

Sistemas de Informação Distribuídos

Licenciaturas em Engenharia Informática e Informática e Gestão de Empresas
2018-2019, Segundo Semestre







Monitorização de Culturas em Laboratório

Mongo DB e Android

Identificação do grupo autor
da especificação (Etapa A):

Identificação do grupo autor da
implementação (Etapas B):

Número	Nome	Foto
77981	André Silva	
73538	Gonçalo Fernandes	
77812	João Aparício	
72809	João Neto	
77561	João Saramago	
77778	Rita Costa	

Número	Nome	Foto
60921	Wang Yang	
72783	André Carvalho	
73353	Bruno Colaço	
74455	Dinis Ferreira	
77667	Maria Dinis	
78014	Mariana Silva	

Instruções

Estas instruções são de cumprimento obrigatório. Relatórios que não cumpram as indicações serão penalizados na nota final.

- Podem (e em várias situações será necessário) ser adicionadas novas páginas ao relatório, mas não podem ser removidas páginas. Se uma secção não for relevante, fica em branco, não pode ser removida;
- Todas as secções têm que iniciar-se no topo de página (colocar uma quebra de página antes);
- A paginação tem de ser sequencial e não ter falha;
- O índice tem de estar atualizado;
- Na folha de rosto (anterior) têm de constar toda a informação solicitada, nomeadamente todas as fotografias de todos os elementos dos dois grupos. É obrigatório que caiba tudo numa única página;
- A formatação das “zonas” (umas sombreadas outras não sombreadas) não pode ser alterada;
- O grupo que primeiro edita o documento (Etapa A) apenas escreve até à secção 1.7, e o outro grupo apenas em todas as outras secções.

Índice

1	Mongo DB.....	7
1.1	Descrição Geral do Procedimento.....	7
1.2	Estrutura da Base de Dados Mongo.....	12
1.3	Periodicidade de Leitura de Sensores e Escrita no Mongo.....	14
1.4	Estrutura da Base de Dados Mysql.....	15
1.5	Periodicidade de Leitura de Mongo e Escrita no MySql.....	17
1.6	Triggers, SP ou eventos no MySql (caso relevante).....	18
1.7	Utilizadores relevantes no Mysql e respectivos privilégios.....	19
	Avaliação Global da Qualidade das Especificações.....	21
1.8	Implementação.....	22
1.8.1	Divergências face ao recebido/especificado.....	22
1.8.2	Código Mongo Implementado (dentro do java).....	23
1.8.3	Código SQL Implementado.....	24
1.8.4	Tempo Médio.....	25
1.8.5	Alertas.....	25
2	Android e Php.....	26
2.1	Esquema da BD Lite Geral.....	26
2.2	Layout Implementado no Android.....	27

Monitorização de Culturas em Laboratório

Um laboratório de investigação de um departamento de biologia necessita de um sistema para monitorizar a evolução de culturas. Mais concretamente, pretende acompanhar a temperatura e luz a que as culturas estão sujeitas, bem como detectar/antecipar potenciais problemas.

Numa estufa estão colocados dois sensores que medem a temperatura e quantidade de luz ambiente (que afeta todas as culturas existentes na estufa).

Periodicamente os investigadores dirigem-se à estufa para efetuarem manualmente várias medições de variáveis (humidade, ph, etc) e registá-las num computador que está localizado na estufa. Cada cultura tem um único investigador responsável e apenas ele pode criar, atualizar e consultar os dados de medições das suas culturas. Esta *proteção de dados* é um aspeto importante do sistema. Nem todas as variáveis necessitam serem lidas e registadas. Para cada cultura o investigador decide quais delas devem ser lidas, e regista no sistema qual o intervalo de valores que considera “normal” para o par variável/cultura.

Por exemplo, para as culturas hidropónicas de pimento e tomate, fazem-se medições do nível de concentração de mercúrio e chumbo. Mas numa cultura de bactérias onde se adicionaram antibióticos o que faz sentido medir é o índice de concentração das bactérias, não faz sentido medir o nível de concentração de mercúrio e chumbo.

Alertas

Existem dois tipos de alertas:

a) alertas resultantes das medições das variáveis. O investigador, quando insere manualmente um valor de uma medição, caso o valor ultrapasse os limites será alertado com um aviso (no próprio computador) e com uma mensagem para o telemóvel (por vezes o investigador pede a um colega para efectuar a medição, sendo por isso aconselhável que o alerta não apareça somente no monitor do computador).

b) Alertas resultantes dos sensores de temperatura e luminosidade. O sistema sabe, para toda a estufa, o intervalo de valores de luminosidade e temperatura adequado (igual para todas as culturas). Se o sensor detectar que os valores vão ser ultrapassados deve notificar por telemóvel o investigador.

