**Universidade Estadual Paulista “Júlio de Mesquita Filho”**

**Ciência da Computação - Projeto da disciplina Engenharia de Software II**

Denis Akira - 161024181

João Vitor Lopes - 141021811

Paulo Hortelan - 141027037

**Documentação Appsico**

Para este projeto foi proposta a construção de um aplicativo móvel para uso em acompanhamentos terapêuticos na psicologia. O objetivo é auxiliar o terapeuta a melhor avaliar seus pacientes, podendo ter registros históricos da evolução deles. Para o paciente, as vantagens estão em ter um canal de comunicação dedicado para o tratamento e poder usar a plataforma para traçar objetivos em conjunto com o terapeuta.

Com o uso de uma ferramenta como esta, também torna-se possível contornar complicações como a lacuna de tempo entre uma sessão e outra, que muitas vezes pode ser longa, fazendo com que a qualidade do tratamento e precisão das avaliações, tanto do terapeuta como do paciente, sejam comprometidas.

**Índice**

1. Ferramentas e técnicas
2. Modelo da análise
3. Modelo de Projeto
4. Testes
5. Resultados e considerações finais

**Ferramentas e Técnicas**

Para o desenvolvimento do aplicativo foram escolhidas tecnologias web muito difundidas atualmente, com documentação detalhada e comunidade ativa, que favoreceu muito o processo de desenvolvimento.

São elas:

**Github**

Para versionamento do código, foi utilizado o Git e Github como repositório. Assim foi possível separar tarefas de desenvolvimento por membro do grupo e trabalhar paralelamente, ganhando tempo.

Com o uso dessa ferramenta, também foi possível seguir práticas de desenvolvimento como revisão de código e seguir um padrão de fluxo de adições de novas funcionalidades.

**Firebase**

Como backend decidimos utilizar o Firebase, que por ser uma opção gratuita, de fácil aprendizagem, que atende bem as necessidades do projeto e que ainda possui fácil integração com as outras tecnologias usadas, torna-se um serviço muito adequado para a construção de protótipos.

Firebase é uma tecnologia disponibilizada pelo Google, como um **B**ackend **A**s **A S**ervice - **BAAS** (Backend como um serviço). Ele possui uma estrutura de armazenamento de dados em NOSql numa estrutura similar à um JSON e traz uma série de funcionalidades e ferramentas (redefinição de senhas, logins, autenticação) que agilizam o processo de desenvolvimento, permitindo que a equipe desenvolvedora foque nas regras de negócio.

É disponibilizado um console gráfico para o moderador do banco de dados, onde é possível atribuir diversas configurações como permissões de usuários.

As opções de pagamento variam de acordo com o uso do firebase em número de usuários do aplicativo e capacidade de armazenamento utilizada. Até 100 acessos simultâneos são permitidos sem cobranças.

Outra vantagem do uso do Firebase é a geração automática de relatórios estatísticos sobre usuários e como eles interagem com a aplicação.

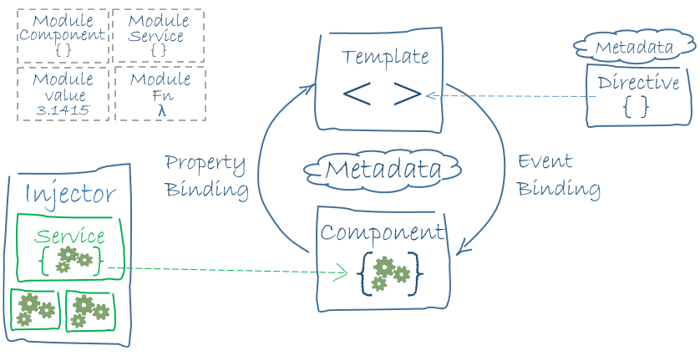
**Ionic 4**

Como framework de desenvolvimento foi escolhido o Ionic, por ser uma opção também gratuita e bem consolidada (versão 4). Sua curva de aprendizado não é muito alta porque utiliza-se tecnologias bem conhecidas como HTML e JavaScript.

O Ionic é um framework de desenvolvimento de aplicações mobile híbridas, que utiliza o Angular como base.

