


CODEHUB BACKLOG							
ID	Nível hierárquico	Funcionalidade	Descrição	Prioridade	Sprint	Status	Legenda
1	Visitante	Criar uma conta	Cria uma conta a partir dos dados fornecidos no processo, salvando os mesmos no banco de dados	Alta	1	Riquelme	Concluído
2	Usuário	Fazer Login	Entra no sistema do CodeHub, salvando as informações da sessão utilizando Cookies	Alta	1	João	Aguardando aprovação
3	Usuário	Fazer Logout	O usuário desconecta-se do sistema, removendo dados da sessão; dados inseridos pelo método login.	Alta	1	Marcus	Bloqueada
4	Usuário	Iniciar novo repositório	Cria o repositório .CodeHub dentro do projeto, permitindo assim que o usuário utilize os demais comandos do CH.	Alta	1	Isaac	Pendente
5	Visitante	Exibir comando de ajuda	Orienta usuários quanto as funcionalidades e exibe os comandos disponíveis na aplicação.	Alta	1	Paloma	Em Desenvolvimento
6	Usuário	Adicionar ao Container	Salva os paths dos arquivos apontados pelo usuário em um compartimento denominado de container	Alta	2	Riquelme	Em teste
7	Usuário	Remover do Container	Modifica o container removendo o path do arquivo que o usuário apontar e reescreve o container.	Alta	2	Paloma	
8	Usuário	Exibir histórico de versões	Lista todo o histórico de versões do projeto ordenado por datas (partindo da mais antiga para a mais recente)	Alta	2	Isaac	
9	Usuário	Criar nova versão	Permite salvar o código em um diretório de versão, associando um hash a este versionamento. Após isso, o container deverá estar vazio e as mudanças efetivadas	Alta	2	Marcus	
10	Usuário	Voltar para versão	Retorna a uma versão determinada: deletando a atual versão do diretório do projeto e trazendo a escolhida de volta	Alta	2	João	
11	Usuário	Remover versão	Deleta a versão informada pelo usuário	Alta	3	João	
12	Usuário	Exibir Container	Lista todos os itens contidos no container	Média	3	Paloma	
13	Usuário	Backup versão removida	Após determinada versão ser removida, ela deve ser salva para permitir sua restauração.	Média	3	Isaac	
14	Usuário	Restaurar versão removida	Recupera a última versão apagada, do backup feito.	Média	3	Riquelme	
15	Administrador	Adicionar outros usuários	Permite que seja adicionado, a um repositório, contribuidores, que possuem permissões para efetuar alterações no projeto	Baixa	3	Marcus	
16	Administrador	Enviar para nuvem	Salvar conteúdo em um Repositório Remoto	Baixa			
17	Usuário	Baixar da nuvem	Get dos dados para o repositório local	Baixa			
18							
19							
20							

## 2. Sprint Backlog

Durante o planejamento, definiu-se uma sequência de ações a serem consideradas, já cientes da existência de possibilidade de serem alteradas durante o desenvolvimento do projeto.

ANÁLISE DE REQUISITOS SPRINT 1	
Funcionais	Não funcionais
Criar uma conta	A aplicação deve funcionar em sistemas operacionais Linux e Windows
Fazer Login	O sistema deve ser implementado na linguagem de programação java
Fazer Logout	O sistema deve ter um banco de dados local
Iniciar novo repositório	Os dados de login devem ser autenticados
Exibir comando de ajuda	O sistema deve responder imediatamente aos comandos do usuário

