UNIVERSIDADE PRESBITERIANA MACKENZIE

CURSO DE CIÊNCIA DA COMPUTAÇÃO

ENGENHARIA DE SOFTWARE

GUILHERME ARAUJO SETTE TIA 41783441

LUIZ HENRIQUE MONTEIRO DE CARVALHO TIA 41719468

JUAN VICTOR DUTRA JUAN TIA 31711081

DOCUMENTAÇÃO APLICATIVO DE FRASES

SÃO PAULO

2019

GUILHERME ARAUJO SETTE TIA 41783441

LUIZ HENRIQUE MONTEIRO DE CARVALHO TIA 41719468

JUAN VICTOR DUTRA JUAN TIA 31711081

DOCUMENTAÇÃO APLICATIVO DE FRASES

Trabalho acadêmico válido como 2ª avaliação intermediária na disciplina Engenharia de Software, ministrada pela Prof.ª. Renata Mendes de Araújo no curso de Ciência da Computação da Universidade Presbiteriana Mackenzie.

SÃO PAULO

2018

1. Introdução

Está é uma documentação de estrutura do software “Aplicativo de frases”. A finalidade do aplicativo é uma central pessoal de frases do usuário com a possibilidade do usuário interagir com o sistema e do usuário setar notificações para armazenar frases.

1.1. Justificativa da Escolha do Tema

A escolha do tema foi dada à necessidade de fornecer uma central para pessoas guardarem frases que o usuário goste como forma de registro.

1.2. Delimitação do Problema

O ponto central do projeto é criar um aplicativo que facilite todo o processo de buscar frases em navegadores ou livros que uma pessoa tem ou goste. Oferecendo uma plataforma apenas como repositório / portfólio de frases onde o usuário salva as frases que goste.

1.3. Objectivos do Projecto

O objetivo do projeto é criar a central para guardar as frases favoritas da pessoa.

1.4. Metodologia de Trabalho e de Desenvolvimento do Software

O modelo de processo a ser usado será o Scrum, pois se adapta ao nosso modelo de dinâmica e trabalho.

O desenvolvimento será feito no modelo Orientado a Objetos para atender aos requisitos de forma mais estruturada.

1.5. Estimativa das atividades do Projeto

Consideramos levar 80 horas para todo o processo de desenvolvimento do projeto. Serão 24 horas para o desenvolvimento e especificação dos requisitos e 56 horas de desenvolvimento.

2. Descrição Geral do Sistema

O sistema será um aplicativo de frases em que será possível armazenar frases no aparelho celular da pessoa em um banco de dados offline usando a memória do celular, será possível criar novas frases para fazer parte do repositório do aplicativo, a edição e exclusão de frases feitas pelo próprio usuário, também serão permitidos. Todo o conteúdo de frases é controlado pelo usuário e é offline. E além disso o usuário vai poder programar notificações para lembrar de armazenar frases.

2.1. Descrição do Problema

As pessoas que queiram guardar frases que gostem estarão engajadas com o aplicativo que facilita essa tarefa trazendo uma plataforma pessoal de fácil acesso com todas as frases que o usuário venha a guardar e queira se lembrar das frases que ele já guardou.

2.2. Principais Envolvidos e suas Características

Pessoas que queiram e que gostam de frases de qualquer cunho e desejam guardar essas frases, isso inclui pessoas de qualquer área, podendo guardar frases como, lembrança, motivação e afins.

2.2.1. Utilizadores do Sistema

O engajamento do sistema se dará em sua grande maioria por Evolução pessoal.