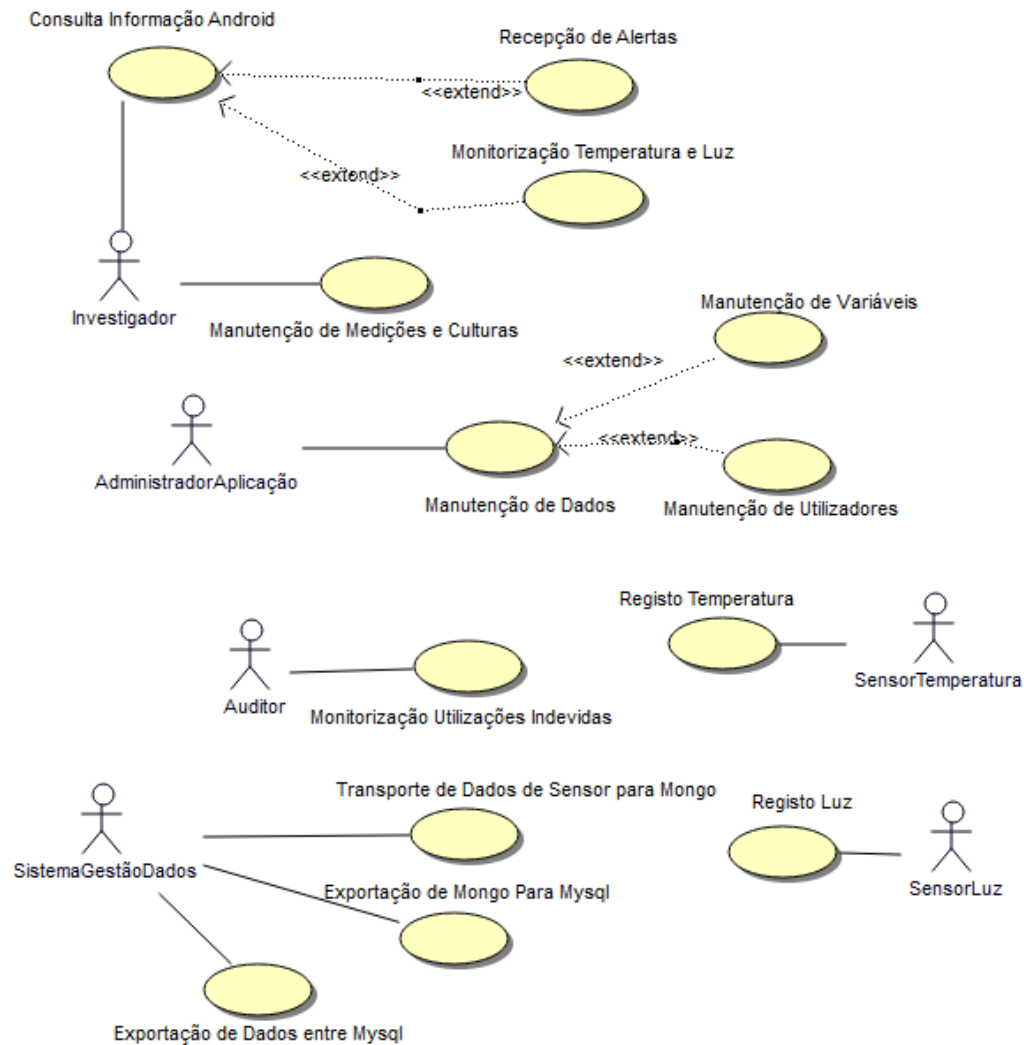
Cada investigador deverá ter a possibilidade de, através de um telemóvel, monitorizar a evolução da temperatura e luminosidade (não apenas a última leitura, mas a evolução na última hora ou horas) e receber os dois tipos de alertas.

