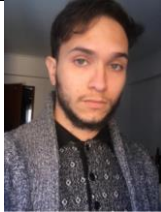
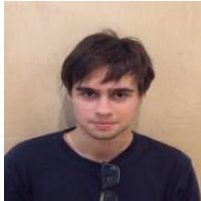
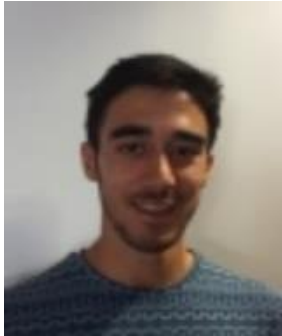





Sistemas de Informação Distribuídos (22/23)

Licenciaturas em Engenharia Informática e Informática e Gestão de Empresas



Monitorização de Ratos em Laboratório





Identificação do grupo : __17__

Número	Nome	Foto
98875	Douglas Lino	
98975	João Franco	
92905	Tiago Castro	
98824	Duarte Laureano	
99776	Diogo Peng	

		
92257	João Pereira	
Especificação: MQTT		

Identificação do grupo que completou documento: 22

Número	Nome	Foto
99352	Ana Mercês Soares dos Reis Moreira	
98528	Gonçalo Miguel Costa Serrano	

99286	Inês Colaço Ascenso	
93040	Inês Mendonça Consolado	
99110	Mariana Coutinho Guerreiro	
98686	Nicole Lopes Nunes	

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 instruções (delimitadas por <>) têm de ser removidas no documento final;
- A paginação tem de ser sequencial e não ter falhas;
- 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;
- A informação apresentada deverá ser suficiente para que o grupo que a receba consiga implementar o que pretendem. 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 aspeto (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), tem acesso à documentação colocado no E-Learning sobre MongoDB, e a mais nada.
- Às vezes os diagramas são uma boa forma de completar a especificação.

Índice

1	Especificação Inicial.....	8
1.1	Transporte de MongoDB para Mysql.....	8
1.1.1	Coleções a criar em cada uma das réplicas do Mongo	8
1.1.2	Descrição Geral do Procedimento.....	8
1.1.3	Migração por MQTT (apenas preencher se for a forma de migração atribuída ao grupo) 9	
1.1.4	Tratamento de medições pré-Mysql.....	10
1.1.4.1	Especificação de Procedimentos e Funções (caso existam).....	10
1.1.5	Tabelas no Mysql.....	11
1.1.6	Tratamento de medições pós-Mysql.....	12
1.1.6.1	Especificação de Triggers (caso existam)	12
1.1.6.2	Especificação de Procedimentos e Funções (caso existam).....	13
1.1.7	Utilizadores Base de Dados Mysql	13
1.1.8	Procedimentos Manutenção Aplicação	15
1.1.8.1	Especificação de Procedimentos	15
1.1.9	Explicação detalhada dos tratamentos aos dados	16
1.1.10	Eventos de suporte à aplicação (caso existam).....	17
1.2	Consulta por HTML/PHP.....	18
1.2.1	Layout dos formulários HTML	18
1.2.2	Associar SP a formulários HTML.....	18
2	Apreciação Crítica à Especificação Recebida.....	19
2.1	Avaliação Global de especificações recebidas	19
3	Especificação entregue ao outro grupo	21
4	1 Especificação Inicial.....	Erro! Marcador não definido.
4.1	1.1 Transporte de MongoDB para Mysql.....	Erro! Marcador não definido.
4.1.1	1.1.1 Coleções a criar em cada uma das réplicas do Mongo .	Erro! Marcador não definido.
4.1.2	1.1.2 Descrição Geral do Procedimento	Erro! Marcador não definido.
4.1.3	1.1.3 Migração por MQTT (apenas preencher se for a forma de migração atribuída ao grupo).....	Erro! Marcador não definido.
4.1.5	1.1.4 Tratamento de medições pré-Mysql.....	Erro! Marcador não definido.

4.1.5.1	1.1.4.1	Especificação de Procedimentos e Funções (caso existam)	Erro! Marcador não definido.
4.1.6	1.1.5	Tabelas no Mysql	Erro! Marcador não definido.
4.1.7	1.1.6	Tratamento de medições pós-Mysql	Erro! Marcador não definido.
4.1.7.1	1.1.6.1	Especificação de Triggers (caso existam)	Erro! Marcador não definido.
4.1.7.2	1.1.6.2	Especificação de Procedimentos e Funções (caso existam)	Erro! Marcador não definido.
4.1.8	1.1.7	Utilizadores Base de Dados Mysql	Erro! Marcador não definido.
4.1.9	1.1.8	Procedimentos Manutenção Aplicação	Erro! Marcador não definido.
4.1.9.1	1.1.8.1	Especificação de Procedimentos	Erro! Marcador não definido.
4.1.10	1.1.9	Explicação detalhada dos tratamentos aos dados....	Erro! Marcador não definido.
4.1.11	1.1.10	Eventos de suporte à aplicação (caso existam)	Erro! Marcador não definido.
4.2	1.2	Consulta por HTML/PHP	Erro! Marcador não definido.
4.2.1	1.2.1	Layout dos formulários HTML	Erro! Marcador não definido.
4.2.2	1.2.2	Associar SP a formulários HTML	Erro! Marcador não definido.
4.3		Auto-Avaliação Global de especificações entregues outro grupo	41
5		Implementação	42
5.1		Transporte de MongoDB para Mysql	42
5.2		Tabelas no Mysql	43
5.3		Tratamento de medições Pré-Mysql	44
5.4		Tratamento de medições pós-Mysql	46
5.4.1		Especificação de Triggers implementados (outro grupo)	46
5.4.2		Código de Triggers implementados (outro grupo)	46
5.4.3		Especificação de Triggers implementados (best off)	46
5.4.4		Código de Triggers implementados (best off)	47
5.4.5		Especificação de Store Procedures implementados (outro grupo)	50
5.4.6		Código Stored Procedures implementados (outro grupo)	50
5.4.1		Especificação de Store Procedures implementados (best off)	51
5.4.2		Stored Procedures implementados (best off)	56
5.5		Utilizadores Base de Dados Mysql	67
5.6		Procedimentos Manutenção Aplicação (best off)	69
5.6.1		Stored Procedures implementados (best off)	69

5.7	Eventos de suporte à aplicação (best off).....	69
5.8	PrintScreen dos formulários HTML (best off).....	69
5.9	PrintScreen do formulários Android (best off).....	74

Monitorização de Ratos em Laboratório

1 Especificação Inicial

1.1 Transporte de MongoDB para Mysql

1.1.1 Coleções a criar em cada uma das réplicas do Mongo

Base de dados - Sensores

Coleções:

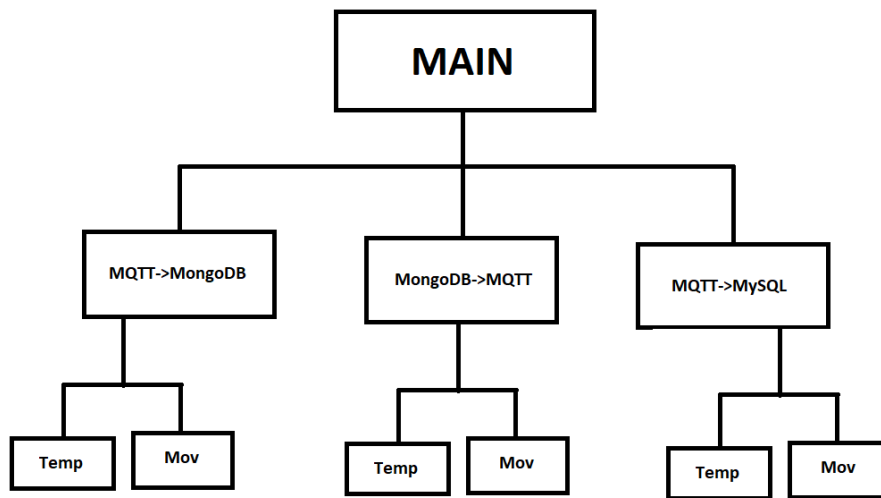
```
MediçõesPassagem { "IDMedicao": "0", "Hora":  
ISODate("2021-11-17T03:19:56.186Z"), "SalaEntrada": "2",  
"SalaSaida": "3"}
```

```
MediçõesTemperatura { "IDMedicao": "0", "Hora":  
ISODate("2021-11-17T03:19:56.186Z"), "Leitura": "20.25",  
"Sensor": "5"}
```

1.1.2 Descrição Geral do Procedimento

- i) "Periodicidade" com que se vai buscar ao Mongo;
1 segundo.
- ii) Como se garante que não envia vezes o mesmo documento/informação;
Cria-se um set nos programas Java que recebe do MongoDB e manda para MQTT, e no que recebe do MQTT e manda para o MySQL. Este set descarta a mensagem que chega se esta já lá existir.
No programa Java que recebe da cloud e envia para MQTT, a cada mensagem recebida vai ser adicionado um id, que vai ser usado para o caso de haver problemas com a data, ou seja, se a data for igual em 2 mensagens cuja informação é igual, mas este id for diferente, então a mensagem é distinta
- iii) Número de "mains" e para cada main o nome, indicar o número de threads, e uma muito breve descrição do que faz.
1 main - Main, 3 threads - MQTT->MongoDB; MongoDB->MQTT; MQTT->MySQL. Cada uma destas threads vai ter duas a correr, uma para as

medições de temperatura, e outra para as medições de passagens. No total estão 9 threads a correr.



1.1.3 Migração por MQTT (apenas preencher se for a forma de migração atribuída ao grupo)

Nome Tópico: g17_mov Cliente: MySQL(User Sensor) QOS: 1

Nome Tópico: g17_temp Cliente: MySQL(User Sensor) QOS: 2

1.1.4 Tratamento de medições pré-Mysql

1.1.4.1 Especificação de Procedimentos e Funções (caso existam)

Nome Procedimento / Função	Parâmetros Entrada	Parâmetros Saída	Valores Retorno	Muito breve descrição
FiltroMongoTemp	Lista de RegistosMongoDBTemp Ou RegistoMongoDBTemp		Lista de RegistosFiltrados Ou RegistoFiltrado	Filtra todos os registos MongoDB da coleção das temperaturas, e verifica se os valores recebidos estão de acordo com as regras estabelecidas no ponto 1.1.9.
FiltroMongoMov	Lista de RegistosMongoMov Ou RegistoMongoMov		Lista de RegistosFiltrados Ou RegistoFiltrado	Filtra todos os registos MongoDB da coleção das passagens, e verifica se os valores recebidos estão de acordo com as regras estabelecidas no ponto 1.1.9.

1.1.5 Tabelas no Mysql

experiencia	medicoespassagem
IDExperiencia (AutoIncrement) (ChavePrimária)	IDMedicao (ChavePrimaria)(autoincrement)
Descricao (text)	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
Investigador (varchar 50)	Hora (timestamp)
DataHora (timestamp)	SalaEntrada (int)
NumeroRatos (int)	SalaSaida (int)
LimiteRatosSala (int)	Ativo (int)
SegundosSemMovimento (int)	ComentarioAdicional(text)
TemperaturaIdeal (decimal 4,2)	
VariacaoTemperaturaMaxima (decimal 4,2)	medicoestemperatura
Ativo (int)	IDMedicao (int) (ChavePrimaria)(autoincrement)
	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
substanciaexperiencia	Hora (timestamp)
IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)	Leitura(decimal 4,2)
CodigoSubstancia (int)	Sensor (int)
NumeroRatos (int)	Ativo (int)
Ativo (int)	ComentarioAdicional(text)
odoresexperiencia	alerta
IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)	IDAlerta (int) (ChavePrimaria) (autoincrement)
sala	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
CodigoOdor (varchar 50)	Hora (timestamp)
Ativo (int)	Sala (int)
	Sensor (int)
Utilizador	Leitura(decimal 4,2)
IDUtilizador (int) (ChavePrimária)(autoincrement)	TipoAlerta (varchar 20)
NomeUtilizador (varchar 100)	Mensagem (varchar 100)
TelefoneUtilizador (varchar 12)	HoraEscrita (timestamp)
EmailUtilizador (varchar 50) Unique	Ativo (int)
TipoUtilizador(varchar 3)	
Ativo (int)	medicoessala
	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
	Sala(int) (ChavePrimária)
	NumeroRatosFinal (int)
	Ativo (int)

ComentarioAdicional:

O campo ComentarioAdicional serve para especificar quando existe um outlier nas medições e para informar o possível delay que a data possa ter após ter sido modificada por ser um valor incompatível

Ativo:

O campo "ativo" irá ter os valores 1 ou 2, 1 para Ativo, 2 para Inativo, este campo é utilizado para demonstrar quando um dado é "eliminado", assim não se perde informação, apenas para certos SP a informação a ser mostrada é só a ativa.

Porquê:

Tipos de Alerta:

-AlertaTemp

-AlertaMov

-AlertaXSegundos

-Alerta10Minutos

1.1.6 Tratamento de medições pós-Mysql

1.1.6.1 Especificação de Triggers (caso existam)

Nome Trigger	Tabela	Tipo de Operação (I,U,D)	Evento (After, Before)	Notas (apenas indicar aquilo que não seja óbvio)
BlockExp	Experiencia	I	Before	Bloquear a criação de uma nova experiência pois está uma a decorrer
AlertaTemp	MedicoesTemperatura	I	After	Usa o SP CriarAlerta para criar um alerta de temperatura Este trigger só vai ativar uma vez a cada 25 segundos
AlertaMov	MedicoesPasagens	I	After	Usa o SP CriarAlerta para criar um alerta de nr de ratos numa sala. Cada vez que é inserido um registo na tabela, é verificado se: nr de entradas na sala desse registo - nr de saídas na sala desse registo < LimiteRatosSala da Experiencia a decorrer
AlertaXSegundos	MedicoesPasagens	I	After	Usa o SP CriarAlerta para criar um alerta de falta de movimento dos ratos.

				Cada vez que é inserido um registo na tabela, conta-se os X segundos definidos na Experiência, e caso não seja inserido nenhum registo nesse tempo, cria o alerta.
Alerta10Minutos	Experiencia	I	After	Usa o SP CriarAlerta para criar um alerta de tempo máximo excedido. Quando é inserida uma nova experiência, o trigger fica ativo e se passado os 10 minutos limite de cada experiência, e esta não tiver acabada, cria o alerta.

