Daily Huddle
Casos de Teste de Software/Sistema

Histórico da Revisão

Data	Versão	Descrição	Autor
<11/01/23>	<1.0>	Especificação dos casos de teste	Grupos Alfa, Beta, Gama, Delta e Epsilon

1. Identificação

[Identificação do software/sistema a ser avaliado, versões dos sistemas e de banco de dados]

1.1

1.2 Referências

Título	Versão	Data	Onde pode ser obtido
Plano Geral de Testes			
Plano de Testes de Software/ Sistema			

[Incluir a identificação de todos os documentos referenciados necessários para a execução dos casos de teste. Para cada documento indicar o título, versão (caso tenha), data e onde o documento pode ser encontrado para consulta. Indicar nesse momento a existência do Plano Geral de Testes que contém as definições gerais dos testes a serem realizados]

2. Casos de Teste

2.1 Caso de Teste : Broker – Testar estabelecimento da conexão entre sistema e broker

2.1.1 *Descrição*

O caso de teste será realizado conectando o Broker ao sistema e verificando se a conexão é estabelecida corretamente. Para isso, serão instanciadas as classes *Broker* e *Sistema* e a função EstabelecerConexao(Broker), da classe *Sistema*, será executada, tendo como argumento a instância criada da classe Broker.

2.1.2 *Pré-condições para o caso*

Broker e sistema devem estar funcionais.

2.1.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1 - Sucesso**	**Cenário 2 - Falha na conexão**
Valor 1	Instância da classe *Broker*, com atributo de status de conexão com valor *falso*, indicando conexão não estabelecida.	Instância da classe *Broker*, com atributo de status de conexão com valor *falso*, indicando conexão não estabelecida.
Valor de saída (Resultado Esperado)	Instância da classe Broker com atributo de status de conexão com valor *verdadeiro*, indicando conexão estabelecida.	Em caso de falha na conexão, a classe Broker com atributo de status de conexão será retornada, com valor *falso*, indicando conexão estabelecida. Uma mensagem de erro deve ser enviada aos usuários do sistema, informando que a conexão com o broker não pode ser estabelecida e que o dashboard não poderá atualizar seus dados e permanecerá exibindo os últimos dados obtidos até que a conexão seja criada. O sistema deve criar entrada no log de erros para registrar o erro de conexão.

Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente **(se falha)**		
Nº Log **(se falha)**		

2.2 Caso de Teste : Administração - Enviar dados

2.2.1 *Descrição*

O caso de teste será realizado via Broker e Sistema,

2.2.2 *Pré-condições para o caso*

Instância das classes Broker e Sistema devem estar iniciadas. O atributo “statusConexao” da instância do Broker precisa estar setado como True.

2.2.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Dados Carregados (sucesso)**	**Falha na Conexão**
Valor 1	função: *Sistema.RequisitarDados(Broker):* Tendo como parâmetro a instância da classe Broker	função: *Sistema.RequisitarDados(Broker):* Tendo como parâmetro a instância da classe Broker
Valor de saída (Resultado Esperado)	Status do sistema alterado para “Dados Carregados”, sinalizando assim o correto carregamento dos dados	Status do sistema alterado para “Falha na Conexão”, sinalizando assim um possível problema com método: *Sistema.EstabelecerConexao(Broker)*
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente **(se falha)**		
Nº Log **(se falha)**		

2.3 Caso de Teste : Administração – Persistir os dados recebidos em um banco de dados

2.3.1 *Descrição*

O caso de teste será realizado com a conexão da classe *Sistema* ao banco de dados através do método PersistirDados(). Neste caso de teste, são instanciadas as classes *Sistema* e *Dashboard*.

2.3.2 *Pré-condições para o caso*

A classe *Dashboard* deve estar instanciada e possuir os atributos equipamentos ou materiais inicializados.