Registo de Acessos

É necessário guardar na base de dados (mysql) o registo de todas as operações de escrita sobre todas as tabelas (quais dados foram alterados/inseridos/apagados, quando e por quem) e o registo de operações de consulta apenas sobre a tabela Medições. Esse registo de alterações (*log*) é *exportado* incrementalmente (apenas informação nova) e periodicamente para uma base de dados autónoma (também mysql). Através dessa base de dados (apenas de consulta) um

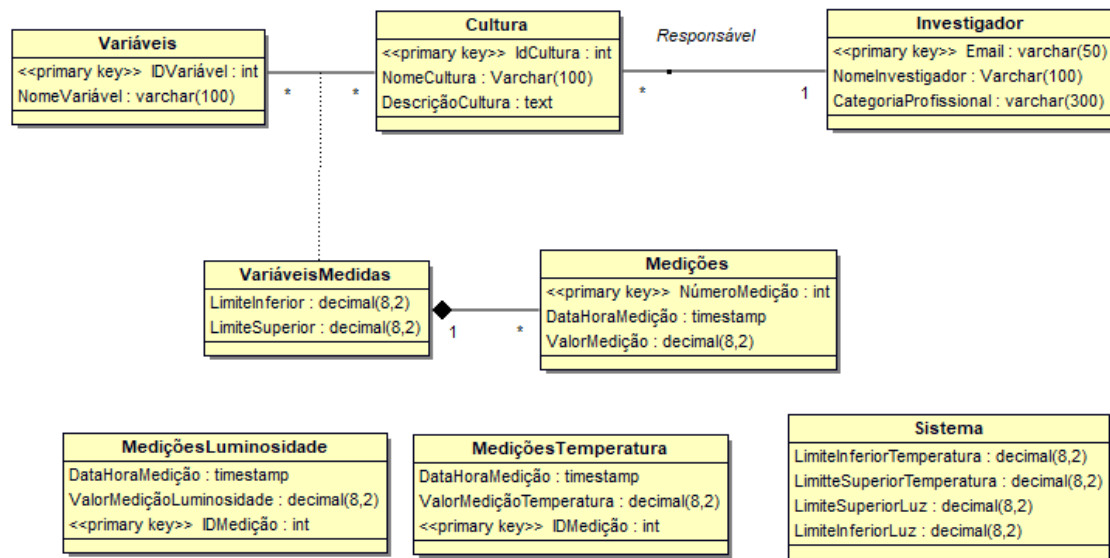
auditor pode analisar se ocorreram utilizações abusivas dos dados (por exemplo, quem é que alterou limites de temperatura de uma cultura, etc.).

Diagrama de Use Case Global



No presente relatório apenas são contemplados os use case “Registo Temperatura”, “Registo Luz”, “Consulta Informação Android”, “Transporte de Dados de Sensor Para Mongo”, “Exportação de Mongo para MySql” e “Exportação de Dados entre Mysql”.

Diagrama de Classes de Suporte à Base de Dados

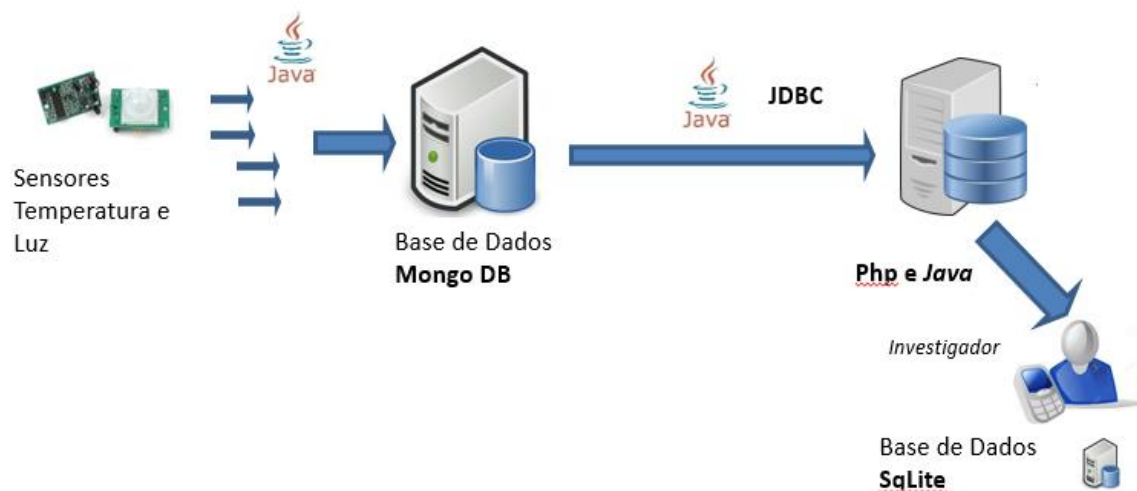


Sensor

Exemplo Mensagens

```
{"sensor":"1","datapassagem ":"2016/12/12","horapassagem ":"18:45:24"}
```

Esquema de Importação e Migração



1 Mongo DB

1.1 Descrição Geral do Procedimento

<Nesta secção deverá ser dada uma descrição genérica sobre a forma como decorre o processo de migração:

- i) Receber a informação dos sensores e guardá-lo numa base de dados MongoDB;
- ii) Exportar de **forma incremental** a informação do MongoDB para a bd Sql Anywhere.

