DESENVOLVIMENTO

O aplicativo AMU produto final do trabalho corrente destina-se a auxiliar a utilização dos medicamentos por grupos de pessoas com alguma comorbidade. Logo, para isso é necessário apresentar algumas bases de que propiciem a adequação de forma mais simples. A figura 200 demostra alguns componentes:

* Cliente
* Barramento de Serviços
* Inteface de Bulário Eletrônico
* Interface de Integração de Usuário
* Interface de Acesso ao Banco de Dados

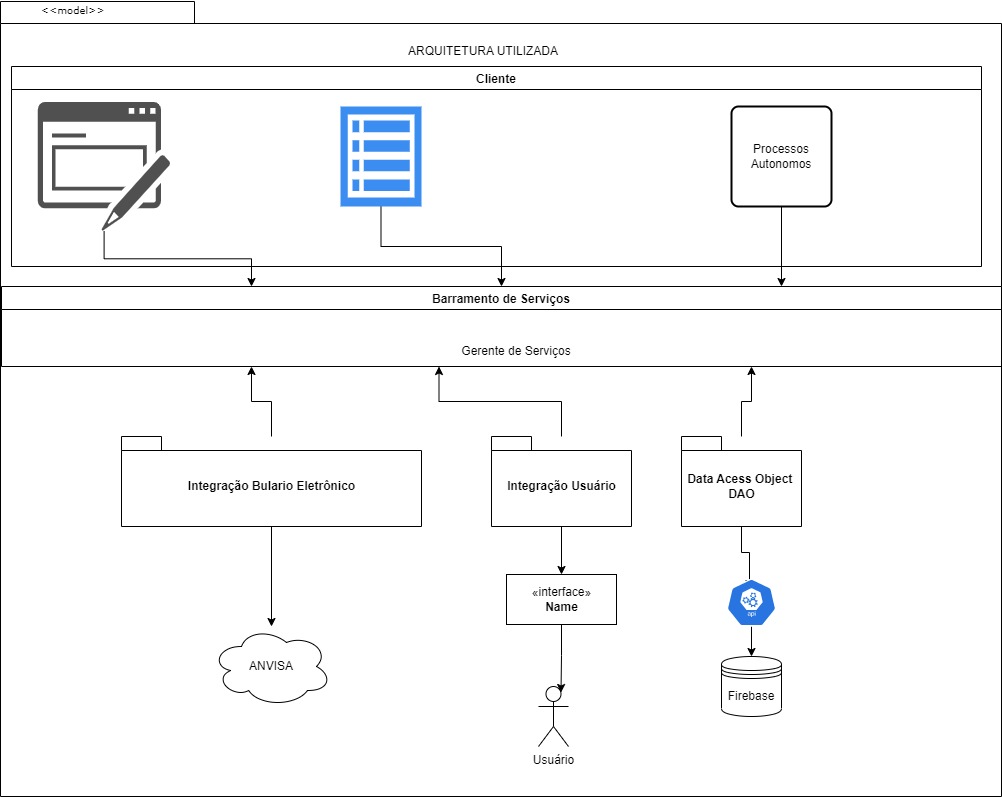


Figura 2000

Explicando a arquitetura de alto nível

1. **Cliente**

Objetivamente falando seria qualquer interface ou recurso utilizado pelo usuário para consumir ou solicitar um serviço do aplicativo. Pode ser um formulário ou uma lista. Assim como pode ser um serviço autônomo monitorando horários por exemplo.

1. **Barramento de Serviços**

Objetiva criar uma interface de acesso a todos os recursos do aplicativo. Para realizar utilizamos um padrão de projeto conhecido como “Facade”. O padrão Facade (pronuncia-se "façade” por ser uma palavra de origem francesa) propõe a criação de uma classe intermediária que serve como uma fachada para que o cliente possa acessar as funcionalidades desejadas. Essa classe encapsula a complexidade da interação entre os diversos componentes e desacopla o cliente das implementações. (1)