Este framework possui um style guide que pode ser acessado neste [link](https://angular.io/guide/styleguide). Resumidamente, trabalha-se com o padrão de arquitetura MVC, onde o model corresponde aos serviços, os componentes correspondem aos controllers e as views são tratadas como templates. Cada componente traz consigo a ideia de módulo, o que exige uma boa modelagem do sistema e sua arquitetura (<https://angular.io/guide/architecture>).

Existe a implementação de funcionalidades bastante úteis como property binding, que atualiza a view automaticamente de acordo com mudanças que ocorrem no model e são “percebidas” pelo controller associado à tal view. Similar à property binding existe o event binding, onde o controller escuta da view ações executadas pelo usuário e permite que seja dado o direcionamento correto ao fluxo de execução da aplicação.



**Modelo da Análise**

A ideia do projeto partiu de um estudante de psicologia, que percebeu a possibilidade de atrelar o acompanhamento psicológico à tecnologia, muito presente na vida cotidiana das pessoas. Existem soluções similares, porém muito mais focadas no paciente. Propõe-se aqui uma aplicação que possa ser utilizada também pelo terapeuta.

Entretanto, até o momento ainda não foram considerados aspectos legais, de ética e de privacidade que podem estar associados à disponibilização de uma solução como essa. Portanto, ainda é incerto se de fato, esta seria uma opção viável de ser posta em produção, mas foi decidido a construção de um protótipo que possa ser validado. Por conta dessas incertezas, neste protótipo constarão apenas funcionalidades voltadas para o paciente.

Para o sistema, foram definidas quatro grandes áreas que podem ser vistas como funcionalidades: controle de humor, pensamentos automáticos, cartões de enfrentamento e uma lista de tarefas.

As funcionalidades descritas a seguir, são conversadas e guiadas pelo profissional e paciente em conjunto, seguindo conceitos e abordagens da vertente TCC - Terapia Cognitivo Comportamental.

**Controle de Humor**

Nessa funcionalidade, o paciente deve ser capaz de definir uma “nota” para como está se sentindo no momento e salvar essa informação. Também deve ser possível acessar registros de humor salvos anteriormente.

**Pensamentos Automáticos**

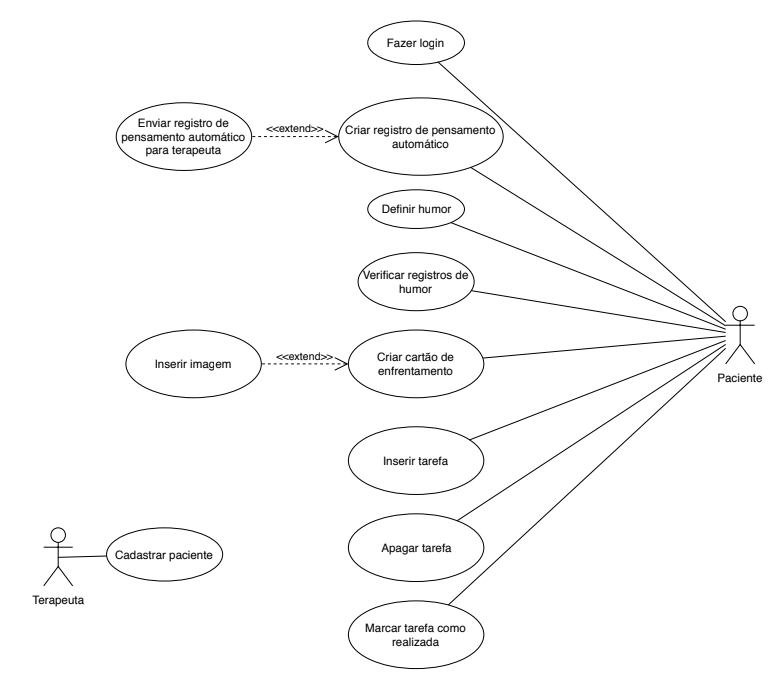
Nessa funcionalidade, o paciente deve ser capaz de registrar situações onde pôde perceber pensamentos automáticos. Para isso ele deve informar a situação em que ocorreu tal pensamento automático, qual foi o pensamento em si e as ações tomadas naquele momento. Armazenar tudo isso como um pensamento automático no banco de dados.

