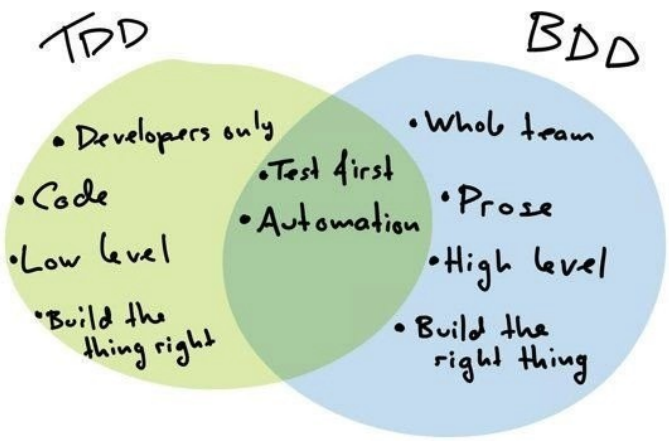
تمرین هفتم آزمون نرم‌افزار[1]

1) آشنایی با توسعه‌ی مبتنی بر رفتار (Behavior Driven Development)



می‌دانیم بعضی از منتقدان توسعه‌ی مبتنی بر آزمون (TDD) راهکار توسعه‌ی مبتنی بر رفتار را پیشنهاد می‌دهند. در این روش تمرکز آزمون‌ها بیشتر روی سناریوهایی است که ارزش کاربردی (Business Value) دارند.

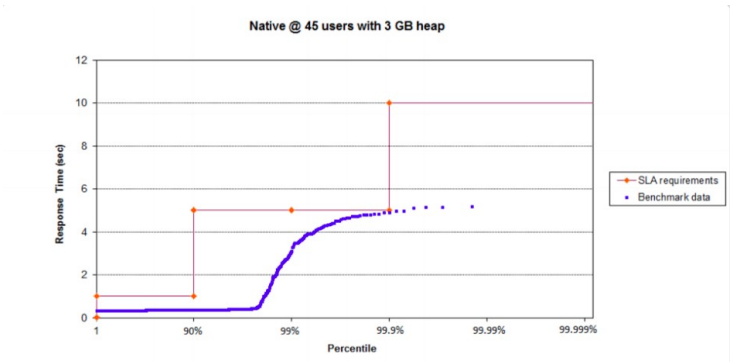
می‌خواهیم با استفاده از ابزار Cucumber که یک چهارچوب برای آزمون مبتنی بر رفتار است، آزمون‌هایی توسعه دهیم.

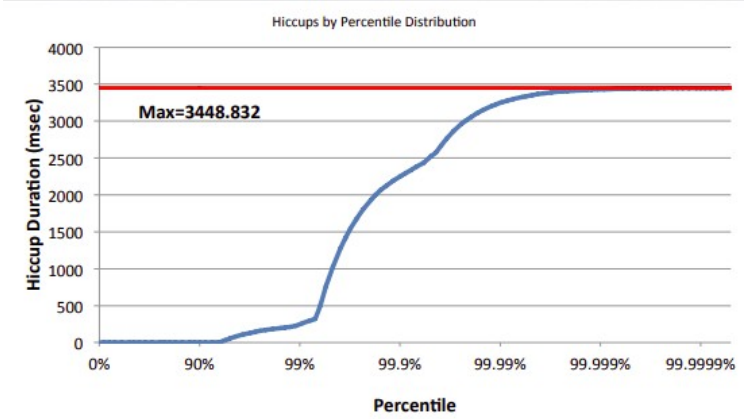
1. برای کلاس PetService آزمون‌هایی مبتنی بر رفتار توسعه دهید. کافی است هر کارکرد این سرویس در یک سناریو فراخوانی شده باشد.
2. یک مرحله (Stage) به خط‌لوله‌ی یکپارچه‌سازی (CI Pipeline) بیافزایید که فقط آزمون مبتنی بر رفتار در آن به صورت پیوسته و خودکار، بعد از مراحل Build و Test انجام شود.