1. **Interface de Integração Bulário Eletrônico**

Esse componente possibilita criar uma ponte entre o serviço e a consulta a fontes de dados de medicamentos. Aqui utilizamos um padrão de projeto chamado “Strategy” com a finalidade de abstrair a fonte de dados que utilizamos. A ideia básica é impedir que o aplicativo conheça os detalhes da consulta de medicamento chegando ao ponto de trocar a fonte de dado e a mudança não gerar nenhum tipo de problema no restante do projeto. (2)

1. **Interface de Integração de Usuário**

Seguindo o mesmo padrão “Strategy” abordado no tópico “Integração de Bulário Eletronico” é necessário criar recursos específicos para os tipos de comorbidades apresentadas. Por exemplo temos uma interface para pessoas com visão reduzida ou mesmo sem visão. Segue a lista das interfaces:

1. Usuario comum
2. Usuario PCD
   1. Usuário com perda de visão ou reduzida
   2. Usuario com perda auditiva
   3. Usuário com TEA
3. Usuario Idoso

Utilizando a interface podemos por exemplo avisar que é horário de medicamento abstraindo a forma que será a saída do aviso. Para alguém com perda de visão o aviso será sonoro. No entanto para alguém com perda auditiva utilizamos a vibração do celular e notificações.

1. **Interface de Acesso a Dados**

O componente de acesso dados além de abstrair o uso da API do firebase possibilita que o usuário tenha acesso em qualquer dispositivo instalado em virtude do Banco de dados está remoto. Utilizamos o padrão de projeto “DAO”. Esse padrão possibilita a troca do banco de dados. Apenas trocando a fábrica da implementação.

1. **Problema analisado no trabalho**

No presente trabalho analisa-se o uso de medicamentos por pessoas com perfil específicos como idosos e pcds

1. **Aplicativo**

**Primeiro Acesso**

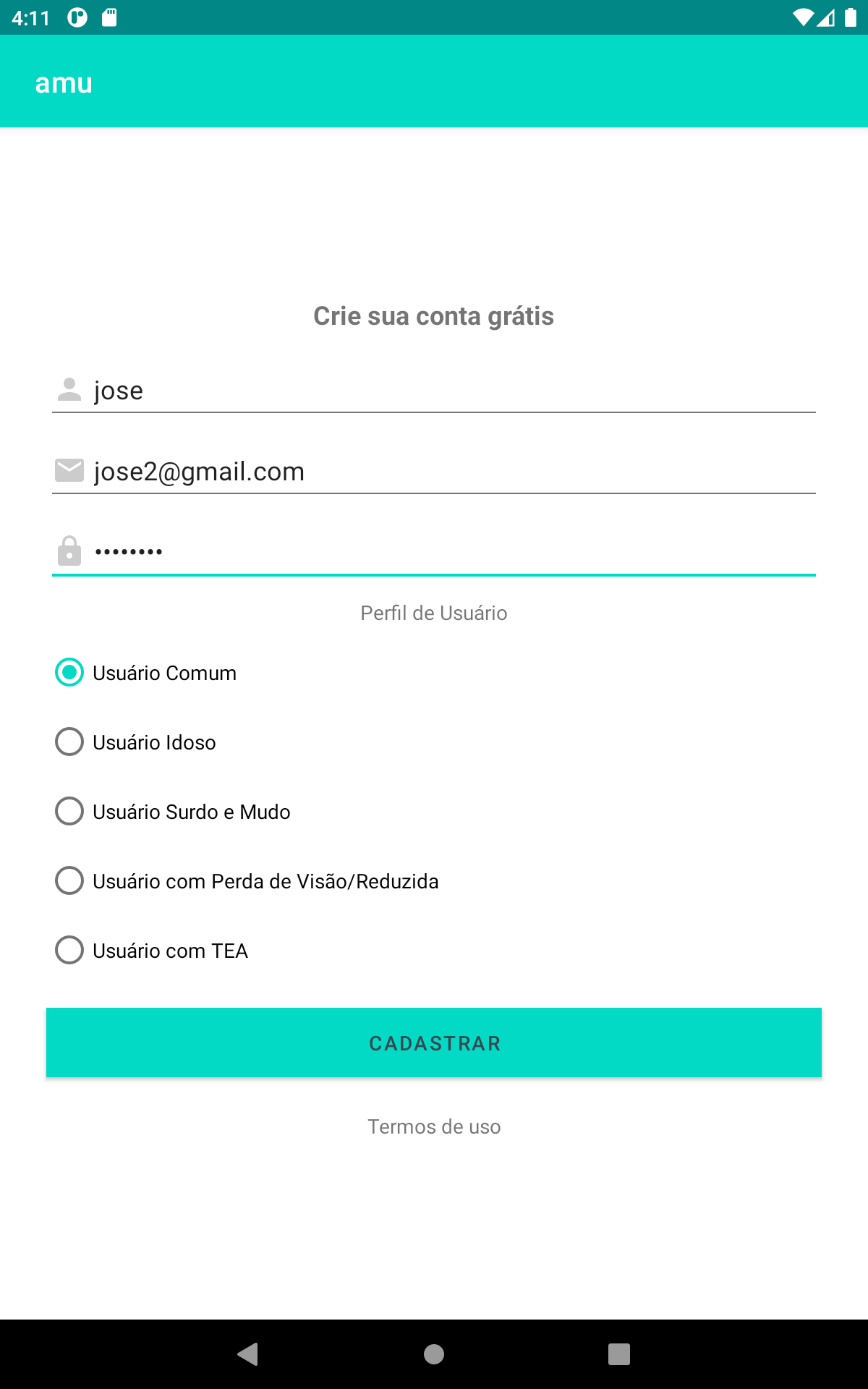
No primeiro acesso do aplicativo alguns slides de boas-vindas são apresentados. Então é apresentado ao usuário criar um cadastro novo ou efetuar o login informando e-mail e senha. Após efetuar o cadastro o aplicativo exibirá a janela principal. Se o usuário possuir um cadastro feito anteriormente escolherá a opção efetuar login.