2.2.2. Desenvolvedores do Sistema

Os Desenvolvedores do sistema serão:

Guilherme Araujo Sette - Desenvolvedor

Luiz Henrique Monteiro de Carvalho - Desenvolvedor

Juan Victor Dutra Juan - Desenvolvedor

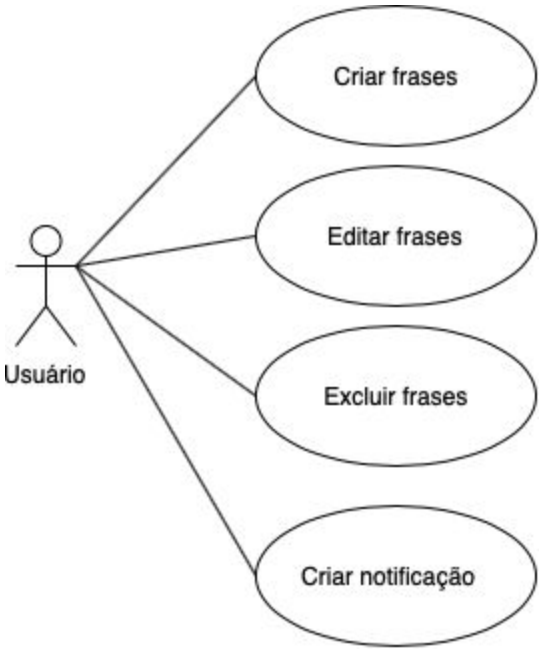
2.3. Regras de Negócio

R1: O usuário só pode criar editar ou excluir frases de autoria própria.

3. Requisitos do Sistema

- Criar Frases
- Edição de frases
- Exclusão de frases
- Criar notificação
- Excluir notificação

