



فهرست مسائل

۱	مسئله ۱
۱	الف
۲	ب
۲	ج
۲	د
۳	مسئله ۲
۳	الف
۳	ب
۴	ج
۵	مسئله ۳
۵	الف
۵	ب
۶	ج
۷	مسئله ۴
۷	الف
۸	ب
۹	ج
۱۰	د

پاسخ مسئله‌ی ۱.

الف

کیفیت نرم افزار به معنای میزان برآورده شدن نیازهای عملکردی و غیرعملکردی سیستم است. دستیابی به کیفیت بالا در مهندسی نرم افزار حیاتی است زیرا مستقیماً بر قابلیت اطمینان، نگهداشت پذیری، و رضایت کاربران تأثیر می‌گذارد. کیفیت پایین نرم افزار می‌تواند منجر به نقص‌های پرهزینه، کاهش بهره‌وری، و نارضایتی کاربران شود.

ابعاد کیفیت مطرح شده توسط گاروین عبارتند از:

- عملکرد (Performance): میزان توانایی سیستم در انجام وظایف مورد نظر. مثال: سامانه مدیریت بانکداری الکترونیکی که تراکنش‌ها را به سرعت پردازش می‌کند.
- ویژگی‌ها (Features): قابلیت‌های اضافی که فراتر از نیازهای اولیه هستند. مثال: اپلیکیشن پیام‌رسانی که علاوه بر ارسال متن، امکان تماس ویدیویی نیز دارد.
- قابلیت اطمینان (Reliability): میزان عملکرد پایدار و بدون خرابی سیستم در طول زمان. مثال: سیستم کنترلی هواپیما که باید همواره بدون نقص کار کند.
- مطابقت (Conformance): میزان تطابق با استانداردها و مشخصات مورد نظر. مثال: نرم افزار حسابداری که مطابق با استانداردهای مالیاتی کشور توسعه یافته است.

- دوام (Durability) : پایداری و طول عمر نرم افزار. مثال: سیستم عامل که به مدت طولانی قابل استفاده باشد.
- قابلیت استفاده (Usability) : میزان سهولت استفاده توسط کاربران. مثال: نرم افزار طراحی گرافیکی با رابط کاربری ساده و شهودی.
- قابلیت سرویس دهی (Serviceability) : سهولت نگهداری و اصلاح نرم افزار. مثال: سامانه مدیریت محتوا (CMS) که به روزرسانی های آسان دارد.

ب

متدولوژی های آزمون نرم افزار را می توان به دو دسته ی آزمون مرسوم و آزمون شی گرا تقسیم کرد.

- سطوح آزمون: در آزمون مرسوم، تست واحدها (Unit-Testing) به صورت تابعی انجام می شود، اما در آزمون شی گرا، تمرکز بر کلاس ها و اشیا است.
- مورد آزمون: در آزمون مرسوم، تست بر روی مازول های منفرد انجام می شود، اما در آزمون شی گرا تعامل بین اشیا و پیام ها بررسی می شود.
- ابزارها: ابزارهای آزمون سنتی شامل JUnit و NUnit می باشند، در حالی که آزمون شی گرا از ابزارهایی مانند JMock و Mockito بهره می برد.
- مزایا: آزمون شی گرا مقیاس پذیری بیشتری دارد و قابلیت نگهداشت بالاتری را ارائه می دهد.

ج

راستی آزمایی (Verification) و صحت سنجی (Validation) دو مرحله ی مهم در آزمون نرم افزار هستند:

- راستی آزمایی: فرآیند بررسی این که آیا نرم افزار مطابق با مشخصات طراحی شده است یا خیر. مثال: بررسی مستندات و اجرای آزمون های واحد.
- صحت سنجی: بررسی اینکه آیا نرم افزار نیازهای کاربر را برآورده می کند یا خیر. مثال: انجام آزمون های پذیرش توسط کاربران.