1.1.6.2 Especificação de Procedimentos e Funções (caso existam)

Nome Procedimento / Função	Parâmetros Entrada	Parâmetros Saída	Valores Retorno	Muito breve descrição

1.1.7 Utilizadores Base de Dados Mysql

Tabela	Tipo de Utilizador		
	AppAdmin	Investigador	Sensor
Experiência	L/U/I/D	L	-

SubstânciaExperiência	L/U/I/D	L	-
OdoresExperiência	L/U/I/D	L	-
Utilizador	L/U/I/D	L	-
MediçõesPassagens	L	L	L
MediçõesTemperatura	L	L	L
Alerta	L	L	-
ParâmetrosAdicionais	L/U/I/D	L	-
MediçõesSalas	L/U/I/D	L	-
Stored Proc.			
CreateUser	X	-	-
DeleteUtilizador	X	-	-
ChangeUser	X	X	-
AddExperiência	X	X	-
DeleteExperiência	X	X	-
ChangeExperiência	X	X	-
CreateAlerta	-	-	-
DeleteAlerta	X	-	-
CreateMedPassagem	-	-	X
CreateMedTemperatura	-	-	X
CreateSubstância	X	X	-
DeleteSubstância	X	X	-
CreateOdor	X	X	-
DeleteOdor	X	X	-
Login	X	X	-

Em que U=Update, I Insert, D- Delete, L=Leitura, X=Executar SP e - sem permissões.

1.1.8 Procedimentos Manutenção Aplicação

1.1.8.1 Especificação de Procedimentos

Nome Procedimento / Função	Parâmetros Entrada	Parâmetros Saída	Valores Retorno	Muito breve descrição
CreateUser	Nome, Telefone, Tipo, Email, Senha e Ativo.			Cria um utilizador de um certo tipo ("role" como p.e. "Investigador") e atribui-lhe um login.
DeleteUtilizador	ID			Muda o campo "Ativo" do utilizador para 2 (Inativo).
ChangeUser	Nome, Telefone, Senha e Ativo.			Alteração de alguns dados de um utilizador.
AddExperiencia	NrRatos, Descricao, Investigador, DataHora, LimiteRatosSala, SegundosSemMovimento, TemperaturaIdeal, VariacaoTemperaturaMaxima e Ativo.			Cria uma experiência associada a um investigador.
DeleteExperiencia	ID			Muda o campo "Ativo" da experiência para 2 (Inativo).
ChangeExperiencia	Descrição, Investigador e Ativo.			Alteração de alguns dados de uma experiência.
CreateAlerta	Hora, Sala, Sensor, Leitura, TipoAlerta, Mensagem, HoraEscrita e Ativo.			Cria um aviso.

DeleteAlerta	ID			Muda o campo "Ativo" do Alerta para 2 (Inativo).
CreateMedPassagem	Hora, SalaEntrada, SalaSaida, Ativo e ComentarioAdicional.			Cria uma medição de passagem.
CreateMedTemperatura	Hora, Leitura, Sensor, Ativo e ComentarioAdicional.			Cria uma medição de temperatura
CreateSubstancia	IDExperiencia, Sala, CodigoOdor e Ativo.			Cria uma substância.
RemoveSubstancia	IDExperiencia.			Muda o campo "Ativo" da Substância para 2 (Inativo).
CreateOdor	IDExperiencia, Sala, CodigoOdor e Ativo.			Cria um odor.
RemoveOdor	IDExperiencia.			Muda o campo "Ativo" do Odor para 2 (Inativo).
Login	Utilizador, Password			Verifica que o utilizador e password recebidos estão associados na tabela utilizadores

1.1.9 Explicação detalhada dos tratamentos aos dados

Todos os dados vindos dos sensores são registados no MongoDB.

Valores errados: No programa Java que lê os dados do MongoDB e manda para MQTT e finalmente para MySQL, é feita uma verificação nos dados se os valores de temperatura são numéricos. Se isto se confirmar, os dados serão enviados para MQTT, caso contrário, não são enviados mas mantêm-se guardados no MongoDB.

No caso de as datas estarem erradas, para ambos os tipos de medições, é feita uma alteração desse valor para a data atual e é adicionado um comentário no campo ComentarioAdicional das tabelas das medições quanto a esse facto, mas estas são enviadas à mesma.

Outliers: Os outliers são verificados no programa que recebe dados vindos do MQTT e escreve no MySQL. Consideramos um outlier na temperatura se a variação de medição em medição for superior a 15° e não se mantiver por mais de 2 medições, exemplo (Medicao1: 9°, Medicao2: 25°, Medicao3: 25°, Medicao4: 9°) este caso é um outlier, mas se Medicao4 fosse 25° não seria um outlier.

Aproximações perigosas de limites: Em MySQL se a temperatura recebida estiver perto dos limites estabelecidos pela experiência atual um alerta é criado.

Atingir e/ou ultrapassar perigosas de limites: Em MySQL se a temperatura ou o número de ratos passarem os limites estabelecidos um alerta é criado.

1.1.10 Eventos de suporte à aplicação (caso existam)

Nome Evento	Local Execução (MySQL/Windows)	Muito breve descrição

1.2 Consulta por HTML/PHP

1.2.1 Layout dos formulários HTML

Login

Password

Enter

Criar

Experiencias	Editar
	.
	.
	.
	.

1.2.2 Associar SP a formulários HTML

ENTER: SP Login

CRIAR: SP AddExperiencia

EDITAR: SP ChangeExperiencia

2 Apreciação Crítica à Especificação Recebida

2.1 Avaliação Global de especificações recebidas

No geral, o grupo considerou que as especificações recebidas não estavam bem explicadas, o que levou a várias dúvidas aquando da leitura e análise do relatório da 1ª fase. Havia algumas secções, mais concretamente a de "Transporte de MongoDB para MySQL" em que havia pouca informação quanto à periodicidade escolhida (1 segundo) e ficámos sem saber o porquê de adotarem essa estratégia. Mais ainda, quando tratam dos dados duplicados nessa mesma secção, o uso de um id como identificador da mensagem e, posteriormente, a data, não é o mais indicado, visto que para isso bastava utilizar a data. Esta abordagem não funciona, muito menos, quando a periodicidade estipulada é de 1 segundo, como definida pelo grupo.

Quanto ao número de threads idealizadas, consideramos que o seu uso é desnecessário e sem coerência; será mesmo preciso recorrer a 9 threads? Para além disso, o grupo ficou sem perceber, mesmo com a especificação dada, o que a secção "Migração por MQTT" é suposto significar. Temos ainda a apontar que não faz sentido todas as tabelas do Modelo Relacional terem um campo "ativo"; porque se o utilizador cometer algum erro e quiser eliminar um determinado dado numa tabela aquando da criação de uma experiência, essa informação será guardada desnecessariamente porque não importará para nada; é um gasto desnecessário de recursos de base de dados. Essa prática é disfuncional e evasiva, não percebemos o seu propósito.

Para concluir, achamos que o relatório não está explícito e carece de especificações o que deixa o grupo um pouco à deriva com o que deverá considerar para futura implementação.

Avaliação Global da Qualidade das Especificações recebidas

Avaliação (A,B,C,D,E) : C

Utilize a seguinte escala:

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

Três principais deficiências de especificação que tiveram impacto mais negativo na qualidade da implementação

Transporte de dados do Mongo para Mysql (periodicidade, dados duplicados, n° de threads)

Valores Errados

Outliers - a abordagem não faz sentido, pouco adaptáveis

Resumo de Avaliações de Qualidade Anteriores (para cada linha assinalar com x em célula correspondente)

	Fraco	Razoável	Bom	Muito Bom
BD Mongo		X		
BD Mysql			X	
Proc. Pré Mysql	X			
Proc. Pós Mysql		X		
Stored P e Funções			X	
Triggers		X		
Utilizadores			X	
Proc. Utilizadores		X		
Eventos	-	-	-	-
HTML		X		

3 Especificação entregue ao outro grupo

Monitorização de Ratos em Laboratório

1 Especificação Inicial

1.1 *Transporte de MongoDB para Mysql*

1.1.1 Coleções a criar em cada uma das réplicas do Mongo

A base de dados criada e utilizada em cada uma das réplicas do Mongo foi apelidada de *labrats*. Numa fase inicial foi pensada a criação de três coleções: duas para as temperaturas medidas por cada sensor e uma relativa ao movimento das portas acionado pelos ratos. Após debate tanto em grupo como com o docente, foi decidido a criação de quatro coleções: uma referente à temperatura registada por ambos os sensores (*Temperaturas*), outra à ativação dos sensores de movimentos (*Movimentos*) e mais duas *MovPorEnviar* e *TempPorEnviar* onde se guardam os dados que ainda não foram enviados. Assim, os dados referentes à coleção *Temperatura* serão separados tal como pensado no momento inicial, mas este processo será tratado posteriormente em java, nas medições pré-MySQL. O intuito das duas últimas coleções é guardar os respetivos dados que estão a ser recebidos, eliminando-os à medida que são enviados. Isto evita o envio de dados duplicados e que não seja perdida informação. É de notar que a existência destas coleções não dispensa a inserção dos dados em cada uma das outras respetivas coleções; inserção esta que deverá ocorrer em simultâneo, à medida que os dados são recebidos.

```

replica [direct: primary] labrats> db.Temperaturas.find()
[
  {
    _id: ObjectId("641058c3f1c1ab1a2487298f"),
    Hora: '2023-03-14 11:21:32.722738',
    Leitura: 14,
    Sensor: 1
  },
  {
    _id: ObjectId("641058c3f1c1ab1a24872990"),
    Hora: '2023-03-14 11:21:32.722738',
    Leitura: 12.499999999999996,
    Sensor: 2
  },
  {
    _id: ObjectId("641058c4f1c1ab1a24872991"),
    Hora: '2023-03-14 11:21:33.724059',
    Leitura: 14,
    Sensor: 1
  },
  {
    _id: ObjectId("641058c4f1c1ab1a24872992"),
    Hora: '2023-03-14 11:21:33.724059',
    Leitura: 12.799999999999997,
    Sensor: 2
  },
]

```

Figura 2

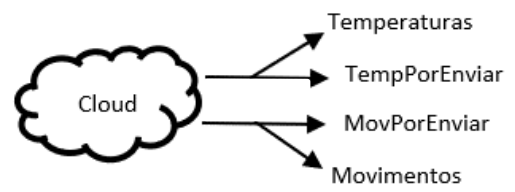


Figura 1

1.1.2 Descrição Geral do Procedimento

i)

Definiu-se que uma possível periodicidade ideal para transferir os dados do MongoDB para o SQL são 5 segundos. Devido ao número elevado de dados recebido por segundo, um atraso menor levaria a uma sobrecarga da base de dados que se pode refletir em dificuldades de visualização dos dados, na aplicação, por parte do investigador. Já um atraso maior pode levar à visualização de resultados menos fidedignos e atuais, que podem ser prejudiciais para a experiência.

ii)

É através das coleções no MongoDB *TempPorEnviar* e *MovPorEnviar* que se garante o envio dos dados apenas uma vez. Isto é garantido pois, durante a migração, apenas se vão buscar os dados a estas coleções. Como estes são apagados após o seu envio, não existe o risco de serem consultados e enviados uma segunda vez. Se porventura for necessário obter os dados de novo, também é possível através das outras duas coleções.

iii)

De maneira a contornar os eventuais problemas causados por falhas no programa Java foi definida a criação de dois mains:

- **Temperaturas-** Este main vai lidar com os dados referentes às medições dos sensores de temperatura. De maneira a não sobrecarregar nenhum processo vão existir 5 threads, cada uma com a sua função:

1.Thread que efetua a passagem de dados do Mongo para o programa onde são filtrados e posteriormente enviados para o SQL.

2.É criada uma thread que controla a periodicidade para transferir os dados, dessa forma filtram-se os dados em conjuntos para podermos tratar de outliers na thread seguinte.

3.Thread responsável por filtrar os dados e verificar se é enviado um alerta.

4. Esta thread serve para controlar o spam, iniciando um timer sempre que for criado um alerta. A thread anterior, se quiser criar um alerta, vai verificar se esta thread está a correr. Se estiver, não cria o alerta, caso contrário cria o alerta e ativa esta thread, iniciando um novo timer.

5. Por fim, existe uma thread que envia os dados para o MySQL.

- **Movimentos**- Este main vai lidar com os dados referentes ao movimento dos ratos. De maneira a não sobrecarregar nenhum processo vão existir 4 threads, cada uma com a sua função:

1. Thread que efetua a passagem de dados do Mongo para o programa onde são filtrados e posteriormente enviados para o SQL.

2. Thread responsável por filtrar os dados e verificar se é enviado um alerta.

3. Esta thread serve para controlar o spam, iniciando um timer sempre que for criado um alerta. A thread anterior, se quiser criar um alerta, vai verificar se esta thread está a correr, se estiver, não cria o alerta, caso contrário cria o alerta e ativa esta thread, iniciando um novo timer.

4. Por fim, existe uma thread que envia os dados para o MySQL.

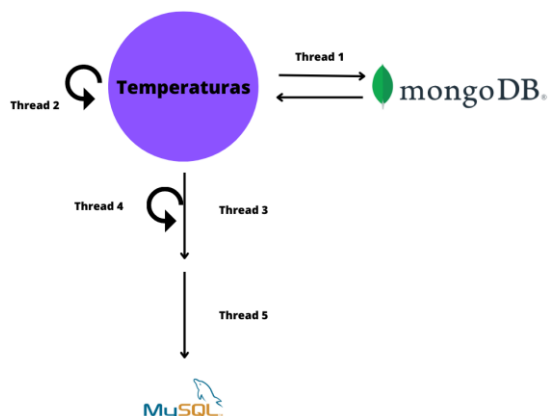


Figura 3

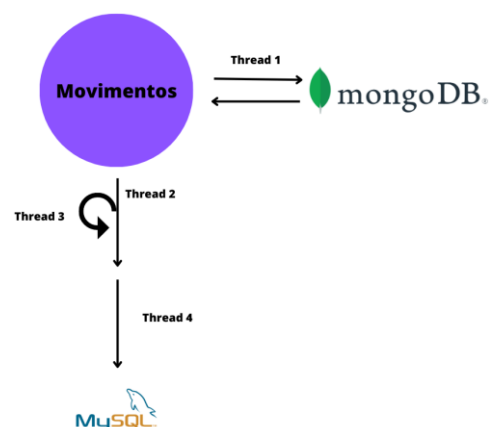


Figura 4

1.1.3 Migração por MQTT (apenas preencher se for a forma de migração atribuída ao grupo)

Nome Tópico: _____ Cliente: _____ QOS: _____

1.1.4 Tratamento de medições pré-Mysql

1.1.4.1 Especificação de Procedimentos e Funções (caso existam)

Nome Procedimento / Função	Parâmetros Entrada	Parâmetros Saída	Valores Retorno	Muito breve descrição
Outliers()	Lista de temperatura de uma zona	-	-	Tratamento de outliers, filtra as temperaturas, retira valores errados (valores que não correspondam ao formato esperado) e valores absurdos (valores que não façam sentido depois das temperaturas anteriores ou antes da temperatura seguinte). A função guarda a última temperatura viável para futuras comparações.
separa_temps()	Lista de temperaturas recebidas das duas zonas	-	Lista de temperaturas filtradas	Função que recebe de 5 em 5 segundos as temperaturas enviadas pelo mongo. Separa as temperaturas por zona e envia as mesmas para a função outlier().
contagem_ratos()	Lista dos movimentos recebidos	-	-	Função que recebe os movimentos enviados pelo mongo. Serve para controlar o número de ratos em cada

				sala, chama a função alerta_ratos() se o número de ratos numa determinada sala se aproximar do número máximo de ratos estabelecido pelo investigador. Na situação de a experiência terminar envia a listagem de ratos por sala para adicionar à tabela medicoessala.
alerta_ratos()	-	-	-	Envia um alerta informando que o número de ratos da sala se está a aproximar do número máximo de ratos. Esta mesma função verifica se a sala já enviou um alerta referente ao número de ratos de determinada sala, para evitar duplicações.
alerta_temperaturas()	Lista de temperaturas filtradas	-	-	Recebe as temperaturas filtradas e analisa as mesmas, verificando se é necessário enviar um alerta amarelo ou vermelho ao investigador. Esta mesma função verifica se enviou um alerta igual ao anterior para evitar duplicações.
Temporizador_experiencia()	-	-	-	Verifica o atributo DataHora de todas as experiências e

				envia um alerta caso falte um minuto para o início da(s) mesma(s). Sendo esta a última oportunidade de o investigador adiar a experiência.
Verifica_parametros()	Lista de movimentos de ratos	-	-	Verifica os parâmetros das mensagens recebidas por mongo referidas aos movimentos de ratos.