3.1. Requisitos Funcionais



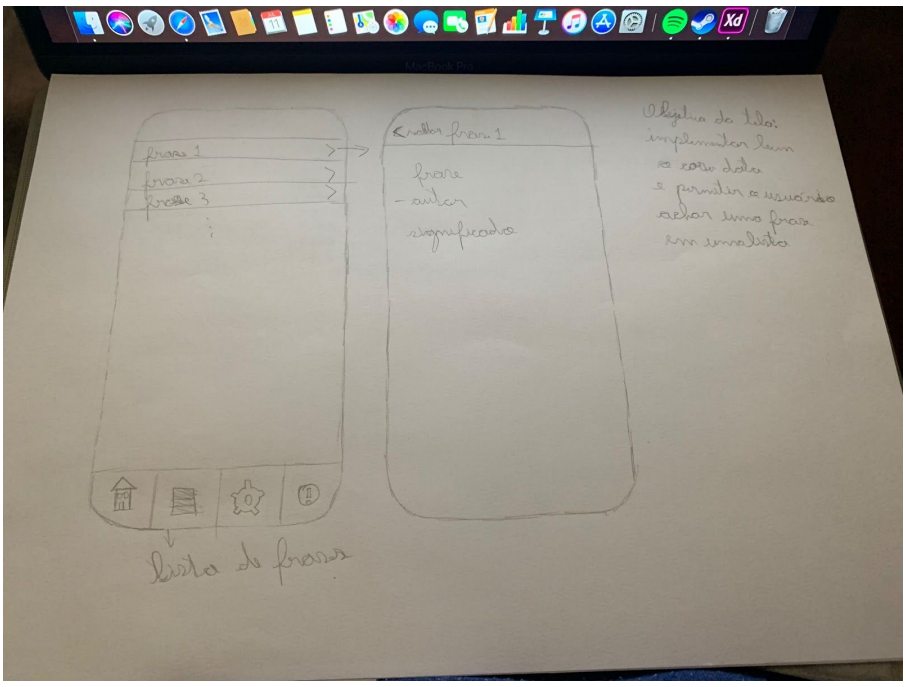
3.1.1. Prioridade de requisitos

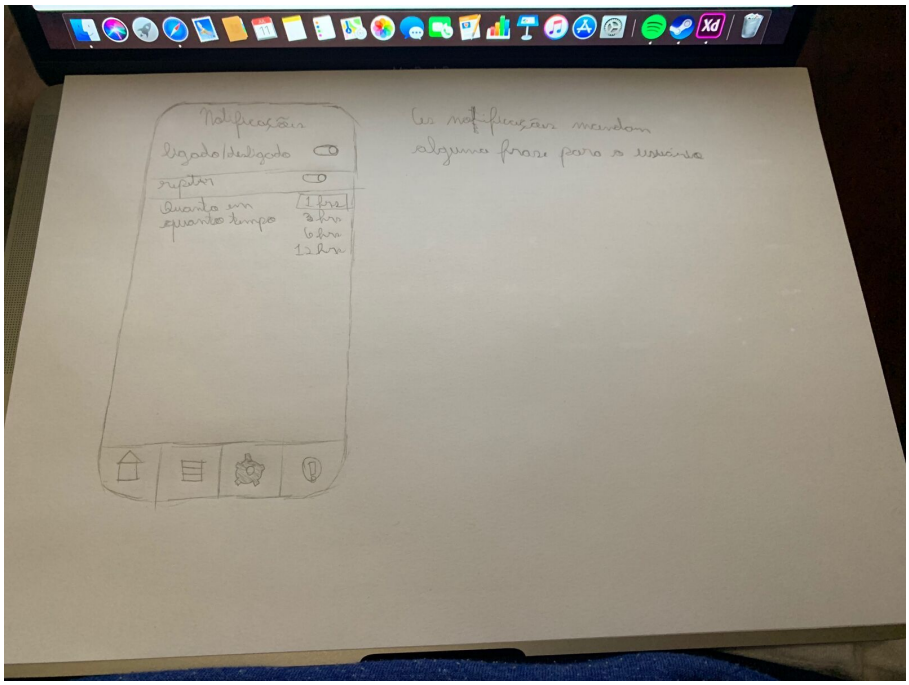
- Criar Frases
- Excluir Frases

3.2. Protótipo

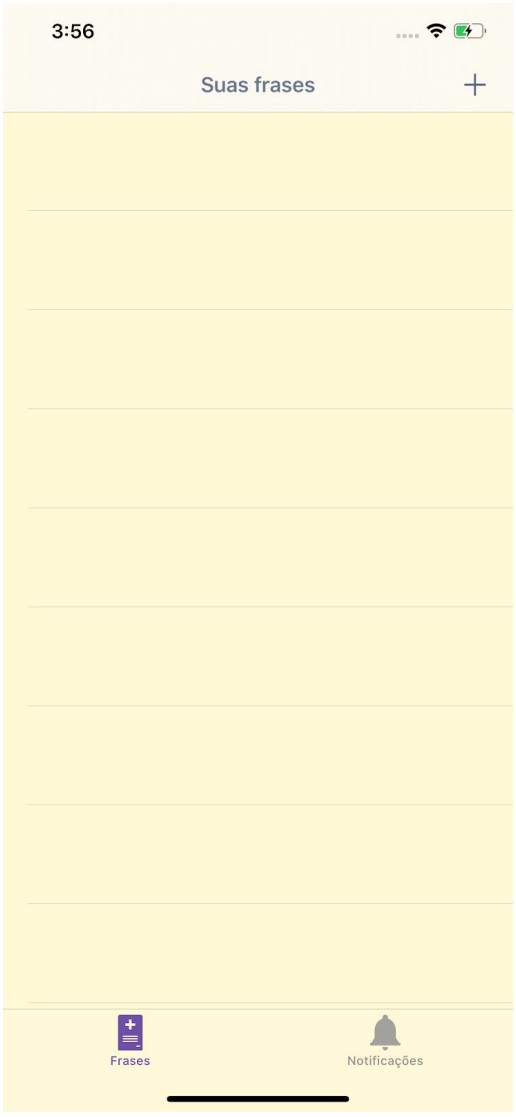
Apresentação da prototipagem das telas.

