Redes Móveis e Sem Fios

Relatório Final

Development of Internet of Things sensor monitoring based on SigFox, Arduino and Android

Bernardo Gomes, 75573 Diogo Martins, 75462

29 de Abril de 2016

Conteúdo

1 Objectivo

O objectivo do projecto é o desenvolvimento de um sistema de monitorização de temperatura.

O sistema, deverá ser baseado num sensor de temperatura associado a um dispositivo *arduino (akeru 3.3)*, que irá comunicar as suas medições a um servidor *SigFox*, armazenando-as na *cloud*.

Na óptica do utilizador, irá ser desenvolvida uma aplicação em ambiente *android*, que fornecerá os dados presentes na *cloud* com uma apresentação *user friendly*. Pretende-se ainda que seja possível que o utilizador registe um novo dispositivo a monitorizar na aplicação, bem como definir alarmes para certos valores de temperatura.

2 Solução encontrada

Tal como referido na secção anterior, a monitorização da temperatura e da qualidade de medição do sensor, irá ser feita pelo utilizador com recurso à aplicação, mas tendo a *cloud SigFox* como intermediária.

Por forma a que os dados de cada utilizador sejam independentes do dispositivo utilizado, foi construída uma base de dados adicional, que guarda informação de *login* bem como dos *devices* e os *thresholds* de alarme de cada utilizador.

A arquitectura será então a apresentada na figura 1:

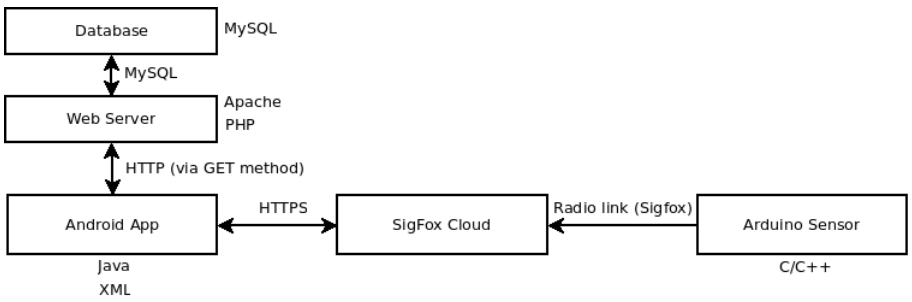

Figura 1: Arquitectura geral

2.1 Aplicação *Android*

A aplicação *Android*, com a qual o utilizador irá ter contacto directo, será constituída por cinco actividades:

- *MainActivity*;
- *CreateLogActivity*;
- *LogsActivity*;
- *NewAlarmActivity*;
- *AddDeviceActivity*.

As relações entre as actividades descritas, encontram-se representadas na figura 2.

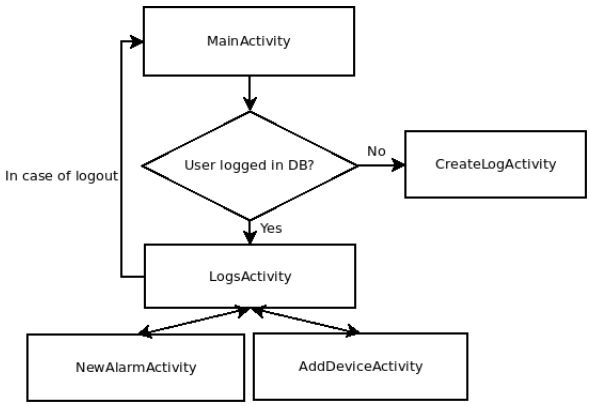

Figura 2: Arquitectura da aplicação *Android*

A *MainActivity* tem como objectivo perguntar ao utilizador o nome com o qual está registado na rede, realizando *queries* à base de dados com as informações do utilzador. No caso de existir registo prévio, as informações são armazenadas num ficheiro que irá funcionar como *cache* na aplicação e é lançada a *LogsActivity*. Caso o utilizador na tenha registo, é lançada a *CreateLogActivity* para registar o novo utilizador.