A informação apresentada deverá ser suficiente para que o grupo que a receba consiga implementar as várias etapas. Deve ser clara e estar bem estruturada em secções. Cabe ao grupo decidir qual a melhor forma de estruturar a exposição.

Apesar de não ser para escrever código, se o grupo considerar que o grupo que vai implementar pode desconhecer algum aspecto (biblioteca, algoritmo, etc.) pode exemplificar/ilustrar a forma de implementação. Considerar que o grupo que vai implementar tem conhecimentos razoáveis de Java (POO e PCD) e relacional, tem acesso à documentação colocado no E-Learning sobre MongoDB, e a mais nada.

Alguns exemplos de informação que poderão constar nos requisitos: utilização ou não de threads em Java, número de Mains Java, parâmetros a utilizar no MQTT, quais as leituras que são descartadas.>

- i) A interação entre os sensores e a base de dados MongoDB deve ter por base uma conexão que use o protocolo MQTT. Desta forma, deve haver uma classe responsável pelo estabelecimento da conexão, que é feito pelo método 'connect' da API do MQTT. Para se poder estabelecer a ligação é necessário definir um cliente, que deve ser uma instância da classe 'MqttClient'. Para a criação desta instância, são necessários os parâmetros: identificador do cliente, identificador da máquina hospedeira e respetivo porto (broker) e uma instância da classe 'MemoryPersistance' para indicar se a ligação deve ou não ser memorizada. Para a conexão, deve ser utilizado o método connect da classe 'MqttClient', que poderá receber como

argumento uma instância da classe `'MqttConnectOptions'`, onde podem ser definidas mais opções.

Depois da conexão estar estabelecida deve ser utilizado o método `'messageArrived'` da interface `'MqttCallback'` que permite receber as mensagens vindas dos sensores. Este método recebe dois argumentos, uma string relativa ao tópico da mensagem e um objeto do tipo `'MqttMessage'`.

Depois de recebidas as mensagens, as mesmas devem ser tratadas para se extrair os dados necessários e inseri-los na base de dados MongoDB. Para isto, é necessário previamente estabelecer a ligação à base de dados MongoDB, sendo por isso necessário criar uma instância da classe `'MongoClient'`. Depois de estabelecida a ligação é necessário colocar os dados sob a forma de documentos e inseri-los.

- ii) Para migrar os dados de forma incremental da base de dados não relacional para a base de dados relacional, deve ser usado um campo `'exportado'`, que é um inteiro e se encontra presente em cada documento da coleção da base de dados MongoDB. Este inteiro é criado a 0 por default e sempre que ocorre a exportação é alterado para 1. Desta forma, são exportados apenas dos dados da coleção que tenham esse mesmo campo a 0.

Os valores das medições devem ser também inseridos numa estrutura de dados Java. Para isso deve ser utilizada a classe medição que deve conter um booleano que indica se a medição já foi ou não foi exportada para base de dados MongoDB.

A migração ocorre no máximo ao fim do tempo previsto para chegada de 3 novas medições. Isto é, supondo que a periodicidade de chegada de novas medições é de 3 segundos, a exportação ocorre quando chegarem 3 novas medições à base de dados MongoDB, ou quando decorrer um período de 10 segundos (tempo previsto para a chegada de 3 medições somado de 1 segundo de margem para pequenos atrasos) desde a última exportação. O Java deve ter um contador que é incrementado quando uma nova medição é enviada para o Mongo e quando esse contador chega a 3 ou decorre o tempo relativo à chegada de 3 novas medições, a

thread responsável pela migração para o relacional é notificada e contador e o timer são postos a 0.

Durante esse período de tempo se for recebida uma medição que despolete um alerta o mesmo deve ser inserido imediatamente na base de dados MongoDB com o respectivo campo que sinaliza o alerta a 1, e a thread responsável pela migração para o relacional deve ser notificada para migrar o alerta para a tabela respectiva, bem como as medições que ainda não tenham sido exportadas. Nesta situação o contador de mensagens enviadas para o Mongo e o timer têm que ser repostos a 0.

Uma medição é considerada um alerta se o valor da luminosidade ou da temperatura se encontrar acima do valor percentual relativo ao limite superior ou abaixo do valor percentual relativo ao limite inferior. Estes valores limite são calculados com base nos atributos 'MargemSegurancaTemperatura' e 'MargemSegurancaLuz' da tabela Sistema, como explicado na secção 1.4.