1.1.5 Tabelas no Mysql

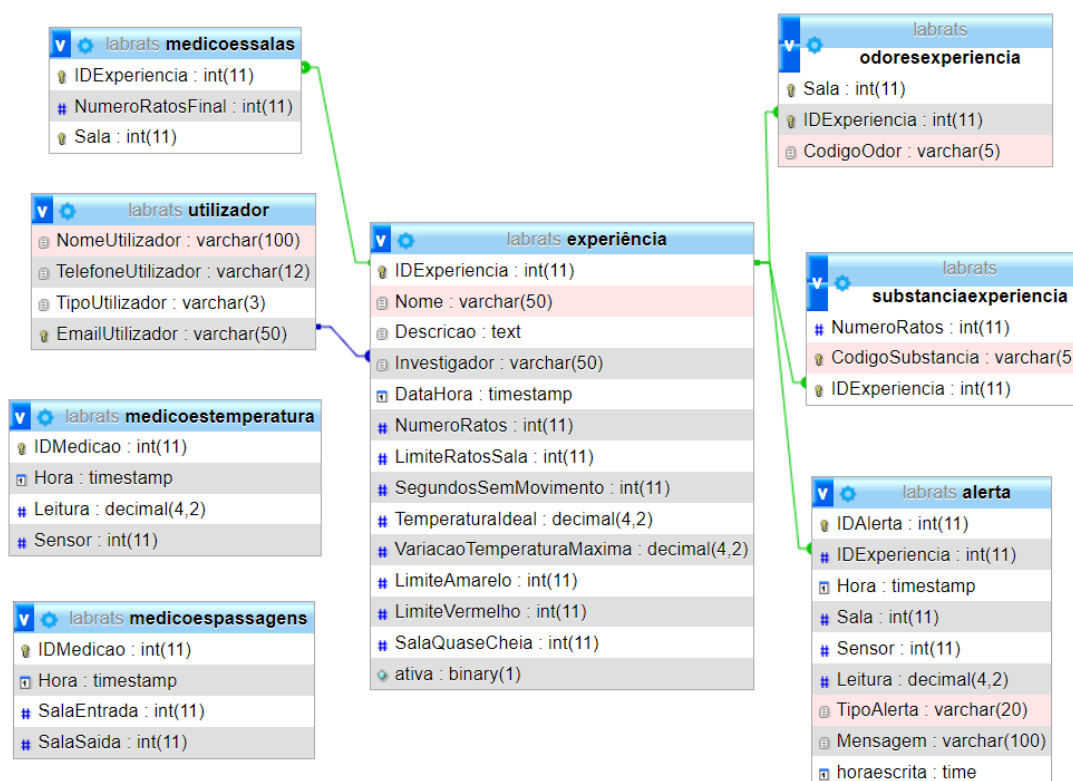


Figura 5

A tabela *experiência* contém os dados que o Investigador escolhe para serem limitadores durante a experiência: "LimiteRatosSala" (indica o número máximo de ratos que uma sala pode conter durante a experiência), "SegundosSemMovimento" (indica o número máximo de segundos entre movimentos) e "VariacaoTemperaturaMaxima" (indica a

variação máxima que a temperatura pode ter). O campo "Investigador" é chave estrangeira do campo "EmailUtilizador" da tabela Utilizador. As tabelas MedicoesPassagens e MedicoesTemperaturas são atualizadas diretamente pelo que é recebido pelo MongoDB, a tabela MedicoesSala é atualizada sempre que é recebida nova informação na tabela MedicoesPassagens, removendo um rato da "SalaEntrada" e adicionando um rato na "SalaSaida". Os campos LimiteAmarelo e LimiteVermelho correspondem a um valor entre 0 e 1 que o Investigador define quando cria a experiência. Estes valores indicam a intensidade com que o alerta é mandado a avisar de que há uma variação de temperatura. Se o valor for pequeno então o alerta é lançado um tempo antes ainda longe do limite. Caso seja um valor mais elevado, perto de 1, então o Investigador é avisado com urgência de que a temperatura está a atingir o limite. É de salientar que o LimiteAmarelo tem que ser inferior ao LimiteVermelho. O campo SalaQuaseCheia permite ao utilizador definir o valor a partir do qual ele considera que a sala começa a aproximar-se da sua capacidade máxima. Por exemplo, se o limite de ratos por sala for 10 e o investigador definir o parâmetro "salaQuaseCheia" com o valor 2 então, quando houver 8 ratos na sala, esta estará quase lotada.

Na tabela Alerta é criado um registo sempre que há demasiados ratos numa sala ou quando a temperatura ultrapassa os limites.

Foi adicionado um campo "ativa" (do tipo binário) inicializado a 0 que indica se a experiencia está ativa(0 representa desativo, 1 indica ativo).

1.1.6 Tratamento de medições pós-Mysql

Após inserir dados novos na base de dados Mysql, é fundamental tratar em conformidade e validar os seguintes valores:

- Quando estamos a avaliar os valores da tabela *MedicoesTemperatura*, iremos verificar se o atributo "Leitura" apresenta um valor de temperatura fora do intervalo estipulado como padrão; se sim, então é lançado um Trigger que avisa o Investigador de que o parâmetro temperatura não está dentro da normalidade. Para além disso, quando é atingido um valor de temperatura anómalo (fora do intervalo

- ver fórmula indicada no ponto 1.1.6.1), a experiência é dada por terminada.

- Na tabela *MedicoesPassagens* é feita uma abordagem semelhante à anteriormente referida, só que neste caso, é relativamente ao número de ratos em cada sala. Se, a quantidade de ratos numa dada sala exceder o número máximo de ratos por sala definido na tabela *Experiencia*, então é lançado um Trigger que notifica o Investigador de que foi excedido o número limite de ratos e que por isso, a experiência irá terminar. Tal acontece porque quando existem demasiados ratos numa sala, o ambiente causa-lhes stress e têm comportamentos que invalidam a experiência.

- Ainda na tabela *MedicoesPassagens*, é necessário confirmar se nenhum rato se movimentou, para dentro ou fora de uma sala, passado X segundos, em todo o labirinto. Caso não tenha ocorrido qualquer movimentação, que indica que todos os ratos estão confortáveis nas suas salas, o Investigador recebe um Trigger em como não há ratos em trânsito e, portanto, a experiência chega ao fim.

Em qualquer um destes cenários, quando a experiência termina, o Investigador é responsável por abrir as portas do labirinto, clicando num botão.

1.1.6.1 Especificação de Triggers (caso existam)

Fórmula para cálculo da Temperatura:

Temperatura máxima=TemperaturaIdeal+VariacaoTemperaturaMaxima;

Temperatura mínima=TemperaturaIdeal-VariacaoTemperaturaMaxima;

Nome Trigger	Tabela	Tipo de Operação (I, U, D)	Evento (After, Before)	Notas (apenas indicar aquilo que não seja óbvio)
TemperaturaForaDoNormal	MedicoesTemperatura	I	Before	Conforme o valor "z" (definido na tabela "Experiencia" no atributo "VariacaoTemperaturaMaxima" pelo Investigador)

ExcessoRat osSala	MedicoesPa ssagens	I	Before	Conforme o valor "y" (definido na tabela "Experiencia" no atributo "LimiteRatosSala" pelo Investigador)
SemMovimen tacao	MedicoesPa ssagens	I	After	Conforme o valor "x" (definido na tabela "Experiência" no atributo "SegundosSemMovime nto" pelo Investigador)

1.1.6.2 Especificação de Procedimentos e Funções (caso existam)

Nome Procediment o / Função	Parâmetros Entrada	Parâmetros Saída	Valores Retorno	Muito breve descrição

1.1.7 Utilizadores Base de Dados Mysql

Após discussão, o grupo decidiu criar 3 utilizadores distintos:

- Administrador

O utilizador do tipo Administrador tem todos os privilégios nas tabelas *Experiencia* e *Utilizador*, de modo a poder controlar todos os mecanismos fulcrais ao desenvolvimento do projeto de monitorização. Em relação às restantes tabelas, estes apenas têm privilégios de leitura.

Consequentemente, podem executar os procedimentos referentes ao seu login, à manutenção de utilizadores e ainda poderá terminar uma experiência.

- Investigador

O utilizador do tipo Investigador apenas poderá visualizar os dados das tabelas. No entanto, tem ainda o poder de, depois de criado, fazer alterações aos seus dados, fazer login, proceder à manutenção das suas experiências, obter as suas experiências e respetivos resultados, começar e terminar uma experiência.

- Migrador

O utilizador do tipo Migrador poderá visualizar a tabela *Utilizador*. Poderá também inserir dados provenientes do MongoDB, nas tabelas de medições e alertas. Os utilizadores deste tipo não podem executar procedimentos.

Sugestão de tabela:

Tabela	Tipo de Utilizador		
	Administrador	Investigador	Migrador
Alerta	L	L	L, I
Experiencia	L, I, D, U	L	-
MedicoesPassagens	L	L	L, I
MedicoesSalas	L	L	L, I
MedicoesTemperatura	L	L	L, I
OdoresExperiencia	L	L	-
SubstanciaExperiencia	L	L	-
Utilizador	L, I, D, U	L	L
Stored Procedure			
CriarUtilizador	X	-	-
RemoverUtilizador	X	-	-
AlterarUtilizador	X	X	-
LoginUtilizador	X	X	-
CriarExperiencia	X	X	-
RemoverExperiencia	X	X	-
EditarExperiencia	X	X	-
ObterExperienciasUtilizador	-	X	-
ObterResultadoFinal	-	X	-
ObterOdoresExperiencia	-	X	-
ObterSubstanciasExperiencia	-	X	-
ComecarExperiencia	-	X	-
TerminarExperiencia	X	X	-
CriarOdor	-	X	-
CriarSubstancia	-	X	-

Em que U=Update, I Insert, D- Delete, L=Leitura, X=Executar SP e - sem permissões.

1.1.8 Procedimentos Manutenção Aplicação

1.1.8.1 Especificação de Procedimentos

Nome Procedimento / Função	Parâmetros Entrada	Parâmetros Saída	Valores Retorno	Muito breve descrição
CriarUtilizador	Email (varchar 50) Tipo (varchar 3) Nome (varchar 100) Telemovel (int 9) Pass (Varchar 50)	-	-	Insere os dados na tabela "Utilizador", sendo necessário fazer a verificação de email e o formato dos dados. É feita a diferenciação entre os vários tipos de utilizador e é-lhes atribuído os seus privilégios.
RemoverUtilizador	EmailUtilizador (varchar 50)	-	-	Remove o utilizador e todas as experiências a ele associadas.
AlterarUtilizador	new_Tipo (varchar 3) new_Nome (varchar 100) new_Telemovel (int 9) Pass (Varchar 50) new_Pass (Varchar 50)	-	-	Atualiza o Tipo e/ou Nome e/ou telemóvel e/ou palavra-passe. Para poder alterar a palavra-passe, é necessário introduzir a antiga. Cada utilizador apenas pode mudar os dados que a ele dizem respeito. No caso de o utilizador ser do tipo Investigador, este não poderá mudar o campo "new_Tipo".

LogInUtilizador	Email (Varchar 50) Pass (Varchar 50)	-	-	Fazer autenticação de um utilizador com as suas credenciais (email e password). É necessário verificar se o email existe (está na base de dados) e a password é a correta.
CriarExperiencia	Nome (Varchar 50) Descricao (Text) Investigador (Varchar 50) DataHora (TimeStamp) NumeroRatos (int) LimiteRatos Sala (int) SegundosSem Movimento (int) Temperatura Ideal (decimal (4,2)) VariacaoTemperaturaMaxima (decimal (4,2)) LimiteAmarelo (decimal (4,2)) LimiteVermelho (decimal (4,2)) SalaQuaseChueia (int)	-	-	O criador da experiência tem de estar registado como utilizador e com privilégios de Investigador ou Administrador. No caso de ser um Investigador, este não poderá criar experiências envolvendo outros Investigadores, ou seja, o campo "Investigador" será automaticamente preenchido com o seu email. No caso do utilizador ser um Administrador, este poderá escolher qual o Investigador que ficará alocado à experiência.
RemoverExperiencia	IDExperiencia (int)	-	-	É necessário confirmar que o email associado ao utilizador que pretende apagar a

				experiência é o email do Investigador responsável pela experiência.
EditarExperiencia	new_Nome (varchar 50) new_Descricao (Text) new_DataHora (TimeStamp) new_NumeroRatos (int) new_LimiteRatosSala (int) new_SegundosSemMovimento (int) new_TemperaturaIdeal (decimal (4,2)) new_VariacaoTemperaturaMaxima (decimal (4,2)) new_LimiteAmarelo (decimal (4,2)) new_LimiteVermelho (decimal (4,2)) new_SalaQuaseCheia (int)	-	-	Atualiza a Descrição e/ou a Data/Hora e/ou NumeroRatos e/ou LimiteRatosSala e/ou SegundosSemMovimento e/ou TemperaturaIdeal e/ou VariacaoTemperaturaMaxima. O utilizador (Admin ou Investigador) só pode alterar a experiência antes desta ser iniciada.
ObterExperienciasUtilizador	-	-	IDExperiencia (int) Nome (varchar 50) Descricao (Text)	Devolve todos os Ids e respetivas Descrições das experiências correspondentes ao utilizador que chama o procedimento.
ObterResultadoFinal	IDExperiencia (int)	-	NumeroRatosFinal (int)	Devolve o número de ratos final e a sala da experiência

			Sala (int)	correspondente, se esta pertencer ao Investigador.
ObterOdoresExperiência	IDExperiencia (int)	-	Sala(int) CodigoOdor(varchar 5)	Devolve os Odores e as respectivas Salas da experiência correspondente, se esta pertencer ao Investigador.
ObterSubstanciasExperiência	IDExperiencia (int)	-	NumeroRatos(int) CodigoSubstancia (varchar 5)	Devolve as substâncias e o número de ratos injetados pela mesma da experiência correspondente, se esta pertencer ao Investigador.
ComecarExperiencia	IDExperiencia (int)	-	-	Procedimento que altera o campo "ativa" para 1 e inicia a experiência. É preciso verificar que o utilizador que executa esta ação corresponde ao Investigador da experiência. (*)
TerminarExperiencia	IDExperiencia (int)	-	-	Procedimento que altera o campo "ativa" para 0, indicando assim que a experiência acabou. A experiência tem que pertencer ao Investigador.
CriarOdor	IDExperiencia (int) CodigoOdor (varchar 5) Sala(int)	-	-	Procedimento que associa um odor a uma dada sala.