A actividade de registo de um utilizador (*CreateLogActivity*), terá apenas três campos de inserção de texto: *username, password* e *devicetype-id*. Estes parâmetros são gravados no ficheiro de texto descrito anteriormente, bem como na base de dados central. De seguida, a aplicação irá lançar a *LogsActivity*.

A actividade de visualização da informação da *Cloud* (*Logs*), é constituída por duas *threads* principais. A primeira consiste na obtenção das mensagens do dispositivo por pedidos HTTPS (GET) periódicos, de acordo com a informação de registo do utilizador e que é actividada por uma *checkbox*. Posteriormente a resposta será enviada para a *thread* principal (da API) que a disponibiliza ao utilizador. No caso de a *checkbox* não estar activa, os pedidos podem ser efectuados apenas ao clicar no botão de pedido de informação. O esquema desta actividade está descrito na figura Figura 3.

Esta actividade tem ainda a opção de registar um novo dispositivo para monitorização, bem como adicionar um novo alarme de temperatura. No caso de um *threshold* de temperatura, lido da base de dados no início da aplicação, ser ultrapassado, a *thread* que realiza o *parsing* da informação deverá lançar uma notificação ao utilizador.

À semelhança da actividade *CreateLogActivity*, as actividades *AddDeviceActivity* e *NewAlarmActivity* serão apenas compostas por campos de texto. Após o registo num ficheiro e na base de dados das informações recolhidas, estas actividades irão retornar no *stack*, voltando à actividade anterior.

No caso de o utilizador pretender fazer *logout*, é lançada a *MainActivity*, por forma a que seja possível iniciar a sessão com outro *username*.

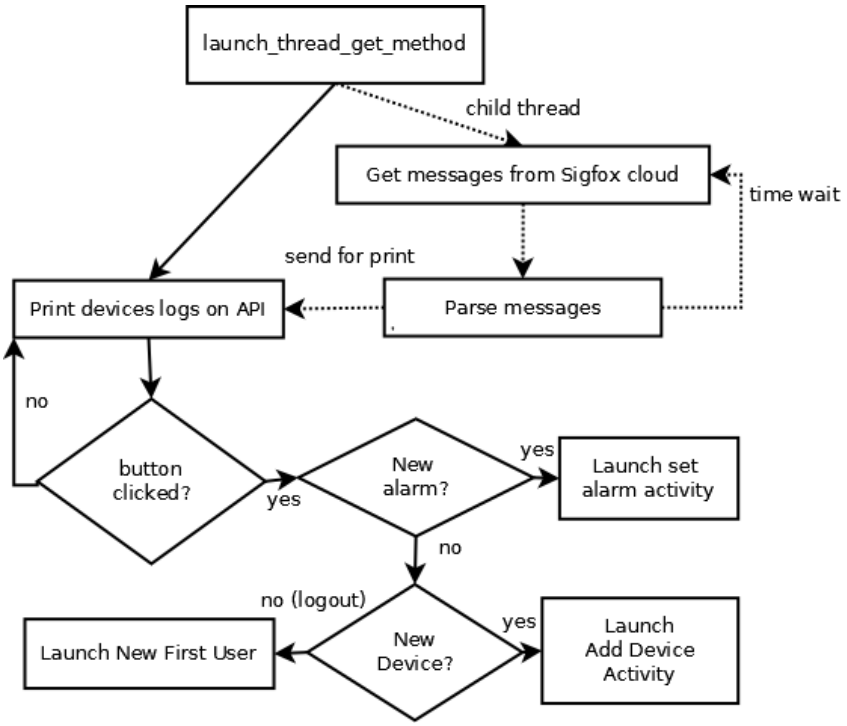

Figura 3: Arquitectura da actividade *Logs*

2.2 Servidor *SigFox*

O papel deste servidor será o armazenamento da informação medida pelo sensor e a recepção de pedidos por parte da aplicação *android*. Consoante o tipo de pedido, irá realizar uma resposta em *JSON* com as informações das medições.