Tanto no primeiro caso quanto no segundo caso o aplicativo exibirá a janela principal.

Quando o usuário já tiver feito login pelo menos uma vez o aplicativo utilizará o recurso FirebaseAuth do serviço de banco de dados Firebase para autenticar o usuário.

O objetivo do Firebase Authentication é facilitar o desenvolvimento de um sistema de autenticação seguro, além de melhorar a experiência de login e integração para os usuários finais. Ele oferece uma solução de identidade completa, compatível com contas de e-mail/senha, autenticação por telefone, login do Google, Twitter, Facebook, GitHub. O login por e-mail e senha foi escolhido por apresentar simplicidade e os e-mails serem únicos em qualquer servidor de e-mail. (Referencia 3)

Dito isto, sempre que o usuário acessar o aplicativo fará login automático apresentando facilidade de uso desde a tela inicial.

**Cadastro de Usuário**

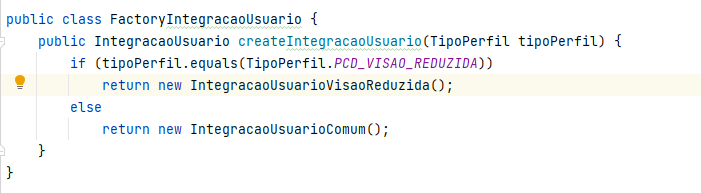
Nesta primeira versão do aplicativo AMU o usuário comum poderá utilizar todas as funcionalidades. O usuário com perda de visão ou reduzida terá através do uso da voz acesso a algumas funcionalidades.

Para usuários idosos, surdos/mudos e TEA não tem funcionalidades adaptadas.

O perfil é o elemento chave na comunicação entre as camadas do aplicativo. O Aplicativo determina através do perfil a forma como os comandos devem ser respondidos, para sustentar esse recurso, utilizamos padrões de Projetos. O principal padrão de projeto adotado foi o Strategy.