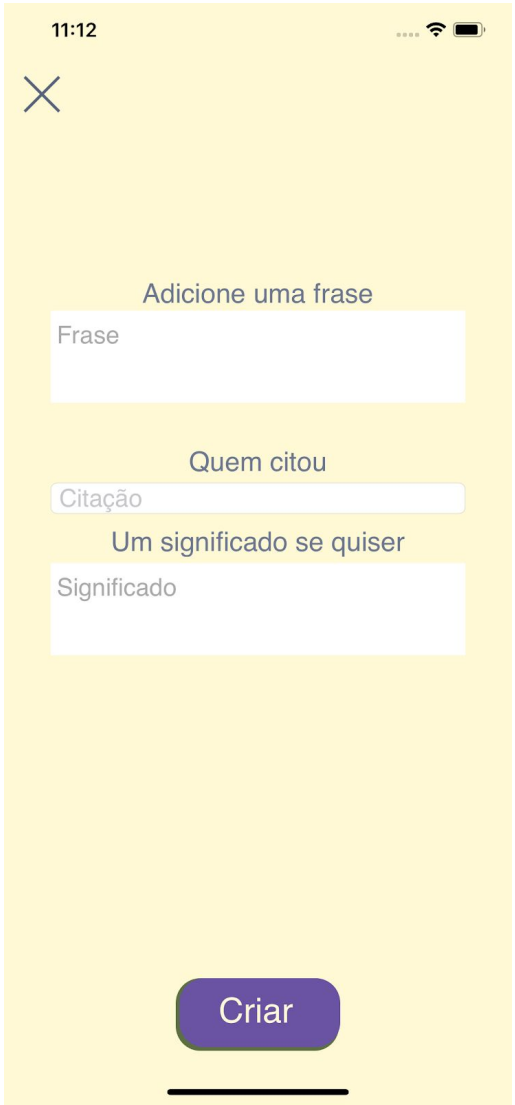
3.2.1 Protótipo de baixa fidelidade

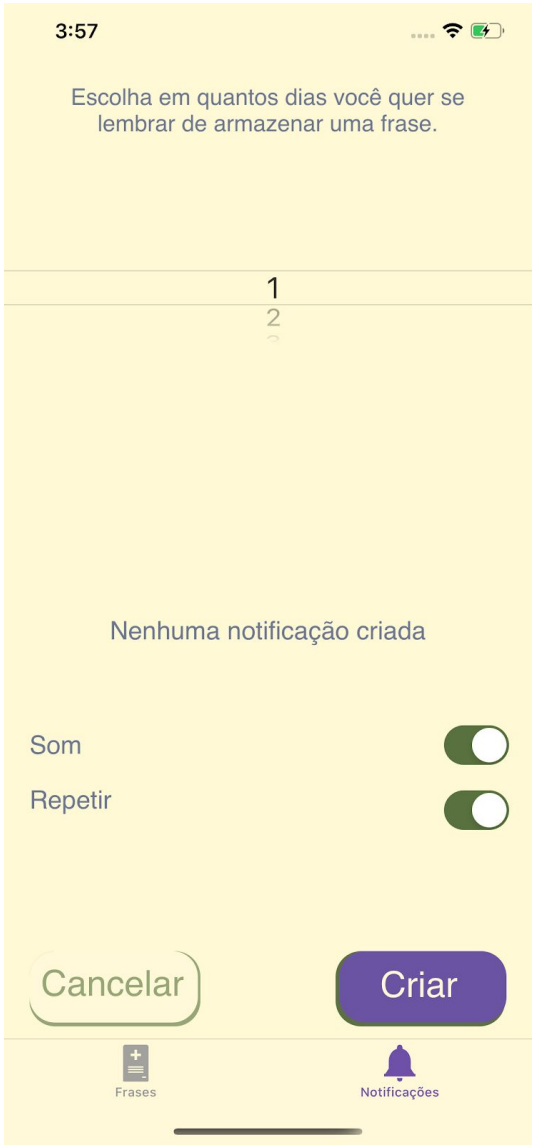



Notificações
ligado/desligado
repetir
Quanto em
quanto tempo
1 hra
3 hra
6 hra
12 hra
As notificações mandam
alguma frase para a usuária