Em termos de implementação para o projecto, foi apenas necessário conhecer a forma como os pedidos devem ser realizados bem como o formato de resposta.

2.3 Sensor Arduino

O sensor deverá realizar medições periodicamente enviando-as para a *cloud SigFox*, via rádio.

3 Detalhes técnicos

Para a implementação da solução descrita anteriormente, para os diferentes módulos recorreu-se às seguintes tecnologias:

Para a aplicação *Android*:

- **Welcome Screen, New First User, Set Alarm e Add Device** - Nestes módulos, pensa-se utilizar a abertura, leitura e escrita de ficheiros, por forma a efectuar/consultar registos (utilizador, dispositivo ou alarme). Para tal, ir-se-á utilizar a Classe *FILE* presente em *android*;

- **Logs** - Neste módulo, será essencial a abertura de uma *thread*, na medida em que a *User Interface* (UI) principal não permite efectuar pedidos periódicos pelo facto de estes a poderem bloquear. Assim, a primeira acção será a abertura de uma *thread*, com recurso à classe *Thread*.

  Para efectuar os pedidos HTTPS, usar-se-á uma de duas classes: *URLConnection* ou *HttpURLConnection*. O formato destes pedidos seguem as normas descritas na REST API-Students fornecida pelo corpo docente.

  Ao receber a resposta, a *thread* deverá ser processada com *parsing* de *JSON* com recurso à classe *JsonReader*. Após o processamento da resposta, irá ser verificado se a temperatura recebida ultrapassa algum *threshold* definido pelo utilizador. Em caso afirmativo, irá ser gerada uma notificação com recurso a um objecto *Notification*.

Para a implementação do sensor de temperatura, recorrer-se-á às bibliotecas associadas ao dispositivo *Akeru 3.3* disponibilizadas pelo *Snootlab*.

4 Verificações da solução encontrada

Do planeamento atrás referido, foi já adiantado parte do trabalho referente à programação da aplicação, nomeadamente as actividades *Welcome Screen, New First User, Set Alarm e Add Device*. De seguida são apresentadas algumas capturas de ecrã destas actividades em versão *beta*.