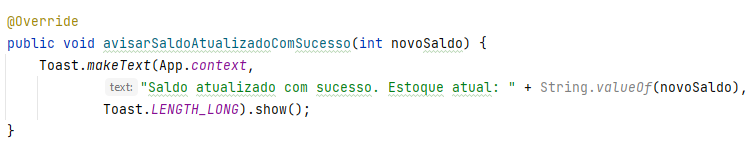
Quando o aplicativo inicia pelo cadastro inicial sabe qual o perfil do usuário. Uma variável global com o nome “integracaoUsuario” então é inicializada por uma fábrica de objetos. Para exemplificar o uso desse recurso, quando o aplicativo dispara um alarme indicando que um medicamento deve ser tomado é feito em duas etapas: através de notificação e por áudio. Isto é necessário para atender quem não pode ler uma notificação. Então o aplicativo fará uma chamada para “integracaoUsuario” que disparará um alarme em audio. A mesma chamada existe para o usuário comum, mas como não tem utilidade para ele não fará nada. Então cada perfil tem uma série de funcionalidades que serão chamadas de acordo com o perfil de usuário, com isso, teremos IntegracaoUsuarioComum ou IntegracaoUsuarioVisaoReduzida e etc.

O fragmento de código abaixo demonstra o uso de uma fábrica de objeto para ser utilizado conforme o perfil.



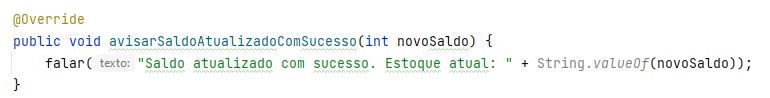
Quanto ao uso do Strategy abaixo temos dois fragmentos de código.

O primeiro fragmento dá suporte ao perfil de usuário comum.



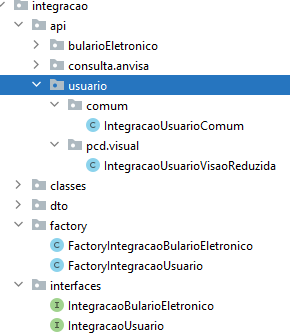
Na imagem acima ao atualizar o saldo de medicamentos uma mensagem é exibida na tela.

O segundo fragmento dá suporte ao usuário com visão reduzida/perda



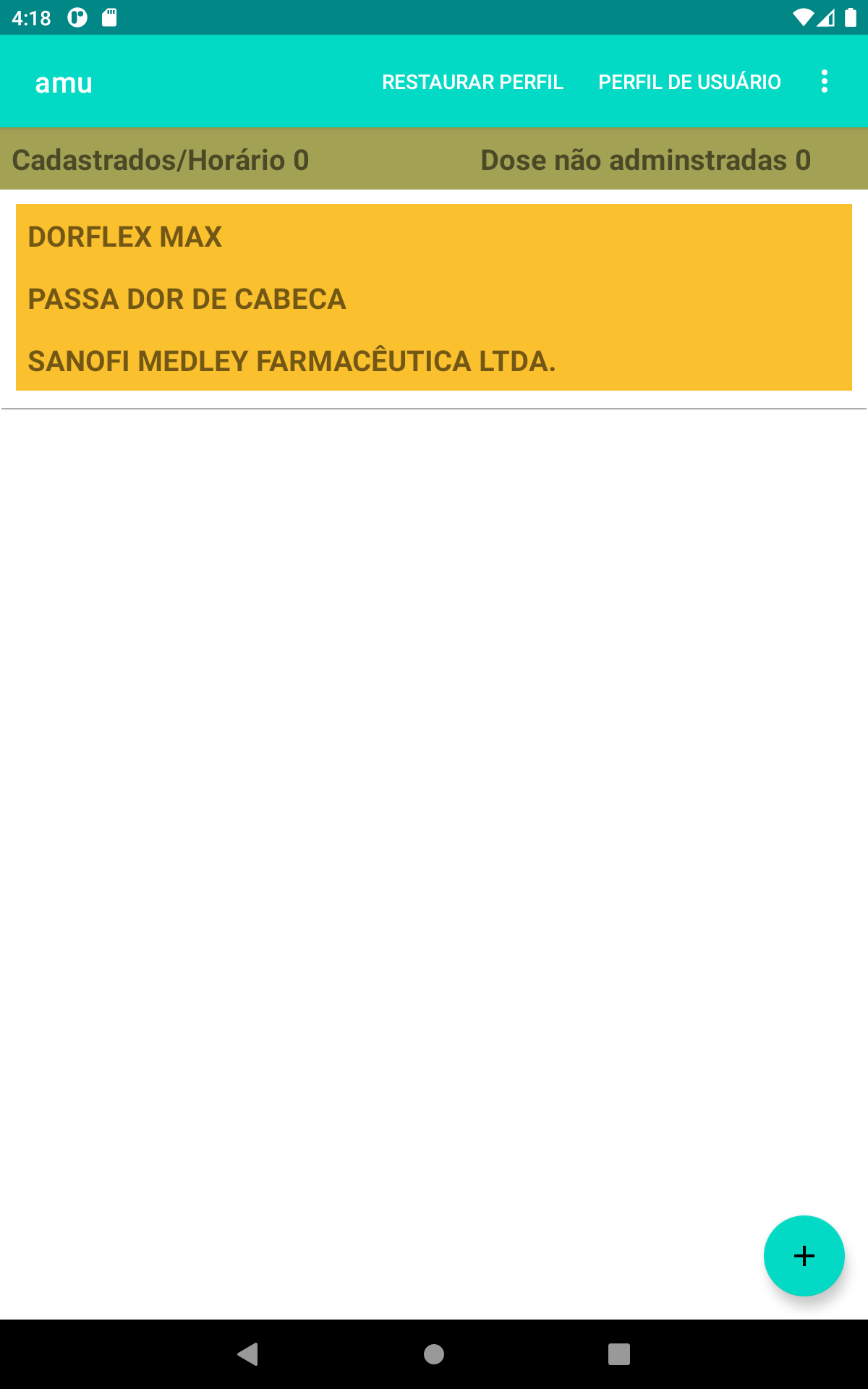
Na imagem acima ao atualizar o saldo de medicamentos um áudio é reproduzido. Assim podemos constatar o uso do padrão Strategy.