Para efetuar a migração, deve ser feita uma conexão à base de dados MongoDB a fim de verificar que dados estão por migrar. Caso tenha ocorrido um erro de qualquer tipo na migração de dados mais antigos, os mesmos devem ser todos migrados pela ordem que foram inseridos na base de dados MongoDB.

Os campos "erroLuminosidade" e "erroTemperatura" constantes na base de dados MongoDB devem ser consultados aquando da migração para o relacional, e no caso de um deles ou ambos terem o valor 1, devem ser inseridas as respectivas medições na respetiva tabela de dados incorretos.

Deve ser criado um mecanismo que garanta que no caso da aplicação ser fechada (exceto falhas de energia ou erros de software), todos os dados que ainda não constam como exportados na base de dados MongoDB são exportados nesse instante.

Para realização do processo de escrita de dados na base de dados MongoDB deverá ser utilizado apenas um Main.

Adicionalmente devem ser utilizadas 3 threads, cujas funções se explicam abaixo:

- 1) Thread responsável por lançar a aplicação e efetuar as ligações às bases de dados e aos sensores.

- 2) Thread responsável por estar sempre à espera de novas medições vindas dos sensores para as inserir uma a uma numa `BlockingQueue`, afim de verificar se os valores recebidos são um erro ou se correspondem a uma situação de alerta. Em seguida deve inserir esses mesmos dados já processados na base de dados MongoDB.
Sempre que surja uma situação de alerta, a medição que provocou esse alerta deve ser imediatamente inserida na base de dados MongoDB com o respetivo campo que sinaliza o alerta a 1 e a thread responsável pela migração de dados da base de dados MongoDB para o relacional deve ser notificada para se proceder à migração dessa medição e de todas as outras que ainda não tenham sido migradas.
- 3) Thread responsável por migrar os dados da base de dados Mongo para a base de dados relacional. Esta thread efetua a migração ao fim de chegarem 3 novas medições, ou findo o período de tempo correspondente à chegada de 3 novas medições somado de uma margem de 1 segundo, ou quando é notificada porque surgiu um situação de alerta ou ainda quando ocorre um fecho da aplicação antes de ter ocorrido a exportação de todos os dados.

Pode dar-se o caso de o sensor enviar medições erradas de temperatura ou luminosidade que podem desencadear falsos alertas. Assim, deve-se tentar perceber se estamos perante um falso alerta através da seguinte estratégia.

Devem ser guardadas numa `BlockingQueue` ou numa estrutura de dados semelhante as últimas 2 medições e a nova medição.

Sempre que uma nova medição chega, a mesma deve ser inserida na `BlockingQueue` e em seguida devem ser comparados os valores das últimas duas medições com o valor recebido, para verificar se o mesmo faz sentido no seguimento dos anteriores.

Estas verificações devem ser feitas recorrendo a uma diferença percentual entre a medição em causa e cada uma das duas medições anteriores. Para exemplificar, tomemos como exemplo a situação em que a variável `'PorcentagemVariacaoTemperatura'` foi definida pelo administrador da aplicação com o valor de 10%, o limite superior de temperatura é 25 graus, o limite inferior é de 15 graus, a medição anteriormente recebida tem o valor de 20 graus e a nova medição tem o valor de 20.5 graus. Se a nova

medição variar em mais do que 1 grau das 2 medições anteriores (10 % de (25-15)) a medição deve ser marcada como errada. Como no exemplo dado esta variação é de meio grau ($|20-20.5| = 0.5$), podemos considerar que a medição recebida apresenta um valor real.

Caso o valor percentual seja em ambos os casos superior à percentagem definida pelo administrador na base de dados relacional, na tabela 'Sistema', que se considera normal para variações entre medições, a respetiva medição deve ser inserida no MongoDB com o respetivo campo que indica que a mesma é um erro com o valor 1, isto é, se o erro for na temperatura, a medição deve ser inserida na base de dados MongoDB com o campo "erroTemperatura" a 1.

Se o valor da medição recebida fizer sentido no seguimento dos anteriores o mesmo deve ser inserido na base de dados MongoDB com os valores que assinalam um erro a 0.