3.2.1 Protótipo de alta fidelidade

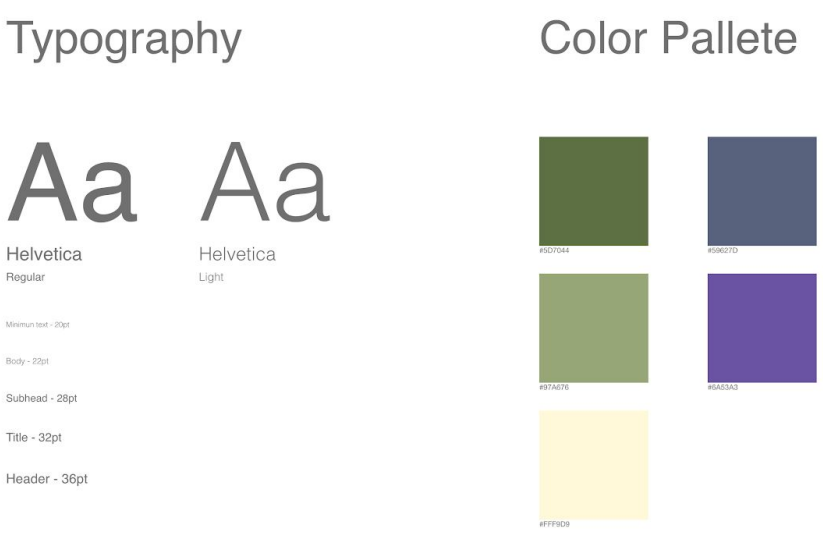


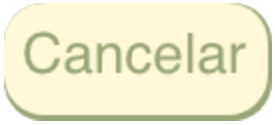

11:12
Adicione uma frase
Frase
Quem citou
Citação
Um significado se quiser
Significado
Criar


3:57
Escolha em quantos dias você quer se lembrar de armazenar uma frase.
1
2
Nenhuma notificação criada
Som
Repetir
Cancelar
Criar
Frases
Notificações

4. Análise e Design

O design do aplicativo será um design minimalista, com botões descritivos e básicos para sua função. Um Style Guide do aplicativo com as cores e fontes usadas:





Criar

Editar

Salvar

4.1. Arquitetura do Sistema O aplicativo será hospedado na base de aplicativos iOS na AppStore. Com uma tabela, para facilitar a busca de frases que são armazenadas pelo usuário. O sistema é um banco de dados local sem conexão offline onde apenas o usuário tem acesso a esse banco com as frases que ele armazenou. E além disso tem a área de criação de notificação onde o usuário pode criar apenas uma notificação para lembrar ele de armazenar uma frase em um intervalo de dias.

