

1. فلسفه تضمین کیفیت در Metrogo

در پروژه Metrogo، تضمین کیفیت صرفاً یک فعالیت فنی برای «کم کردن باگ» نیست، بلکه یک الزام حیاتی برای بقای محصول است Metrogo. در نقطه‌ای از تجربه کاربر قرار دارد که خطا قابل چشم‌پوشی نیست: لحظه عبور از گیت مترو و انجام پرداخت. در این نقطه، هر خطا مستقیماً به ازدحام، نارضایتی عمومی، بی‌اعتمادی به سیستم و حتی مداخله اپراتور انسانی منجر می‌شود.

به همین دلیل، کیفیت در Metrogo به عنوان «قابلیت اعتماد در شرایط واقعی» تعریف می‌شود، نه صرفاً عملکرد صحیح در شرایط ایده‌آل. سیستم باید در ساعات اوج ترافیک، با اینترنت ناپایدار، روی گوشی‌های میان‌رده و در تعامل با زیرساخت‌های بیرونی (PSP)، سامانه شهری (همچنان رفتار قابل پیش‌بینی داشته باشد).

فلسفه QA در Metrogo بر این فرض استوار است که:

هر فیچر جدید بدون تست، یک بدهی پنهان است

هر بدهی پنهان، در سیستم پرداخت شهری چند برابر هزینه ایجاد می‌کند

کیفیت اگر زود مدیریت نشود، دیرتر قابل اصلاح نخواهد بود

2. استراتژی کلی تست در Metrogo

استراتژی تست Metrogo به صورت لایه‌ای طراحی شده است تا ریسک‌ها در نزدیکترین نقطه به منبع خطا شناسایی شوند. این استراتژی ترکیبی از تست‌های خودکار و سناریوهای دستی هدفمند است که متناسب با ماهیت MVP و محدودیت منابع طراحی شده‌اند.

در Metrogo، تست‌ها صرفاً برای «اثبات درست کار کردن سیستم» نوشته نمی‌شوند، بلکه برای پاسخ به این سؤال کلیدی هستند:

«اگر فردا این بخش را تغییر دهیم، چه چیزی ممکن است بشکند؟»

به همین دلیل، تمرکز تست‌ها روی مسیرهای بحرانی (Critical User Journeys) است، نه روی پوشش ظاهری همه کدها. مسیرهایی مانند:

تولید QR برای ورود به مترو

اعتبارسنجی پرداخت

جلوگیری از دوبار استفاده از یک مجوز

مدیریت خطا در صورت قطع ارتباط

3. تست واحد (Unit Testing) در Metrogo

تست واحد در Metrogo اولین خط دفاعی در برابر خطاهای منطقی است. این تست‌ها منطق داخلی سیستم را بدون وابستگی به شبکه، دیتابیس یا سرویس‌های شهری بررسی می‌کنند. هدف اصلی این تست‌ها جلوگیری از ورود خطاهای ساده ولی پرریسک به لایه‌های بالاتر سیستم است.

در Metrogo، تست واحد به‌طور خاص روی موارد زیر تمرکز دارد:

قوانین اعتبارسنجی پرداخت

منطق تولید و انقضای QR

بررسی وضعیت مجوز کاربر

تصمیم‌گیری‌های شرطی مرتبط با سناریوهای خطا

هر تغییری که رفتار یکی از این منطق‌ها را تغییر دهد، بدون تست واحد قابل پذیرش نیست. این سیاست باعث می‌شود توسعه‌دهنده قبل از اعمال تغییر، دقیقاً بداند چه رفتاری انتظار می‌رود و چه رفتاری نباید تغییر کند.

4. پوشش تست (Test Coverage) و معیارهای آن

در پروژه Metrogo، Test Coverage به عنوان یک ابزار کنترلی استفاده می شود، نه یک هدف نمایشی. هدف رسیدن به عدد بالا برای گزارش نیست، بلکه تمرکز روی پوشش بخش هایی است که خرابی آنها بیشترین آسیب را وارد می کند.

هدف پوشش تست واحد برای منطق های بحرانی Metrogo حداقل ۷۰٪ تعریف شده است. این عدد به صورت آگاهانه انتخاب شده تا بین سرعت توسعه MVP و کنترل ریسک تعادل برقرار شود. کاهش ناگهانی Coverage به عنوان یک هشدار در نظر گرفته می شود و پیش از انتشار نسخه جدید باید بررسی شود.