1.2 Estrutura da Base de Dados Mongo

<Nome da base de Dados e das coleções

Listar algumas linhas exemplificativas da informação guardada na (s) coleção(ões). Usar o comando `find().pretty()` sem critérios>

A base de dados deverá ter o nome 'sensores' e deverá ter apenas uma coleção denominada 'medicoes' com a seguinte estrutura.

```
> db.medicoes.find().pretty();
{
  "_id" : ObjectId("5cba51b2e8b32b13d1942538"),
  "timestamp" : "2019-04-06 00:22:56",
  "temperatura" : "35",
  "luminosidade" : "500",
  "alertaTemperatura" : "1",
  "exportado" : "1"
}
{
  "_id" : ObjectId("5cba51dde8b32b13d1942539"),
  "timestamp" : "2019-04-06 00:32:56",
  "temperatura" : "35",
  "luminosidade" : "5005",
  "alertaTemperatura" : "1",
  "alertaLuminosidade" : "1",
  "exportado" : "1"
}
{
  "_id" : ObjectId("5cba51fae8b32b13d194253a"),
  "timestamp" : "2019-04-06 00:33:56",
  "temperatura" : "35",
  "luminosidade" : "5005",
  "erroLuminosidade" : "1",
  "exportado" : "0"
}
{
  "_id" : ObjectId("5cba5233e8b32b13d194253b"),
  "timestamp" : "2019-04-06 00:35:56",
  "temperatura" : "250",
  "luminosidade" : "503",
  "erroTemperatura" : "1",
  "exportado" : "0"
}
{
  "_id" : ObjectId("5cba5258e8b32b13d194253c"),
  "timestamp" : "2019-04-06 00:39:56",
  "temperatura" : "250",
  "luminosidade" : "5033",
  "erroTemperatura" : "1",
  "erroLuminosidade" : "1",
  "exportado" : "1"
}
```

Os campos "erroTemperatura" e "erroLuminosidade" indicam se o valor de temperatura ou de luminosidade correspondem a um erro (se tiver o valor 1), para fazer a distinção se os mesmos devem ser inseridos nas respectivas tabelas de medições corretas ou incorretas. Quando o campo existe é porque sabemos que houve um erro. Por outro lado, quando o campo não existe é porque não houve um erro.

Os campos "alertaTemperatura" e "alertaLuminosidade" indicam se o valor de temperatura ou de luminosidade corresponde a um alerta (se tiver o valor 1).

Da mesma forma que nos erros, se o campo de alerta de luminosidade ou de temperatura existir e tiver o valor 1 é porque a respectiva medição corresponde a um alerta, se o campo não existir é porque essa medição não despoletou nenhum alerta.

1.3 Periodicidade de Leitura de Sensores e Escrita no Mongo

<Explicar de que forma e com que periodicidade o Java recebe informação dos sensores e exporta para Mongo>

Os sensores enviam novas medições para o Java, sendo que as mesmas são recebidas através do método 'messageArrived' da interface 'MqttCallback' que deverá constar numa classe com uma thread dedicada a receber novas medições. Assim que uma nova medição chega ao Java, deve ser feito o parse da estrutura JSON recebida, e a informação depois de tratada deve ser registada de imediato numa BlockingQueue ou numa estrutura de dados com um comportamento semelhante para se verificar se os valores recebidos são um erro. Se as medições forem um erro devem ser inseridas na base de dados MongoDB com o respetivo campo que indica se a medição é um erro com o valor 1.

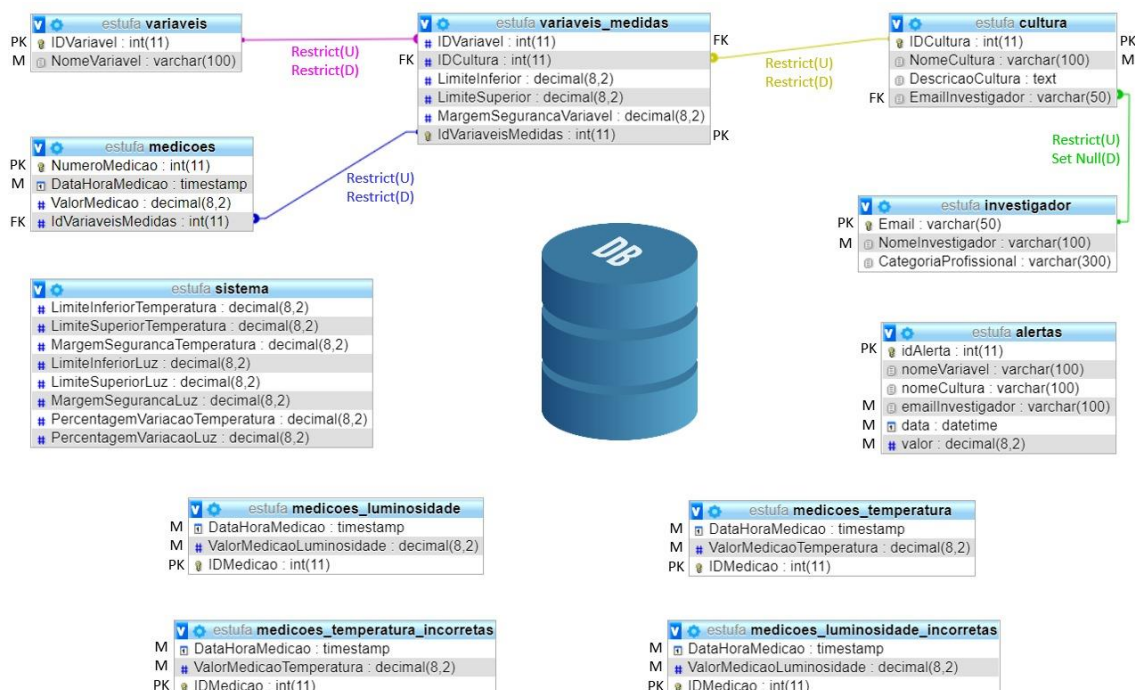
Se os valores das medições despoletarem um alerta devem ser inseridos na base de dados MongoDB com o respetivo campo que indica se a medição é um alerta com o valor 1 e a thread responsável pela migração para o relacional deve ser de imediato notificada para proceder à migração do alerta e de todos os dados ainda não migrados. Nesta situação o contador de mensagens que foram enviadas para o MongoDB deve ser posto a 0, bem como o timer relativo ao tempo decorrido desde a última migração.