هر دو پروسه برای تضمین کیفیت بالا ضروری هستند، زیرا راستی آزمایی از تطابق با طراحی و صحت سنجی از تطابق با نیازهای واقعی اطمینان حاصل می کند.

د

چندین اقدام می تواند تأثیر آزمون رگرسیون را بهبود ببخشد:

- استفاده از آزمون های خود کار برای کاهش هزینه و زمان آزمون.
- انتخاب مجموعه ی بهینه آزمون ها برای اجرای مجدد، به جای اجرای کل مجموعه.
- استفاده از آزمون های افزایشی به جای اجرای مجدد کل سیستم.
- ترکیب آزمون رگرسیون با تحلیل تأثیر تغییرات برای تمرکز بر قسمت های حساس نرم افزار.

پاسخ مسئله‌ی ۲.

الف

یکپارچگی در فرآیند تست نرم افزار به معنای بررسی نحوه تعامل و همکاری میان ماژول ها و اجزای مختلف یک سامانه نرم افزاری است. این تست اطمینان حاصل می کند که اجزا به درستی با یکدیگر ارتباط دارند و داده ها بین آنها به صورت صحیح منتقل می شوند.

اهمیت تست های یکپارچگی برای سامانه های پیچیده به دلایل زیر است:

- تضمین سازگاری بین ماژول های مختلف.
- کشف خطاهای ناشی از تعامل اجزا که در تست های واحد قابل تشخیص نیستند.
- افزایش قابلیت اطمینان سامانه و کاهش مشکلات احتمالی در محیط عملیاتی.

تفاوت های کلیدی بین تست های واحد و تست های یکپارچگی:

- تست واحد بر روی یک ماژول یا تابع مجزا تمرکز دارد، در حالی که تست یکپارچگی نحوه تعامل چندین ماژول را بررسی می کند.
- تست واحد معمولاً توسط توسعه دهندگان انجام می شود، اما تست یکپارچگی بیشتر توسط تیم تست انجام می شود.
- در تست واحد، از شبیه سازی (Mock) برای بررسی عملکرد مستقل یک ماژول استفاده می شود، اما در تست یکپارچگی، سیستم در محیط واقعی تری تست می شود.

ب

دو استراتژی مهم تست یکپارچگی عبارتند از:

- ادغام بیگ بنگ: در این روش، تمام ماژول های سامانه به طور همزمان با یکدیگر ادغام و سپس تست می شوند. این روش برای سامانه های پیچیده ریسک بالایی دارد، زیرا خطاهای متعددی به طور همزمان پدیدار می شوند و یافتن منبع اصلی مشکل دشوار است.
- ادغام افزایشی: در این روش، ماژول ها به صورت تدریجی و مرحله به مرحله ادغام و تست می شوند. این روش شامل دو رویکرد اصلی است:
 - ادغام از بالا به پایین: ابتدا ماژول های سطح بالا تست می شوند و سپس ماژول های سطح پایین اضافه می شوند.
 - ادغام از پایین به بالا: ابتدا ماژول های سطح پایین ادغام و تست شده و سپس به سطح بالاتر گسترش می یابند.

برای تست سامانه بانک اطلاعاتی، استراتژی ادغام افزایشی مناسب تر است، زیرا:

- کاهش ریسک خطاهای پیچیده و امکان شناسایی زودهنگام مشکلات.
- امکان انجام تست های مرحله به مرحله برای اطمینان از عملکرد صحیح هر بخش.
- ساده تر بودن اشکال زدایی نسبت به روش بیگ بنگ.

ج

سناریوی اول:

- مرحله ۱: کاربر اطلاعات ورود را در سامانه ورود وارد می کند.
- مرحله ۲: سامانه ورود، اطلاعات کاربر را به سامانه مدیریت داده ها ارسال می کند.
- مرحله ۳: سامانه مدیریت داده ها بررسی می کند که آیا اطلاعات کاربر معتبر است یا خیر.
- مرحله ۴: در صورت تأیید، مجوز ورود داده شده و اطلاعات به کاربر نمایش داده می شود.