**Cartão de Enfrentamento**

Nessa funcionalidade, o paciente deve ser capaz de armazenar cartões visuais, juntamente de uma frase e imagem de sua escolha. Também deve ser possível ver os cartões salvos.

**Lista de Tarefas**

Aqui, terapeuta e paciente definem tarefas com data limite que devem ser realizadas pelo paciente. O usuário deve ser capaz de adicionar uma nova tarefa à lista, remover uma tarefa da lista, editar uma tarefa da lista e marcar uma tarefa como realizada.

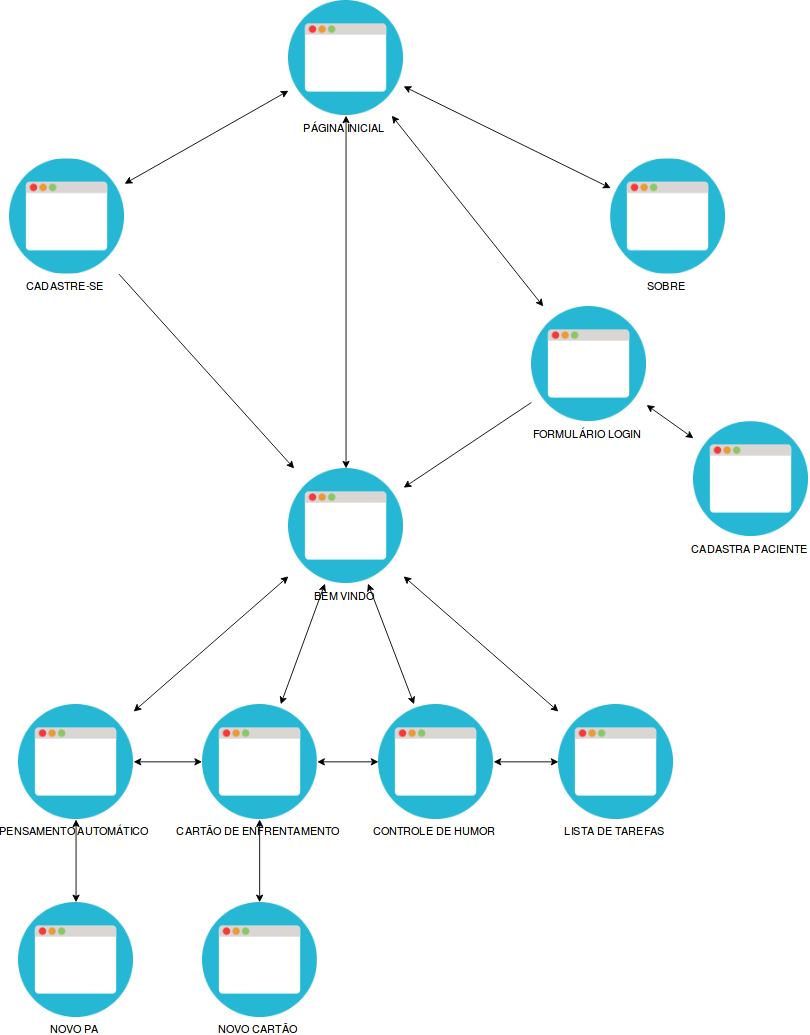


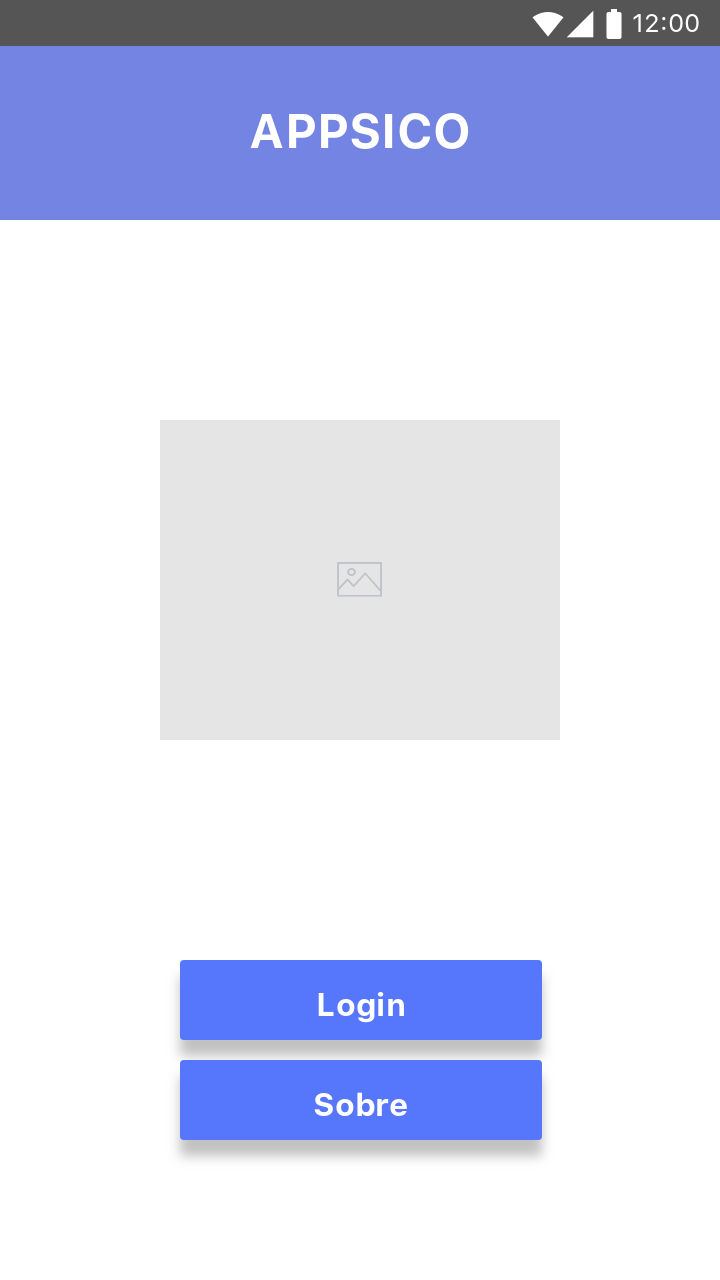
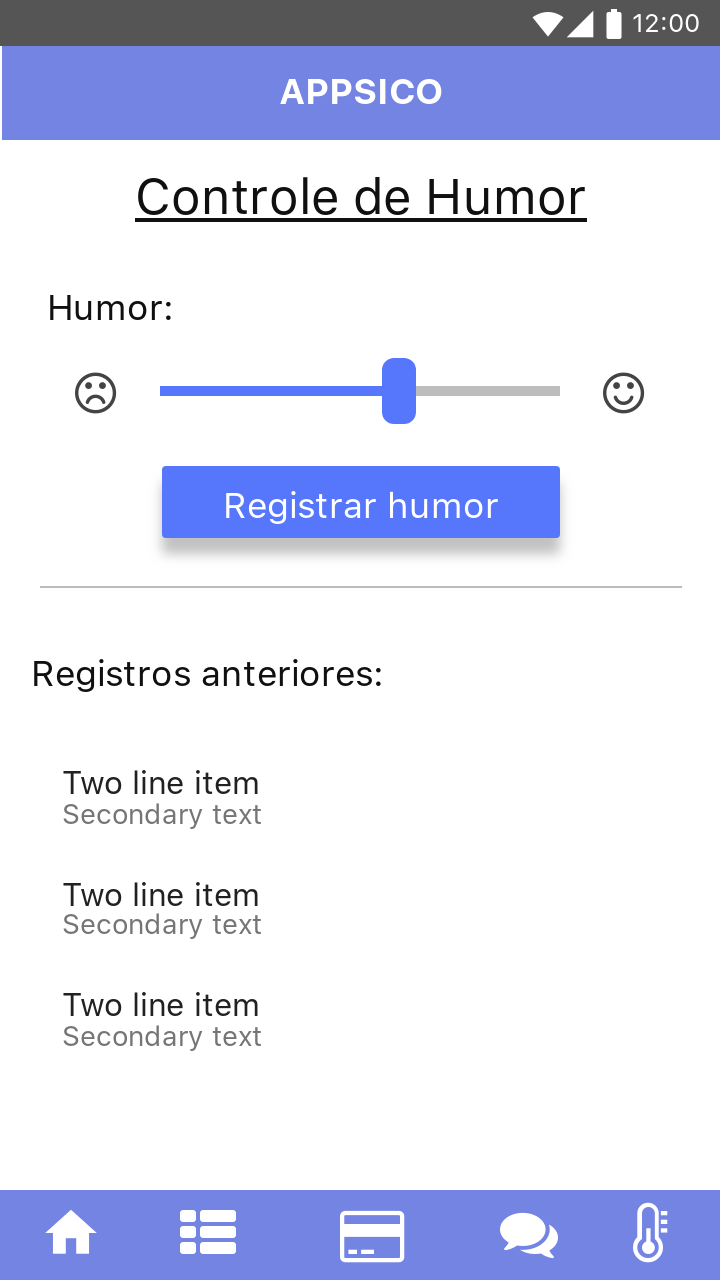
**Modelo de Projeto**

