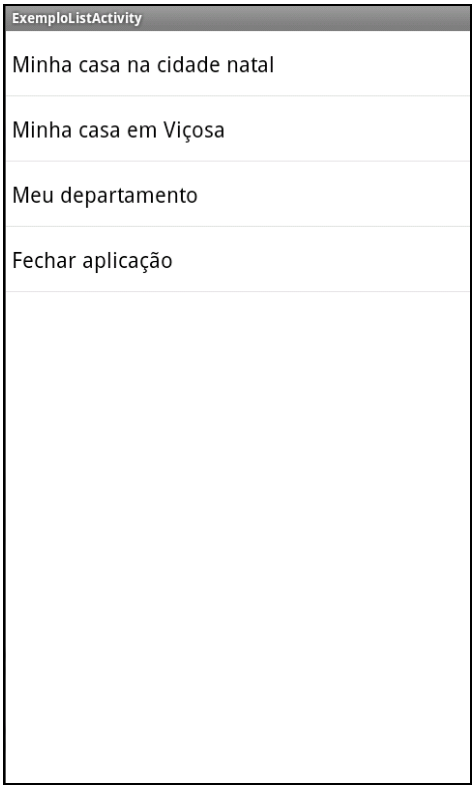
Universidade Federal de Viçosa – Departamento de Informática
INF311 – Programação para dispositivos móveis
Prof. Lucas Francisco da Matta Vegi

- Atividade prática 03 -

O objetivo desta atividade prática é desenvolver um aplicativo que exiba alguns pontos geográficos importantes relacionados ao seu programador. Este aplicativo deve conter um menu principal com as seguintes opções:

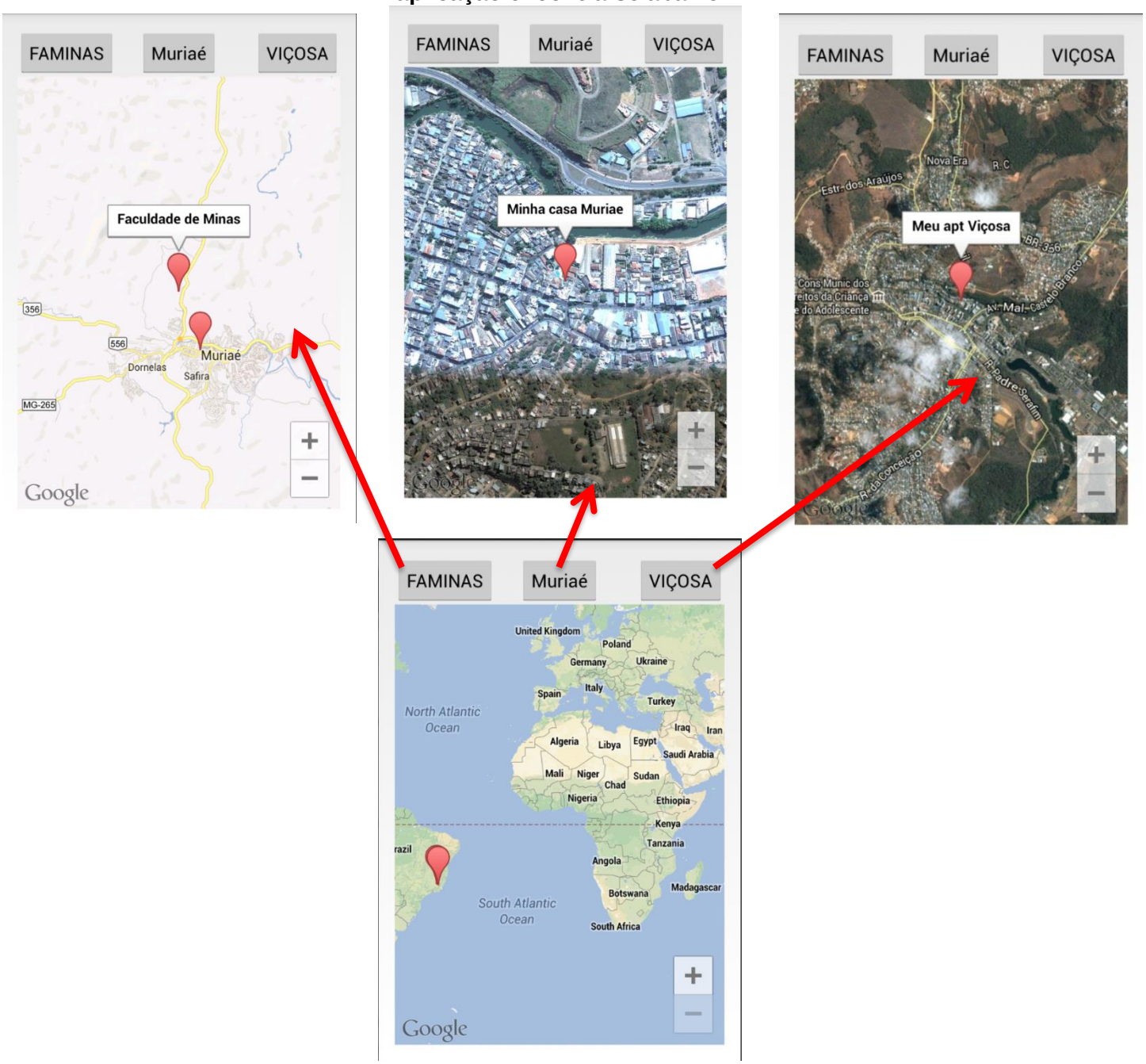


a) Atividade 1: **Quando o usuário clicar em uma das três primeiras opções do menu da aplicação, um mapa centralizado em um marcador (ponto) referente à respectiva localização geográfica deverá ser mostrado.**

Uma única `Activity` além do menu principal deverá ser utilizada nesta atividade, logo, no momento em que uma opção do menu for clicada, um parâmetro diferente deverá ser passado para esta segunda `Activity` (mapa e três botões). Dependendo do valor deste parâmetro passado entre as telas, uma das três possíveis coordenadas geográficas será utilizada para centralizar o mapa no ponto em questão.

Na segunda `Activity`, responsável por mostrar o mapa, deverão estar contidos também três botões distintos, sendo cada um deles responsável por centralizar o mapa em uma das três possíveis localizações trabalhadas no aplicativo.

Um exemplo de parte do comportamento da segunda `Activity` da aplicação encontra-se abaixo:



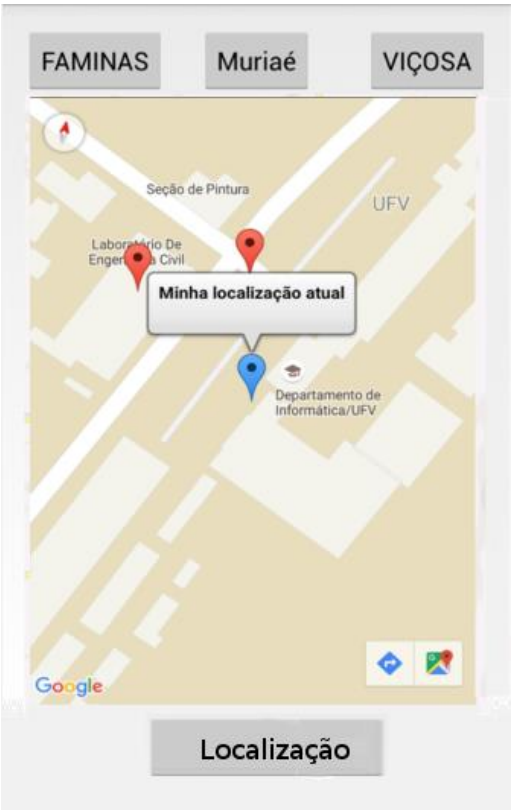
OBSERVAÇÃO:

- O seu aplicativo deverá nomear o botão “Muriaé” com o nome da sua cidade natal e o botão “FAMINAS” com o nome “DPI/UFV”.
- DICA: Para obter as posições de latitude e longitude desses três marcadores fixos que serão adicionados ao mapa, use o Google Maps (https://www.google.com/maps). Ao clicar com o botão direito do mouse em um mapa do Google, ele te informar latitude e longitude do ponto em questão. Você poderá copiar esses valores e levar para o seu código.
- Aproveite a oportunidade para testar diferentes tipos de mapas (Normal, Híbrido, Satélite...etc) e diferentes valores de zoom.