با اجرای این دستور در ترمینال یا در قسمت Maven در Intellij Idea می‌توانید آزمون رفتار را اجرا نمایید:

```
mvn compiler:testCompile
mvn -Dtest="bdd.BDDEntryPointAcceptanceTest" test
```

2) مفاهیم آزمون کارایی





در این قسمت قصد داریم پس از مرور اجمالی مفاهیم آزمون کارایی، یک آزمون برای این کار توسعه دهیم.

1. راجع به مفهوم Service Level Agreement تحقیق نموده و معنی نمودار اول را بیان نمایید.
2. فرض کنید نمودار خیالی دوم توزیع تاخیر سامانه‌ی ut.ac.ir را نشان می‌دهد (البته که در واقع چنین نیست). چقدر احتمال دارد یک کاربر در یک مراجعه به سامانه، تاخیری بیشتر از یک ثانیه تجربه کند؟ (راهنمایی: این آدرس را در حالی که Network Recording مرورگر فعال است فراخوانی کنید و تعداد درخواست‌ها را در یک مراجعه مشاهده نمایید).
3. در گزارشی خیالی آمده است: «تاخیر ۹۹٪ percentile این سامانه زیر نیم ثانیه است». بر اساس پاسخ سوال قبل، آیا این ادعا برای تضمین کیفیت این سامانه کافی است؟ چرا؟

---

1 طراح: کیارش آذرنیا

3) آشنایی با آزمون کارایی (Performance Test)

1. تفاوت Stress Test و Load Test که انواع آزمون کارایی هستند را بیان نمایید.
2. دو تا از APIهای PetClinic را به دلخواه انتخاب نمایید و با استفاده از ابزار K6 برای آن‌ها آزمون کارایی طراحی و اجرا نمایید. **(*)**

جهت سهولت انجام کار می‌توانید سرور و تست، هر دو را همزمان روی سیستم لوکال اجرا نمایید (هرچند می‌دانیم این کار نتایج تست را تحت تاثیر می‌گذارد). اسکریپت‌های پیاده‌سازی شده را در فلدری با نام performance در مخزن کامیت نمایید و تصاویر نتایج تست خود را در گزارش درج نمایید. هر پیکربندی‌ای برای stageها و thresholdها بنا به طراحی تست شما پذیرفته است.

نکات

•متن کدها را از مخزن تمرینات دریافت نمایید و قبل از انجام از بروز بودن آن نسبت به شاخه‌ی مربوط به این تمرین (ca7) اطمینان حاصل کنید. توجه نمایید که باید شاخه‌ی تمرین را در مخزنی که قبلا ساخته‌اید و دسترسی نوشتن را به کاربر uttest داده‌اید، بروزرسانی نمایید.

•توجه کنید که خط لوله‌ی یکپارچگی پیوسته (CI Pipeline) برای کامیت شما موفق باشد.

•موارد ستاره‌دار (*) امتیازی و نمره این تمرین تا ۱۱۰٪ قابل محاسبه است.

•هم‌افزایی (اشتراک ایده، لینک‌های مفید، مشارکت در بحث‌های مربوطه، کمک به حل مشکلاتی مثل کانفیگ لازم برای IDE) در گروه کلاس توصیه می‌شود.

- https://bitbucket.org/YOUR_NAME/pet-clinic/src/COMMIT_ID

لینک‌های مفید

- Cucumber Tutorial https://cucumber.io/docs/guides/10-minute-tutorial
- Gherkin Language https://cucumber.io/docs/gherkin/reference/
- K6 Docs https://github.com/loadimpact/k6
- Informative Talk on Performance Testing By Gil Tene https://www.dideo.ir/v/yt/_fiV3JUXylM/how-i-learned-to-stop-worrying-and-love-misery-by