خطاهای محتمل:

- نامعتبر بودن اطلاعات ورود (ورود نادرست رمز عبور).
- اختلال در ارتباط میان سامانه ورود و سامانه مدیریت داده ها.
- ارسال اطلاعات نادرست از سامانه ورود به سامانه مدیریت داده ها.

سناریوی دوم:

- مرحله ۱: یک تراکنش مالی توسط کاربر انجام می شود.
- مرحله ۲: سامانه مدیریت تراکنش ها جزئیات تراکنش را پردازش و ذخیره می کند.
- مرحله ۳: سامانه مدیریت تراکنش ها داده های مرتبط را به سامانه گزارش دهی ارسال می کند.
- مرحله ۴: سامانه گزارش دهی داده ها را دریافت کرده و یک گزارش مالی تولید می کند.

خطاهای محتمل:

- تأخیر یا عدم ارسال اطلاعات از سامانه مدیریت تراکنش ها به سامانه گزارش دهی.
- نمایش نادرست اطلاعات تراکنش در گزارش ها.
- ناهماهنگی در پردازش داده ها و ایجاد گزارش های ناقص.

پاسخ مسئله‌ی ۳.

الف

عناصر اصلی تضمین کیفیت نرم افزار شامل موارد زیر است:

- برنامه ریزی کیفیت: تعیین استانداردها و روش های ارزیابی کیفیت در مراحل مختلف توسعه.
 - تضمین کیفیت: فرآیندها و فعالیت هایی که اطمینان حاصل می کنند نرم افزار مطابق با استانداردها و الزامات تعیین شده است.
 - کنترل کیفیت: شامل فعالیت های تست و بررسی که برای شناسایی و رفع خطاهای نرم افزار انجام می شود.
 - بهبود مستمر کیفیت: تجزیه و تحلیل عملکرد گذشته و اعمال تغییرات برای افزایش کیفیت.
- تضمین کیفیت به عنوان یک فعالیت چتری در نظر گرفته می شود زیرا تمامی مراحل چرخه عمر نرم افزار را در بر می گیرد، از جمله:

- مرحله نیازسنجی: بررسی نیازها برای جلوگیری از ابهامات و تناقضات.
- طراحی: ارزیابی معماری و طراحی برای بهینه سازی ساختار نرم افزار.
- پیاده سازی: بررسی کدها و انجام آزمون های خودکار برای شناسایی خطاها.
- آزمون و اعتبارسنجی: انجام تست های واحد، یکپارچگی و پذیرش برای اطمینان از صحت عملکرد.
- استقرار و نگهداری: ارزیابی عملکرد سیستم و مدیریت تغییرات.

ب

وظایف اصلی تیم تضمین کیفیت شامل موارد زیر است:

- تعریف استانداردها و فرآیندهای کیفیت.
 - طراحی و اجرای تست های مختلف نرم افزار.
 - تحلیل نتایج آزمون ها و ارائه گزارش های کیفی.
 - بررسی و مدیریت مشکلات و پیشنهاد راه حل های بهبود.
 - اجرای ممیزی های کیفی و بهبود فرآیندهای توسعه.
- مدیریت تغییرات نقش مهمی در تضمین کیفیت دارد زیرا هر تغییری در نرم افزار ممکن است بر کیفیت آن تأثیر بگذارد. این فرآیند شامل موارد زیر است:

- بررسی و ارزیابی تأثیر تغییرات بر عملکرد کلی نرم افزار.
 - اجرای آزمون های رگرسیون برای اطمینان از عدم ایجاد مشکلات جدید.
 - مستندسازی تغییرات برای شفافیت و قابل ردیابی بودن فرآیند.
 - مدیریت بازخورد کاربران و اصلاح نواقص احتمالی.
- تأثیر مدیریت تغییرات بر کیفیت نرم افزار شامل افزایش پایداری، کاهش خطاها و بهبود تجربه کاربری است.

