Universidade de Brasília Faculdade UnB Gama

Universidade de Brasília

Testes de Software - 2020/2

Trabalho Final - Entrega 1

Grupo 5

Andre Aben Athar de Freitas - 17/0056155
João Gabriel de Campos de Matos - 18/0042238
João Vitor Lopes de Farias - 18/0020251
José Aquiles Guedes de Rezende - 16/0010331
Lucas Medeiros Rosa - 17/0039803
Luis Gustavo Avelino de Lima Jacinto - 15/0016310
Marcos Felipe de Almeida Souza - 18/0066382
Pedro Vítor de Salles Cella - 17/0113060
Sara Conceição de Sousa Araújo Silva - 16/0144752

1. Tema e contextualização

O aplicativo Stay Safe foi um projeto desenvolvido por alunos das disciplinas de Métodos de Desenvolvimento de Software e Engenharia de Produto de Software do curso de graduação em Engenharia de Software da Universidade de Brasília.

O Stay Safe é um aplicativo mobile para Android que mostra, através de mapas e estatísticas, informações sobre a segurança de um lugar. Os dados exibidos são obtidos das Secretarias de Segurança Pública (SSP) e dos usuários do aplicativo que podem contribuir registrando uma ocorrência da qual foram vítimas ou testemunhas, e avaliando a segurança dos bairros que conhecem.

O aplicativo tem seu *backend* composto por dois micro-serviços. O primeiro deles é o *User Service*, uma *API REST,* que trata todos os dados dos usuários, ocorrências reportadas e avaliações de bairros feitas, enviando-os para serem armazenados em um banco de dados relacional.

O segundo micro-serviço é o *Secretary Service* que obtém as estatísticas de crimes dos sites das SSPs de São Paulo e do Distrito Federal por meio de *crawlers* e as armazena em um banco de dados não relacional. Nesse serviço também ficam os dados populacionais das cidades de cada estado. Esses dados são expostos em uma API do serviço para que o *front-end* tenha acesso. Por fim, o aplicativo usa a API do Google Maps para exibir os mapas das funcionalidades.

Para limitar o escopo de acordo com o tempo disponível, este trabalho está focado apenas na funcionalidade de mapa de calor das estatísticas.

2. Problema

É preciso, primeiro, definir o que pode ser considerado problema de desempenho para então dizer se há problemas desse tipo na funcionalidade em questão e como mitigá-los.

2.1 Objetivos

Geral:

- Testar as etapas da funcionalidade de forma individual, a fim de identificar o pior desempenho dentre elas e desenvolver uma solução para este ponto.

Específicos:

- Avaliar o desempenho para carregar as coordenadas;
- Avaliar o desempenho para fazer requisição na API;
- Avaliar o desempenho para plotar gráfico.