- Utilize os métodos `makeText().show()` da classe `Toast` para exibir alertas na tela contendo o nome da opção escolhida no menu da primeira tela.

b) Atividade 2: **Faça uma alteração no exercício anterior**, acrescentando um botão abaixo do mapa da segunda tela, que ao ser pressionado adicione um marcador AZUL no mapa, referente à posição atual do usuário do aplicativo. O mapa deverá ser centralizado nesse novo marcador. Observe que a cada vez que o botão for pressionado, a posição do usuário poderá ter sofrido variações, logo, sempre que ocorrer um evento de clique neste botão, o marcador deverá ser substituído por um novo e o mapa deverá ser novamente centralizado na nova posição.

Um exemplo do comportamento da aplicação encontra-se abaixo:

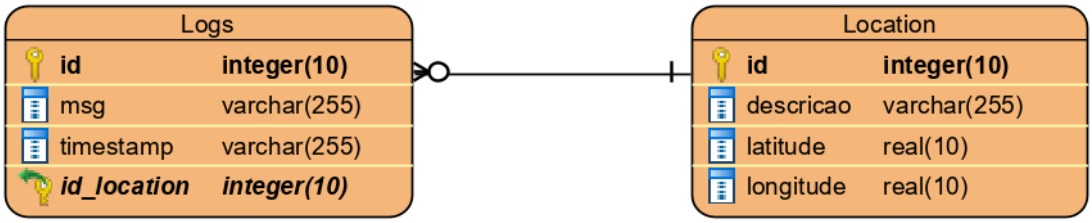


OBSERVAÇÃO:

- Utilize os métodos `makeText().show()` da classe `Toast` para exibir alertas na tela informando a qual distância (m) em linha reta o usuário do app se encontra em relação à sua casa em viçosa quando o botão “Localização” é pressionado.
- Utilize como auxílio, um ponteiro para um objeto da classe `Marker` para apontar para o marcador atual da posição.
- Utilize o método `remove()` da classe `Marker` para apagar um objeto marcador do mapa quando necessário.
- Para acrescentar um marcador azul, utilize os seguintes comandos:

```
map.addMarker(new MarkerOptions().position(ATUAL).title("Minha
localização atual")
     .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptor
     Factory.HUE_AZURE)));
```

c) Atividade 3: **Modifique a seu aplicativo para que o banco de dados a seguir seja criado no momento em que ele for aberto pela primeira vez após a sua instalação**:



Onde na imagem acima foram utilizados os tipos `VARCHAR`, no Android deve ser usado o tipo `TEXT`, devido às características do SQLite. Além disso, desconsidere os valores informados entre parênteses quando for criar o seu banco. Por exemplo, `INTEGER(10)` deverá ser apenas `INTEGER`.

Passo 1) Durante a criação do banco de dados, a tabela `Location` já deverá ser populada, cadastrando as localizações que são usadas para mostrar os três marcadores vermelhos do mapa. Uma vez feito isso, o seu código deverá ser adaptado para não mais definir os valores de latitude e longitude de maneira *hardcoded*, mas sim realizando buscas no banco de dados.

Passo 2) No menu da primeira `Activity`, onde é mostrada uma lista de opções para o usuário, uma tupla deverá ser persistida na tabela `Logs` sempre que o usuário clicar em uma opção do menu. A chave primária da tabela `Logs` (`id`) é AUTOINCREMENT. A coluna `msg` deverá conter uma `String`, como por exemplo “Viçosa”, “DPI” ou “Cidade Natal”. Isso vai depender da opção escolhida pelo usuário. A coluna `timestamp` deverá ter o horário em que o log foi registrado. Em Java, esse valor pode ser obtido da seguinte forma:

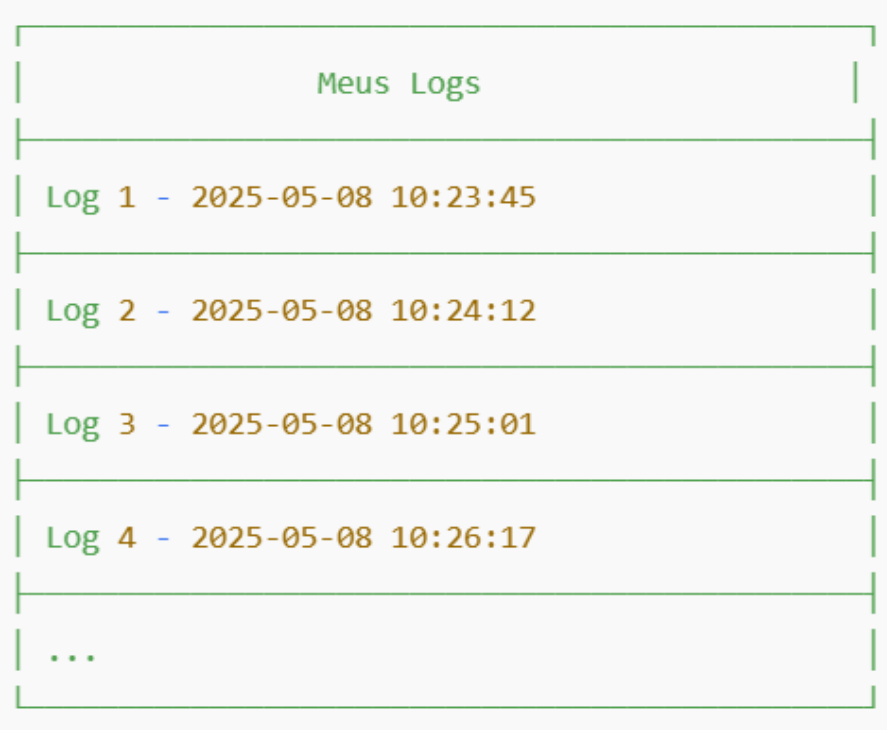
```
import java.time.Instant;

Instant timestamp = Instant.now(); // Exemplo: 2025-05-08T14:25:30.123Z
```

Como a coluna `timestamp` é do tipo `TEXT`, lembre-se de concatenar o valor de `Instant` com uma `String` vazia (“”) antes de persistir no banco.

Por fim, a coluna `id_location` é uma chave estrangeira para a tabela `Location`, portanto ela deverá ser relacionada com a tupla da tabela `Location` equivalente com a opção do menu escolhida pelo usuário.

Passo 3) Na Tela principal do programa (`ListActivity`), você deverá acrescentar mais um item na lista chamado “Relatório”. Ao clicar nesse novo item, o aplicativo deverá navegar até uma nova `Activity` (Report.java), que será descrita melhor no passo 4.

Passo 4) A nova tela criada no passo anterior deverá ser também uma `ListActivity`. O seu papel será o de listar dados de todas as tuplas persistidas na tabela `Logs` do banco, sendo cada tupla um item dessa lista. A seguir é apresentado um *wireframe* dessa tela.



Observe que nos locais onde temos *Log 1*, *Log 2*, etc.. deverá ser exibido o valor da coluna `msg` da respectiva tupla. Ao lado desse valor, o `timestamp` da tupla deverá ser concatenado precedido por um hífen (-). Esse *wireframe* representa apenas a ideia da tela, no entanto você deverá deixá-la com o comportamento visual similar à da primeira tela do app.

Passo 5) Nessa tela de relatório, quando um item da lista for clicado, você deverá usar os métodos `makeText().show()` da classe `Toast` para exibir um alerta na tela informando a `latitude` e `longitude` do local associado ao log clicado. Observe que esses dados deverão ser recuperados via um INNER JOIN entre as duas tabelas do banco.

ENTREGA DAS ATIVIDADES:

Ao final da atividades, suba o seu projeto para um repositório público do GitHub e informe a URL desse repositório pelo PVAnet Moodle.