ج

مدیریت خطاها و تجزیه و تحلیل آن‌ها نقش حیاتی در بهبود کیفیت نرم افزار دارد. فرآیند مدیریت خطا شامل:

- شناسایی و ثبت خطاها در طول فرآیند تست.
 - تحلیل ریشه‌ای مشکلات برای درک دلایل اصلی آن‌ها.
 - اولویت‌بندی و اصلاح خطاها برای بهینه‌سازی عملکرد.
 - نظارت بر روند اصلاحات و ارزیابی تأثیر آن‌ها.
- فعالیت‌های تضمین کیفیت به کاهش ریسک‌های ایجاد نرم افزار کمک می‌کنند، از جمله:
- انجام تست‌های مستمر برای جلوگیری از بروز مشکلات در محیط عملیاتی.
 - استفاده از رویکردهای مهندسی کیفیت برای جلوگیری از ایجاد خطاها.
 - مستندسازی فرآیندها برای افزایش قابلیت نگهداری و بهبود مداوم.
 - تحلیل بازخورد کاربران برای شناسایی نقاط ضعف و ارتقای نرم افزار.

مثال: در یک سامانه مدیریت بانک اطلاعاتی، تیم تضمین کیفیت پس از شناسایی خطای تأخیر در پردازش گزارش‌ها، یک تحلیل ریشه‌ای انجام داده و متوجه شده که پایگاه داده بهینه‌سازی نشده است. با اجرای تغییرات لازم در شاخص‌های پایگاه داده، زمان پردازش کاهش یافته و عملکرد سامانه بهبود یافته است.

پاسخ مسئله‌ی ۴.

الف

در دنیای نرم افزار و فناوری، کیفیت یکی از مهم ترین عوامل موفقیت شرکت ها محسوب می شود. بسیاری از شرکت های مطرح، با وجود داشتن بازار گسترده و نوآوری، به دلیل نادیده گرفتن برخی از فاکتورهای کلیدی کیفیت، متحمل شکست های سنگینی شده اند. در این مقاله به بررسی سه شرکت مشهور که به علت ضعف در طراحی و کیفیت نرم افزاری شکست خوردند، می پردازیم.

۱. نوکیا (Nokia)

حوزه فعالیت: تولید گوشی های موبایل و تجهیزات مخابراتی
محصولات: تلفن های همراه کلاسیک و هوشمند
ایراد: عدم توجه به تجربه کاربری و به روز رسانی سیستم عامل
نوکیا به عنوان یکی از غول های صنعت موبایل، در سال های اولیه دهه ۲۰۰۰ سلطه ی کاملی بر بازار داشت. اما عدم تمرکز کافی بر سیستم عامل های هوشمند و تجربه ی کاربری مناسب باعث شد تا در رقابت با اندروید و iOS شکست بخورد.

۲. بلک بری (BlackBerry)

حوزه فعالیت: تولید تلفن های هوشمند و تجهیزات امنیتی
محصولات: گوشی های مجهز به صفحه کلید فیزیکی و نرم افزارهای امنیتی
ایراد: نادیده گرفتن طراحی کاربر پسند و محدودیت اکوسیستم نرم افزاری
بلک بری به دلیل تمرکز بیش از حد بر امنیت و عدم توجه کافی به اکوسیستم نرم افزارهای کاربردی و طراحی بصری جذاب، سهم بازار خود را از دست داد.