### Sprint 1:

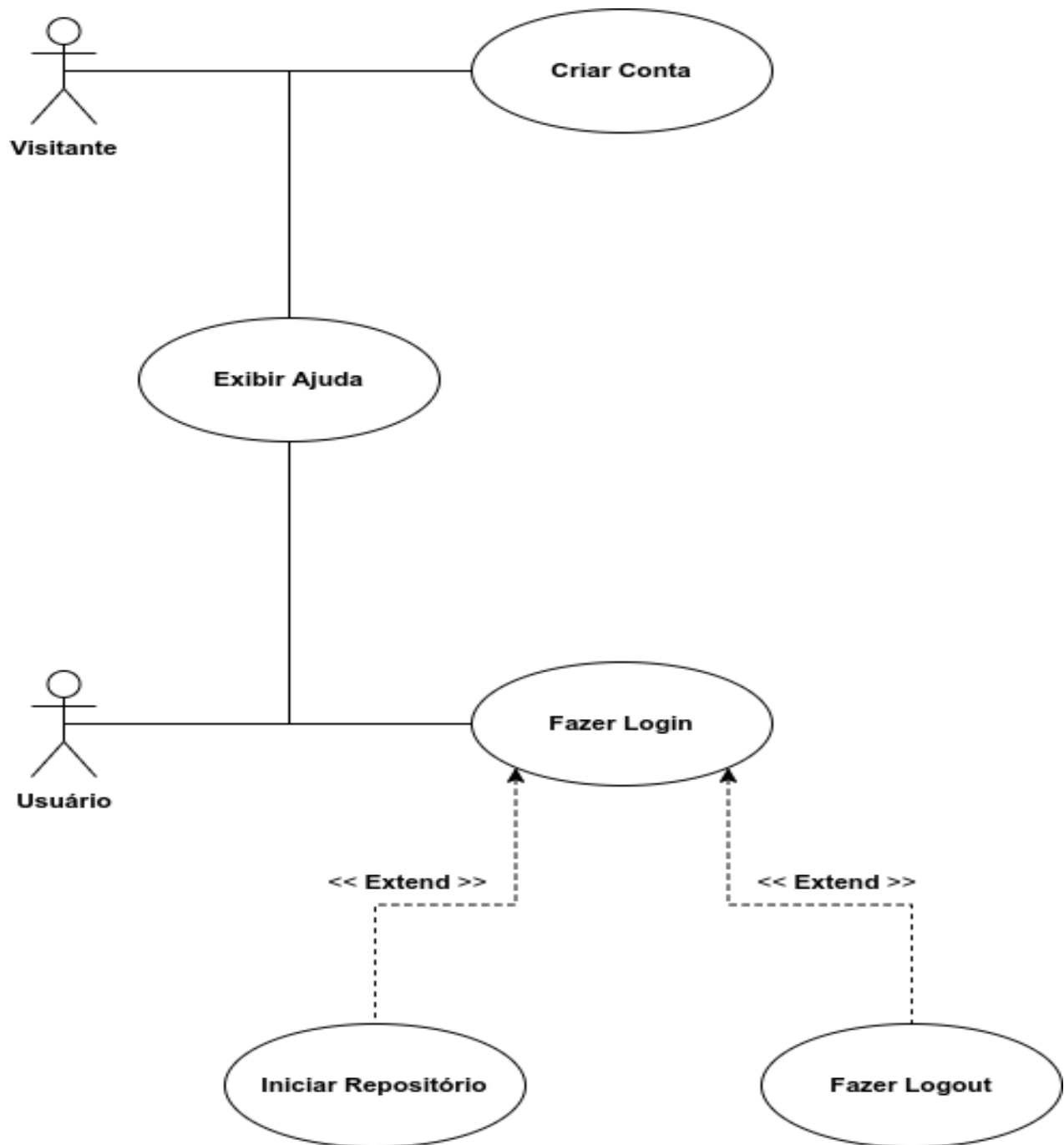
Tempo estimado: 15 dias

Tempo decorrido: 10 dias

Entrega: 16/05/2022

Relator: Professora Luciana

Funcionalidade	Prioridade	Responsável
Criar conta	Alta	Riquelme
Fazer login	Alta	João Gabriel
Fazer logout	Alta	Marcus Vinícius
Exibir ajuda	Alta	Paloma
Iniciar repositório	Alta	Isaac Santiago



### **3.2. Cenários de caso de uso:**

#### **Cenário 01:**

**Nome do Cenário:** Criar conta

**Ator:** Visitante

**Pré-condição:** Não possui.

#### **Fluxo normal:**

1. Visitante insere comando "--registrar";
2. Sistema solicita dados de registro:
  - *"Nome: "*
  - *"Email: "*
  - *"Senha: "*
3. Visitante insere dados de registro;
4. Sistema verifica se os dados de registros inseridos estão de acordo com o formato de autenticação (*regex*);
5. Sistema solicita confirmação do dado de registro senha;
  - *"Confirmacao da senha: "*
6. Sistema armazena o registro do usuário no Banco de Dados;
7. Sistema exibe a mensagem: *"Conta cadastrada com sucesso!"*.