CriarSubstancia	IDExperiencia (int) NumeroRatos (int) CodigoSubstancia (varchar 5)	-	-	Procedimento que permite ao Investigador injetar com uma dada substância um dado número de ratos.
-----------------	--	---	---	---

Na tentativa de executar algum dos procedimentos acima descritos e ocorrer uma falha, quer porque os parâmetros não estão corretos ou não tem os privilégios necessários para os fazer, o utilizador é informado da impossibilidade de executar o procedimento em causa.

(*) Especificação do campo DataHora no procedimento ComeçarExperiencia:

Quando o utilizador (Administrador ou Investigador) cria uma experiência é necessário que indique uma data e hora para que esta inicie. No entanto, como se trata de um caso prático que envolve aspetos exteriores à vontade do Investigador (problemas com portas, ar condicionados, sensores, etc) é possível iniciar a experiência antes ou depois da "DataHora" estipulada.

Caso o Investigador pretenda dar início à experimentação antes da data definida, ao clicar no botão que inicializa o estudo, o campo "DataHora" é automaticamente alterado para o instante em que o botão é apertado (current_time).

Se, por algum motivo, a experiência não puder ser realizada na data e hora pré-estabelecidas então, quando faltar 1 minuto para o início da experiência, o Investigador receberá um aviso que lhe indicará que a experiência está prestes a começar e que se quiser, poderá adiá-la. Caso contrário a experiência começa. O Investigador terá acesso ao botão de iniciar a experimentação e nessa altura o campo DataHora será atualizado para o instante em que tal ação acontece (current_time).

1.1.9 Explicação detalhada dos tratamentos aos dados

No que toca a tratamento de outliers e de valores errados vindos do mongo propomos ser tratado no java. Onde recebemos as temperaturas de 5 em 5 segundos e tratamos de filtrar as mesmas usando as funções `separa_temps()` e `Outliers()` anteriormente referidas. Os movimentos dos ratos recebem um tratamento semelhante, usando a função `verifica_parametros()`, onde os parâmetros de movimentos são verificados.

Em relação aos alertas, o grupo decidiu recorrer a quatro:

- Amarelos- subida/descida da temperatura e aproximação ao limite amarelo definido pelo investigador;
- Vermelhos - subida/descida de temperatura e aproximação ao limite de temperatura definido pelo investigador;
- Verde - Começo de uma experiência;
- Roxo - limite de ratos próximo de ser atingido.

Os alertas amarelo e vermelho, referentes a temperatura, são manipuláveis pelo investigador graças aos atributos "`limiteAmarelo`" e "`limiteVermelho`" definidos na criação de experiências. Os mesmos calculam-se da seguinte forma:

Limites amarelos:

- $\text{superior} = \text{temperatura_ideal} + \text{limiteAmarelo} * \text{variacaoTemperaturaMaxima}$
- $\text{inferior} = \text{temperatura_ideal} - \text{limiteAmarelo} * \text{variacaoTemperaturaMaxima}$

Limites vermelhos:

- $\text{superior} = \text{temperatura_ideal} + \text{limiteVermelho} * \text{variacaoTemperaturaMaxima}$
- $\text{inferior} = \text{temperatura_ideal} - \text{limiteVermelho} * \text{variacaoTemperaturaMaxima}$

O alerta verde refere-se ao início da experiência. Este serve para alertar o respetivo investigador que o início de uma experiência está perto (um minuto de distância) e dá a última oportunidade de se alterar a data prevista da experiência.

Se nada ocorrer (se o investigador não pressionar o botão de cancelar experiência) a experiência começa.

Por último, o alerta roxo vai avisar o investigador quando o número de ratos de uma sala está próximo do limite de ratos por sala definido. Este é também manipulável pelo investigador, ou seja, o mesmo consegue definir quando é que é avisado da proximidade do limite graças ao parâmetro "salaQuaseCheia", que define essa diferença. Por exemplo, se o limite de ratos por sala for 10 e o investigador definir o parâmetro "salaQuaseCheia" com o valor 2 então o alerta é lançado quando existirem 8 ratos na mesma sala.

1.1.10 Eventos de suporte à aplicação (caso existam)

Nome Evento	Local Execução (MySQL/Windows)	Muito breve descrição

1.2 Consulta por HTML/PHP

1.2.1 Layout dos formulários HTML

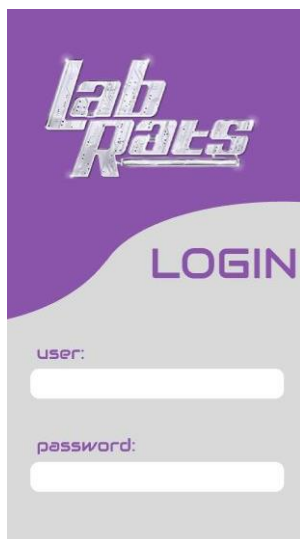


Figura 6



Figura 7



Figura 8

Figura 9

Figura 10

Figura 11

Figura 12

Figura 13

1.2.2 Associar SP a formulários HTML

Login (fig. 6): O utilizador inicia sessão com o e-mail e a password que lhe foi atribuída. SP: LoginUtilizador.

Ecrã principal (fig. 7): Ecrã para onde o utilizador é redirecionado após o login. Através do ecrã principal é possível executar quatro operações: visualizar e editar experiências, criar experiências, visualizar o histórico de alertas e ainda consultar detalhes relativos ao utilizador. SP: sem SPs.

O campo **Experiências (fig. 8)** dá acesso a três tipo de experiências:

- Ativas: é possível consultar todas as experiências que estão ativas. Após carregar no símbolo "+" conseguimos visualizar o ecrã com detalhes da experiência com o SP: ObterExperienciasUtilizador. Para além disso é possível editar e remover essa experiência com os SPs: EditarExperiencia e RemoverExperiencia, respetivamente. Ao carregar no ícone editar o utliizador tem acesso a um novo ecrã **"Editar experiência" (fig. 10)** onde pode guardar as alterações feitas.
- Agendadas: é possível consultar todas as experiências que estão agendadas. Após carregar no símbolo "+" conseguimos visualizar o ecrã com **detalhes da experiência (fig. 9)** com o SP: ObterExperienciasUtilizador. Para além disso é possível editar e remover essa experiência com os SPs: EditarExperiencia e RemoverExperiencia, respetivamente. Aqui é dada a possibilidade ao utilizador de começar a experiência antes do tempo previsto através do SP: ComecarExperiencia. Ao carregar no ícone editar o utilizador tem acesso a um novo ecrã **"Editar Experiência" (fig. 10)** onde pode guardar as alterações feitas.
- Histórico de experiências: contém uma lista de todas as experiências já terminadas e inativas. É possível visualizar para além dos detalhes da experiência a conclusão da mesma.

O campo **criar experiências (figura 7)**: dá acesso a uma página **(fig. 13)** de **nova experiência** onde é possível inserir os dados para uma experiência nova. SP: CriarExperiencia.

O campo **histórico de alertas (figura 7)** dá acesso aos **alertas** recebidos pelo utilizador em todas as experiências **(fig. 12)**.

Detalhes do user (fig. 11): visualização de informações do utilizador e possibilidade de alterar alguns campos. SP: AlterarUtilizador.

É importante referir que em todos os ecrãs existe a possibilidade de fazer logout e de retornar à página anterior exceto no ecrã principal e no login.

3.1 Auto-Avaliação Global de especificações entregues outro grupo

Auto-Avaliação Global da Qualidade das especificações que entregaram ao outro grupo (descrita nos parágrafos anteriores)

Avaliação (A,B,C,D,E) : A

Utilize a seguinte escala:

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

4 Implementação

4.1 Transporte de MongoDB para Mysql

A – Especificação recebida outro grupo

Justificação das alterações

Decidimos implementar a periodicidade definida pelo outro grupo, de 1 segundo, devido ao facto de termos arranjado uma nova forma de filtrar os outliers, não precisando assim de esperar 5 segundos para os poder comparar.

B – Best Off

Justificação das alterações

Continuámos com a nossa abordagem em relação aos documentos duplicados, usando uma coleção para os movimentos por enviar e para as temperaturas por enviar, removendo os documentos dessa coleção assim que lidos, evitando assim ler duas vezes o mesmo documento.

Em relação às threads, fizemos alterações à nossa estratégia inicial. Em vez de utilizar várias threads, decidimos não usar nenhuma, visto que não nos trazia nenhum benefício, muito pelo contrário. Decidimos, portanto, usar duas mains, uma para os movimentos e outra para as temperaturas.

4.2 Tabelas no Mysql

A – Especificação recebida outro grupo

experiencia	medicoespasagem
IDExperiencia (AutoIncrement) (ChavePrimária)	IDMedicao (ChavePrimaria)(autoincrement)
Descricao (text)	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
Investigador (varchar 50)	Hora (timestamp)
DataHora (timestamp)	SalaEntrada (int)
NumeroRatos (int)	SalaSaida (int)
LimiteRatosSala (int)	Ativo (int)
SegundosSemMovimento (int)	ComentarioAdicional(text)
TemperaturaIdeal (decimal 4,2)	
VariacaoTemperaturaMaxima (decimal 4,2)	medicoestemperatura
Ativo (int)	IDMedicao (int) (ChavePrimaria)(autoincrement)
	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
substanciaexperiencia	Hora (timestamp)
IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)	Leitura(decimal 4,2)
CodigoSubstancia (int)	Sensor (int)
NumeroRatos (int)	Ativo (int)
Ativo (int)	ComentarioAdicional(text)
odoresexperiencia	alerta
IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)	IDAlerta (int) (ChavePrimaria) (autoincrement)
sala	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
CodigoOdor (varchar 50)	Hora (timestamp)
Ativo (int)	Sala (int)
	Sensor (int)
Utilizador	Leitura(decimal 4,2)
IDUtilizador (int) (ChavePrimária)(autoincrement)	TipoAlerta (varchar 20)
NomeUtilizador (varchar 100)	Mensagem (varchar 100)
TelefoneUtilizador (varchar 12)	HoraEscrita (timestamp)
EmailUtilizador (varchar 50) Unique	Ativo (int)
TipoUtilizador(vvarchar 3)	
Ativo (int)	medicoessala
	IDExperiencia (int) (ChavePrimária)(Chave Estrangeira)
	Sala(int) (ChavePrimária)
	NumeroRatosFinal (int)
	Ativo (int)

Figura 14

Justificação das alterações

Todas as tabelas são constituídas por um campo “Ativo”, que verifica a “eliminação” das mesmas, ou seja, quando estas não estão a decorrer são dadas como inativas, em detrimento da experiência a decorrer. Como descrito anteriormente, nesse caso apenas a experiência ativa seria mostrada em certos procedimentos. Como melhoria, foi retirado o campo “Ativo” de todas as tabelas, exceto a tabela “experiencia”, e não é utilizada a ideia de “eliminar uma experiência”, mas sim de preservar os valores desta, para futura avaliação por parte do investigador associado. No caso de uma experiência estar ativa, sabe-se que todos os registos de outras tabelas a ela associados farão parte da mesma e, portanto, será desnecessária a utilização do mesmo. Adicionalmente, este campo foi convertido a booleano, de modo a ser mais fácil a sua diferenciação. Quando este se encontra a 1 a experiência estará ativa, enquanto se o seu valor for 0 a experiência estará inativa.

Em relação ao campo “CampoAdicional” em certas tabelas, considerou-se a sua não execução, pois os dados errados e outliers são tratados em pré-mysql e, portanto, nunca chegarão à base de dados.

B – Best Off

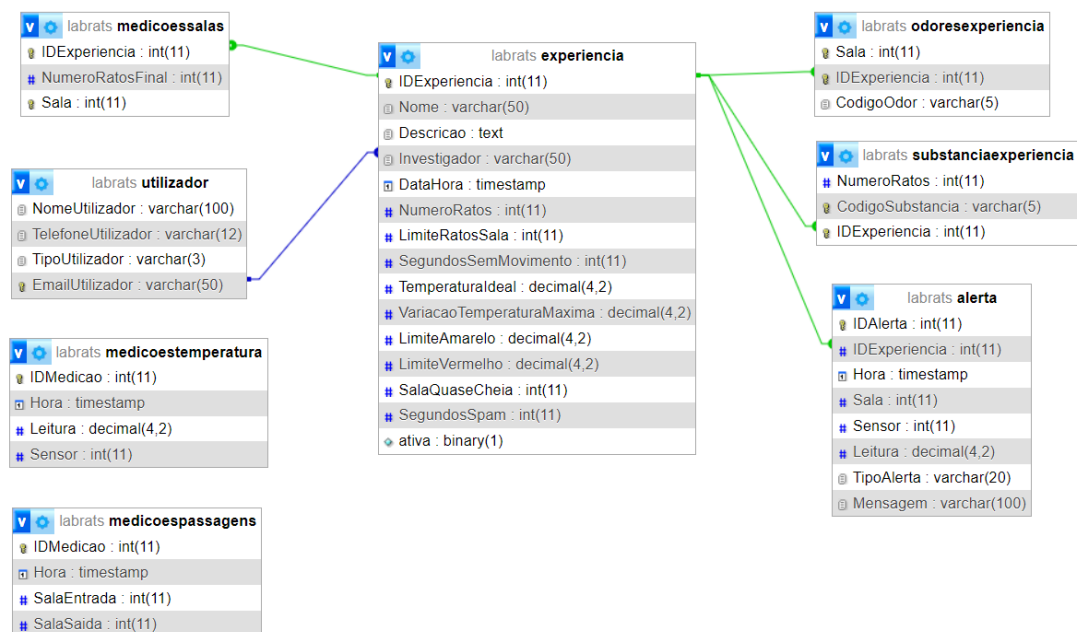


Figura 15

Justificação das alterações

Na tabela “experiencia” foi adicionado o campo “SegundosSpam”, campo que define o tempo mínimo, em segundos, entre alertas do mesmo tipo, de modo a não incomodar em demasia o Investigador com a mesma informação, se este o desejar.