2.3.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1 - Sucesso**	**Cenário 2 - Falha na conexão**
Valor 1	Classe *sistema* instanciada e a função *Sistema.PersistirDados()* recebe os atributos da classe *Dashboard* inicializados.	Classe *sistema* instanciada e a função *Sistema.PersistirDados()* recebe os atributos da classe *Dashboard* inicializados.
Valor de saída (Resultado Esperado)	Instância da classe *Sistema* é capaz de inserir atributos da classe *DashBoard* no próprio atributo de classe bancoDados, e o array é modificado após a execução da função PersistirDados().	Caso o atributo bancoDados não seja modificado ao final da execução do método PersistirDados(). Neste cenário, uma mensagem de erro deve notificar a administração de que os dados enviados pela classe *Dashboard* não foram salvos pela classe *Sistema*.
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente (se falha)		
Nº Log (se falha)		

2.4 Caso de Teste : Dashboard - Autenticar usuário

2.4.1 *Descrição*

O caso de teste será feito tendo em vista o teste da função Receber_Login(), para isso será necessário um dashboard, para visualizar se o mesmo faz o correto registro dos dados

2.4.2 *Pré-condições para o caso*

Necessário um Dashboard funcional.

2.4.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor 1	Instância da classe dashboard, a qual recebe dados do usuário através da função Receber_Login(Dados).	
Valor de saída (Resultado Esperado)	Registro textual (Json) do login e senha exatamente da forma que foram digitados.	
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente (se falha)		
Nº Log (se falha)		

Caso de Teste : Dashboard - Autenticar usuário

2.4.4 Descrição

O caso de teste será feito tendo em vista o teste do método Envia_dados(), para isso será necessário um dashboard, para visualizar se o mesmo fez o correto envio dos dados

2.4.5 Pré-condições para o caso

Necessário um Dashboard e Sistema funcionais.

2.4.6 Conjunto de valores

2.4.7

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	Instância da classe dashboard, a qual recebe dados do usuário através do método Receber_Login(Dados), em seguida os mesmos são enviados ao sistema através do método Envia Dados()
Valor de saída (Resultado Esperado)	Sistema recebendo hash condizente com o que foi digitado no dashboard
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

Caso de Teste : Dashboard - Autenticar usuário

2.4.8 Descrição

O caso de teste será feito tendo em vista o teste do método Logar_Usuario(), para isso será necessário que o sistema esteja operacional

2.4.9 Pré-condições para o caso

Sistema funcional.

2.4.10 Conjunto de valores

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor 1	O método de autenticação Logar_Usuario(dados) deverá receber um hash correspondendo a dados de login válidos.	O método de autenticação Logar_Usuario(dados) deverá receber um hash correspondendo a dados de login inválidos.
Valor de saída (Resultado Esperado)	Retorno de autenticação aprovada para determinado usuário	Retorno de autenticação negada para determinado usuário
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha	Sucesso	Falha
Nº Ambiente (se falha)		
Nº Log (se falha)		

2.5 Caso de Teste : Broker –

2.5.1 *Descrição*

2.5.2 *Pré-condições para o caso*

2.5.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.6 Caso de Teste : Administração – Atualizar Estado dos Equipamentos

2.6.1 *Descrição*

O caso de teste será realizado via o Dashboard atualizando, em sua interface, o estado dos equipamentos. Para isso, será instanciada a classe *Dashboard*, que possui um atributo *Equipamentos* do tipo array. A função AtualizarEstadoEquipamentos(código), da classe *Dashboard*, será executada, tendo como argumento o código de um equipamento.

2.6.2 *Pré-condições para o caso*

A rede da aplicação deve estar funcional.

2.6.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor 1	A classe Dashboard executa a função *AtualizarEstadoEquipamentos(código),* com o atributo código estando	A classe Dashboard executa a função *AtualizarEstadoEquipamentos(código),* com o atributo código não estando

	presente em um objeto do array *Equipamentos*.	presente em um objeto do array *Equipamentos*.
Valor de saída (Resultado Esperado)	Função *AtualizarEstadoEquipamentos(código)* retorna e exibe o estado atual do equipamento.	Função *AtualizarEstadoEquipamentos(código)* retorna um erro e exibe uma mensagem informando que não foi possível atualizar o estado do equipamento.
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		

2.7 Caso de Teste : Broker – Solicitar Dados de Temperatura e Umidade

2.7.1 *Descrição*

O caso de teste será feito com o Sistema enviando a mensagem de solicitação de dados de temperatura e umidade ao dispositivo ESP32 para verificar se o status do sistema é atualizado da maneira esperada. As classes *Broker* e *Sistema* serão instanciadas e a função RequisitarDados(Broker), da classe *Sistema,* será executada, tendo como argumento a instância criada da classe Broker.