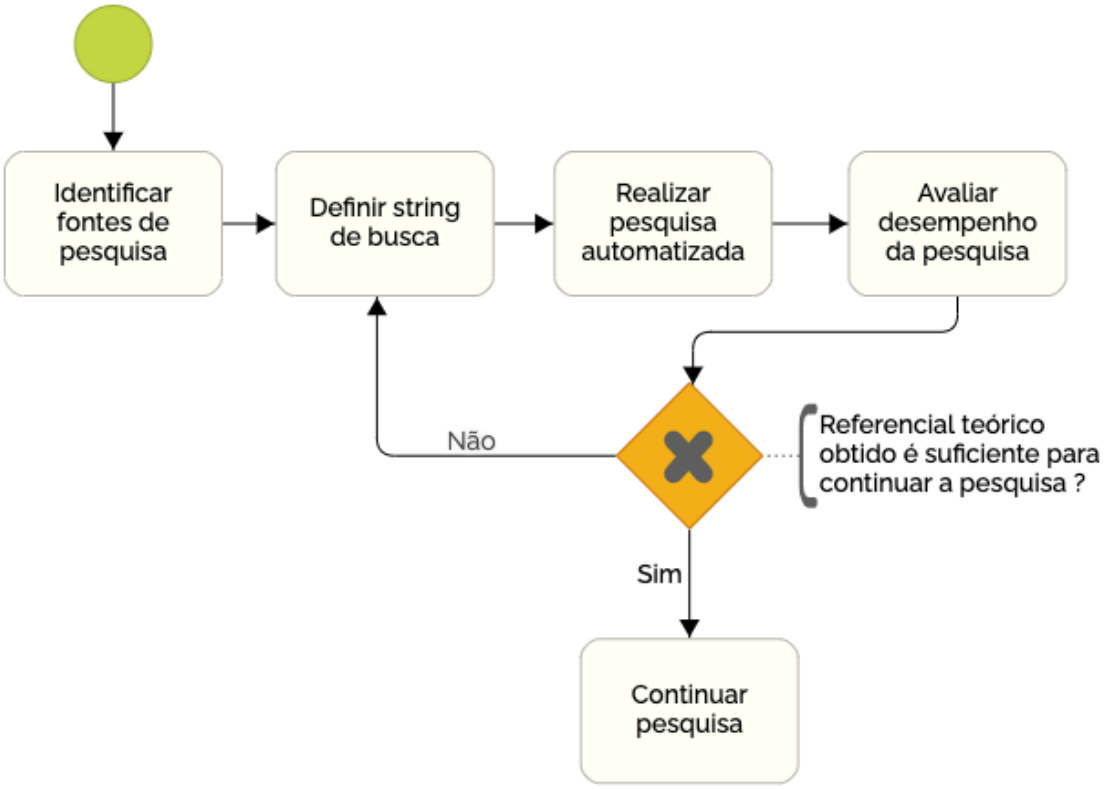
2.2 Questões de Pesquisa

- Se encontrar um defeito no desempenho, como resolver este defeito?
- O que pode ser considerado um defeito no desempenho?

3. Referencial Teórico

O objetivo do referencial teórico é pesquisar e analisar estudos acadêmicos que abordam possíveis soluções para o tema escolhido no trabalho final, no caso tem-se o objetivo de analisar o desempenho de uma plotagem gráfica de um mapa utilizando de um aplicativo.

Assim, a estratégia para obtenção do referencial teórico pode ser visualizada na imagem a seguir:



HEFLO

1. **Identificar fontes de pesquisa:** buscar na literatura para identificação dos locais de publicação relevantes;
2. **Definir *string* de busca:** Podem ser definidas de modo subjetivo (com base em conhecimento de domínio e experiências passadas) ou objetivo (uma análise de frequência das informações de citação dos artigos é realizada, seguida de uma análise estatística das palavras ou frases mais frequentes);
3. **Realizar pesquisa automatizada:** as bibliotecas digitais são pesquisadas usando as *strings* definidas;

4. **Avaliar desempenho da pesquisa:** os resultados da pesquisa são avaliados para garantir que a pesquisa foi suficiente para continuação do presente trabalho.

A pesquisa utilizará as bases *Scopus, ACM Digital Library*, *IEEE Xplore* e SciELO, utilizando o seguintes critérios de inclusão (CI) e exclusão (CE) para filtrar os artigos

- **CI 1:** Artigos escritos em inglês ou português;
- **CI 2:** Artigos de acesso gratuito ou de acesso concedido pela universidade;
- **CI 3:** Artigos que tratem do tema de avaliação de defeitos no desempenho de um software;
- **CE 1:** Artigos que fogem da concordância com a *string* de busca.

4. Avaliar o desempenho para obter as coordenadas

Para que cada cidade no mapa de calor esteja com a cor correta é preciso que todas estejam bem delimitadas então é preciso ter as coordenadas que formam os limites geográficos de cada cidade.

A forma que o *front-end* da aplicação obtém as coordenadas pode ser problemática, principalmente para o estado de São Paulo que possui muitos municípios. Um número grande de coordenadas pode levar um tempo não satisfatório para o usuário final.

O resultado esperado da avaliação do desempenho ao carregar as coordenadas é rastrear possíveis defeitos nessa operação, assim como propor possíveis soluções baseadas nos resultados dos testes, a fim de propiciar o melhor o uso do aplicativo. Os dados e informações serão derivados dos resultados das aplicações dos artefatos no contexto de obtenção das coordenadas dos limites das cidades.

