**به نام خدا**

# آزمون پایان‌ترم گروه ۲- پاسخنامه

## مهندسی نرم‌افزار – نیمسال اول ۴۰۲

- تاریخ آزمون: چهارشنبه ۴ / ۱۱ / ۴۰۲
- زمان شروع: ۹:۰۰
- مدت زمان آزمون: ۱۵۰ دقیقه


دانشکده مهندسی کامپیوتر – دانشگاه شریف

مدرس: دکتر مهران ریواده

۱- سه نوع مختلف هزینه‌هایی که ممکن است در راه افزایش کیفیت نرم‌افزار مجبور شویم پرداخت کنیم را توضیح دهید و در چارچوب یک شرکت مبتنی بر تجارت الکترونیک (E-commerce) مثالی از هر کدام بیاورید.

**پاسخ:**

**۱) هزینه‌های پیشگیری (Prevention costs):**

هزینه‌های پیشگیری برای جلوگیری یا به حداقل رساندن نقص‌ها و مشکلات در طول چرخه‌ی عمر ایجاد نرم‌افزار پرداخت می‌شوند. این هزینه‌ها با فعالیت‌هایی مرتبط‌اند که هدفشان جلوگیری از رخ دادن خطاها و حصول اطمینان از رعایت استانداردهای کیفیت در نرم‌افزار می‌باشد. می‌توان به موارد زیر به عنوان مثال‌هایی از این فعالیت‌ها اشاره کرد.

- در حوزه آموزش:

مثال: برگزاری برنامه‌های آموزشی برای توسعه‌دهندگان، تست‌کنندگان و سایر اعضای تیم در زمینه‌ی اصول کدنویسی امن به منظور جلوگیری از آسیب‌پذیری‌هایی که ممکن است منجر به نقض اطلاعات شوند.

هزینه‌ها: هزینه‌های مواد آموزشی، مدرسان و زمان صرف شده توسط کارمندان در جلسات آموزش.

- در حوزه بهبود فرآیند:

مثال: پیاده‌سازی یک سامانه پردازش سفارش استاندارد برای جلوگیری از بروز خطاها در انجام سفارشات.

هزینه‌ها: تجزیه و تحلیل فرآیند، مستندسازی و ایجاد یا انتخاب ابزارهایی برای پشتیبانی از فرآیند بهبود یافته

- در حوزه برنامه‌ریزی کیفیت:

مثال: ایجاد یک برنامه سنجش کیفیت جامع برای پلتفرم تجارت الکترونیک به منظور اطمینان از اینکه استانداردهای عملکرد، امنیت و قابل استفاده بودن (Usability) را رعایت می‌کند.

هزینه‌ها: منابع انسانی برای برنامه‌ریزی اینکه چگونه باید سنجش کیفیت انجام شود و اینکه اصلاً چه معیارهایی برای کیفیت باید انتخاب شوند، زمان اختصاص یافته برای این برنامه‌ریزی‌ها و همچنین زمان اختصاص یافته برای اینکه برنامه‌های مدنظر اجرا شوند.

**۲) هزینه‌های ارزیابی (Appraisal costs):**

هزینه‌های ارزیابی در طول مراحل مختلف چرخه‌ی عمر ایجاد نرم‌افزار برای شناسایی و تصحیح عیوب، حصول اطمینان از اینکه محصول نهایی با استانداردهای کیفیت مشخص مطابقت دارد، پرداخت می‌شود. فعالیت‌های ارزیابی بر بازرسی، بازبینی و آزمون نرم‌افزار متمرکز شده‌اند تا هر مشکلی را در مراحل اولیه، شناسایی و برطرف کنند. می‌توان به موارد زیر به عنوان مثال‌هایی از این فعالیت‌ها اشاره کرد.

- در حوزه آزمون نرم‌افزار:

مثال: انجام آزمون دقیق وب‌سایت تجارت الکترونیک، شامل آزمون صحت عملکرد منطقی و مورد نظر وبسایت، آزمون امنیتی و آزمون استرس و لود.

هزینه‌ها: ابزارهای تست، محیط‌های تست و زمان و تلاش صرف شده در برنامه‌ریزی، اجرا و تجزیه و تحلیل آزمون.

- در حوزه بررسی‌ها و نقدها:

مثال: انجام Code review و بازبینی‌ها برای شناسایی و اصلاح مشکلات احتمالی در ماژول پردازش پرداخت.

هزینه‌ها: زمان و تلاش اعضای تیم درگیر در فرآیند بازبینی.