2.7.2 *Pré-condições para o caso*

Conexão entre sistema e broker devem estar estabelecidas

2.7.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor 1	A classe Sistema executa a função *RequisitarDados(Broker)* com o atributo código estando presente em um objeto do array *Equipamentos*.	A classe Sistema executa a função *RequisitarDados(Broker)* com o atributo código não estando presente em um objeto do array *Equipamentos*.
Valor de saída (Resultado Esperado)	A função *RequisitarDados(Broker)* retorna e imprime o status do sistema de “Dados Recebidos” .	A função *RequisitarDados(Broker)* retorna um erro e imprime o status do sistema de “Perda de Conexão ” .
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		

2.8 Caso de Teste : Administração – Publicar parâmetros de limites de temperatura e umidade

2.8.1 *Descrição*

O caso de teste é realizado com a conexão entre o sistema e o dashboard. Neste caso de uso, são instanciadas as classes *Sistema* e *Dashboard*. Os métodos publicarParamUmidade() e publicarParamTemperatura() da classe *Sistema* são testados.

2.8.2 *Pré-condições para o caso*

O atributo bancoDados da classe *Sistema* deve ter sido previamente inicializado com dados da classe *Dashboard*.

2.8.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1 - Sucesso**	**Cenário 2 - Falha na conexão**
Valor 1	Não se aplica.	Não se aplica.
Valor de saída (Resultado Esperado)	Classe *Sistema* foi capaz de imprimir os parâmetros de temperatura e umidade contidos no atributo bancoDados através dos métodos publicarParamTemperatura() e publicarParamUmidade(), respectivamente .	Classe *Sistema* não foi capaz de se comunicar com a classe *Dashboard* e os dados não puderam ser exbidos. Uma mensagem de erro é emitida para a administração.
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente (se falha)		
Nº Log (se falha)		

2.9 Caso de Teste : Broker –

2.9.1 *Descrição*

2.9.2 *Pré-condições para o caso*

2.9.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.10 Caso de Teste : Broker –

2.10.1 *Descrição*

2.10.2 *Pré-condições para o caso*

2.10.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.11 Caso de Teste : Dashboard - Testar acesso ao site do Dashboard.

2.11.1 *Descrição*

Um dos membros da equipe de testes deve se logar no sistema com uma conta de testes simulando o acesso de um profissional de saúde e tentará acessar a interface do Dashboard.

2.11.2 *Pré-condições para o caso*

O dashboard do huddle deve ser disponibilizado em um servidor de testes, para que sua interface possa ser acessada com requisições HTTP por um navegador.

2.11.3 *Conjunto de valores*

Não se aplica.

2.12 Caso de Teste : Dashboard – Atualizar dados de Equipamentos

2.12.1 *Descrição*

O caso de teste será realizado na interface do Dashboard, onde o ator “Técnico de Equipamentos” pode atualizar os dados dos equipamentos.

2.12.2 *Pré-condições para o caso*

O usuário deve estar logado, e identificado como “Técnico de Equipamentos” no banco de dados do Sistema. Conexão do Sistema com o Equipamento deve estar estabelecida.

2.12.3 *Conjunto de valores*

Conjunto de Valores			
Cenários	**Dados Atualizados**	**Novo Equipamento**	**Conexão Perdida**
Interação inicial	Técnico Seleciona equipamento na interface do dashboard e insere dados atualizados	Técnico Cria novo equipamento na interface do Dashboard e insere seus dados	Técnico Seleciona equipamento na interface do dashboard e insere dados atualizados
Função acionada	função: *Dashboard.AtualizarDadosEquipamentos(codigo):* Tendo como parâmetro o código referente ao equipamento que deve ser atualizado	função: *Dashboard.AtualizarDadosEquipamentos(codigo):* Tendo como parâmetro o código referente ao novo equipamento que deve ser atualizado	função: *Dashboard.AtualizarDadosEquipamentos(codigo):* Tendo como parâmetro o código referente ao equipamento que deve ser atualizado
Valor de saída (Resultado Esperado)	Status do sistema alterado para “Dados Atualizados”, sinalizando assim sucesso na atualização	Status do sistema alterado para “Dados Atualizados”, sinalizando assim sucesso na criação e atualização	Status do sistema alterado para “Conexão perdida”, sinalizando assim um possivel problema com o método: *Sistema.EstabelecerConexao(Broker)*
Resultado Obtido (se diferente do esperado)			
Sucesso /Falha			
Nº Ambiente			

(se falha)			
Nº Log (se falha)			

2.13 Caso de Teste : Dashboard – Atualizar Dados de Materiais

2.13.1 *Descrição*

O caso de teste será realizado com a conexão de um Farmacêutico ou Técnico de Esterilização. Para isso, serão instanciadas as classes *Farmaceutico, TecnicoEsterilizacao*, *Dashboard* e *Sistema*. O método AtualizarEstadoMateriais(codigo), da classe *Dashboard,* será executado e receberá como parâmetro o código do material a ser atualizado.