۳. یاهو (Yahoo)

حوزه فعالیت: ارائه خدمات اینترنتی، موتور جستجو، ایمیل و تبلیغات آنلاین
محصولات: موتور جستجوی یاهو، یاهو میل، یاهو مسنجر
ایراد: عدم نوآوری در طراحی سرویس های دیجیتال و تجربه کاربری ضعیف
ياهو نتوانست خود را با تغییرات سریع در بازار تطبیق دهد و طراحی نامناسب محصولات باعث شد که کاربران به سمت رقابایی مانند گوگل و جیمیل مهاجرت کنند.
بر اساس مدل مک مال، این شرکت ها سه فاکتور مهم کیفیت را نادیده گرفتند:

- تجربه کاربری (Usability): طراحی ضعیف رابط کاربری و پیچیدگی در استفاده باعث کاهش رضایت کاربران شد.
- نگهداشت پذیری (Maintainability): سیستم های نرم افزاری ناکارآمد که به سختی قابلیت بروزرسانی داشتند، منجر به از دست رفتن مزیت رقابتی شد.
- انعطاف پذیری (Flexibility): عدم تطابق با تغییرات بازار و روندهای نوظهور فناوری، موجب شکست در رقابت شد.

این شرکت ها هزینه های مختلفی را متحمل شدند که باعث شد نتوانند شکست خود را جبران کنند:

- هزینه‌ی از دست دادن بازار: نوکیا و بلک‌بری میلیاردها دلار از سهم بازار خود را از دست دادند، زیرا کاربران به گزینه‌های بهتری روی آوردند.
- هزینه‌ی از دست دادن اعتماد مشتریان: هنگامی که کاربران متوجه شدند این برندها نمی‌توانند نیازهای آن‌ها را برآورده کنند، دیگر به آن‌ها بازنگشتند.
- هزینه‌ی توسعه‌ی ناکارآمد: تلاش برای نجات کسب و کار با سرمایه‌گذاری در فناوری‌های جدید، اما بدون درک عمیق از نیاز بازار، به هدر رفتن منابع منجر شد.

ب

در دنیای نرم افزار و فناوری، کیفیت یکی از مهم‌ترین عوامل موفقیت شرکت‌ها محسوب می‌شود. بسیاری از شرکت‌های مطرح، باوجود داشتن بازار گسترده و نوآوری، به دلیل نادیده گرفتن برخی از فاکتورهای کلیدی کیفیت، متحمل شکست‌های سنگینی شده‌اند. در این مقاله به بررسی سه شرکت مشهور که به علت ضعف در طراحی و کیفیت نرم‌افزاری شکست خوردند، می‌پردازیم. همچنین به بررسی شرکت‌هایی می‌پردازیم که با وجود نادیده گرفتن برخی فاکتورهای کیفیت، همچنان در بازار موفق هستند. نمونه‌هایی از شرکت‌های موفق باوجود ضعف در کیفیت:

۱. مایکروسافت (Microsoft)

حوزه فعالیت: توسعه نرم افزار و سیستم عامل

محصولات: ویندوز، آفیس، آژور

ایراد: وجود مشکلات امنیتی و تجربه کاربری نه‌چندان ایده‌آل در برخی نسخه‌ها
مایکروسافت با وجود عرضه نسخه‌هایی از ویندوز که از نظر عملکرد و امنیت دچار مشکل بودند (مانند ویندوز ویستا)، همچنان توانست با عرضه‌ی نسخه‌های بهبود یافته جایگاه خود را حفظ کند.

۲. فیسبوک (Meta)

حوزه فعالیت: شبکه‌های اجتماعی و فناوری

محصولات: فیسبوک، اینستاگرام، واتساپ

ایراد: نگرانی‌های مربوط به حریم خصوصی و داده‌های کاربران
با وجود رسوایی‌های مربوط به حریم خصوصی، فیسبوک توانست با ارائه ویژگی‌های جدید و خرید پلتفرم‌های محبوبی مانند اینستاگرام، همچنان جزو شرکت‌های موفق باقی بماند.