**۳) هزینه‌های خرابی (Failure costs):**

هزینه‌های خرابی به هزینه‌هایی اطلاق می‌شود که در نتیجه نقص یا مشکلاتی است که در طول فرآیند ایجاد، شناسایی و اصلاح نمی‌شوند و منجر به مشکلاتی در محصول نهایی نرم‌افزار یا استقرار آن می‌شوند. این هزینه‌ها از عدم رعایت استانداردهای

کیفیت ناشی می‌شوند و می‌توانند به صورت داخلی (در فرآیند ایجاد) یا خارجی (پس از عرضه نرم‌افزار به کاربران) رخ دهند. می‌توان به موارد زیر به عنوان مثال‌هایی از فعالیت‌های مرتبط با این نوع هزینه‌ها اشاره کرد.

- در حوزه هزینه‌های داخلی خرابی (Internal Failure Costs):

مثال: شناسایی و رفع اشکالات در عملکرد سبد خرید قبل از اینکه پلتفرم تجارت الکترونیک به مشتریان منتشر شود.

هزینه‌ها: بازآرایی (Refactor)، رفع اشکال و زمان و تلاش مرتبط با نیاز به حل مسائل در طول فاز پیاده‌سازی.

- هزینه‌های خرابی خارجی (External Failure Costs):

مثال: مواجهه با مشکلات پشتیبانی مشتری و پردازش بازگشت محصول به دلیل یک عیب نرم‌افزاری که منجر به ارسال نادرست محصولات شده است.

هزینه‌ها: منابع پشتیبانی مشتری، بازپرداخت یا مرجوعی و یا هر گونه جبران مافات برای مشتریان و خطر آسیب به اعتبار شرکت.

- از دست دادن فرصت تجاری:

مثال: نارضایتی مشتری منجر به کاهش فروش و تأثیر منفی بر حصار بازار می‌شود.

هزینه‌ها: هزینه‌های نامشهود مرتبط با از دست دادن فرصت درآمد و خسارت بلندمدت احتمالی به برند.

( صفحه ۴۲۲ تا ۴۲۴ - بخش ۱۹,۳,۲ - کتاب پرسمن ادیشن ۸)

۲- تصور کنید یک تیم ایجاد نرم‌افزار در حال برگزاری یک جلسه‌ی بازبینی فنی رسمی (FTR[1]) برای مستندات طراحی یک ماژول حیاتی پروژه‌ی خود است. این تیم از ۱۵ عضو شامل اعضای تیم ایجاد، آزمون و مدیران پروژه تشکیل شده است. در طول جلسه، مدیر بازبینی مکرراً به اشتباهات طراحی اشاره می‌کند و آن‌ها را مستقیماً به نیرویی که مستندات را تهیه کرده است، نسبت می‌دهد. لحن او انتقادی است و منجر به ناراحتی اعضای تیم می‌شود. جلسه با یک دستور کار کوتاه شروع شده اما به سرعت به بحث‌های غیر مرتبط با اسناد طراحی منحرف می‌شود. اعضای تیم، موضوعات مختلف را مطرح می‌کنند و جلسه تمرکز خود را از هدف اصلی که بازبینی طرح است، از دست می‌دهد. مسائل متفاوت توسط اعضای مختلف تیم مطرح می‌شوند و بحث‌های طولانی مدتی در جریان است. همچنین شرکت‌کنندگان در مورد شدت هر مشکل بحث می‌کنند. مدیر بازبینی اصرار دارد که هر مشکل شناسایی شده را فوراً حل کند و جلسه FTR را به یک جلسه حل مسئله تبدیل می‌کند. با توجه به خط مشی‌هایی (Guidelines) که برای بازبینی فنی رسمی داریم:

A. حداقل سه مورد از خط مشی‌هایی که در این سناریو نقض شده‌اند را به وضوح شناسایی کنید و شرح دهید.

B. برای هر خط مشی شناسایی شده در بخش قبل، توضیح دهید چه اقداماتی باید انجام می‌شد تا با شیوه‌های توصیه شده‌ی FTR هماهنگ شود.

