن.
2. **ریسک تکنولوژی جدید:**
   - *برنامه کاهش:*
     - برگزاری آموزش‌های تخصصی برای تیم در مورد تکنولوژی جدید.
     - اختصاص منابع بیشتر برای تحقیق و توسعه.
     - پیاده‌سازی نمونه‌های اولیه (Prototyping) برای کاهش عدم قطعیت‌ها و بررسی قابلیت استفاده.
   - *برنامه واکنش:*
     - اگر تکنولوژی جدید کارایی لازم را نداشت، به تکنولوژی جایگزینی که قبلاً بررسی و آماده شده است، تغییر مسیر داده شود.
3. **ریسک وابستگی‌های تیم‌های خارجی:**
   - *برنامه کاهش:*
     - تنظیم قراردادهای دقیق و شفاف با تیم خارجی، شامل تعهد به کیفیت و زمان‌بندی.
     - ایجاد زمان‌بندی‌های شفاف و هماهنگ با کل پروژه.
     - برگزاری جلسات منظم پیگیری برای نظارت بر پیشرفت کار.
   - *برنامه واکنش:*
     - در صورت عدم همکاری مناسب، تأمین منابع از تیم‌های داخلی یا انتخاب یک تأمین‌کننده جایگزین.
4. **مزیت برنامه‌ها:**
   - **برنامه کاهش:** احتمال وقوع ریسک را کاهش داده و به جلوگیری از تأخیر یا مشکلات دیگر کمک می‌کند.
   - **برنامه واکنش:** تضمین می‌کند که حتی در صورت وقوع ریسک، تأثیرات منفی آن بر پروژه حداقل بوده و پروژه بتواند ادامه پیدا کند.

این چارچوب جامع به تیم کمک می‌کند تا ریسک‌ها را به‌صورت استراتژیک مدیریت کرده و به اهداف پروژه دست یابد.

سوال چهارم

پاسخ:

الف) شرکت "فودلاین" می‌تواند از اصول «انتزاع» و «پنهان‌سازی اطلاعات» در طراحی ماژول مدیریت سفارش به روش زیر استفاده کند:

۱. انتزاع (Abstraction):

- **تمرکز بر ویژگی‌های اصلی:**

تیم توسعه می‌تواند تمرکز خود را روی ویژگی‌های کلیدی ماژول مدیریت سفارش، مانند دریافت اطلاعات سفارش، بررسی موجودی، و به‌روزرسانی وضعیت سفارش (در حال پردازش، آماده برای ارسال، تحویل داده‌شده) قرار دهد و از جزئیات غیرضروری مانند پروتکل‌های انتقال داده یا فرمت ذخیره‌سازی پایگاه داده صرف‌نظر کند.

- **استفاده از انتزاع داده و رویه‌ای:**
  - انتزاع داده: طراحی یک کلاس "Order" که تنها ویژگی‌های کلیدی مثل شناسه سفارش، وضعیت، و جزئیات مشتری را نمایش دهد.
  - انتزاع رویه‌ای: طراحی API هایی مانند updateOrderStatus(orderId, status) یا getOrderDetails(orderId) که جزئیات پیاده‌سازی داخلی را از سایر بخش‌های سیستم مخفی کند.

۲. پنهان‌سازی اطلاعات (Information Hiding):

- **مخفی کردن جزئیات پیاده‌سازی:**

ماژول مدیریت سفارش می‌تواند تنها رابط‌های ضروری (مانند متدهای API) را در دسترس دیگر ماژول‌ها قرار دهد. جزئیاتی مانند نحوه ذخیره‌سازی اطلاعات در پایگاه داده یا الگوریتم‌های خاص پردازش سفارش‌ها از دید سایر ماژول‌ها پنهان باقی می‌ماند.

- **ایجاد لایه‌های مشخص:**

تیم توسعه می‌تواند لایه‌هایی جداگانه برای پردازش داده‌ها، مدیریت پایگاه داده، و ارتباط با واسط کاربری طراحی کند. برای مثال، لایه پایگاه داده تنها با لایه پردازش تعامل دارد و سایر بخش‌ها به آن دسترسی مستقیم ندارند.

۳. بهبود کیفیت نرم‌افزار:

- **ماژولاریتی (Modularity):**
انتزاع و پنهان‌سازی اطلاعات به ماژولار شدن سیستم کمک می‌کنند. هر بخش از سیستم می‌تواند مستقل طراحی شود و تغییرات در یک ماژول (مانند پایگاه داده) به سایر بخش‌ها تأثیری نمی‌گذارد.
- **ساده‌تر شدن نگهداری (Maintainability):**
به دلیل محلی‌سازی تغییرات، تیم توسعه می‌تواند به‌راحتی تغییرات یا بهبودهایی را در یک بخش خاص اعمال کند بدون اینکه سایر بخش‌ها نیاز به تغییر داشته باشند.
- **افزایش انعطاف‌پذیری (Flexibility):**
اگر "فودلاین" بخواهد سیستم را گسترش دهد (مثلاً افزودن قابلیت سفارش‌های گروهی)، انتزاع و پنهان‌سازی اطلاعات این امکان را فراهم می‌کنند که تغییرات به‌صورت کنترل‌شده انجام شوند و سیستم دچار خطاهای پیش‌بینی‌نشده نشود.

با رعایت این اصول، ماژول مدیریت سفارش "فودلاین" می‌تواند به سیستمی پایدار، انعطاف‌پذیر، و آسان برای نگهداری تبدیل شود.

ب) حفظ استقلال عملکردی

۱. قابلیت اطمینان:

ماژولاریتی (Modularity):

هر ماژول یک وظیفه مشخص مانند مدیریت انبار، پیشنهادات یا پردازش سفارش دارد. این استقلال باعث می‌شود هر بخش به صورت جداگانه تست شده و خطاهای یک بخش بر دیگران تاثیر نگذارد. مثال:

اگر ماژول پیشنهاد غذا دچار خطا شود، دیگر بخش‌ها مانند ثبت سفارش همچنان عملکرد عادی خواهند داشت.

۲. مقیاس‌پذیری (Scalability):

معماری میکروسرویس:

هر ماژول به‌صورت مستقل اجرا و مقیاس‌بندی می‌شود. برای مثال، اگر تعداد کاربران در بخش پیشنهادات افزایش یابد، تنها منابع این بخش گسترش می‌یابد. مثال:

ماژول پیشنهادات با الگوریتم‌های یادگیری ماشین طراحی می‌شود و از طریق API با ماژول ثبت سفارش تعامل دارد.

مثال پیاده‌سازی در سیستم فودلاین:

- ماژول پیشنهاد غذا:

تحلیل سفارش‌های قبلی مشتریان و ارائه پیشنهادات مرتبط از طریق الگوریتم‌های یادگیری ماشین. این ماژول اطلاعات را از دیتابیس دریافت کرده و فقط نتیجه تحلیل را به واسط کاربری ارسال می‌کند.

- ماژول مدیریت انبار:

مدیریت موجودی اقلام غذا و ارسال اطلاعات موجودی به ماژول سفارش. برای مثال، هنگام ثبت سفارش، موجودی بررسی شده و نتیجه (موجود یا ناموجود بودن) بازگردانده می‌شود.

- ماژول ثبت سفارش:

پردازش اطلاعات مشتری و سفارش و ارتباط با سایر ماژول‌ها برای نهایی‌سازی فرآیند سفارش.