1.4 Estrutura da Base de Dados Mysql

<Apenas as tabelas relevantes para esta fase. Utilizar formato de relacional do relatório anterior

<Listar algumas linhas exemplificativas da informação guardada na (s) tabela(s).

Sugestão: ter uma tabela para armazenar os alertas que vão ser consultados a partir do android.>



A tabela alertas deve conter a hora e data em que a medição que despoletou o alerta foi feita, o que causou o alerta, ou seja, a variável cujos valores limite foram ultrapassados, o nome da cultura que está associado a essa variável, o email do investigador responsável e o valor que despoletou o alerta.

Um exemplo do que poderia aparecer na tabela alertas é o que se encontra na imagem seguinte.

	idAlerta	nomeVariavel	nomeCultura	emailInvestigador	data	valor
<input type="checkbox"/> Editar Copiar Apagar	5	PH	Batatas	pedro@gmail.com	2019-04-01 00:00:00	7.80
<input type="checkbox"/> Editar Copiar Apagar	6	CHUMBO	Cebola	carlos@gmail.com	2019-04-13 00:00:00	3.57

As tabelas 'medicoes_temperatura_incorretas' e 'medicoes_luminosidade_incorretas' devem ser utilizadas para registar as medições que se considerem incorretas, segundo a lógica explicada na primeira secção, sendo que as mesmas

não devem ser inseridas na tabela de medições temperatura ou medições luminosidade.

Na tabela sistema foram inseridas as colunas 'MargemSegurancaTemperatura' e 'MargemSegurancaLuz' e na tabela variáveis medidas foi inserida a coluna 'MargemSegurancaVariavel'.

Estes atributos correspondem à percentagem a que as medições têm que estar dos valores limite para despoletar o alerta. Supondo que o limite superior de temperatura é 40 graus e o limite inferior é de -10 graus e a respetiva margem de segurança é de 10%, os alertas devem ser despoletados assim que os valores das medições estejam a 5 ou menos graus do limite superior ou inferior, ou seja, caso a temperatura seja de 35 ou mais graus, ou de -5 ou menos graus.

Foram também inseridos os atributos 'PercentagemVariacaoTemperatura' e 'PercentagemVariacaoLuz' que correspondem às percentagens consideradas normais pelos técnicos responsáveis para as variações entre medições próximas no tempo.

Estes valores devem ser parametrizáveis por parte do administrador da aplicação, conferindo assim mais flexibilidade ao sistema e um maior controlo por parte dos investigadores e do administrador da aplicação sobre os alertas que os mesmos pretendem receber.

1.5 Periodicidade de Leitura de Mongo e Escrita no MySql

<Explicar de que forma e com que periodicidade o Java recebe informação do mongo e exporta para o MYSql.>

A exportação para o relacional deve ser feita depois de terem sido recebidas 3 novas medições, ou caso haja algum atraso na chegada das mesmas, no período de tempo mínimo correspondente à chegada de 3 medições. Isto é, supondo que os sensores enviam dados de 3 em 3 segundos, a exportação deve ser feita de 3 em 3 medições recebidas ou ao fim de 10 segundos (tempo necessário para receber 3 medições somado de uma margem de 1 segundo para pequenos atrasos) no caso de as 3 medições não terem chegado no tempo previsto.