#### **Fluxos alternativos:**

1. Nome, email ou senha não se adequam ao formato de autenticação:
  - 1.1. Sistema exibe mensagem descrevendo o motivo da invalidez:
    - Caso nome inválido:  
*"Quantidade de caracteres invalida!"*  
*"O nome do usuario deve conter de 5 a 20 caracteres!"*
    - Caso email inválido:  
*"Email invalido!"*  
*"Exemplo de email valido: exemplo@dominio.com"*
    - Caso senha inválida:  
*"Senha invalida!"*  
*"Deve conter ao menos 8 digitos"*

*"Deve conter ao menos uma letra maiúscula"*

*"Deve conter ao menos uma letra minúscula"*

*"Deve conter ao menos um caracter"*

- 1.2. Sistema solicita nova inserção do campo inválido e volta ao fluxo inicial.

2. Senhas não coincidentes:

- 2.1. Sistema exibe mensagem de alerta: *"Senhas incompatíveis!"*.

- 2.2. Sistema solicita nova inserção do campo inválido e volta ao fluxo inicial.

3. Email já cadastrado:

- 3.1. Sistema exibe mensagem de e-mail já cadastrado:  
*"Email ja cadastrado!"*.

**Pós-condição:** Dados de registro (nome, email, senha) salvos no Banco de Dados, ou seja, conta do usuário criada com sucesso.

**Cenário 02:**

**Nome do Cenário:** Fazer login

**Ator:** Usuário

**Pré-condição:** 01. Criar conta

**Fluxo normal:**

1. Usuário insere comando "--acessar";
2. Sistema solicita dados de registro:
  - *"Email: "*
  - *"Senha: "*
3. Usuário insere email e senha;
4. Sistema verifica se os dados inseridos estão presentes na base dados;
5. Sistema cria contexto de acesso (cookie);
6. Usuário acessa sistema;
7. Sistema exibe mensagem de boas vindas:  
*"Email e senha corretos, bem-vindo ao CodeHub "*.

**Fluxos alternativos:**

Email ou senha não presentes na base de dados ou fora do formato de autenticação (regex):

1. Sistema exibe mensagem de não efetuação do login:  
*"Email ou senha incorretos, tente novamente".*

**Pós-condição:** Contexto de acesso (cookie) criado.

**Cenário 03:**

**Nome do Cenário:** Fazer Logout

**Ator:** Usuário

**Pré-condição:** 02. Fazer login

**Fluxo normal:**

1. Usuário insere comando "--sair";
2. Sistema remove contexto de acesso (cookie);
3. Sistema exibe mensagem de saída com sucesso:  
*"Sessao encerrada! Volte em breve!"*

**Fluxos alternativos:**

Usuário não está logado:

1. Sistema exibe mensagem de aviso da necessidade de logar:  
*"Voce deve efetuar o acesso para poder sair de uma conta!"*

**Pós-condição:**

1. Contexto de acesso (cookie) removido.
2. Usuário é deslogado do sistema.

#### **Cenário 04:**

**Nome do Cenário:** Iniciar repositório

**Ator:** Usuário

**Pré-condição:** 02. Fazer login

**Fluxo normal:**

1. Dentro do diretório do projeto que deseja versionar, o usuário deve executar o comando "--iniciar";
2. Sistema checka se o repositório atual não foi iniciado anteriormente, ou seja, se não existe um diretório '.CodeHub' dentro do diretório do projeto atual;
3. Sistema gera diretório '.CodeHub' no diretório do projeto, contendo em seu interior os diretórios 'versoes' e 'container'.
4. Após criar a pasta '.CodeHub', o sistema exibe a mensagem:  
*"CodeHub inicializado com sucesso!"*

**Fluxos alternativos:**

Caso o diretório '.CodeHub' já exista no repositório atual:

1. Sistema exibe mensagem de erro ao usuário:  
*"O CodeHub ja foi inicializado!"*

**Pós-condição:** Diretório '.CodeHub' é iniciado dentro de projeto, contendo em seu interior os diretórios 'versoes' e 'container'.

#### **Cenário 05:**

**Nome do Cenário:** Exibir ajuda

**Ator:** Usuário, Visitante

**Pré-condição:** Não possui.

**Fluxo normal:**

1. Usuário/Visitante insere comando "--ajudar" ou não passa argumentos na chamada do programa;
2. Sistema exibe a mensagem:
  - *"Precisa de ajuda? Podemos te ajudar!"*
3. Sistema exibe uma lista com os comandos existentes:

- *--ajuda : exibe janela com todos os comandos"*
- *--registrar : registra um novo usuario"*
- *--acessar : acessa conta existente"*
- *--sair : encerra sessao"*
- *--iniciar : inicia um repositório local"*

**Fluxos alternativos:** Não possui.

**Pós-condição:** Não possui.