C. به نظر شما، نقض هر یک از این خط مشی‌های شناسایی شده چه تاثیری در روند جلسه و به دنبال آن در روند اجرای پروژه‌ی تیم خواهد داشت؟

[1] Formal Technical Review

**پاسخ:**

**بخش A سوال:**

خط‌مشی‌هایی که در این سناریو نقض شده‌اند عبارتند از:

۱) تیم بازبینی از ۱۵ عضو تشکیل شده؛ تعداد شرکت‌کنندگان را باید محدود کنیم.
۲) اشاره مکرر مدیر بازبینی به اشتباهات توسعه‌دهنده‌ها و لحن انتقادی: باید محصول را بازبینی کنیم، نه توسعه‌دهنده‌ی آن را.
۳) منحرف شدن جلسه به بحث‌های غیر مرتبط با اسناد طراحی: باید یک دستور کار تنظیم کرده و آن را حفظ کنیم.
۴) مطرح شدن مسائل متفاوت توسط اعضای مختلف تیم و شکل‌گیری بحث‌های طولانی: باید بحث و مخالفت‌ها را محدود کنیم.
۵) اصرار مدیر بازبینی به حل هر مشکل شناسایی‌شده: باید حوزه‌های مشکلات را بیان کنیم، اما سعی نکنیم همه‌ی مشکلات را حل کنیم.

هر سه موردی از موارد بالا ذکر شده باشد، صحیح است.

**بخش B و C سوال:**

اقداماتی که گفته می‌شود باید برای ۳ مورد بخش قبل باشند.

دو عقل بهتر از یک عقل کار می‌کند، ولی دیگر لزوماً به ۱۵ نفر هم نیاز نیست. باید تعداد اعضای درگیر را به حداقل لازم برسانیم. وقتی تعداد نفرات بیش از حد باشد، جلسه به راحتی منحرف می‌شود و خط‌مشی‌های دیگر هم راحت‌تر نقض می‌شوند.

جلسه FTR شامل افراد و عزت‌نفس آن‌ها می‌شود. به اشتباهات باید به آرامی اشاره شود. لحن جلسه باید آزاد و سازنده باشد. نیت کسی، به خصوص مدیر بازبینی، نباید ایجاد شرمساری یا تحقیر باشد. اگر جلسه به طور نادرست هدایت شود، می‌تواند باعث پیش آمدن دلخوری و سوتفاهم بین اعضا شده و در ادامه‌ی همکاری آن‌ها مشکل ایجاد کند.

یکی از مشکلات کلیدی هر نوع جلسه‌ای، انحراف است. یک جلسه‌ی FTR باید در مسیری مشخص و طبق برنامه پیش رود. مدیر بازبینی مسئولیت حفظ این مسیر و برنامه‌ی جلسات را بر عهده دارد و باید در هنگام انحراف، تذکر دهد و مشکل را برطرف کند. در صورت انحراف، نتیجه‌های به دست آمده ممکن است معتبر نباشند، جلسه بیش‌تر از برنامه طول بکشد و وقتی که می‌توانست مفید بگذرد، تلف شود.

هنگامی که یک موضوع توسط یکی از اعضا مطرح می‌شود، ممکن است توافق همگانی در مورد شدت و اهمیت آن وجود نداشته باشد. به جای صرف وقت برای بحث در مورد این موضوع، باید برای بحث بیش‌تر در خارج از جلسه ثبت شود و از آن گذر کرد. پیامدهای این مورد می‌تواند مشابه مورد ۳ باشد.

بازبینی، جلسه‌ی حل مسئله نیست. حل یک مشکل اغلب می‌تواند توسط تولیدکننده‌ی آن به تنهایی یا فوقش با کمک یک فرد دیگر انجام شود. حل مسئله باید به بعد از جلسه بازبینی موکول شود. از پیامدها این مورد نیز می‌توان به پیامدهای مورد ۳ اشاره کرد.

برای پیامدها و تاثیر نقض هر یک از خط‌مشی‌های شناسایی شده چه در روند جلسه و چه در روند اجرای پروژه‌ی، هر موردی که منطقی و درست باشد قابل قبول است.

(صفحه ۴۴۳ تا ۴۴۴ - بخش ۲۰,۶,۳ - کتاب پرسمن ادیشن ۸)

۳- در یک شرکت نرم‌افزاری یکی از تیم‌ها پس از استقرار با شرایط زیر مواجه می‌شود:

مصطفی باید در یک وظیفه، تغییراتی در صفحه پرداخت میداد. او این تغییرات را طبق نیازمندی تعریف شده توسط مدیر محصول، پیاده‌سازی کرد. این نیازمندی عبارت بود از:

تاکنون، کاربر در صفحه پرداخت نمی‌توانست درگاه پرداخت را انتخاب کند. می‌خواهیم این ویژگی را اضافه کنیم تا کاربر در صفحه پرداخت بتواند درگاه مورد نظر خود را انتخاب نماید.