2.13.2 *Pré-condições para o caso*

Devem estar instanciadas as classes *Farmaceutico, TecnicoEsterilizacao* e *Dashboard*. *Farmaceutico* e *TecnicoEsterilizacao* precisam estar validados e inserir um codigo de material como input.

2.13.3 *Conjunto de valores*

Conjunto de Valores			
Cenários	**Dados Atualizados**	**Novo Equipamento**	**Conexão Perdida**
Interação inicial	Técnico de Esterilização ou Farmacêutico selecionam material e inserem os novos parâmetros de segurança	Técnico de Esterilização ou Farmacêutico criam novo material na interface do Dashboard e insere seus dados	Técnico de Esterilização ou Farmacêutico selecionam material e inserem os novos parâmetros de segurança
Função acionada	função: *Dashboard.AtualizarEstadoMateriais(codigo):* Tendo como parâmetro o código referente ao material que deve ser atualizado	função: *Dashboard.AtualizarEstadoMateriais(codigo):* Tendo como parâmetro o código referente ao novo material que deve ser atualizado	função: *Dashboard.AtualizarEstadomateriais(codigo):* Tendo como parâmetro o código referente ao equipamento que deve ser atualizado
Valor de saída (Resultado Esperado)	Status do sistema alterado para “Dados Atualizados”, sinalizando assim sucesso na atualização	Status do sistema alterado para “Dados Atualizados”, sinalizando assim sucesso na criação e atualização	Status do sistema alterado para “Conexão perdida”, sinalizando assim um possível problema com o método: *Sistema.EstabelecerConexao(Broker)*
Resultado Obtido (se diferente			

do esperado)			
Sucesso/Falha			
Nº Ambiente (se falha)			
Nº Log (se falha)			

2.14 Caso de Teste : Dashboard – Testar logar no sistema

2.14.1 *Descrição*

O caso de teste será realizado preenchendo os campos de login e senha e pressionando o botão entrar e verificando se logou corretamente. Para isso, serão instanciadas as classes *Credencial, Usuário* e *Sistema* e os Preencher_Login(Login, Senha), Validar_Credencial(Login, Senha) e o método Logar_Usuario(Usuario), que serão executados.

2.14.2 *Pré-condições para o caso*

O usuário deve ter sido cadastrado previamente no Sistema.

2.14.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor (Login e Senha)	Profissional do hospital preenche login e senha e clica no botão entrar	Profissional do hospital preenche login e senha e clica no botão entrar
Valor de saída (Resultado Esperado)	Login e Senha existem no sistema e correspondem a mesma credencial. Logar no sistema no perfil correspondente a esta credencial.	Login e Senha não correspondem à nenhuma credencial. Sistema exibe mensagem: “Login ou senha incorretos."
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente (se falha)		
Nº Log (se falha)		

2.15 Caso de Teste :

2.15.1

2.15.2 *Pré-condições para o caso*

2.15.3 *Conjunto de valores*

2.16 Caso de Teste : Dashboard - Testar exibição de dados de equipamentos pelo Dashboard

2.16.1 *Descrição*

Um dos membros da equipe de testes deve se logar no sistema com uma conta de testes simulando o acesso de um profissional de saúde e tentará acessar a interface do Dashboard. Após o login, o usuário deve tentar visualizar dados de um equipamento pelo Dashboard.

2.16.2 *Pré-condições para o caso*

O dashboard do Huddle deve ser disponibilizado em um servidor de testes, para que sua interface possa ser acessada com requisições HTTP por um navegador.

2.16.3 *Conjunto de valores*

Não se aplica.

2.17 Caso de Teste : Equipamentos – Marcar status “para limpeza”

2.17.1 *Descrição*

O caso de teste será realizado pelo Usuário qualificado para verificar o funcionamento da flag “Para limpeza”, presente nos dispositivos IoT no formato de um LED.

2.17.2 *Pré-condições para o caso*

Deve existir objetos das classes Sistema, Broker e Dashboard. Usuário deve estar logado no Sistema e cadastrado como “Profissional de Saúde” ou “Gestor do Setor”.