گزارش Coverage به صورت خودکار در فرآیند CI تولید و ذخیره می شود و به عنوان Test Evidence قابل ارائه است. این گزارش ها نشان می دهند که تست ها صرفاً در حد ادعا نیستند و واقعاً اجرا شده اند.

5. تست یکپارچه (Integration Testing) در Metrogo

تست‌های یکپارچه در Metrogo برای بررسی تعامل بین اجزای اصلی سیستم طراحی شده‌اند؛ به‌ویژه در نقاطی که Backend، منطق پرداخت و وضعیت کاربر به هم می‌رسند. این تست‌ها بررسی می‌کنند که قراردادهای بین سرویس‌ها به‌درستی رعایت شده و تغییر در یک بخش باعث رفتار غیرمنتظره در بخش دیگر نشده است.

در Metrogo، تست‌های یکپارچه روی سناریوهایی مانند زیر تمرکز دارند:

ارسال درخواست پرداخت و دریافت پاسخ معتبر

هماهنگی بین وضعیت کاربر و مجوز عبور

مدیریت خطا در صورت پاسخ نامعتبر از سرویس بیرونی

این تست‌ها نقش مهمی در کاهش ریسک خطاهایی دارند که در تست واحد قابل شناسایی نیستند.

6. تست End-to-End (E2E) در پروژه Metrogo

در پروژه Metrogo، تست End-to-End حیاتی‌ترین لایه تضمین کیفیت محسوب می‌شود، زیرا این تست‌ها دقیقاً همان تجربه‌ای را شبیه‌سازی می‌کنند که کاربر نهایی در محیط واقعی مترو با آن مواجه است. برخلاف تست واحد یا یکپارچه که رفتار اجزای سیستم را به‌صورت جداگانه بررسی می‌کنند، تست E2E رفتار «سیستم به‌عنوان یک کل» را می‌سنجد.

در Metrogo، کاربر در یک بازه زمانی بسیار کوتاه (چند ثانیه) انتظار دارد بتواند وارد مترو شود. هیچ فرصتی برای retry پیچیده، پیام خطای طولانی یا رفتار مبهم وجود ندارد. بنابراین تست‌های E2E دقیقاً روی این لحظات حساس متمرکز شده‌اند.

سناریوی اصلی E2E در Metrogo شامل مراحل زیر است: کاربر اپلیکیشن را باز می‌کند، وضعیت حساب او بررسی می‌شود، QR تولید می‌شود، QR توسط گیت یا سیستم شهری خوانده می‌شود، اعتبار پرداخت تأیید می‌گردد و در نهایت مجوز عبور صادر می‌شود. هر شکست در هرکدام از این مراحل باید قابل تشخیص، قابل لاگ‌گیری و قابل مدیریت باشد.

هدف تست E2E در این پروژه پاسخ به این سؤال است: «آیا Metrogo در شرایط نزدیک به واقعیت، بدون دخالت انسانی، کار می‌کند؟»

به همین دلیل، تعداد تست‌های E2E محدود اما بسیار هدفمند انتخاب شده‌اند. این تست‌ها روی مسیرهای غیرحیاتی نوشته نمی‌شوند، زیرا هزینه نگهداری بالایی دارند و ارزش یادگیری پایینی ایجاد می‌کنند.

7. تست رگرسیون (Regression Testing) در Metrogo

با رشد Metrogo و اضافه شدن تدریجی قابلیت‌ها، ریسک بازگشت باگ‌های قدیمی به شدت افزایش می‌یابد. در سیستم‌های پرداخت شهری، بازگشت یک باگ قدیمی می‌تواند بسیار پرهزینه‌تر از معرفی یک باگ جدید باشد، زیرا معمولاً در شرایط واقعی و پرتراфик ظاهر می‌شود.

تست رگرسیون در Metrogo به عنوان یک سیاست سیستماتیک تعریف شده است، نه یک فعالیت موردی. هر باگی که شناسایی و رفع می‌شود، الزاماً باید منجر به اضافه شدن یک تست رگرسیون شود. این سیاست تضمین می‌کند که:

هزینه هر باگ فقط یکبار پرداخت شود

سیستم به مرور زمان پایدارتر شود

تیم از تکرار خطاهای گذشته درس بگیرد

در Metrogo، تست‌های رگرسیون ترکیبی از تست‌های واحد، یکپارچه و E2E هستند که روی رفتارهای تثبیت شده سیستم تمرکز دارند؛ به ویژه رفتارهایی که مستقیماً به پرداخت، اعتبار کاربر و عبور از مترو مرتبط‌اند.