او پس از پیاده‌سازی، برای کد خود تست نوشت و به صورت دستی نیز از صحت کارکرد در محیط کاربر با استقرار قناری (deployment Canary) اطمینان حاصل نمود. روز بعد از استقرار نهایی این تغییرات، حجم زیادی پیام از سمت تیم پشتیبانی دریافت شد که کاربران به هنگام ورود به صفحه پرداخت، سوالاتی دارند و نمی‌توانند به صورت شفاف این کار را انجام دهند. همچنین بعضی مواقع خطای نامشخصی هنگام انتخاب درگاه پرداخت تعدادی از بانک‌ها مشاهده می‌کنند. در نهایت با بالا رفتن تعداد تماس‌های پشتیبانی، تیم مجبور به Revert تغییرات می‌شود. با توجه به این سناریو که در جلسه Post Mortem مطرح شده است:

A. کدام عوامل تضمین کیفیت نقض شده‌اند؟

B. چه تغییراتی در این فرآیند باید داده شود تا مشکل مشابهی پیش نیاید؛ آیا می‌توانستیم تعریف وظیفه را بهتر کنیم یا مستندات بیشتری اضافه کنیم، یا فرآیند QA را بهبود دهیم یا ... ؟

**پاسخ اول:**

**بخش A سوال:**

۱) Completeness (کامل بودن نیازمندی‌ها): نیازمندی‌های تعریف شده برای ویژگی جدید به اندازه کافی جزئیات نداشته‌اند، به خصوص در مورد رفتار کاربردی و غیرکاربردی درگاه‌های پرداخت مختلف.

۲) Clarity (شفافیت نیازمندی‌ها): از آنجا که کاربران پس از استقرار با سردرگمی مواجه شده‌اند، می‌توان نتیجه گرفت که نیازمندی‌ها به صورت شفاف و قابل فهم برای تمام ذینفعان تعریف نشده‌اند و ابهام در تعریف نیازمندی‌ها وجود دارد.

۳) User Interface Design (طراحی رابط کاربری): درواقع با توجه به سناریو، Design مشخصی برای این تسک وجود نداشته. بنابراین سردرگمی کاربر در نتیجه نبود یک طراحی UI هم می‌تواند باشد.

۴) Test Effectiveness (کارآمدی آزمون): با توجه به خطاهای نامشخصی که کاربران پس از استقرار تجربه کرده‌اند، می‌توان گفت که تست‌های انجام شده نتوانسته‌اند تمامی موارد استفاده و شرایط خطا را پوشش دهند. همینطور تست باید توسط تیم QA انجام می‌گرفت نه صرفاً بدست یکی از توسعه‌دهنده‌ها.

**بخش B سوال:**

برای جلوگیری از بروز مشکلات مشابه، تغییرات زیر در فرآیندها پیشنهاد می‌شود:

۱) تعریف دقیق‌تر نیازمندی‌ها: نیازمندی‌ها باید با جزئیات بیشتر و با در نظر گرفتن تمامی حالات ممکن تعریف شوند.

۲) مستندسازی و طراحی UI: طراحی رابط کاربری باید به صورت دقیق مستند شود و قبل از پیاده‌سازی باید توسط کاربران بررسی و تایید شود.

۳) بهبود فرآیند QA: تست‌ها باید به گونه‌ای باشند که تمامی مسیرهای کاربری و حالات خطا را پوشش دهند و باید شامل تست‌های کاربردی (Functional) و غیرکاربردی (Non-Functional) باشند. همینطور باید تیم یا شخصی به غیر از توسعه دهنده فرایند QA را انجام دهد.

(صفحه ۴۵۵ – بخش ۲۱,۴,۲ - کتاب پرسمن ادیشن ۸)

**پاسخ دوم:**

**بخش A سوال:**

۱) Change Management: یکی از مسائل اصلی، نحوه مدیریت تغییرات در فرآیندی است که کاربران برای انجام دادن عملیات خاصی دنبال می‌کنند. تغییرات اعمال شده در پروژه موجب ایجاد خطاهایی شده است که نشان‌دهنده عدم اجرای صحیح مدیریت تغییرات است.

۲) Reviews and audits: فرآیند بازبینی نهایی، توسط توسعه‌دهنده‌ای انجام شده که خود بر روی ویژگی‌های جدید کار کرده است. علاوه بر این، کد بعد از ایجاد توسط سایر اعضای تیم مورد بازبینی قرار نگرفته است.