O campo “horaescrita” foi retirado da tabela “alerta”, devido à não utilização do mesmo.

4.3 Tratamento de medições Pré-Mysql

A – Especificação recebida outro grupo

Nome Procediment o / Função	Parâmetros Entrada	Valores Retorno (função)	=, Novo, Alterado

Justificação das alterações

Texto justificativo

B - Best Off

Nome Procedimento / Função	Parâmetros Entrada	Valores Retorno (função)	=, Novo, Alterado
SendObject	Alerta ou dadomovimento ou dadotemp	- (envia para a BD MySql os dados)	Novo
toDadoMov	String(dado) lido do MongoDB	- (cria o dadoMovimento, valida-o, vê se é necessário enviar alerta e envia o suposto (alerta e/ou dadoMovimento))	Novo
toDadoTemp	String(dado) lido do MongoDB	Equivalente à função supra	Novo
ConnectToMongo	-	- Faz a conexão ao MongoDB e chama as duas funções anteriores para cada dado recebido	Novo

Justificação das alterações

Achámos melhor começar do zero, pois embora estivessem corretas, ao tentar implementar qualquer uma das duas especificações, deparámo-nos com erros ou falta de eficiência na implementação. Criámos também diversas funções auxiliares e classes para um melhor tratamento dos dados e as mesmas são usadas pelas funções anteriormente referidas, nomeadamente, "CreateDado()", "IsSalaCheia()", classe "AlertTester", "AddAlerta()", "IsOutlier()", entre outras.

4.4 Tratamento de medições pós-Mysql

A – Especificação recebida outro grupo

4.4.1 Especificação de Triggers implementados (outro grupo)

Nome Trigger	Tabela	Tipo de Operação (I,U,D)	Evento (After, Before)	=, Novo, Alterado

4.4.2 Código de Triggers implementados (outro grupo)

```
1. Nome Trigger: _____  
// Breve Descrição  
Código
```

```
2. Nome Trigger: _____  
// Breve Descrição  
Código
```

```
3. Nome Trigger: _____  
// Breve Descrição  
Código
```

Justificação das alterações

Texto justificativo

B – Best Off

4.4.3 Especificação de Triggers implementados (best off)

Nome Trigger	Tabela	Tipo de Operação (I,U,D)	Evento (After, Before)	=, Novo, Alterado
TemperaturaForaDoNormal	medicoeste mperatura	I	After	Alterado

ExcessoRatosSala	medicoessalas	U	After	Alterado
SemMovimentacao	alerta	I	After	=
AtualizarNumeroRatosSala	medicoessagens	I	After	Novo

4.4.4 Código de Triggers implementados (best off)

1. Nome Trigger: TemperaturaForaDoNormal

Este trigger verifica a leitura da temperatura recebida na tabela "medicoestemperatura". Se esta estiver fora do valor máximo de variação da temperatura, definido pelo utilizador na criação da experiência, a experiência será terminada.

```
CREATE TRIGGER `TemperaturaForaDoNormal` AFTER INSERT ON
`medicoestemperatura`
FOR EACH ROW BEGIN
    DECLARE id INT;
    DECLARE var DECIMAL(4,2);
    DECLARE temp DECIMAL(4,2);
    DECLARE max_temp DECIMAL(4,2);
    DECLARE min_temp DECIMAL(4,2);

    SELECT IDExperiencia INTO id FROM experiencia WHERE
ativa = 0x01;
    SELECT VariacaoTemperaturaMaxima INTO var FROM
experiencia WHERE IDExperiencia = id;
    SELECT TemperaturaIdeal INTO temp FROM experiencia
WHERE IDExperiencia = id;
    SET max_temp = temp + var;
    SET min_temp = temp - var;
    IF (NEW.Leitura >= max_temp OR NEW.Leitura <= min_temp)
THEN
        UPDATE experiencia SET ativa = 0x00 WHERE
IDExperiencia = id;
    END IF;
END
```

2. Nome Trigger: ExcessoRatosSala

Este trigger verifica o número de ratos em cada sala, com exceção da sala 1 que pode ter o número máximo de ratos. No caso de o número de ratos numa dada sala ser superior ao limite de ratos por sala, a experiência termina.

```

CREATE TRIGGER `ExcessoRatosSala` AFTER UPDATE ON
`medicoessalas`
FOR EACH ROW BEGIN
    DECLARE id INT;
    DECLARE lim INT;

    SELECT IDEXperiencia INTO id FROM experiencia WHERE
ativa = 0x01;
    SELECT LimiteRatosSala INTO lim FROM experiencia WHERE
IDExperiencia = id;

    IF (NEW.NumeroRatosFinal >= lim AND NEW.Sala != 1) THEN
        UPDATE experiencia SET ativa = 0x00 WHERE
IDExperiencia = id;
    END IF;
END

```

3. Nome Trigger: SemMovimentacao

Este trigger verifica o número de segundos sem movimento de ratos. Ao fim do tempo, o utilizador recebe um alerta que lhe indica que atingiu o máximo de tempo sem movimento. Neste momento, surge ao utilizador a oportunidade de prolongar o tempo até novo movimento, com o procedimento "AlterarSegundosExp". No caso de este não querer prolongar o tempo, a experiência termina.

```

CREATE TRIGGER `SemMovimentacao` AFTER INSERT ON `alerta`
FOR EACH ROW
BEGIN
    DECLARE id INT;
    DECLARE seg DECIMAL(4,2);
    DECLARE segs INT;
    DECLARE segi DECIMAL(4,2);

    IF(NEW.TipoAlerta = 'Roxo ') THEN

        SELECT IDEXperiencia INTO id FROM experiencia
WHERE ativa = 0x01;

        SELECT SegundosSemMovimento INTO seg FROM
experiencia WHERE IDEXperiencia = id;

        SELECT SegundosSpam INTO segs FROM experiencia
WHERE IDEXperiencia = id;

        SET segi = CAST((seg - segs) AS DECIMAL(4,2));

        IF(NEW.Leitura >= segi AND NEW.Leitura < (CAST(seg
AS DECIMAL(4,2)))) THEN

```



```

        UPDATE alerta SET Mensagem = CONCAT(Mensagem, ',
altere os SegundosSemMovimento se quiser prolongar a
experiência') WHERE IDAlerta= NEW.IDAlerta;
    END IF;

    IF(new.Leitura = (CAST(seg AS DECIMAL(4,2)))) THEN
        UPDATE experiencia SET ativa = 0x00 WHERE
IDExperiencia = id;
    END IF;
END IF;
END

```

4. Nome Trigger: AtualizarNumeroRatosSala

Este trigger tem como função atualizar o campo "NumeroRatosFinal", a cada mudança de salas na tabela "medicoespasagens", de modo a ter sempre atualizado o número de ratos em cada sala.

```

CREATE TRIGGER `AtualizarNumeroRatosSala` AFTER INSERT ON
`medicoespasagens` FOR EACH ROW
BEGIN
DECLARE experiencia_id INT;
SELECT IDExperiencia INTO experiencia_id FROM experiencia
WHERE ativa = 0x01;

IF ((SELECT COUNT(*) FROM medicoessalas WHERE Sala =
new.SalaEntrada)>0) THEN
UPDATE medicoessalas
SET NumeroRatosFinal = CASE
    WHEN NumeroRatosFinal + 1 < 0 THEN 0
    ELSE NumeroRatosFinal + 1
END
WHERE Sala = NEW.SalaEntrada;
ELSE
INSERT INTO medicoessalas
(IDExperiencia,NumeroRatosFinal,Sala) VALUES
(experiencia_id,1,new.SalaEntrada);
END IF;
UPDATE medicoessalas
SET NumeroRatosFinal = CASE
    WHEN NumeroRatosFinal - 1 < 0 THEN 0
    ELSE NumeroRatosFinal - 1
END
WHERE Sala = NEW.SalaSaida;
END

```

Justificação das alterações

O trigger “TemperaturaForaDoNormal” tem a mesma funcionalidade e características anteriormente especificadas, mas ocorre depois de serem inseridos os valores (AFTER). Tal acontece de forma a serem guardados os valores de término, de modo a o investigador conseguir entender o porquê de a experiência ter terminado e poder tirar as suas próprias conclusões.

Da mesma forma, o trigger “ExcessoRatosSala” sofreu a mesma alteração. Adicionalmente, foi corrigido o tipo de operação para “Update”, pois este trigger é executado sempre que o valor “NumeroRatosFinal” na tabela “medicoessalas” é alterado. No término da experiência, o valor final será o definido na tabela.

4.4.5 Especificação de Store Procedures implementados (outro grupo)

<Listar todos os SP e funções implementados. Na última coluna indicar se foi implementado tal como especificado, se é novo, ou se foi alterada a especificação>

A – Especificação recebida outro grupo

Nome Procedimento / Função	Parâmetros Entrada	Valores Retorno (função)	=, Novo, Alterado

Justificação das alterações

Texto justificativo

4.4.6 Código Stored Procedures implementados (outro grupo)

```
1. Nome SP: CriarUtilizador
// Breve Descrição
Código

2. Nome SP: EditarExperiência
// Breve Descrição
Código
```

4.4.7 Especificação de Store Procedures implementados (best off)

Nome Procedimento / Função	Parâmetros Entrada	Valores Retorno (função)	=, Novo, Alterado
CriarUtilizador	Email (varchar 50) Tipo (varchar 3) Nome (varchar 100) Telemovel (int 9) Pass (varchar 50)	-	=
RemoverUtilizador	EmailUtilizador (varchar 50)	-	=
EditarUtilizador	EmailUtilizador (varchar 50) new_EmailUtilizador (varchar 50) new_Nome (varchar 50) new_Telefone (int 9) new_Pass (varchar 50)	-	=
CriarExperienciaInv	Nome (varchar 50) Descricao (text) NumeroRatos (int) LimiteRatosSala (int) SegundosSemMovimento (int) TemperaturaIdeal (decimal (4,2)) VariacaoTemperaturaMaxim	-	=

	a (decimal (4,2)) LimiteAmarelo (decimal (4,2)) LimiteVermelho (decimal (4,2)) SalaQuaseCheia (int) SegundosSpam (int)		
CriarExperienciaAdm	Investigador (varchar 50) Nome (varchar 50) Descricao (text) NumeroRatos (int) LimiteRatosSala (int) SegundosSemMovimento (int) TemperaturaIdeal (decimal (4,2)) VariacaoTemperaturaMaxima (decimal (4,2)) LimiteAmarelo (decimal (4,2)) LimiteVermelho (decimal (4,2)) SalaQuaseCheia (int) SegundosSpam (int)	-	=
RemoverExperienciaInv	IDExperiencia (int)	-	=
RemoverExperienciaAdm	IDExperiencia (int)	-	=

EditarExperienciaInv	IDExperiencia (inv) new_Nome (varchar 50) new_Descricao (text) new_NumeroRatos (int) new_LimiteRatosSala (int) new_SegundosSemMovimento (int) new_TemperaturaIdeal (decimal (4,2)) new_VariacaoTemperaturaMaxima (decimal (4,2)) new_LimiteAmarelo (decimal (4,2)) new_LimiteVermelho (decimal (4,2)) new_SalaQuaseCheia (int) new_SegundosSpam (int)	-	=
EditarExperienciaAdm	IDExperiencia (int) new_Investigador (varchar 50) new_Nome (varchar 50) new_Descricao (text) new_DataHora (timeStamp) new_NumeroRatos (int) new_LimiteRatosSala (int)	-	=

	new_Segundos SemMovimento (int) new_Temperat uraIdeal (decimal (4,2)) new_Variacao TemperaturaM axima (decimal (4,2)) new_LimiteAm arelo (decimal (4,2)) new_LimiteVe rmelho (decimal (4,2)) new_SalaQuas eCheia (int) new_Segundos Spam (int)		
CriarOdor	IDExperienci a (int) CodigoOdor (varchar 5) Sala (int)	-	=
CriarSubstan cia	IDExperienci a (int) NumeroRatos (int) CodigoSubsta ncia (varchar 5)	-	=
ComecarExper iencia	IDExperienci a (int)	-	=
TerminarExpe riencia	IDExperienci a (int)	-	=
ObterExperie nciasUtiliza dor	-	IDExperienci a (int) Nome (varchar 50) Descricao (text) Investigador (varchar 50) NumeroRatos (int) LimiteRatosS	=

		ala (int) SegundosSemM ovimento (int) TemperaturaI deal (decimal (4,2)) VariacaoTemp eraturaMaxim a (decimal (4,2)) LimiteAmarel o (decimal (4,2)) LimiteVermel ho (decimal (4,2)) SalaQuaseChe ia (int) SegundosSpam (int) Ativa (binary)	
ObterOdoresE xperiencia	IDExperienci a (int)	CodigoOdor (varchar 5) Sala (int)	=
ObterSubstan ciasExperien cia	IDExperienci a (int)	CodigoSubsta ncia (varchar 5) NumeroRatos (int)	=
ObterResulta doFinal	IDExperienci a (int)	NumeroRatosF inal (int) Sala (int)	=
AlterarSegun dosExp	IDExperienci a (int) SomaSegs (int)	-	Novo

Justificação das alterações

Os procedimentos com "Inv", para o investigador e "Adm", para o administrador, foram especificados em conjunto na fase 1, mas definidos separadamente na fase 2.