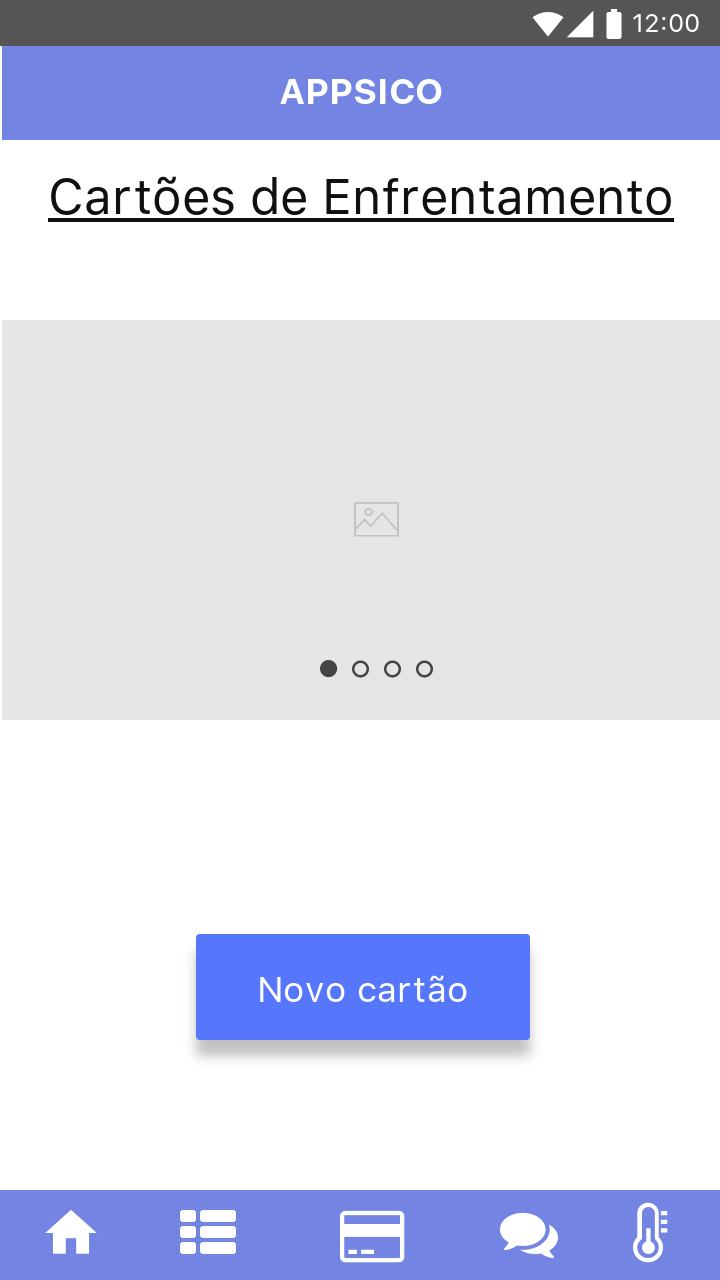
**Interface**

Para o design de interfaces do projeto, foi utilizada o Mockinbot como ferramenta de prototipação de telas por ser uma ferramenta com planos gratuitos e que permite muita flexibilidade. O design escolhido foi o de abas na parte inferior da tela, muito utilizado em aplicações com poucas funcionalidades, facilitando a navegação e com visual agradável.