2.17.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Sucesso**	**Falha**
Valor 1	Profissional de Saúde / Gestor do Setor seleciona o equipamento da interface do Dashboard e redefine seu status “para limpeza”	Profissional de Saúde / Gestor do Setor seleciona o equipamento da interface do Dashboard e redefine seu status “para limpeza”

Valor de saída (Resultado Esperado)	função: *Equipamento.MarcarLimpeza(EtiquetaRFID):* Possui como parâmetro a identificação RFID do equipamento	função: *Equipamento.MarcarLimpeza(EtiquetaRFID):* Possui como parâmetro a identificação RFID do equipamento
Resultado Obtido (se diferente do esperado)	Equipamento tem LED referente à limpeza aceso.	O equipamento permanece inalterado. Apresentando assim, problema na comunicação com o equipamento, ou defeito no funcionamento do LED.
Sucesso/Falha		
Nº Ambiente **(se falha)**		
Nº Log **(se falha)**		

2.18 Caso de Teste : Equipamentos – Remover status “para limpeza”

2.18.1 *Descrição*

O caso de teste será realizado pelo Usuário qualificado para verificar o funcionamento da flag “Para limpeza”, presente nos dispositivos IoT no formato de um LED.

2.18.2 *Pré-condições para o caso*

Deve existir objetos das classes Sistema, Broker e Dashboard. Usuário deve estar logado no Sistema e cadastrado como “Profissional da Equipe de Limpeza”.

2.18.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Sucesso**
Valor 1	Profissional da Equipe de Limpeza seleciona o equipamento na interface do dashboard e remove o seu status “para limpeza”
Valor de saída (Resultado Esperado)	função: *Equipamento.RemoverLimpeza(EtiquetaRFID):* Possui como parâmetro a identificação RFID do equipamento Apaga o LED e executa a função EnviarRelatório(broker) da classe Equipamentos.
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.19 Caso de Teste : Broker –

2.19.1 *Descrição*

2.19.2 *Pré-condições para o caso*

2.19.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.20 Caso de Teste : Broker –

2.20.1 *Descrição*

2.20.2 *Pré-condições para o caso*

2.20.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.21 Caso de Teste : Equipamentos – Autenticar RFID

2.21.1 *Descrição*

O caso de teste será realizado aproximando uma Etiqueta RFID ao Leitor RFID e verificar se o status dos botões será liberado. Para isso, serão instanciadas as classes: Equipamento, Leitor RFID, Etiqueta RFID e Usuário. Além de relacionar um Equipamento a um Leitor RFID e uma Etiqueta RFID a um Usuário. Por fim, o método EmitirSinal(), da classe *Etiqueta RFID*.

2.21.2 *Pré-condições para o caso*

O status da conexão entre o Broker e o Sistema devem estar estabelecidos, o Usuário deve estar conectado a uma etiqueta RFID e o Equipamento conectado ao Leitor RFID

2.21.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor 1	O Usuário deverá aproximar adequadamente uma Etiqueta RFID autenticada do Leitor RFID, disparando o método EmitirSinal()	O Usuário deverá aproximar de forma inadequada ou uma Etiqueta RFID não autenticada do Leitor RFID, disparando o método EmitirSinal()
Valor de saída (Resultado Esperado)	A classe do Equipamento que está relacionada ao Leitor RFID deve ativar os botões de alteração de status	Instância da classe Leitor RFID deve exibir a mensagem para aproximar a etiqueta com uma menor distância, disparando o método ExibirErroLeitura()
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente (se falha)		
Nº Log (se falha)		

2.22 Caso de Teste : Broker – Testar Remover Equipamento do Inventário

2.22.1 *Descrição*

O teste funcional envolve desde a chamada Equipamentos.RemoverEquipamento(EtiquetaRFID) por um funcionário com as devidas permissões até o envio da mensagem de atualização de estado do equipamento pelo dispositivo IoT com destino ao Broker

2.22.2 *Pré-condições para o caso*

Objeto da classe Equipamentos deve estar instanciado e com estado de 'Disponível'. Objeto da classe Broker deve estar instanciado. Funcionário deve estar autenticado via RFID.