4.4.8 Stored Procedures implementados (best off)

1. Nome SP: CriarUtilizador

Procedimento que permite criar um utilizador, atribuindo as ROLES especificadas no argumento "tipo" ao mesmo. (Argumentos possíveis e respectivas ROLES: Adm - Administrador da App, Inv - Investigador, Mig - Migrador) É utilizado o email como identificador do utilizador.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`CriarUtilizador`(IN `email` VARCHAR(50), IN `tipo`
VARCHAR(3), IN `nome` VARCHAR(100), IN `telemovel`
INT(12), IN `pass` VARCHAR(50))
BEGIN
IF(email LIKE '%@%') THEN
INSERT INTO
utilizador(NomeUtilizador, TelefoneUtilizador,
TipoUtilizador, EmailUtilizador)
VALUES(nome, telemovel, tipo, email);
SET @Create = CONCAT('CREATE USER "',email, '" IDENTIFIED
BY "', pass, '";');
PREPARE statement FROM @Create;
EXECUTE statement;
IF (tipo LIKE 'Inv') THEN
SET @query = CONCAT('GRANT Investigador TO "', email,
'";');
PREPARE statement FROM @query;
EXECUTE statement;
SET @query =CONCAT('SET DEFAULT ROLE Investigador FOR "',
email, '";');
PREPARE statement FROM @query;
EXECUTE statement;
ELSEIF (tipo LIKE 'Adm') THEN
SET @query = CONCAT('GRANT AdminApp TO "', email, '";');
PREPARE statement FROM @query;
EXECUTE statement;
SET @query =CONCAT('SET DEFAULT ROLE AdminApp FOR "',
email, '";');
PREPARE statement FROM @query;
EXECUTE statement;
ELSEIF (tipo LIKE 'Mig') THEN
SET @query = CONCAT('GRANT Migrador TO "', email, '";');
PREPARE statement FROM @query;
EXECUTE statement;
SET @query =CONCAT('SET DEFAULT ROLE Migrador FOR "',
email, '";');
PREPARE statement FROM @query;
```



```

EXECUTE statement;
END IF;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Email tem de possuir um formato
válido';
END IF;
END$$
DELIMITER;

```

2. Nome SP: CriarExperienciaInv

Procedimento que permite ao Investigador criar uma experiência.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`CriarExperienciaInv`(IN `NomeExp` VARCHAR(50), IN
`DescricaoExp` TEXT, IN `NumeroRatosExp` INT(11), IN
`LimiteRatosSalaExp` INT(11), IN
`SegundosSemMovimentoExp` INT(11), IN
`TemperaturaIdealExp` DECIMAL(4,2), IN
`VariacaoTemperaturaMaximaExp` DECIMAL(4,2), IN
`LimiteAmareloExp` DECIMAL(4,2), IN `LimiteVermelhoExp`
DECIMAL(4,2), IN `SalaQuaseCheiaExp` INT(11), IN
`SegundosSpamExp` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
INSERT INTO experiencia(Nome, Descricao, Investigador,
NumeroRatos, LimiteRatosSala, SegundosSemMovimento,
TemperaturaIdeal, VariacaoTemperaturaMaxima,
LimiteAmarelo, LimiteVermelho, SalaQuaseCheia,
SegundosSpam)
VALUES(NomeExp, DescricaoExp, @nome, NumeroRatosExp,
LimiteRatosSalaExp, SegundosSemMovimentoExp,
TemperaturaIdealExp, VariacaoTemperaturaMaximaExp,
LimiteAmareloExp, LimiteVermelhoExp, SalaQuaseCheiaExp,
SegundosSpamExp);
END$$
DELIMITER;

```

3. Nome SP: CriarExperienciaAdm

Procedimento que permite ao Administrador a criação de uma experiência e atribuição da mesma a um Investigador.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`CriarExperienciaAdm`(IN `InvestigadorExp` VARCHAR(50),

```

```

IN `NomeExp` VARCHAR(50), IN `DescricaoExp` TEXT, IN
`NumeroRatosExp` INT(11), IN `LimiteRatosSalaExp`
INT(11), IN `SegundosSemMovimentoExp` INT(11), IN
`TemperaturaIdealExp` DECIMAL(4,2), IN
`VariacaoTemperaturaMaximaExp` DECIMAL(4,2), IN
`LimiteAmareloExp` DECIMAL(4,2), IN `LimiteVermelhoExp`
DECIMAL(4,2), IN `SalaQuaseCheiaExp` INT(11), IN
`SegundosSpamExp` INT(11))
BEGIN
IF ((SELECT COUNT(*) FROM utilizador WHERE
InvestigadorExp = EmailUtilizador AND TipoUtilizador =
"Inv")>0) THEN
INSERT INTO experiencia(Investigador, Nome, Descricao,
NumeroRatos, LimiteRatosSala, SegundosSemMovimento,
TemperaturaIdeal, VariacaoTemperaturaMaxima,
LimiteAmarelo, LimiteVermelho, SalaQuaseCheia,
SegundosSpam)
VALUES(InvestigadorExp, NomeExp, DescricaoExp,
NumeroRatosExp, LimiteRatosSalaExp,
SegundosSemMovimentoExp, TemperaturaIdealExp,
VariacaoTemperaturaMaximaExp, LimiteAmareloExp,
LimiteVermelhoExp, SalaQuaseCheiaExp, SegundosSpamExp);
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Impossível criar experiência.
Verifique que existe o utilizador pedido';
END IF;
END$$
DELIMITER;

```

4. Nome SP: EditarExperienciaInv

Procedimento que permite ao Investigador alterar qualquer uma das suas experiências inativas (no caso, alterar os parâmetros que consideramos alteráveis).

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`EditarExperienciaInv`(IN `IDExp` INT(11), IN `nomeExp`
VARCHAR(50), IN `descr` TEXT, IN `numR` INT(11), IN
`LimRa` INT(11), IN `SegSem` INT(11), IN `Temp`
DECIMAL(4,2), IN `var` DECIMAL(4,2), IN `LimA`
DECIMAL(4,2), IN `LimV` DECIMAL(4,2), IN `salaQ` INT(11),
IN `SegSpam` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF (((SELECT count(*) FROM experiencia WHERE
IdExperiencia = idExp AND Investigador = @nome)>0))THEN

```

```

        IF(((SELECT count(*) FROM experiencia WHERE
IdExperiencia = idExp AND Investigador = @nome AND ativa
= 0x00 )>0)) THEN
            UPDATE experiencia SET
                Nome= IFNULL(nomeExp, Nome),
                Descricao = IFNULL(descr, Descricao),
                NumeroRatos = IFNULL(numR, NumeroRatos),
                LimiteRatosSala = IFNULL(LimRa, LimiteRatosSala),
                SegundosSemMovimento = IFNULL(SegSem,
SegundosSemMovimento),
                TemperaturaIdeal = IFNULL(Temp,
TemperaturaIdeal),
                VariacaoTemperaturaMaxima      = IFNULL(var,
VariacaoTemperaturaMaxima),
                LimiteAmarelo = IFNULL(LimA, LimiteAmarelo),
                LimiteVermelho = IFNULL(LimV, LimiteVermelho),
                SalaQuaseCheia = IFNULL(salaQ, SalaQuaseCheia),
                SegundosSpam = IFNULL(SegSpam, SegundosSpam)
            WHERE IdExperiencia = idExp;
        ELSE
            SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Impossível editar experiência, a
mesma está a decorrer.';
        END IF;
    ELSE
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Impossível editar experiência, não
tem permissões para tal.';
    END IF;
END$$
DELIMITER;

```

5. Nome SP: EditarExperienciaAdm

Procedimento que permite ao Administrador alterar qualquer experiência inativa.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`EditarExperienciaAdm`(IN `IDExp` INT(11), IN `NomeInv`
VARCHAR(50), IN `nomeExp` VARCHAR(50), IN `descr` TEXT,
IN `numR` INT(11), IN `LimRa` INT(11), IN `SegSem`
INT(11), IN `Temp` DECIMAL(4,2), IN `var` DECIMAL(4,2),
IN `LimA` DECIMAL(4,2), IN `LimV` DECIMAL(4,2), IN
`salaQ` INT(11), IN `SegSpam` INT(11))
BEGIN
    IF (((SELECT count(*) FROM experiencia WHERE
IdExperiencia = idExp)>0))THEN
        IF(((SELECT count(*) FROM experiencia WHERE
IdExperiencia = idExp AND ativa = 0x00 )>0)) THEN

```

```

UPDATE experiencia SET
    Investigador = IFNULL(NomeInv, Investigador),
    Nome = IFNULL(nomeExp, Nome),
    Descricao = IFNULL(descr, Descricao),
    NumeroRatos = IFNULL(numR, NumeroRatos),
    LimiteRatosSala = IFNULL(LimRa, LimiteRatosSala),
    SegundosSemMovimento = IFNULL(SegSem,
SegundosSemMovimento),
    TemperaturaIdeal = IFNULL(Temp,
TemperaturaIdeal),
    VariacaoTemperaturaMaxima      = IFNULL(var,
VariacaoTemperaturaMaxima),
    LimiteAmarelo = IFNULL(LimA, LimiteAmarelo),
    LimiteVermelho = IFNULL(LimV, LimiteVermelho),
    SalaQuaseCheia = IFNULL(salaQ, SalaQuaseCheia),
    SegundosSpam = IFNULL(SegSpam, SegundosSpam)
WHERE IdExperiencia = idExp;
ELSE
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Impossível editar experiência, a
mesma está a decorrer.';
END IF;
ELSE
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Impossível editar experiência, a
mesma não existe.';
END IF;
END$$
DELIMITER;

```

6. Nome SP: CriarOdor

Procedimento que ao receber o ID de uma experiência associa um odor à sala especificada.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`CriarOdor`(IN `IdExp` INT(11), IN `SalaExp` INT(11), IN
`CodigoOdorExp` VARCHAR(5))
BEGIN
    SET @nome:= REPLACE(session_user(), '@localhost', '');
    IF ((SELECT COUNT(*) FROM experiencia WHERE IdExperiencia
= IdExp AND Investigador = @nome) > 0) THEN
    IF ((SELECT COUNT(*) FROM odoresexperiencia WHERE
IdExperiencia = IdExp AND SalaExp=odoresexperiencia.Sala)
= 0) THEN
    INSERT INTO odoresexperiencia(Sala, IDExperiencia,
CodigoOdor)
VALUES(SalaExp, IdExp, CodigoOdorExp);
    ELSE

```

```

SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'A sala já tem um odor associado.';
END IF;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Condições inválidas.';
END IF;
END$$
DELIMITER;

```

7. Nome SP: CriarSubstancia

Procedimento que dado um ID de experiência adiciona o número de ratos injetados com a substância dada. Caso a dupla substância e ID já exista na tabela o número de ratos dado é somado ao já existente.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`CriarSubstancia`(IN `IdExp` INT(11), IN `NumeroRatosExp`
INT(11), IN `CodigoSubstanciaExp` VARCHAR(5))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF ((SELECT COUNT(*) FROM experiencia WHERE IdExperiencia
= IdExp AND Investigador = @nome) > 0) THEN
IF ((SELECT count(*) FROM substanciaexperiencia WHERE
IdExperiencia = IdExp AND
CodigoSubstanciaExp=substanciaexperiencia.CodigoSubstanci
a)=0) THEN
INSERT INTO substanciaexperiencia(NumeroRatos,
IDExperiencia, CodigoSubstancia)
VALUES(NumeroRatosExp, IdExp, CodigoSubstanciaExp);
ELSE
UPDATE substanciaexperiencia SET NumeroRatos =
NumeroRatos + NumeroRatosExp WHERE IdExperiencia = IdExp
AND CodigoSubstancia = CodigoSubstanciaExp;
END IF;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Condições inválidas.';
END IF;
END$$
DELIMITER;

```

8. Nome SP: ComecarExperiencia

Procedimento que, dado um ID de uma experiência, começa a mesma. Caso uma experiência já esteja a decorrer, não permite o começo de outra.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`ComecarExperiencia`(IN `id_experiencia` INT)
BEGIN
DECLARE ratos INT;
IF ((SELECT count(*) from experiencia WHERE ativa =
0x01)=0) THEN
UPDATE experiencia SET DataHora = CURRENT_TIMESTAMP(),
ativa = 0x01 WHERE IDExperiencia = id_experiencia;
SELECT NumeroRatos INTO ratos FROM experiencia WHERE
ativa = 0x01;
INSERT INTO medicoessalas (IDExperiencia,
NumeroRatosFinal, Sala) VALUES (id_experiencia,ratos,1);
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Não é possível começar uma
experiência, já há uma a decorrer.';
END IF;
END$$
DELIMITER;

```

9. Nome SP: TerminarExperiencia

Procedimento que termina uma experiência dando o ID referente à mesma.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`TerminarExperiencia`(IN `id_experiencia` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF ((SELECT COUNT(*) FROM experiencia WHERE IDExperiencia
= id_experiencia AND ativa = 0x01 AND Investigador =
@nome)>0) THEN
UPDATE experiencia SET ativa = 0x00 WHERE IDExperiencia =
id_experiencia;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Impossível terminar a experiência.';
END IF;
END$$
DELIMITER;

```

10. RemoverExperienciaInv

Procedimento que permite ao Investigador remover uma das suas experiências inativas.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`RemoverExperienciaInv`(IN `id_experiencia` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF ((SELECT COUNT(*) FROM experiencia WHERE IDExperiencia
= id_experiencia AND experiencia.Investigador = @nome AND
ativa = 0x00) > 0 ) THEN
DELETE FROM experiencia WHERE IDExperiencia =
id_experiencia;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Impossível apagar experiência';
END IF;
END$$
DELIMITER;

```

11. Nome SP: RemoverExperienciaAdm

Procedimento que permite ao Administrador remover uma experiência inativa.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`RemoverExperienciaAdm`(IN `id_experiencia` INT(11))
BEGIN
IF ((SELECT COUNT(*) FROM experiencia WHERE IdExperiencia
= id_experiencia AND ativa = 0x00) > 0) THEN
DELETE FROM experiencia WHERE IDExperiencia =
id_experiencia;
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Impossível apagar experiência, a
mesma está a decorrer, termine-a primeiro.';
END IF;
END$$
DELIMITER;

```

12. Nome SP: RemoverUtilizador

Procedimento que permite ao Administrador remover Investigadores ou Migradores.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`RemoverUtilizador`(IN `email` VARCHAR(50))
BEGIN
IF (((SELECT TipoUtilizador FROM utilizador

```

```

WHERE EmailUtilizador = email) = 'Inv') OR ((SELECT
TipoUtilizador FROM utilizador WHERE EmailUtilizador =
email) = 'Mig')) THEN
SET @query = CONCAT("DROP USER '", email, "';");
PREPARE stmt FROM @query;
EXECUTE stmt;
DELETE FROM utilizador WHERE emailUtilizador = email;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Nao tem permissoes para remover este
utilizador';
END IF;
END$$
DELIMITER;

```

13. Nome SP: EditarUtilizador

Procedimento que permite ao Investigador alterar os seus dados.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`EditarUtilizador`(IN `Email` VARCHAR(50), IN `new_Email`
VARCHAR(50), IN `new_Nome` VARCHAR(100), IN
`new_Telefone` INT(12), IN `new_Pass` VARCHAR(50))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF ((SELECT count(*) FROM utilizador WHERE @nome =
EmailUtilizador) > 0) THEN
    IF new_Pass IS NOT NULL THEN
        SET @NewPass = CONCAT('SET PASSWORD FOR "',
session_user() , '" = PASSWORD(', new_Pass, ');');
        PREPARE statement FROM @NewPass;
        EXECUTE statement;
    END IF;
    IF new_Email IS NOT NULL THEN
        SET @NewMail = CONCAT('RENAME USER"',
session_user() , '" TO', new_Email, ';');
        PREPARE statement FROM @NewMail;
        EXECUTE statement;
    END IF;
    UPDATE utilizador SET
        EmailUtilizador = IFNULL(new_Email,
EmailUtilizador),
        NomeUtilizador = IFNULL(new_Nome,
NomeUtilizador),
        TelefoneUtilizador = IFNULL(new_Telefone,
TelefoneUtilizador)
        WHERE EmailUtilizador = @nome;
ELSE

```