Segue o fluxo de navegação do aplicativo e alguns designs de telas:

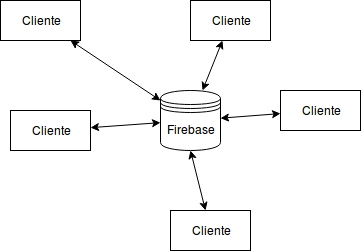


**Arquitetura**

Em termos gerais de arquitetura de dados, o funcionamento geral do sistema será dado por escritas e leituras no banco de dados. Por isso utilizamos a arquitetura centrada em dados, onde múltiplos clientes podem se conectar ao firebase e as alterações feitas por cada um deles são propagadas à rede em tempo real.



Na aplicação em si, a arquitetura foi muito guiada pelo tipo de interface e framework de desenvolvimento escolhidos. Cada uma das abas representa uma funcionalidade ou módulo, que são desenvolvidos visando alta coesão. Classes de suporte foram construídas para serviços de autenticação de usuário, que precisa estar compartilhado entre todos os componentes, por exemplo.

**Banco de Dados**

Para o banco de dados Firebase, foi necessário mapear no problema as entidades e relacionamentos entre elas, e em seguida, modelá-las no padrão NoSQL.

A modelagem do banco de dados não relacional deve seguir algumas técnicas que visam otimizar a performance das queries:

* É recomendado evitar o aninhamento muito profundo de nós (objetos) no json. Isso evita que dados desnecessários sejam retornados nas queries. Ao buscar dados no documento, todos os nós filhos são recuperados também.
* É recomendado um processo de desnormalização para que as queries sejam eficientes. Isso traz consequências como dados duplicados e exige um tratamento cuidadoso de inserções e exclusões de dados para que não surjam inconsistências.

Foram mapeados os seguintes atributos de entidades e relacionamentos:

* Terapeuta: nome, email, senha e lista de pacientes.
* Paciente: nome, email, senha, terapeuta, lista de tarefas, lista de pensamentos automáticos, lista de cartões de enfrentamento e lista de registros de humor.
* Pensamento automático: pensamento, ação tomada e situação do ocorrido.
* Humor: valor numérico que representa o humor do paciente.
* Cartão de enfrentamento: imagem e texto para o cartão.
* Tarefa: título, descrição, boolean que indica se foi realizada ou não e data limite.

**Componentes**

Os componentes do sistema foram facilmente identificados pela descrição do problema.

**Página Inicial**

* Descrição: Será a tela apresentada à todos os usuários. Daqui pode-se ir para 3 outras páginas (entrar, cadastre-se e sobre) por meio de botões.
* Serviços: Aqui deve ser criada uma instância do Firebase para ser usada ao longo do app.
* Outros: Nenhum.

**Página Sobre**

* Descrição: Página que apresentará dados da aplicação que podem ser relevantes aos usuários.

**Página Cadastre-se**

* Descrição: Página onde um usuário pode se cadastrar através de um formulário. Para tal cadastro será necessário que o terapeuta tenha definido que o acesso deste paciente é válido.

O formulário deverá ser validado (botão de submit se torna “clicável” apenas se a validação estiver correta).

* Serviços: Ao tentar realizar o cadastro, é necessário que se verifique se o email informado para sign up está associado à algum terapeuta.
* Serviços: Armazenar o usuário no firebase.
* Outros: Se o paciente não estiver liberado pelo terapeuta, deverá ser emitido um modal com uma mensagem de erro.

**Página Entrar**

* Descrição: Nesta página há o formulário de login que deverá ser validado (botão de submit se torna “clicável” apenas se a validação estiver correta).
* Serviços: Verificar se tal usuário existe no firebase e tentar autenticá-lo.
* Serviços: Recuperar o token de autenticação do usuário para que ele possa realizar as ações desejadas no app.
* Outros: Modal informando erros possíveis (usuário não existente, senha incorreta).