4.2. Modelo do Domínio

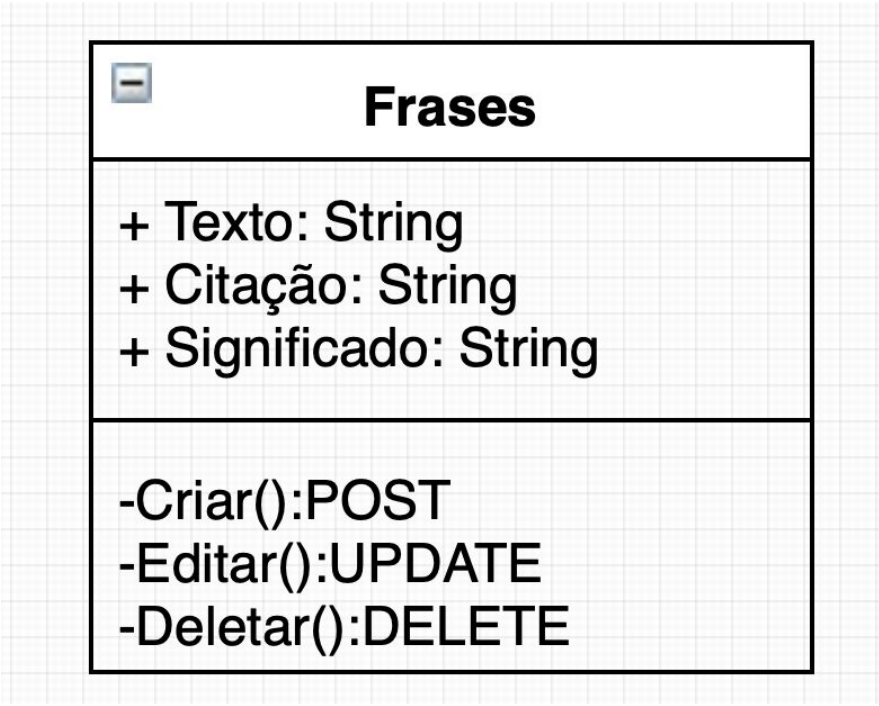
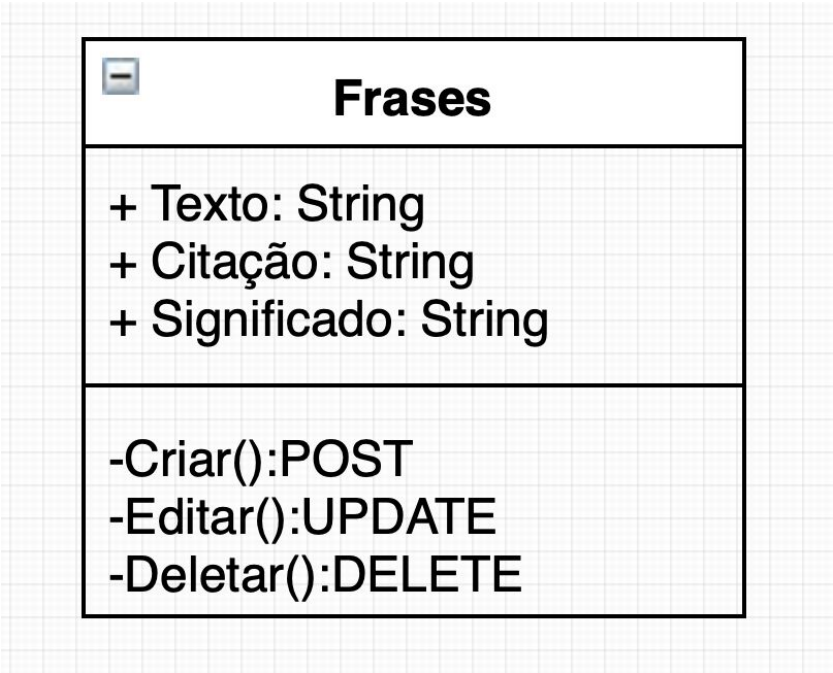


Diagrama de classes

4.3. Modelo de Dados

Modelo de dados simples com Banco de dados local com 1 classe e 3 atributos.

4.3.1. Modelo Lógico da Base de Dados



O modelo é simples e tem apenas a interação do usuário podendo criar, editar e deletar frases. O modelo utiliza um banco de dados local chamado de CoreData. Onde uma classe é instanciada com os atributos de Texto, Citação e Significado que tem os métodos que serão utilizados pelo usuário de poder criar, editar e deletar frases.

4.3.2. Criação Física do Modelo de Dados

Primeiramente é feito uma classe no banco de local com 3 atributos. Esses atributos permitem a interação com métodos CRUD de GET, POST, UPDATE e DELETE.