```

        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Condições inválidas.';
END IF;
END$$
DELIMITER;

```

14. Nome SP: AlterarSegundosExp

Procedimento que é chamado quando o Investigador pretende alterar o campo "SegundosSemMovimento", para prolongar a experiência.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`AlterarSegundosExp`(IN `IDExp` INT, IN `SomaSegs` INT)
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF ((SELECT count(*) FROM experiencia WHERE IdExperiencia
= idExp AND Investigador = @nome) > 0) THEN
    UPDATE experiencia SET SegundosSemMovimento =
SegundosSemMovimento + SomaSegs WHERE IdExperiencia =
idExp;
ELSE
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Impossível editar
SegundosSemMovimento, não tem permissões para tal.';
END IF;
END$$
DELIMITER;

```

15. Nome SP: ObterExperienciasUtilizador

Procedimento que, chamado pelo Investigador, retorna todas as suas experiências.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`ObterExperienciasUtilizador`()
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF ((SELECT count(*) from experiencia WHERE
experiencia.Investigador = @nome)>0) THEN
SELECT * from experiencia WHERE experiencia.Investigador
= @nome;
ELSE
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Você não tem
experiências.';
END IF;
END$$

```

```
DELIMITER;
```

16. NomeSP: ObterOdoresExperiencia

Procedimento que, chamado pelo Investigador, retorna o(s) Odor(es) da experiência pedida.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`ObterOdoresExperiencia`(IN `IDExp` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF (@nome = (SELECT Investigador FROM experiencia
WHERE IDExperiencia = IDExp))THEN
SELECT CodigoOdor,Sala FROM odoresexperiencia
WHERE IDExperiencia = IDExp;
ELSE
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Experiência não existe';
END IF;
END$$
DELIMITER;
```

17. Nome SP: ObterResultadoFinal

Procedimento que, chamado pelo Investigador, retorna o Resultado Final de uma experiência, caso a mesma tenha terminado.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`ObterResultadoFinal`(IN `IDExp` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF (@nome = (SELECT Investigador FROM experiencia
WHERE IDExperiencia = IDExp))THEN
IF EXISTS (SELECT * FROM experiencia WHERE
IDExperiencia = IDExp AND Investigador = @nome AND ativa
= 0x00 AND DataHora != NULL) THEN
SELECT NumeroRatosFinal,Sala FROM
medicoessalas WHERE IDExperiencia = IDExp;
ELSE
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'A
experiência está a decorrer.';
END IF;
ELSE
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'A
experiência não existe';
END IF;
END
```

```

END$$
DELIMITER;

18. Nome SP: ObterSubstanciasExperiencia

Procedimento que, chamado pelo Investigador, retorna a(s)
Substância(s) da experiência pedida.

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`ObterSubstanciasExperiencia`(IN `IDExp` INT(11))
BEGIN
SET @nome:= REPLACE(session_user(), '@localhost', '');
IF (@nome = (SELECT Investigador FROM experiencia
WHERE IDExperiencia = IDExp)) THEN
SELECT CodigoSubstancia, NumeroRatos FROM
substanciaexperiencia WHERE IDExperiencia = IDExp;
ELSE
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
'Experiência não existe';
END IF;
END$$
DELIMITER;

```

4.5 Utilizadores Base de Dados Mysql

Best Off

Tabela	Tipo de Utilizador		
	Administrador	Investigador	Migrador
Alerta	L	L	L, I
Experiencia	L, I, D, U	L	L
MedicoesPassagens	L	L	L, I
MedicoesSalas	L	L	L, I
MedicoesTemperatura	L	L	L, I
OdoresExperiencia	L	L	-
SubstanciaExperiencia	L	L	-
Utilizador	L, I, D, U	L	L
Stored Proc.			
CriarUtilizador	X	-	-
RemoverUtilizador	X	-	-
EditarUtilizador	X	-	-
CriarExperienciaInv	-	X	-
CriarExperienciaAdm	X	-	-

RemoverExperienciaInv	-	X	-
RemoverExperienciaAdm	X	-	-
EditarExperienciaInv	-	X	-
EditarExperienciaAdm	X	-	-
ObterExperienciasUtilizador	-	X	-
ObterResultadoFinal	-	X	-
ObterOdoresExperiencia	-	X	-
ObterSubstanciasExperiencia	-	X	-
CriarOdor	-	X	-
CriarSubstancia	-	X	-
ComecarExperiencia	X	X	-
TerminarExperiencia	X	X	-
AlterarSegundosExp	-	X	-

Justificação das alterações

Em relação à tabela “experiencia”, considerámos que seria necessário a visualização dos dados por parte dos utilizadores do tipo “Migrador”, de modo a ser possível recolher dados para a especificação em java.

Em relação ao procedimento anteriormente designado como “LoginUtilizador”, considerámos que este não seria aplicado diretamente na base de dados, mas aquando do início de sessão dos utilizadores para preenchimento dos formulários php/html, bem como no início de sessão dos utilizadores para visualização dos dados na aplicação android.

Em relação ao procedimento “ComecarExperiencia”, decidimos atribuir a execução do mesmo também aos utilizadores do tipo “Administrador”, em caso de necessidade ou a pedido do investigador.

Como complemento, decidimos adicionar o procedimento “AlterarSegundosExp”, que permite ao investigador adicionar segundos, no final da experiência. Ou seja, após o expectável término da experiência por um dos motivos definido, o utilizador poderá prolongar a mesma, dando como argumento o número de segundos a acrescentar até nova verificação das possíveis causas do término desta.

Os procedimentos “CriarExperiencia”, “RemoverExperiencia” e “EditarExperiencia” podem ser executados tanto pelos utilizadores do tipo “Investigador” como pelos do tipo “Administrador”. Nestes casos, foram criados dois procedimentos, um para cada tipo de utilizador, sendo que no caso de ser um “Investigador” este verifica se é este o investigador associado à experiência e, apenas nesse caso, deixa o utilizador executar o procedimento.

4.6 Procedimentos Manutenção Aplicação (best off)

Best Off

Nome Procedimento / Função	Parâmetros Entrada	Valores Retorno (função)	=, Novo, Alterado

4.6.1 Stored Procedures implementados (best off)