2.22.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor RFID	RFID corresponde a um funcionário habilitado a utilizar o equipamento	RFID corresponde a um funcionário não habilitado a utilizar o equipamento
Valor de saída (Resultado Esperado)	LED do dispositivo IoT encontra-se aceso, alternadamente. Estado do equipamento definido como 'Em uso'. A mensagem de atualização de estado do objeto da classe Equipamentos é enviada com destino ao Broker, imediatamente.	LED do dispositivo IoT permanece apagada. Estado do equipamento permanece como 'Disponível'.
Resultado Obtido (se diferente do esperado)		

2.23 Caso de Teste : Broker –

2.23.1 *Descrição*

2.23.2 *Pré-condições para o caso*

2.23.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.24 Caso de Teste : Broker – Testar Devolver Equipamento ao Local de Concentração

2.24.1 *Descrição*

O caso de teste será realizado devolvendo o equipamento ao local de concentração e sinalizando ao sistema que o equipamento foi devolvido. Para isso, serão instanciadas as classes *Broker, RFID, Equipamento, Usuário, PlacaESP32* e *Sistema* e o método Devolver_Equipamento(ID_Equipamento), da classe *Equipamento*, será executado, tendo como argumento o identificador do equipamento.

2.24.2 *Pré-condições para o caso*

O usuário deve estar autenticado via RFID (caso de teste 2.21) e o equipamento deve ter sido retirado do local de concentração (caso de teste 2.22) além das instâncias supracitadas estarem integradas e funcionais.

2.24.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	A instância do objeto usuário, devidamente autenticada via RFID, possuindo um equipamento com estado “em uso”, e com a localização no local de concentração constatado pela placaEsp32 chama por mensagem o método devolver_equipamento(ID_Equipamento)
Valor de saída (Resultado Esperado)	LED do dispositivo IoT se apaga. Estado do equipamento altera para ‘Disponível’. A mensagem de atualização do estado do objeto é enviada ao Broker.
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.25 Caso de Teste : Broker –

2.25.1 *Descrição*

2.25.2 *Pré-condições para o caso*

2.25.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.26 Caso de Teste :

2.26.1 *Descrição*

2.26.2 *Pré-condições para o caso*

2.26.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.27 Caso de Teste : Equipamentos – Testar Desativar alarme

2.27.1 *Descrição*

O caso de teste envolve tanto um teste funcional quanto um teste de controle de acesso e segurança. Para o teste funcional, definida a pré-condição de que o alarme encontra-se em estado ‘Ativado’, o teste envolve desde a solicitação de desativação de alarme, através da chamada ao método Equipamentos.DesativarAlarme(EtiquetaRFID) em um objetivo da classe Equipamentos, até a desativação visual e sonora do alarme presente no dispositivo IoT, considerando ainda o envio da mensagem de atualização de estado do dispositivo IoT ao Broker. Para o teste de controle de acesso e segurança, o teste envolve checar o nível de hierarquia do solicitante, uma vez que apenas o Gerente do Setor poderá desativar o alarme de forma bem sucedida.

2.27.2 *Pré-condições para o caso*

Alarme de algum equipamento deve estar com estado de ‘Ativado’ e funcionário deve estar autenticado. O objeto da classe Broker deve estar instanciado..

2.27.3 *Conjunto de valores*

Conjunto de Valores		
Cenários	**Cenário 1**	**Cenário 2**
Valor RFID	RFID corresponde a um Gestor do Setor	RFID corresponde a algum funcionário, mas não a um Gestor do Setor.
Valor de saída (Resultado Esperado)	Alarme sonoro e visual no dispositivo IoT encontram-se desligados. Estado do equipamento definido como 'Disponível'. A mensagem de atualização de estado do objeto da classe Equipamentos é enviada com destino ao Broker, imediatamente.	Alarme sonoro e visual no dispositivo IoT permanecem ligados
Resultado Obtido (se diferente do esperado)		
Sucesso/Falha		
Nº Ambiente (se falha)		
Nº Log (se falha)		

2.28 Caso de Teste : Broker –

2.28.1 *Descrição*

2.28.2 *Pré-condições para o caso*

2.28.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	

2.29 Caso de Teste : Broker –

2.29.1 *Descrição*

2.29.2 *Pré-condições para o caso*

2.29.3 *Conjunto de valores*

Conjunto de Valores	
Cenários	**Cenário 1**
Valor 1	
Valor de saída (Resultado Esperado)	
Resultado Obtido (se diferente do esperado)	
Sucesso/Falha	
Nº Ambiente (se falha)	
Nº Log (se falha)	