Para dar suporte a estrutura mencionada acima foi necessário dividir o projeto em camadas. Abaixo temos uma imagem que exibe a estrutura da camada de Api-integração



### Janela Principal

Como o objetivo do presente trabalho é ajudar pessoas com alguma dificuldade em fazer uso de medicamentos e nesta primeira versão serão atendidos um público com dificuldade visual reduzida/perda foram desenvolvidos alguns comados de voz. Partimos da premissa que tal perfil precisa de um cuidador para fazer algumas etapas preliminares as quais iremos abordar em tópico anexo. Tais comandos estão disponíveis desde a tela principal exibida abaixo:



Comandos de voz disponíveis após o usuário tocar em qualquer área da tela principal. O aplicativo informa que o usuário poderá falar o comando reproduzindo um pequeno bip. Após escutar o comando falado o aplicativo informa que a captura foi encerrada e então processa a voz:

### Falando - ícones de pessoas grátis Lista de Medicamentos

O AMU lerá as informações principais de cada medicamento previamente cadastrado

### Falando - ícones de pessoas grátis Detalhe do Medicamento “apelido”

O AMU localizará na lista o medicamento que corresponde ao apelido cadastrado. Após localizar o medicamento exibirá o detalhe lendo as principais informações do medicamento e informando que está na janela de detalhe do medicamento.

### Falando - ícones de pessoas grátis Administrar

O AMU trocará o perfil do usuário temporariamente possibilitando que o cuidando faça a administração dos dados cadastrados. Entre esses itens está: Cadastrar medicamento, cadastrar horários, cadastrar compra ou informar uma redução na quantidade de medicamento disponível. Poderá alterar o apelido do medicamento entre outros dados. Após efetuar a mudança basta clicar no botão “Restaurar perfil”.

### Falando - ícones de pessoas grátis Sair

### O AMU será fechado.

Para termos acesso a todos os recursos de voz foi necessário utilizar alguns recursos importantes relativos a plataforma Android, bem como fundamentos do desenvolvimento móvel. O recurso de reconhecimento de voz, reprodução de voz e mesmo o uso do firebase são normalmente assincronos.

Devido a essa caracteristica precisamos “assinar” metodos para receber as respostas ao final da execução. Para issofoi necessário o uso do recuso de CallBacks.