۳. تسلا (Tesla)

حوزه فعالیت: خودروسازی الکتریکی و انرژی پاک

محصولات: خودروهای الکتریکی، پنل‌های خورشیدی

ایراد: مشکلات مربوط به کنترل کیفیت در تولید خودرو
با وجود گزارش‌های متعدد از مشکلات تولیدی و کیفیتی در خودروهای تسلا، این شرکت توانسته است با نوآوری‌های خود و محبوبیت برند، به فعالیت خود ادامه دهد.

بر اساس مدل مک‌مال، این شرکت‌ها سه فاکتور مهم کیفیت را نادیده گرفتند:

- امنیت (Security): مایکروسافت و فیسبوک در برخی محصولات خود با چالش‌های امنیتی و حفاظت از داده‌های کاربران روبرو بوده‌اند.

- قابلیت اطمینان (Reliability): تسلا به دلیل مشکلات کنترل کیفیت، در برخی مواقع با کاهش اعتماد کاربران مواجه شده است.
- پایداری (Stability): برخی نسخه‌های نرم‌افزاری، مانند ویندوز ویستا، مشکلات زیادی از نظر عملکردی و پایداری داشتند.

این شرکت‌ها هزینه‌های مختلفی را متحمل شدند اما توانستند آن‌ها را جبران کنند:

- هزینه‌ی مالی: فیسبوک و مایکروسافت جریمه‌های سنگینی بابت نقض حریم خصوصی و ضعف‌های امنیتی پرداخت کردند.
- هزینه‌ی از دست دادن اعتماد کاربران: تسلا با وجود مشکلات کیفیتی، توانسته با نوآوری و تبلیغات قوی، برند خود را حفظ کند.
- هزینه‌ی توسعه‌ی مجدد: مایکروسافت با عرضه نسخه‌های بهبودیافته، مانند ویندوز ۱۰، توانست نواقص محصولات قبلی را برطرف کند.

شکست برخی شرکت‌ها و موفقیت برخی دیگر با وجود ضعف‌های کیفیتی نشان می‌دهد که عواملی مانند نوآوری، بازاریابی و پاسخ سریع به نیازهای کاربران می‌تواند بر ضعف‌های فنی غلبه کند. شرکت‌هایی که به موقع ضعف‌های خود را اصلاح کنند، همچنان می‌توانند جایگاه خود را حفظ کنند.

ج

در دنیای نرم‌افزار و فناوری، کیفیت یکی از مهم‌ترین عوامل موفقیت شرکت‌ها محسوب می‌شود. بسیاری از شرکت‌های مطرح، با وجود داشتن بازار گسترده و نوآوری، به دلیل نادیده گرفتن برخی از فاکتورهای کلیدی کیفیت، متحمل شکست‌های سنگینی شده‌اند. در این مقاله به بررسی سه شرکت مشهور که به علت ضعف در طراحی و کیفیت نرم‌افزاری شکست خوردند، می‌پردازیم. همچنین به بررسی شرکت‌هایی می‌پردازیم که با وجود نادیده گرفتن برخی فاکتورهای کیفیت، همچنان در بازار موفق هستند.

معاوضات انجام شده توسط این شرکت‌ها

شرکت‌های موفق که برخی فاکتورهای کیفیت را نادیده گرفته‌اند، این کار را در ازای مزایای دیگری انجام داده‌اند:

- مایکروسافت: کاهش هزینه‌های توسعه و تسریع در عرضه نسخه‌های جدید نرم‌افزار.
- فیسبوک: حفظ مدل تبلیغاتی سودآور به بهای نگرانی‌های مربوط به حریم خصوصی.
- تسلا: افزایش سرعت نوآوری در فناوری‌های خودرویی به بهای کنترل کیفیت نامناسب.