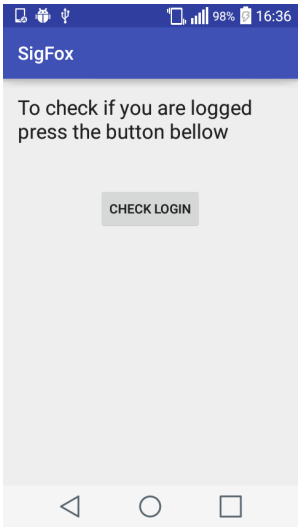

Figura 4: Welcome Screen

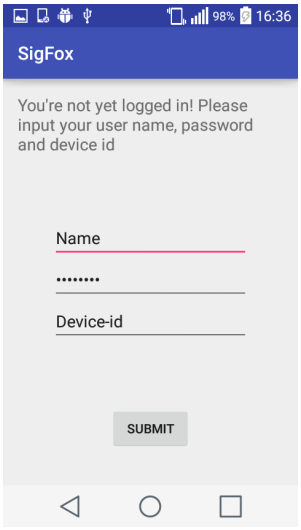

Figura 5: Registo de um novo utilizador

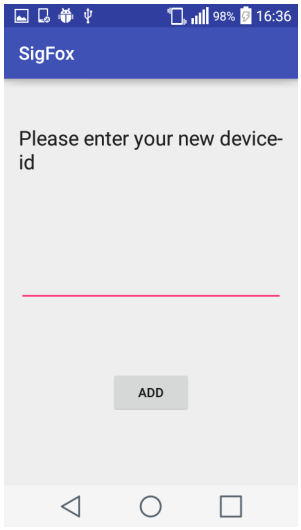

Figura 6: Registo de um novo dispositivo

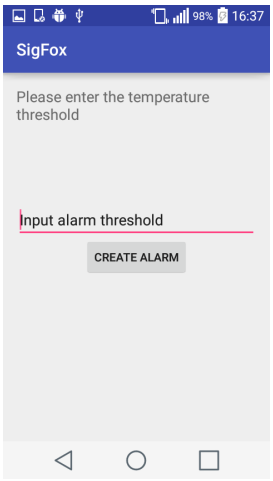

Figura 7: Registo de um novo alarme

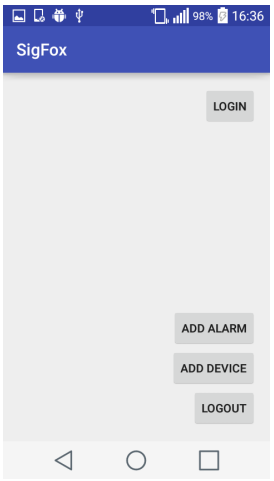

Figura 8: Actividade *Logs*

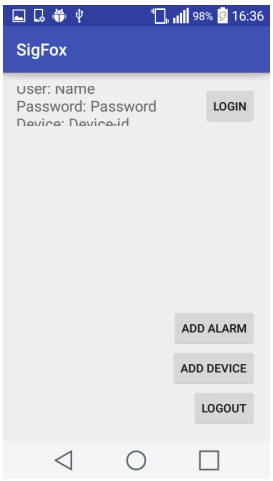

Figura 9: Actividade *Logs* com *login*

Relativamente à obtenção das leituras do sensor de arduíno a partir da *cloud Sigfox*, foi realizado um programa à parte por forma a testar apenas esta funcionalidade, tal como representado nas figuras abaixo. O programa, além de recolher os dados, realiza o *parse* do *JSON*.

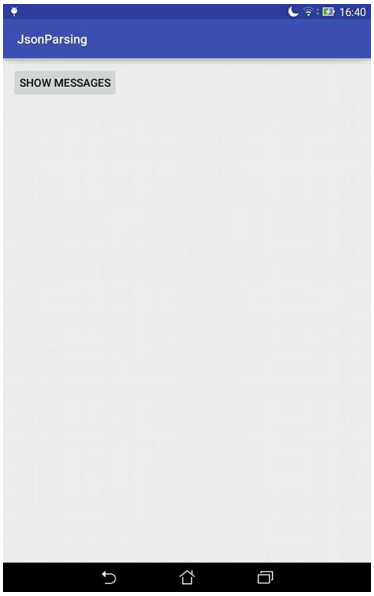

Figura 10: Botão para obtenção da informação da *cloud*

Figura 11: Recolha da informação com *parse*

Será agora necessária a interligação dos dois programas e a activação das notificações, por forma a que a aplicação fique terminada.

Relativamente ao sensor, foi apenas realizado o trabalho de pesquisa, sendo iniciada a implementação quando a aplicação estiver concluída.

5 Alterações face ao planeamento inicial

6 Pontos críticos

Referências

[1] [FILE16] http://developer.android.com/reference/java/io/File.html, Março 2016

[2] [THREAD16] http://developer.android.com/reference/java/lang/Thread.html, Março 2016

[3] [HTTPURLCONNECTION16]

http://developer.android.com/reference/java/net/HttpURLConnection.html, Março 2016

[4] [URLCONNECTION16]

http://developer.android.com/reference/java/net/URLConnection.html, Março 2016

[5] [JSON16] http://developer.android.com/reference/android/util/JsonReader.html, Março 2016

[6] [NOTIFICATION16]

http://developer.android.com/training/notify-user/build-notification.html, Março 2016

[7] [AKERU16] https://github.com/Snootlab/Akeru, Fevereiro 2016