Trata-se de uma função que é executada quando algum evento acontece ou depois que algum código chega ao estado desejado. **Também conhecido como função de retorno**, o Callback cria regras dentro de outras funções para que sejam utilizadas no futuro. Normalmente, ele age de forma assíncrona, ou seja, não é executado imediatamente. A aplicação continuará rodando enquanto espera o momento certo da sua execução.

O Callback é muito comum na linguagem Java, por exemplo, durante a busca de dados ou reprodução de mídia. Isso porque permite ao programador especificar o que deve ocorrer quando a execução acabar.

As notificações (ou “**callbacks**”) permitem que o aplicativo seja informado quando uma transação tiver seu status alterado. Dessa forma, podemos identificar quando um dado solicitado ja foi devolvido pelo banco de dados firebase ou quando uma captura de voz já foi obtida.

Para fazer uso deste importante recurso na programação orientada a objeto foi necessário fazer uso massivo de interfaces.

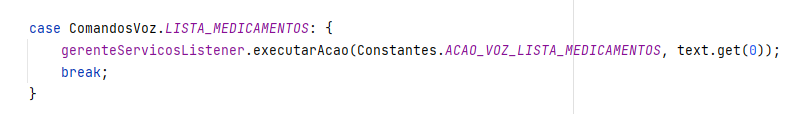
Alguns fragmentos de código abaixo mostram o uso de um desses esquemas.

Por exemplo para obter a lista de medicamentos a apartir do uso da voz:

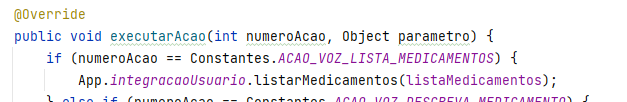
Fragmento 1



Fragmento 2



Fragmento 3



Nos fragmentos acima precisamos destacar os elementos chaves:

GerenteServicoListener e Constantes que determinam a entrada do comando e ação resultado. Vamos detalhar o fluxo:

Primeiramente vemos no fragmento 1 que durante a definição do formulário PrincipalActivity implementados a interface GerenteServicoListener. Essa classe fará o papel do call-back.

O fragmento 2 ocorre após a classe responsável pela escuta dos comandos de voz definirem que o comando é “Lista de Medicamentos”. Cada comando de voz tem uma ação de voz conectada. No caso atual é reproduzir em áudio a lista de medicamentos. Nesse momento uma chamada Callback ocorre para executar o próximo fragmento.

O fragmento 3 então aciona a variável atual que representa o perfil do usuário. Então o recurso de áudio lerá os itens que constam na lista de medicamentos exibida.

### Detalhes do Medicamento

Após o usuário selecionar um medicamento da lista será exibida a tela de detalhes dos medicamentos. Nela encontra-se as principais funcionalidades que atendem o uso do medicamento: Registro de Utilização do Medicamento, compra de medicamento e horário de medicamentos.

### 

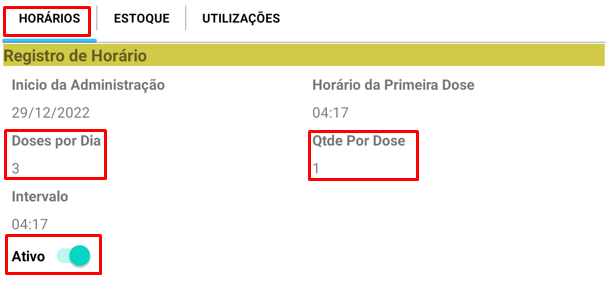
Primeiramente iremos fragmentar a tela em diversos pedaços e abordar o que cada funcionalidade e após isso abordar como a camada de serviço interage com as funcionalidades.



Funcionalidade disponível somente se o apelido ou a cor ou a quantidade de medicamentos por embalagem for alterada.



Funcionalidade sinaliza ao aplicativo que uma dose foi tomada pelo usuário do medicamento atual. São feitas algumas críticas para tornar o processo válido:

* Horário precisa está cadastrado e ativo. O fragmento de tela abaixo demonstra o cadastro ativo de um medicamento.
* 
* O saldo do medicamento precisa ser maior ou igual a quantidade de medicamento por dose.