Algumas métricas ajudarão a avaliar os resultados e na elaboração de uma boa solução:

- Métricas de comportamento em relação a tempo
  - Tempo de Resposta
  - Vazão
- Métricas de comportamento em relação a recursos
  - Utilização de recursos de memória

4.1 Estudos empíricos que serão realizados

A execução de algumas estratégias alternativas pode ajudar na escolha da melhor solução:

1. O front–end fazer requisições para uma API aberta de mapas

2. Armazenar a lista de coordenadas no banco de dados do StaySafe e criar *endpoints* para acessá-la.
3. Armazenar as coordenadas em um arquivo estático.

Para cada estratégia serão realizados testes e serão obtidas as métricas descritas na seção 8 para que elas sejam comparadas.

5. Avaliar o desempenho para fazer requisição na API

A avaliação do desempenho ao realizar requisição na API pretende rastrear os possíveis defeitos encontrados neste processo, de maneira que os resultados de testes possibilitem o desenvolvimento de possíveis soluções. Os dados utilizados na avaliação serão obtidos a partir dos resultados das aplicações dos artefatos no contexto de execução de requisição à API.

As métricas a seguir ajudarão a avaliar os resultados e na elaboração de uma boa solução sobre eficiência de desempenho:

Métricas de comportamento em relação a tempo

- Tempo de Resposta
- Tempo de Espera

- Métrica relacionados a experiência do usuário
  - Latência

5.1 Estudos empíricos que serão realizados

Possíveis alternativas de testes sobre a requisição na API que podem auxiliar no desenvolvimento da solução:

1. Executar a aplicação sob monitoramento de uma ferramenta que permita a verificação do atraso no tempo de resposta.
2. Realizar testes na requisição da API quando o sistema estiver trabalhando com pouco e grande volume de dados.

Para cada estratégia serão realizados testes e serão obtidas as métricas descritas na seção 8 para que elas sejam comparadas.

6. Avaliar o desempenho para plotar gráfico

Considerando possíveis gargalos nos motores gráficos, ao plotar as informações fornecidas pelo aplicativo, se fazem necessários testes tanto de uso de recursos quanto de tempo de resposta. Com os resultados desses testes em mãos podemos descartar ou confirmar as hipóteses de gargalo de desempenho no módulo em questão.

Para avaliar o desempenho desse módulo serão usadas as seguintes métricas:

- Métricas de comportamento em relação a tempo
  - Tempo de Resposta
- Métricas relacionadas ao uso de recursos

- Utilização CPU
- Utilização Memória

6.1 Estudos empíricos que serão realizados

Dentre as possibilidades de testes em relação ao motor gráfico da aplicação podemos usar alternativas como:

1. Executar testes usando o front-end em um ambiente neutro e monitorado.
2. Realizar os testes com e sem a aceleração via placa de vídeo dedicada (GPU) e descrever as mudanças de no resultado

Para cada estratégia serão realizados testes e serão obtidas as métricas descritas na seção 8 para que elas sejam comparadas.

7. Proposta de solução

Para cada um dos objetivos específicos serão aplicados os seguintes tópicos.

7.1 Proposta de artefatos da pesquisa

Alguns artefatos podem ser úteis para avaliar o desempenho:

- Goal Question Metric - GQM (processo para abordagem de métricas em Engenharia de Software).
- Plano de testes (documento em que ficará registrada a abordagem sistemática - tipos de testes, quando executá-los, etc - para o teste de um sistema).

7.2 Proposta de aplicação dos Artefatos

Os artefatos são complementares entre si e podem ser aplicados simultaneamente. Os testes serão executados por membros do grupo em ambientes controlados e podem ajudar a responder às perguntas do GQM.

8. Métricas

8.1 Métricas de comportamento em relação ao tempo

8.1.1 Tempo de Resposta

Nome	Tempo de Resposta
Característica	Eficiência de Desempenho
SubCaracterística	Comportamento do Tempo
Descrição	Quanto tempo leva para plotar as informações fornecidas pelo aplicativo para o usuário

Função de Medição	• X = Tb - Ta • Tb = Tempo de recebimento da resposta. • Ta = Tempo da solicitação do usuário.
Método	1. Registrar o tempo gasto da solicitação até o recebimento da requisição. 2. Registrar tempo de resposta. 3. Calcular tempo de resposta. 4. Registrar resultados.