**Página Cadastrar Paciente**

* Descrição: Componente dedicado ao terapeuta, cuja única finalidade é, cadastrar o paciente que concordou em utilizar o aplicativo no tratamento através de um endereço de email válido (botão de submit se torna “clicável” apenas se o email passar na validação). Em versões futuras este módulo poderá se tornar um painel administrativo para os terapeutas.
* Serviços: Armazenar o endereço de email do paciente no firebase e associá-lo ao terapeuta.
* Outros: Modals informando erros como email já cadastrado.

**Página Bem Vindo**

* Descrição: Componente de apresentação da ferramenta ao usuário (pacientes). Aqui é possível que o paciente possa ver uma prévia da lista de tarefas definida conjuntamente por ele e o terapeuta em sessão. É possível também enviar uma mensagem via email ao terapeuta (apenas mensagem validada, não vazia). É possível também navegar para outras páginas através de abas.
* Serviços: Envio de mensagem ao terapeuta e busca das tarefas com prazo mais próximo da data atual para visualização.

**Página Pensamento Automático**

* Descrição: Neste módulo é possível ver uma lista do histórico de pensamentos automáticos salvos pelo usuário e registrar um novo PA (formulário só pode ser submetido se passar pela validação dos campos).
* Serviços: Recuperar do firebase os x últimos PAs associados ao usuário.
* Serviços: Armazenar no firebase um novo PA associado ao usuário.
* Outros: A criação de um novo PA é feita através de um modal.

**Página Cartão de Enfrentamento**

* Descrição: Neste módulo é possível visualizar uma lista de cartões de enfrentamento salvos pelo usuário e criar um novo CE (validação).
* Serviços: Recuperar do firebase os x últimos CEs associados ao usuário.
* Serviços: Armazenar no firebase um novo CE associado ao usuário.
* Serviços: Armazenar no firebase storage a imagem escolhida pelo usuário (câmera ou arquivo) e associá-la ao CE.
* Outros: Nesse componente do sistema, foram usados plugins para habilitar o uso de recursos nativos do hardware, como câmera.
* Outros: Modals para criação do cartão, falha ao acessar câmera, falha no armazenamento do CE.

**Página Controle de Humor**

* Descrição: Página onde será possível definir um “range” que corresponde ao humor do usuário (bom/ruim) e salvá-lo. Visualizar registros de humor salvos.
* Serviços: Recuperar do firebase registros de humor associados ao usuário.
* Serviços: Armazenar um novo registro de humor.
* Outros: Modal de erros de acesso ao firebase.

**Página Lista de Tarefas**

* Descrição: Página onde deverá haver uma lista de tarefas à serem combinadas entre paciente e terapeuta. O usuário pode marcar como realizada, excluir ou adicionar uma tarefa (validada).
* Serviços: Recuperar tarefas associadas ao usuário.
* Serviços: Armazenar uma nova tarefa.
* Outros: Modals de erros como data limite para realização da tarefa inválida.

**Testes**

Os testes unitários da aplicação foram simples e executados na construção dos módulos, na fase de codificação pelo desenvolvedor responsável pelo módulo. No momento de juntar um módulo feito localmente ao repositório remoto, foi usada uma política de revisão de código onde um desenvolvedor deve aprovar o recebimento da nova funcionalidade.

Ao final do desenvolvimento dos módulos, foi executada uma revisão de integração entre módulos para então partir para a geração de um executável. Todos esses passos foram facilitados pelo uso do Git.

Uma vez que a aplicação é um protótipo simples e com fluxos de execução curtos, os testes resumiram-se basicamente a validar formulários e a verificar o envio de informações ao Firebase.

**Resultados e Considerações Finais**

Com este protótipo torna-se possível a validação da ideia do aplicativo com uma amostra de usuários. Pode-se avaliar a satisfação deles em relação à diversos aspectos como fluxo de navegação, facilidade de uso, desempenho. Com essas informações, a viabilidade do projeto e do modelo de negócios pode ser estudada.

Do ponto de vista acadêmico, foi um projeto essencial onde pudemos ter uma visão macro do processo de desenvolvimento de software. Avaliando questões de desenvolvimento, processos, gestão de equipe e de negócios.

Como próximos passos, é necessário corrigir possíveis erros que passaram despercebidos e considerar a construção de uma área de administrador para uso dos terapeutas.