دلایل ارائه‌ی محصولات پر ایراد

سه دلیل اصلی که شرکت‌ها معمولاً محصولات پر ایراد ارائه می‌دهند، شامل موارد زیر است:

- فشار زمانی برای ورود به بازار: رقابت شدید باعث می‌شود شرکت‌ها محصولات ناپایدار را برای زودتر در دسترس قرار دادن، عرضه کنند.
- کاهش هزینه‌های توسعه: بعضی شرکت‌ها برای بهینه‌سازی هزینه‌ها، تست‌های کیفی کافی را انجام نمی‌دهند.
- تمرکز بر ویژگی‌های جدید به جای بهبود کیفیت: بسیاری از شرکت‌ها برای جذب مشتریان جدید، روی نوآوری تمرکز کرده و کیفیت را به عنوان اولویت دوم در نظر می‌گیرند.

شکست برخی شرکت‌ها و موفقیت برخی دیگر با وجود ضعف‌های کیفیتی نشان می‌دهد که عواملی مانند نوآوری، بازاریابی و پاسخ سریع به نیازهای کاربران می‌تواند بر ضعف‌های فنی غلبه کند. شرکت‌هایی که به موقع ضعف‌های خود را اصلاح کنند، همچنان می‌توانند جایگاه خود را حفظ کنند.

د

درستی (Correctness)

مثال: اطمینان از نمایش صحیح اطلاعات محصولات و خدمات.

راهکار پیاده‌سازی: استفاده از اعتبارسنجی داده‌ها در فرم‌های ورودی و بررسی صحت اطلاعات بارگذاری شده از طریق فیلترها و قوانین اعتبارسنجی.

قابلیت اطمینان (Reliability)

مثال: سرورهای پایدار و جلوگیری از قطعی‌های ناگهانی.

راهکار پیاده‌سازی: بهره‌گیری از معماری متمرکز بر تحمل خطا (Fault-Tolerance) و پایگاه داده‌های توزیع شده برای افزایش قابلیت اطمینان.

کارایی (Efficiency)

مثال: بارگذاری سریع صفحات و نمایش بهینه تصاویر محصولات.

راهکار پیاده‌سازی: بهینه‌سازی درخواست‌های پایگاه داده، استفاده از فشرده‌سازی تصاویر، و بهره‌گیری از سامانه کشینگ.

قابلیت استفاده (Usability)

مثال: طراحی رابط کاربری ساده و قابل فهم برای کاربران مبتدی.

راهکار پیاده‌سازی: استفاده از اصول طراحی رابط کاربری کاربرمحور (User-Centered-Design) و انجام تست‌های تجربه کاربری.

نگهداری پذیری (Maintainability)

مثال: امکان بروزرسانی آسان نرم افزار و افزودن قابلیت‌های جدید.

راهکار پیاده‌سازی: پیاده‌سازی معماری ماژولار، مستندسازی کامل کدها، و استفاده از اصول برنامه‌نویسی شیء‌گرا.

قابلیت انتقال (Portability)

مثال: اجرای نرم افزار بر روی سیستم‌عامل‌های مختلف مانند اندروید و iOS.

راهکار پیاده‌سازی: استفاده از فریمورک‌های چندسکویی مانند Flutter یا React-Native برای توسعه هم‌زمان بر روی پلتفرم‌های مختلف.

در صورتی که مجبور به ارائه‌ی نرم‌افزاری با حداقل امکانات باشیم، باید فاکتورهایی را انتخاب کنیم که بیشترین تأثیر را در عملکرد و رضایت کاربران داشته باشند. این فاکتورها عبارت‌اند از:

- درستی (Correctness): اگر اطلاعات نادرست نمایش داده شوند، کاربران اعتماد خود را از دست خواهند داد.
- قابلیت اطمینان (Reliability): اگر نرم‌افزار پایدار نباشد و دچار قطعی شود، کاربران آن را ترک خواهند کرد.
- قابلیت استفاده (Usability): تجربه‌ی کاربری خوب باعث جذب کاربران و افزایش تعامل آن‌ها می‌شود.
- کارایی (Efficiency): سرعت و عملکرد بهینه برای کاربران حیاتی است، زیرا نرم‌افزار کند تجربه‌ی کاربری نامطلوبی را ایجاد می‌کند.