Interpretação	• Quanto menor melhor.
	  • 0,XXX segundos é péssimo
	  • 0,XXX segundos é regular
	  • 0,XXX segundos é bom
	  • 0,XXX segundos é excelente.

8.1.2 Tempo de Espera

Nome	Tempo de espera
Característica	Eficiência de Desempenho
SubCaracterística	Comportamento do Tempo
Descrição	Quanto tempo demora desde o envio de uma requisição, até a conclusão da mesma?
Função de Medição	$X = B - A$ • A = tempo quando se inicia um trabalho. • B = tempo que o trabalho é completo.
Método	1. Definir o tempo que uma requisição for enviada. 2. Definir o tempo em que a requisição foi concluída. 3. Realizar o cálculo da métrica e observar o tempo necessário para a requisição ser efetuada. 4. Registrar resultados.

8.1.3 Vazão

Nome	Vazão
Característica	Eficiência de Desempenho
SubCaracterística	Comportamento do Tempo
Descrição	Quantas tarefas podem ser realizadas por período de tempo?
Função de Medição	• A = Número de tarefas concluídas. • T = Período de tempo observado (minutos) = (A /T) * 100%
Método	1. Definir um período de tempo X. 2. Definir um número de tarefas a serem executadas. 3. Observar quantas tarefas são realizadas dentro do período de minutos definidos. 4. Registrar resultados.

Interpretação	• Quanto maior melhor.
	  • 0 < X <= 25% é péssimo
	  • 25% < X <= 50% é regular
	  • 50% < X <=75% é bom
	  • 75% < X <= 100% é excelente.

8.2 Métricas relacionadas ao uso de recursos

8.2.1 A Utilização de CPU

Nome	Utilização de CPU
Característica	Eficiência de Desempenho
SubCaracterística	Utilização de Recursos
Descrição	Qual a porcentagem do poder total de processamento que a tarefa está consumindo?
Função de Medição	• A = Quantidade de CPU utilizada ao se realizar uma tarefa. • B = Quantidade de CPU disponível. • X = (A/B) * 100 %

Método	1. Definir uma tarefa X 2. Verificar a quantidade de CPU disponível. 3. Analisar a quantidade de CPU gasta ao se realizar a tarefa X.
Interpretação	● Quanto maior melhor. ● 0 < X <= 25% é péssimo ● 25% < X <= 50% é regular ● 50% < X <=75% é bom ● 75% < X <= 100% é excelente.

8.2.2 A Utilização de memória

Nome	Utilização de memória
Característica	Eficiência de Desempenho
SubCaracterística	Utilização de Recursos
Descrição	Qual a utilização de memória (KB) do sistema ao se realizar uma determinada tarefa?
Função de Medição	● A = Quantidade de Memória utilizada ao se realizar uma tarefa. ● B = Quantidade de memória disponível. ● X = (A/B) * 100 %
Método	2. Definir uma tarefa X 3. Verificar a quantidade de memória disponível. 4. Analisar a quantidade de memória gasta ao se realizar a tarefa X.
Interpretação	● Quanto maior melhor. ● 0 < X <= 25% é péssimo ● 25% < X <= 50% é regular ● 50% < X <=75% é bom ● 75% < X <= 100% é excelente.

8.3 Métrica relacionados a experiência do usuário

8.3.1 Latência

Nome	Latência
Característica	Eficiência de Desempenho

SubCaracterística	Experiência do usuário
Descrição	Quanto é o atraso que ocorre quando ocorre a requisição de dados?
Função de Medição	• Como os atrasos muitas vezes podem ser indetectáveis ao olho humano, é preciso de uma ferramenta de monitoramento para detectar os possíveis atrasos.
Método	1. Definir qual ferramenta de monitoramento será utilizada. 2. Utilizar a ferramenta de monitoramento. 3. Registrar resultados.