8. تست امنيت (Security Testing) با تمرکز بر پرداخت شهری

امنيت در Metrogo یک ویژگی جانبی نیست، بلکه بخشی از کیفیت پایه سیستم است. هر نقص امنیتی در یک سیستم پرداخت شهری می‌تواند منجر به سوءاستفاده مالی، از دست رفتن اعتماد عمومی و مداخله نهادهای رگولاتوری شود.

تست‌های امنیتی در Metrogo روی ریسک‌های مشخص و واقعی تمرکز دارند، نه روی چک‌لیست‌های تئوریک. مهمترین محورهای تست امنيت شامل موارد زیر است:

جلوگیری از دوبار استفاده از یک QR (Double Spending)

جلوگیری از Replay Attack

کنترل دسترسی به API ها

اعتبارسنجی ورودی‌ها و جلوگیری از تزریق داده مخرب

این تست‌ها به‌صورت ترکیبی از تست خودکار و بررسی‌های دستی انجام می‌شوند. در سطح MVP ، هدف رسیدن به امنيت مطلق نیست، بلکه جلوگیری از ریسک‌های فاجعه‌بار است. هر ضعف امنیتی شناسایی‌شده به‌عنوان باگ بحرانی دسته‌بندی می‌شود و بدون رفع آن، انتشار نسخه جدید مجاز نیست.

9. فرآیند مدیریت باگ (Bug Management Policy) در Metrogo

در Metrogo ، باگ صرفاً یک خطای فنی محسوب نمی‌شود، بلکه یک ریسک عملیاتی و اعتباری است. به همین دلیل، مدیریت باگ به‌صورت ساختاریافته و با اولویت‌بندی شفاف انجام می‌شود.

باگ‌ها بر اساس دو معیار اصلی دسته‌بندی می‌شوند:

شدت (Severity): میزان آسیب فنی

اثر (Impact): میزان تأثیر بر کاربر و عملیات مترو

باگ‌هایی که منجر به اختلال در پرداخت، عبور یا اعتماد کاربر شوند، به‌عنوان باگ بحرانی شناخته می‌شوند و هیچ فیچر جدیدی جایگزین رفع آن‌ها نخواهد شد. سیاست تیم این است که سرعت توسعه هرگز نباید کیفیت مسیر اصلی کاربر را قربانی کند.

10. فرآیند Release و کنترل کیفیت انتشار

در Metrogo ، انتشار نسخه جدید یک رویداد فنی ساده نیست، بلکه یک تصمیم پرریسک است که باید با شواهد پشتیبانی شود. هیچ نسخه‌ای بدون عبور موفق از تست‌های تعریف‌شده وارد محیط نزدیک به عملیاتی نمی‌شود.

فرآیند Release شامل مراحل زیر است: اجرای خودکار تست‌های واحد، یکپارچه و رگرسیون، بررسی گزارش Coverage ، مرور تغییرات بحرانی و تأیید نهایی. اگر هرکدام از این مراحل با شکست مواجه شود، انتشار متوقف می‌شود.

این فرآیند به تیم اجازه می‌دهد با اطمینان بیشتری تغییرات را اعمال کند و از ورود خطاهای پرهزینه به محیط واقعی جلوگیری نماید.

11. Test Evidence و گزارش‌های اجرای تست

یکی از تفاوت‌های اصلی این پروژه با رویکردهای تزئینی QA، تأکید بر شواهد قابل ارائه است. در Metrogo، هر ادعای مربوط به تست باید با داده و گزارش پشتیبانی شود.

گزارش‌های اجرای تست، Coverage، شکست‌ها و موفقیت‌ها ذخیره می‌شوند و به‌عنوان Test Evidence قابل ارائه هستند. این شواهد نشان می‌دهند که کیفیت به‌صورت سیستماتیک مدیریت می‌شود، نه بر اساس اعتماد شفاهی.

وجود این شواهد به تیم اجازه می‌دهد:

تصمیم‌های آگاهانه‌تری بگیرد

ریسک‌ها را زودتر شناسایی کند

و به‌ذی‌نفعان نشان دهد که Metrogo برای محیط واقعی آماده می‌شود

جمع‌بندی نهایی (Metrogo-Specific)

استراتژی QA و تست در Metrogo بر این اصل بنا شده که کیفیت پیش‌نیاز رشد است، نه مانع آن. در یک سیستم پرداخت شهری، نبود تست به‌معنای انتقال ریسک به کاربر و شهر است. این سند نشان می‌دهد که تیم Metrogo این ریسک را آگاهانه مدیریت می‌کند، نه اینکه آن را نادیده بگیرد.