4.3.3. Dicionário de Dados

Inicialmente o banco de dados local vai vir despopulado. Dependendo apenas do usuário popula-lo.

4.4. Ambiente de Desenvolvimento

Linguagem Swift com banco de dados local e CoreData. Um MacBook com a ferramenta XCode. Sem necessidade de equipamentos de rede. E Adobe XD para a criação de elementos gráficos do software. GitHub para repositório do aplicativo.

4.5. Sistemas e componentes externos utilizados

Nenhum componente externo é necessário para o desenvolvimento desse software. Apenas os citados no Ambiente de Desenvolvimento.

5. Implementação

BackEnd:

A implementação deve ser feita em Swift no XCode com modelagem do banco de dados local com apenas uma classe com 3 atributos. E devem ser implementados métodos CRUD nesse banco de dados.

A outra parte de implementação serão as notificações que usarão uma biblioteca própria da linguagem swift chamada LocalNotifications onde serão feitas as configurações do usuário para criar uma notificação para lembrá-lo de armazenar uma frase. Apenas uma notificação pode ser criada por vez e o será possível ativar ou desativar o som desta notificação e ativar ou desativar a repetição desta notificação. A notificação ao ser criada vai poder ser cancelada também a qualquer momento pelo aplicativo.

FrontEnd:

Com a ferramenta de XCode modelar as telas do aplicativo e interface gráfica com os componentes criados no Adobe XD.

Organização do código e implementação o geral:

Usar o modelo Model View Controller (MVC) para organização de classes que vão cuidar do comportamento de cada objeto criado seja no FrontEnd seja no BackEnd.
Model é responsável pelo banco de dados local e métodos CRUD.
View é responsável pelas características dos objetos visíveis na tela do FrontEnd.
Controller é responsável pela comunicação e comportamento entre Model e Controller.

O controle de versionamento e implementação de features vai ser feito via GitHub com GitFlow como ferramenta de gestão de configuração.

6. Testes

Os testes serão feitos tendo como base os métodos CRUD. E com base nas configurações de notificação que criarmos.

6.1. Plano de Testes

Os testes que precisam ser feitos nos sistema são de persistência de dados após a utilização dos métodos CRUD no sistema. Sempre que um método do CRUD testado é preciso ver se houve a persistência dos dados alterados.

O resultado esperado é que sempre que um método CRUD for utilizado o sistema persista esses dados.

E devem ser feitos testes também com a central de notificações para garantir que elas estejam sendo criadas, estejam funcionando e estejam sendo deletadas também.

6.2. Execução do Plano de Testes

Teste de criação de frase no Banco: ao utilizar o CREATE uma frase com os 3 atributos deve ser criada e é esperado que a frase esteja no banco após essa criação.

Teste de recuperação de frase no Banco: ao Criar, Deletar ou Editar uma frase é feito um GET sempre que a tela do aplicativo mudar e é esperado que ocorra uma atualização na tela e as frases devem aparecer para o usuário.

Teste de deletar a frase do Banco: ao deletar a frase com o método DELETE a frase deve desaparecer do banco de dados.

Teste ao editar a frase do Banco: ao editar uma frase específica com o método UPDATE é esperado que a frase que tenha sido editada tenha mudado de acordo com as alterações feitas.

Para os testes de notificação serão necessários 5 testes: Um para quando a notificação

estiver com Som ativo e Repetição ativo, um para quando estiver apenas com o som ativo, um para quando estiver apenas com repetição ativa, um para quando estiver com os dois desativados e um onde a notificação tenha sido cancelada.

Todos os testes de notificação que tem a repetição ativa precisam ser maiores pois é preciso ver se realmente a notificação está sendo repetida.

7. Implantação

O principal motivo para o funcionamento do software é a verificação e confirmamento de atualização do banco e logo em seguida a atualização da tela cada vez que é feito alguma alteração no banco de dados. E o funcionamento de notificação também.