Contudo, caso surja um valor para uma medição que esteja dentro da percentagem de segurança definida na base de dados relacional, e caso esse valor faça sentido (diferença percentual entre cada uma das duas medições anteriores e a nova medição dentro da percentagem definida pelo administrador da aplicação como sendo 'salto normal') com pelo menos um dos dois valores anteriormente recebidos, o alerta deve ser imediatamente inserido na tabela de alertas e as medições que ainda não foram exportadas devem ser exportadas nesse instante para a respetiva tabela. Caso surja uma nova medição que suscite alerta, o procedimento anterior só deve ser feito se tiverem decorrido pelos menos 12 segundos desde o último alerta.

1.6 Triggers, SP ou eventos no MySql (caso relevante)

<Especificar que triggers ou SP pretendem que sejam implementados (por exemplo, para alertas).>

Devem ser criados triggers que verifiquem para cada uma das medições manuais inseridas na base de dados relacional se estamos perante uma situação de alerta, e caso isto se verifique inserir o alerta na tabela de alertas.

Assim, estes triggers devem ser do tipo 'After Insert' na tabela de variáveis medidas e para cada uma das novas medições devem verificar se os valores registados estão dentro da margem de segurança definida para a variável em causa no atributo 'MargemSegurancaVariavel', seguindo o mesmo procedimento especificado para as medições de temperatura e luminosidade. Caso isto se verifique deve registar o alerta na respetiva tabela.

1.7 Utilizadores relevantes no Mysql e respetivos privilégios

<Utilizar formato de tabela do relatório anterior.>

Tabela	Tipo de Utilizador			
	Investigador	Administrador ou Aplicação	JavaUser	PhpUser
sistema	L	E, L	-	-
variaveis	E, L	L	-	-
investigador	L	E, L	-	-
medicoes	E, L	E, L	E	-
variaveis_medidas	E, L	E, L	-	-
medicao_luminosidade	E, L	E, L	E	-
medicao_temperatura	L	E, L	E	-
alertas	L	L	E	L
cultura	E, L	E, L	-	-
medicoes_temperatura_incorretas	L	L	E	-
medicoes_luminosidade_incorretas'	L	L	E	-
Stored Proc.				

O utilizador JavaUser deve ter privilégios para inserir medições nas tabelas medições luminosidade e medições temperatura, inserir medições nas tabelas de medições incorretas de temperatura e luminosidade e inserir alertas na respetiva tabela.

O utilizador PhpUser é responsável pelas ligações à base de dados para efetuar migrações e consultas dos alertas para serem enviados em Android.

Avaliação Global da Qualidade das Especificações recebidas

Avaliação (A,B,C,D,E) : _____

Utilize a seguinte escala:

A: - 1 – 5 valores B: 6 – 9 valores C: 10 – 13 Valores D: 14 – 17 valores E: 18 – 20 valores

Análise crítica (clareza, completude, rigor):

1.8 Implementação

1.8.1 Divergências face ao recebido/especificado

<Indicar as divergências relevantes (ignorar pequenos detalhes de implementação) face ao especificado pelo próprio grupo e face ao especificado pelo outro grupo, nomeadamente as que consideram que permitiu chegar a uma solução melhor.

Tem de ficar claro:

- (i) que ideias aproveitaram da própria especificação;
- (ii) que ideias aproveitaram da especificação do outro grupo;
- (iii) que ideias novas foram introduzidas.

>

1.8.2 Código Mongo Implementado (dentro do java)

<Listar todo o código Mongo utilizado no processo, quer para importar, quer para exportar. O código tem de ser comentado para que se torne legível para quem sabe uns rudimentos de MongoDB. Fragmentos de código java apenas serão mostrados para dar algum contexto.>

1.8.3 Código SQL Implementado

<Listar todo o código SQL utilizado no processo de colocação de inserção nas tabelas SQL Anywhere. O código tem de ser comentado para que se torne legível para quem sabe SQL. Os comentários não podem ser redundantes, colocar apenas o essencial. Indicar triggers ou eventos no lado MySQL, se existirem.>

1.8.4 Tempo Médio

<Indicar o tempo médio que demora um valor do sensor a chegar a base de dados Mysql.>

1.8.5 Alertas

<Exemplificar alguns alertas gerados automaticamente.>

2 Android e Php

2.1 Esquema da BD Lite Geral

<Modelo relacional implementado no Android, tabelas e atributos>

2.2 Layout Implementado no Android

<PrintScreen de um exemplo de interacção>