۳) Testing (کارآمدی تست): با توجه به خطاهایی که کاربران پس از استقرار با آن مواجه شده‌اند، به نظر می‌رسد که تست‌های انجام شده نتوانسته‌اند تمامی سناریوهای استفاده و شرایط خطا را پوشش دهند. همچنین، تست‌ها باید توسط تیم تضمین کیفیت (QA) و نه تنها توسط یکی از توسعه‌دهندگان انجام شود.

**بخش B سوال:**

برای جلوگیری از بروز مشکلات مشابه، تغییرات زیر در فرآیندها پیشنهاد می‌شود:

۱) تعریف دقیق‌تر نیازمندی‌ها: نیازمندی‌ها باید به صورت دقیق‌تری تعریف شوند و تمامی حالات ممکن را در بر بگیرند تا مدیریت تغییرات به شکل موثرتری صورت گیرد.

۲) مستندسازی و طراحی UI: طراحی رابط کاربری باید به دقت مستند شود و قبل از پیاده‌سازی باید توسط کاربران بررسی و تایید شود.

۳) بهبود فرآیند QA: تست‌ها باید به گونه‌ای باشند که تمامی مسیرهای کاربری و حالات خطا را پوشش دهند و باید شامل تست‌های کاربردی (Functional) و غیرکاربردی (Non-Functional) باشند. همینطور باید تیم یا شخصی بغیر از توسعه دهنده فرایند QA را انجام دهد.

(صفحه ۴۵۰ – ۴۵۱ - کتاب پرسمن ادیشن ۸)

هر یک از پاسخ‌های اول یا دوم بیان شده باشد، صحیح است.

۴- قطعه کد مقابل برنامه جست‌وجوی دودویی در یک آرایه را نمایش می‌دهد.

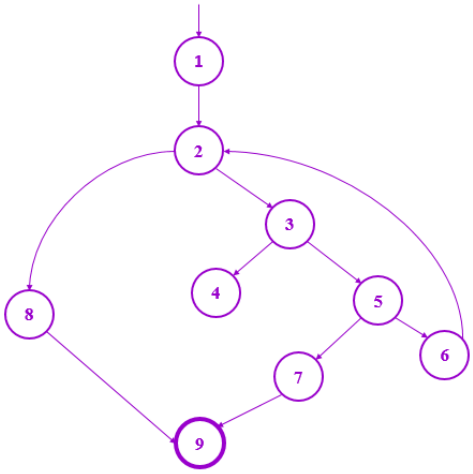
A. گراف کنترل جریان برنامه را رسم نمایید.

B. با استفاده از پوشش نود (Node Coverage) و پوشش یال (Edge Coverage) برای این برنامه، مورد آزمون ( Test Case) طراحی کنید.

**پاسخ:**

گراف کنترل جریان برنامه به شکل زیر است:

```
int binsearch(int x, int v[], int n)
{
    int low, high, mid;
1   low = 0;
    high = n - 1;
    while (low <= high) |2
    {
3       mid = (low + high)/2;
        if (x < v[mid])
            high = mid - 1;  |4
5       else if (x > v[mid])
            low = mid + 1;   |6
7       else return mid;
    }
    return -1; |8
} |9
```




توضیح گراف: برای هر بخش از کد یک نود گراف را در نظر می‌گیریم. نود دوم که مربوط به while است دو حالت دارد، یا شرط برقرار است و به نود ۳ می‌رود، یا برقرار نیست و به نود ۸ می‌رود. از نود سوم که یک if است، اگر شرط برقرار باشد وارد نود ۴ شده و چون در شروط دیگر داخل while صدق نمی‌کند به نود دوم برمی‌گردد و اگر شرط برقرار نباشد وارد نود ۵ می‌شود. در نود ۵ هم اگر شرط برقرار باشد به نود ۶ می‌رود و دوباره به اول while برمی‌گردد، در غیر این صورت به نود ۷ می‌رود. (رسم گراف بدون توضیحات قابل قبول است.)

حال بر اساس گراف و پوشش‌های خواسته شده به طراحی موارد آزمون می‌پردازیم.