7.1. Manual de Implantação

A instalação do sistema deve ser feita na AppStore ao clicar no botão de “Instalar” na tela do aplicativo na loja.

8. Refletindo sobre a arquitetura do software

Como mencionado anteriormente será utilizado o modelo de arquitetura de software de MVC. O MVC se encaixa bem na arquitetura de aplicativos pois separa bem todas as partes lógicas e de telas/visuais do aplicativo. Essa separacao e organizacao deixam o processo mais efetivo e organizado.

Aqui estão os prós e contras do MVC:

Prós:

Desenvolvimento simultâneo - vários desenvolvedores podem trabalhar simultaneamente no model, controler e view.

Coesão alta - o MVC permite o agrupamento lógico de ações relacionadas em um controler. As views de um modelo específico também são agrupadas.

Baixo acoplamento - A própria natureza da estrutura MVC é tal que existe um baixo acoplamento entre models, views ou controlers.

Facilidade de modificação - Devido à separação de responsabilidades, o desenvolvimento ou modificação futura é mais fácil, ou seja, a escalabilidade do produto aumenta.

Várias views para um modelo - Os models podem ter várias views.

Contras

Navegabilidade de código - A navegação da estrutura pode ser complexa porque introduz novas camadas de abstração e requer que os usuários se adaptem aos critérios de decomposição do MVC.

Várias representações - decompor um recurso em três artefatos causa dispersão. Portanto, exigindo que os desenvolvedores mantenham a consistência de várias representações ao mesmo tempo.

Vendo todas os prós e contras a equipe decidiu que pode se adaptar bem ao modelo MVC e para a demanda do aplicativo esse modelo se encaixa bem.

9. Manual do Utilizador

Para adicionar uma frase existe um botão com forma de (+) no lado direito superior. Ao clicar no botão de mais (+) uma tela irá aparecer onde ele pode adicionar uma frase escrevendo nos campos de texto, citação e significado e clicar em salvar para guardar essa frase ou clicar em um botão com forma de X para cancelar o adicionamento de uma frase no canto esquerdo superior.

Para deletar a frase: na tela com a tabela da segunda aba do aplicativo deslizar o dedo da direita para a esquerda em cima da frase que deseja deletar que um botão em vermelho de deletar vai aparecer. Clicando no botão a frase será deletada. Para cancelar essa ação basta clicar em qualquer lugar fora da área do botão ou arrastar da esquerda para a direita na frase selecionada.

Para editar a frase: na tela com a tabela da segunda aba do aplicativo clique em uma frase que deseja editar e a frase vai aparecer em uma tela nova. Nesta tela nova um botão no canto direito inferior aparece para poder editar a frase. Ao clicar nesse botão uma tela com a frase editável aparece e é possível editar a frase. Para salvar as alterações é preciso clicar no botão inferior na tela. Caso queira cancelar a edição clicar em um botão com X no canto superior esquerdo da tela.

Para criar uma notificação basta ir na segunda aba do aplicativo e selecionar em qual o intervalo de dias que se deseja criar as notificação e clicar no botão criar. Apenas uma notificação pode ser criada por vez e se forem criadas mais notificações a última e mais recente sempre substitui a que foi criada deixando apenas uma. A notificação pode ter som e pode ser repetida no intervalo de dias selecionado. A notificação pode ser cancelada a qualquer momento com o botão de cancelar notificação.

10. Considerações Finais e Recomendações

A aplicabilidade do aplicativo é deixar o usuário no controle de um banco de frases que ele pode criar e ser algo pessoal. Limitações do aplicativo não ser online e ser muito simples. Inovação em criar um aplicativo de frases totalmente moldáveis. Outros projetos de cunhos motivadores poderiam ter as funções desse aplicativo. O software no futuro poderia trazer frases novas de um banco online todos os dias também. E caso o usuário queira compartilhar uma frase ele poderia a implementação online seria o melhor caminho para melhorar o software.