```
1. Nome SP: CriarUtilizador
// Breve Descrição
Código

2. Nome SP: EditarExperiência
// Breve Descrição
Código
```

Justificação das alterações

Texto justificativo

4.7 Eventos de suporte à aplicação (best off)

Best Off

Nome Evento	Local Execução (MySQL/Windows)	Muito breve descrição
-	-	-

4.8 PrintScreen dos formulários HTML (best off)

Best Off

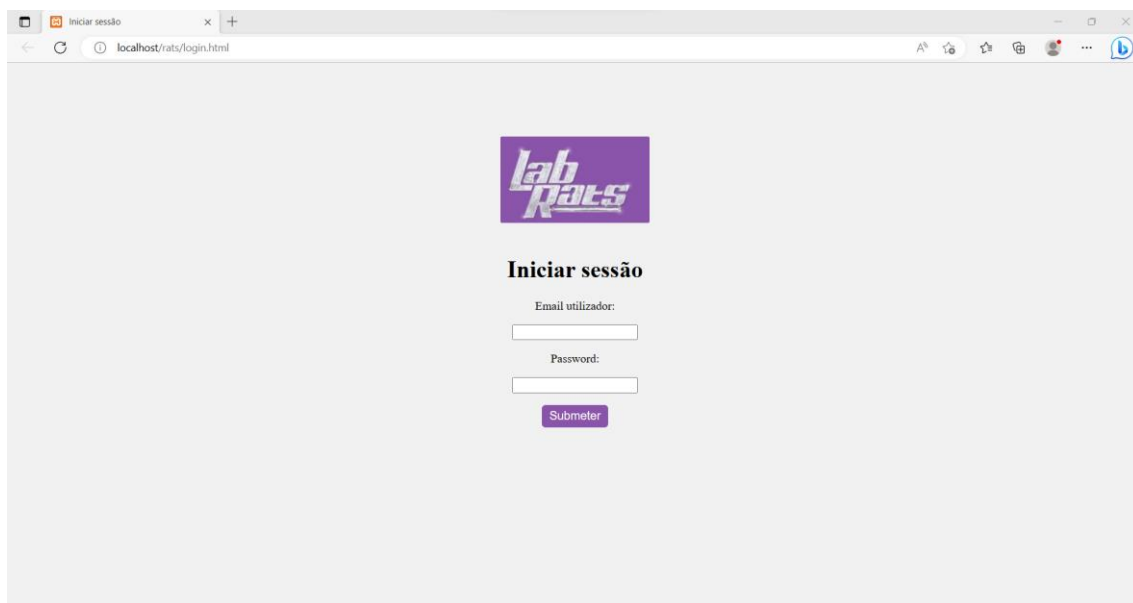


Figura 16 – Ecrã para iniciar sessão

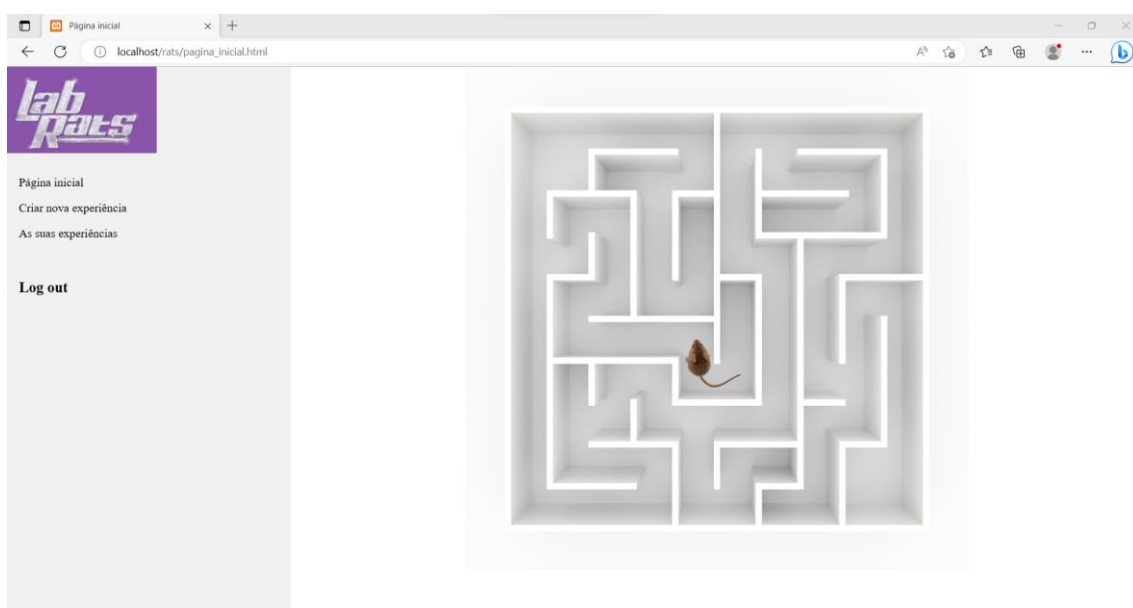


Figura 17 – Ecrã de Página Inicial

Labrats - Criar nova experiência

Nome Experiência:

Descrição Experiência:

Número de ratos Experiência:

Límite de ratos por sala Experiência:

Límite de segundos sem movimento Experiência:

Temperatura ideal Experiência:

Variação de temperatura máxima Experiência:

Aviso amarelo Experiência:

Aviso vermelho Experiência:

Aviso de sala quase cheia Experiência:

Tempo mínimo entre avisos (em segundos) Experiência:

Figura 18 – Ecrã para Criar Experiência

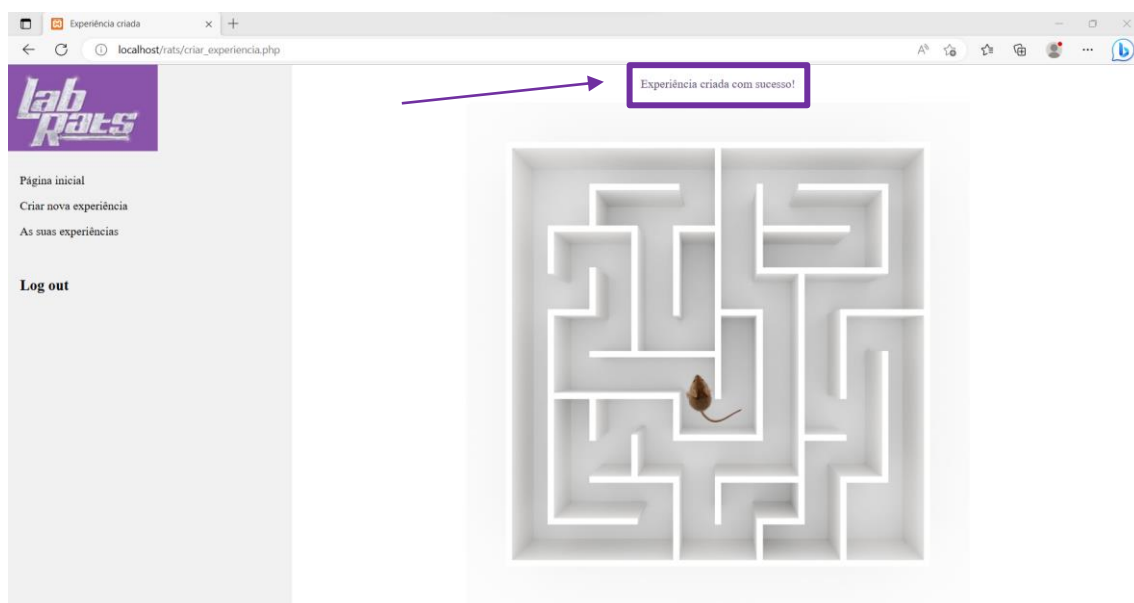


Figura 19 – Ecrã de experiência criada

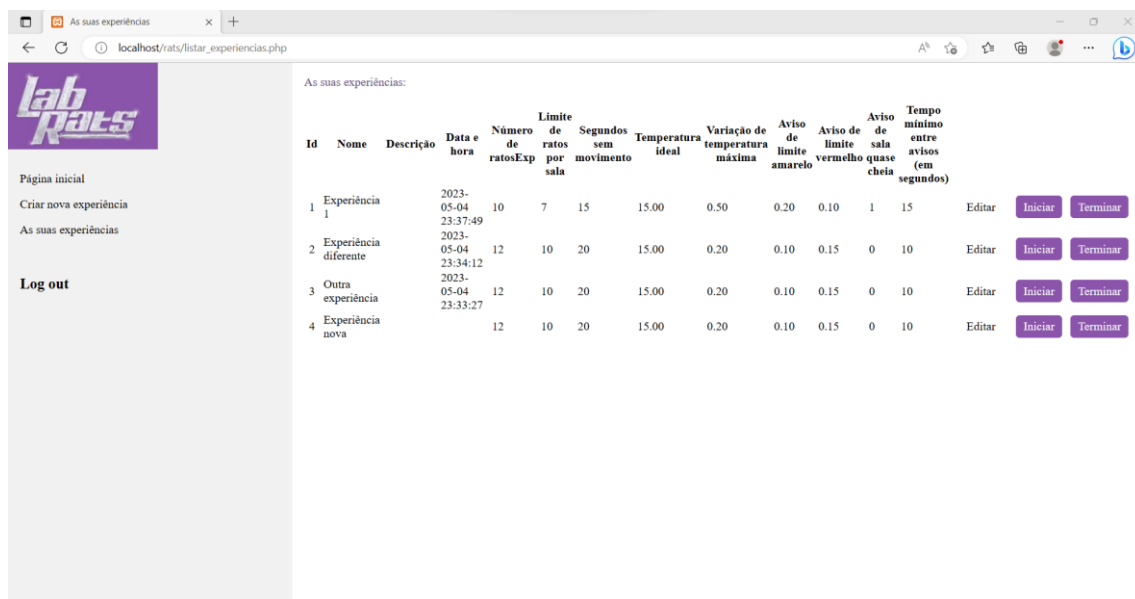


Figura 20 - Ecrã de listagem das experiências do utilizador

Labrats - Alterar experiência 4

Experiência selecionada: 4

Nome Experiência:

Descrição Experiência:

Número de ratos Experiência:

Límite de ratos por sala Experiência:

Límite de segundos sem movimento Experiência:

Temperatura ideal Experiência:

Variação de temperatura máxima Experiência:

Aviso amarelo Experiência:

Aviso vermelho Experiência:

Aviso de sala quase cheia Experiência:

Tempo mínimo entre avisos (em segundos) Experiência:

Figura 21 - Ecrã para alterar experiência

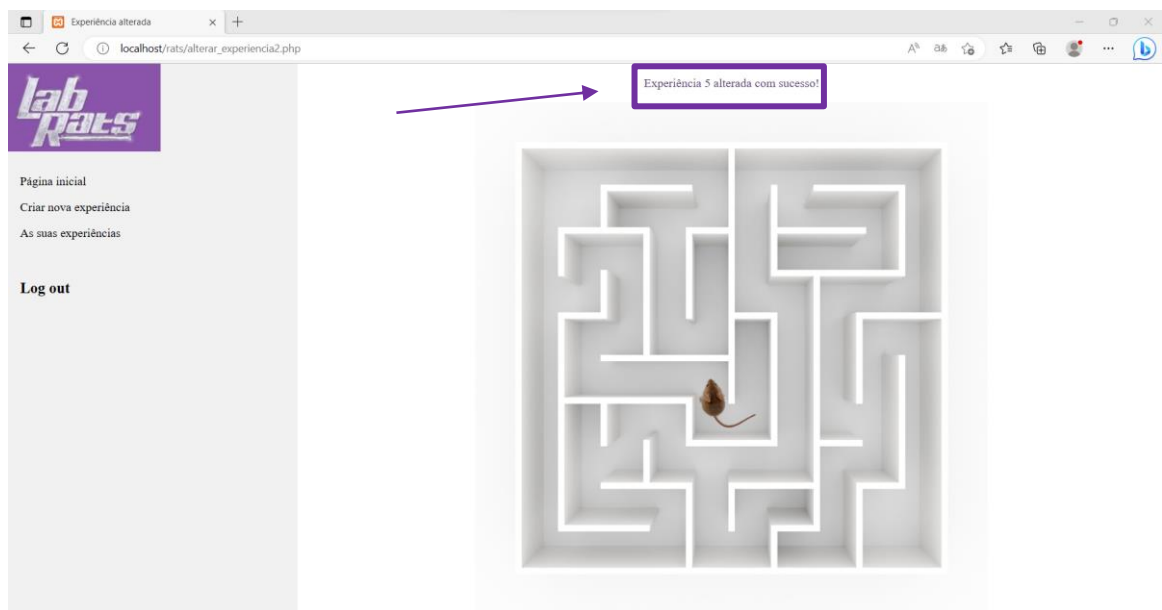


Figura 22 - Ecrã de experiência alterada

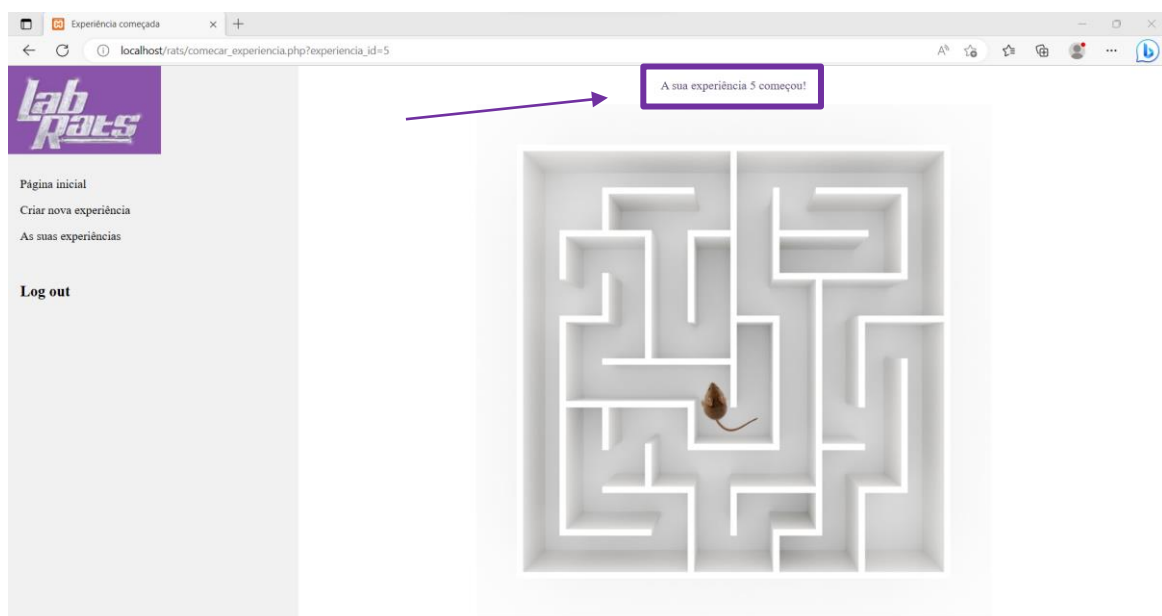


Figura 23 – Ecrã que indica que dada experiência começou

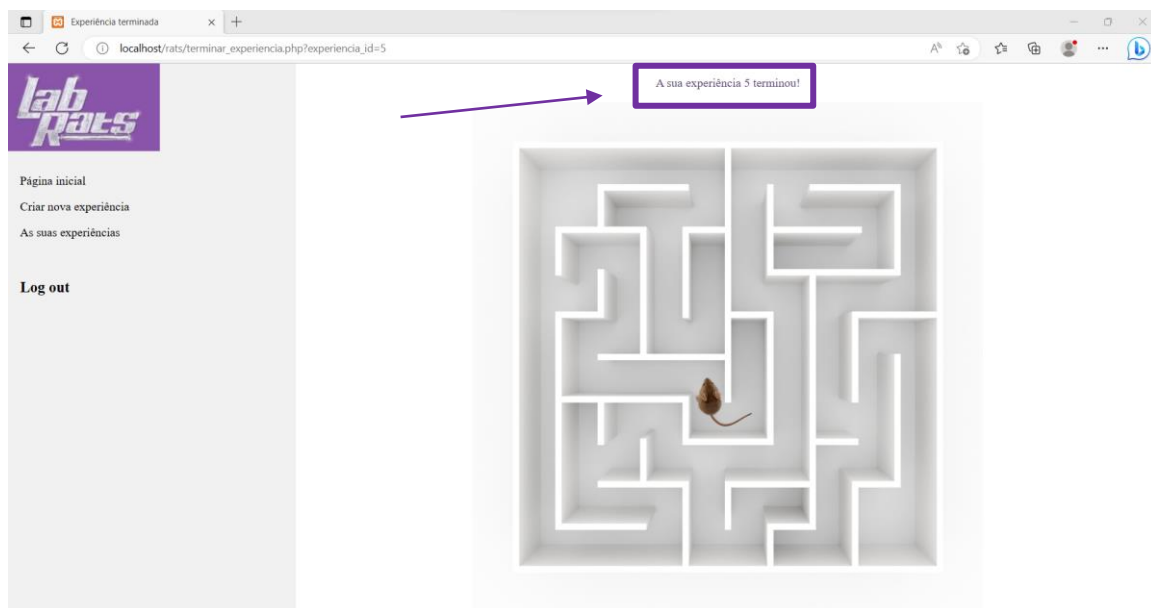


Figura 24 – Ecrã que indica quando uma experiência termina

4.9 PrintScreen do formulários Android (best off)

Best Off

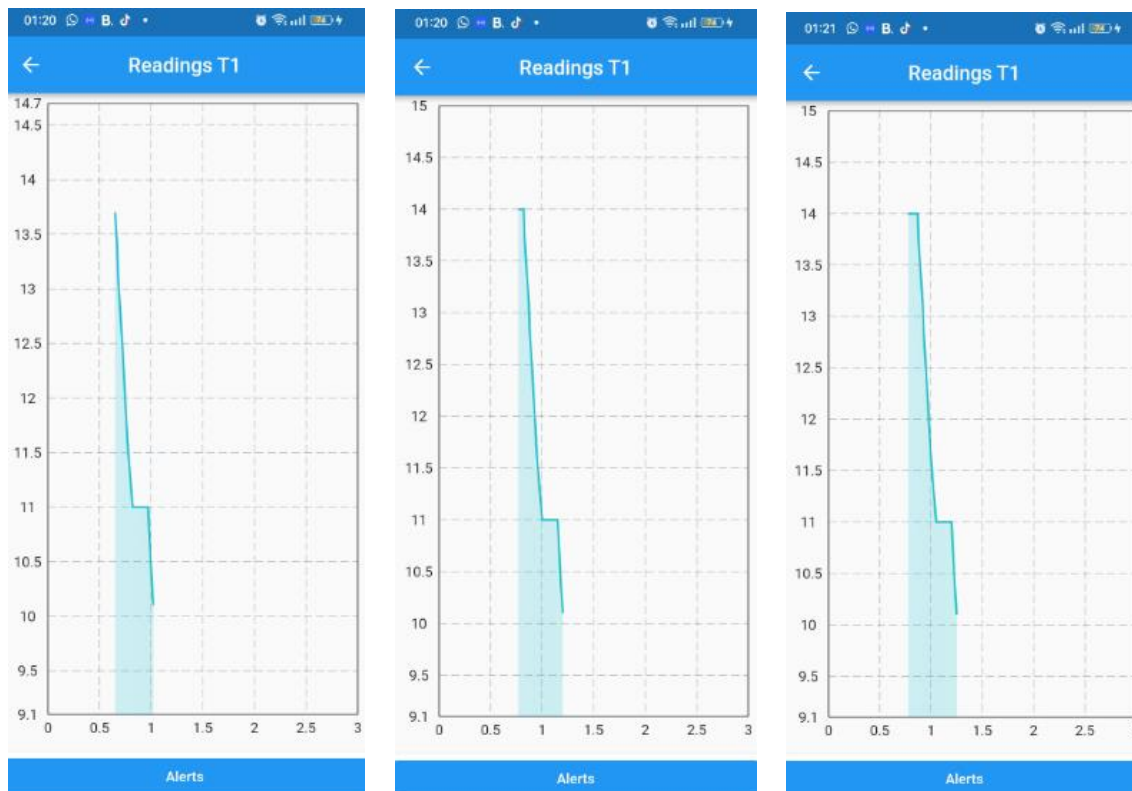


Figura 25 – Gráfico de evolução de temperaturas ao longo do tempo

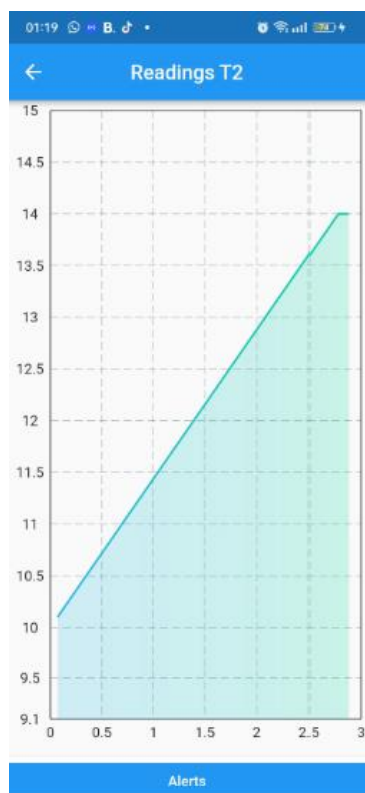


Figura 26 – Gráfico de evolução de temperaturas ao longo do tempo

The screenshot shows a mobile application interface with a blue header bar titled 'Alertas'. Below the header is a table with a single column labeled 'Mensagem'. The table contains ten rows of alert messages. The first eight rows are 'Alerta Amarelo: sensor 1 atingiu a temperatura 5.0.', and the next two rows are 'Alerta Amarelo: sensor 2 atingiu a temperatura 11.599'. At the bottom of the screen is a blue bar with three icons: a thermometer, a sensor, and a mouse. Below these icons are the labels 'Temp Sensor 1', 'Temp Senso...', and 'Mouses/...'.

Mensagem
Alerta Amarelo: sensor 1 atingiu a temperatura 5.0.
Alerta Amarelo: sensor 1 atingiu a temperatura 5.0.
Alerta Amarelo: sensor 1 atingiu a temperatura 5.0.
Alerta Amarelo: sensor 2 atingiu a temperatura 11.599
Alerta Amarelo: sensor 1 atingiu a temperatura 5.0.
Alerta Amarelo: sensor 1 atingiu a temperatura 5.0.
Alerta Amarelo: sensor 1 atingiu a temperatura 5.3.
Alerta Amarelo: sensor 1 atingiu a temperatura 5.6.
Alerta Amarelo: sensor 1 atingiu a temperatura 5.8999
Alerta Amarelo: sensor 2 atingiu a temperatura 11.599
Alerta Amarelo: sensor 1 atingiu a temperatura 6.4999
Alerta Amarelo: sensor 2 atingiu a temperatura 9.1999

Figura 27 - Ecrã com alertas de temperatura