**Node coverage:**
Test Requirements = {1, 2, 3, 4, 5, 6, 7, 8, 9}
Test paths: { [1,2,3,5,7,9], [1,2,3,4,2,8,9], [1,2,3,5,6,2,8,9]}

**Edge coverage:**
Test Requirements = {1-2, 2-3, 2-8, 3-4, 3-5, 4-2, 5-6, 5-7, 6-2, 7-9, 8-9}
Test paths: {[1,2,3,5,7,9], [1,2,3,4,2,8,9], [1,2,3,5,6,2,8,9]}

| مسیر آزمون | یال‌های پوشش داده شده |
|---|---|
| [1,2,3,5,7,9] | 1-2 , 2-3 , 3-5 , 5-7 , 7-9 |
| [1,2,3,4,2,8,9] | 3-4 , 4-2 , 2-8 , 8-9 |
| [1,2,3,5,6,2,8,9] | 5-6 , 6-2 |

❖ گرچه برخی از مسیرها یال‌های دیگری را نیز پوشش داده‌اند، برای تفکیک یال‌هایی که هر یک به طور خاص پوشش می‌دهند از آوردن یال‌های پوشش داده شده خودداری کردیم.

با توجه به اینکه مسیرهای آزمون در هر دو حالت یکسان هستند، جدول زیر را به طور کلی برای موارد آزمون در نظر می‌گیریم:

| ردیف | x | v | n | مسیر آزمون |
|---|---|---|---|---|
| 1 | 2 | [1, 2, 3] | 3 | [1,2,3,5,7,9] |
| 2 | 1 | [2] | 1 | [1,2,3,4,2,8,9] |
| 3 | 3 | [2] | 1 | [1,2,3,5,6,2,8,9] |

در صورتی که مسیرهای دیگر بیان شده باشد و یا موارد آزمون متفاوت در نظر گرفته شده باشند، قابل قبول است به شرط آنکه موارد آزمون، همه مسیرهای آزمون را پوشش داده باشند.

(اسلایدهای آزمون نرم‌افزار - فصل ۲۲ و۲۳ و ۲۴)

۵- احتمالاً کپل‌خان را می‌شناسید؛ معمولاً در اغلب امتحانات من (استاد) حضور دارد. این ترم به این فکر افتاده تا به همراه سه نفر از دوستانش یعنی تپل و مپل و خپل یک Startup ایجاد کند. او پس از مشورت با گروه ۱ مهندسی نرم‌افزار، می‌خواهد با شما (گروه ۲) نیز به بررسی Startup یک موتور جستجو بپردازد. ایده این است: همهٔ انسان‌ها گاهی نیاز دارند تا در مواردی که با آن مواجه شده‌اند قوانین را مورد جستجو قرار دهند. می‌خواهیم یک موتور جستجو ایجاد کنیم که برای جستجوی قوانین کاربرد دارد. در بررسی اولیه، کپل‌خان موارد زیر را بدست آورده است:

- ✓ امکان به دست آوردن قوانین کشور وجود دارد.
- ✓ امکان به دست آوردن رأی‌هایی که قضات در پرونده‌ها صادر کرده‌اند وجود دارد.
- ✓ امکان به دست آوردن کتاب‌های مختلف قانون وجود دارد.

هدف تیم ایجاد صفحه‌ای برای جستجو است. برای نتیجه جستجو می‌تواند از هوش مصنوعی هم استفاده کند. مثلاً زمان طرح سوال با توجه به پرونده‌های قبلی و آرای صادر شده قبلی، گزینه مناسبی پیشنهاد شود. با توجه به این توضیحات، به سوالات کپل‌خان که در زیر آمده است، پاسخ دهید.

A. بر اساس روش‌های استخراج نیازمندی‌ها که در این درس آموخته‌اید، ۸ مورد از نیازمندی‌های این نرم‌افزار را بنویسید.
B. برای انجام این پروژه چه متدولوژی را پیشنهاد می‌کنید؟ چرا؟
C. چه نیازمندی غیرعملکردی در این سیستم وجود دارد؟
D. چه سطوحی از آزمون برای این سیستم مورد نیاز است؟ چرا؟

۶- با توجه به توضیحات ارائه شده در پروژه مد نظر کپل‌خان به سوالات زیر نیز پاسخ دهید.

A. آیا معماری میکروسرویس را به کپل‌خان پیشنهاد می‌کنید؟ چرا؟
B. به نظر شما Decomposition میکروسرویس‌ها باید با چه الگویی انجام شود. هر الگویی که پیشنهاد می‌دهید را کمی توضیح داده و سه نمونه از میکروسرویس‌های پیشنهادی را نام ببرید.
C. آیا ممکن است برای این سیستم CQRS کاربرد پیدا کند؟ مثال بزنید.
D. چه روش Messagingی برای این سیستم پیشنهاد می‌کنید؟ روش انتخابی خود را توضیح دهید.

موفق باشید

تیم آموزش مهندسی نرم‌افزار