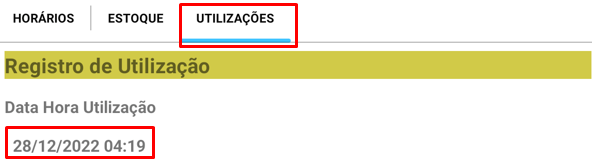
Após a validação o aplicativo obterá o ultimo saldo e criará um novo registro de saldo com o abatimento do saldo atual pela quantidade utilizada.

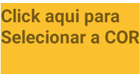
Também fará um registro com o horário da utilização. Os dois fragmentos da tela demonstram o que foi falado no texto acima.

O primeiro fragmento exibe o controle de saldo de estoque de um medicamento:

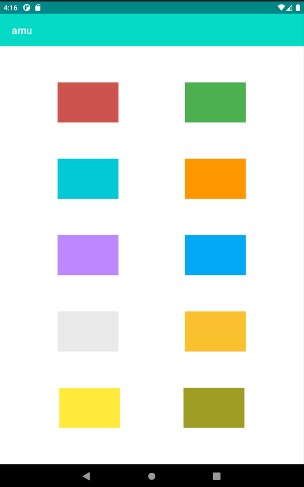


O segundo fragmento exibe o registro de uma utilização do medicamento:





Funcionalidade importante para usuários idosos já que a cor do medicamento pode ajudar a selecionar facilmente qual o medicamento. Ao clicar no botão uma paleta simples de cores será exibida. A lista de cores é reduzida para tornar mais simples o processo.

O usuário poderá apenas clicar em uma das cores e assim mudar a forma como é apresenta a linha na lista de medicamentos.



Funcionalidade que tem o objetivo de dar entrada no estoque de medicamentos. Com base no que foi informado na caixa de texto “estoque” será adicionado no saldo atual.



Funcionalidade que tem o objetivo de dar saída no estoque de medicamento. Com base no que foi informado na caixa de texto “estoque” será subtraído do saldo atual.

**Dicionário**

**ESB**

Um ESB, ou barramento de serviço corporativo, é um padrão pelo qual um componente de software centralizado realiza integrações a sistemas back-end (e conversões de modelos de dados, conectividade profunda, roteamento e solicitações) e disponibiliza essas integrações e conversões como interfaces de serviço para reutilização por novos aplicativos. O padrão ESB é geralmente implementado usando um tempo de execução de integração e um conjunto de ferramentas especialmente projetados que garantem a melhor produtividade possível.

**Strategy**

O Strategy é um padrão que deve ser utilizado quando uma classe possuir diversos algoritmos que possam ser utilizados de forma intercambiável. A solução proposta pelo padrão consiste em delegar a execução do algoritmo para uma instância que compõe a classe principal. Dessa forma, quando a funcionalidade for invocada, no momento de execução do algoritmo, será invocado um método da instância que a compõe. A Figura 1.4 apresenta um diagrama que mostra a estrutura básica do padrão.

**Referências**

1. Design Patterns com Java – Projeto orientado a objetos guiado por padrões
   * Pagina 233
2. Design Patterns com Java – Projeto orientado a objetos guiado por padrões.
   * Pagina 200
3. Autenticação no Utilizando Firebase
   * Acessado em 28/12/2022
   * <https://firebase.google.com/products/auth?gclid=CjwKCAiA76-dBhByEiwAA0_s9To21pNpQy3x9y79Uyl8YgNx3iwk0U0Ox8dx45IfKmPpNjXxG22eVhoCC60QAvD_BwE&gclsrc=aw.ds>
4. Uso de Polimorfismo em java
   * Acessado em 28/12/2022
   * <https://www.devmedia.com.br/uso-de-polimorfismo-em-java/26140>
5. Definição de Barramento de Serviços
   * Acessado em 07/12/2022
   * <https://www.ibm.com/br-pt/cloud/learn/esb#:~:text=Um%20ESB%2C%20ou%20barramento%20de,como%20interfaces%20de%20servi%C3%A7o%20para>
6. Definção de CallBack
   * Acessado em 30/12/2022
   * <https://www.jivochat.com.br/blog/ferramentas/o-que-e-callback.html>
   * <https://pt.stackoverflow.com/questions/27177/o-que-%C3%A9-callback>