	PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
	Disciplina: Laboratório de Desenvolvimento de Aplicações Móveis e Distribuídas
	Professor: Cleiton Tavares e Ilo Amy Saldanha Rivero
	Curso: Engenharia de Software **Semestre:** 2/2023

NOME DO PROJETO: SQUAD PLANNER

INTEGRANTES E RESPONSABILIDADES

Bárbara Menezes (Desenvolvedora Designer e responsável pela documentação)
Eduardo Guimarães (Analista)
Nando Tupinambá (Analista)

PROPOSTA DE DESENVOLVIMENTO

Objetivo do App:

O app tem como objetivo simplificar e organizar o planejamento de encontros entre amigos, permitindo que membros criem e gerenciem eventos, confirmem presença, visualizem os próximos encontros, etc. Trata-se basicamente de um “calendário compartilhado”, mas em formato diferente e com layout mais atrativo.

Público alvo do app:

Quaisquer grupos de pessoas que se encontrem com certa recorrência (amigos, famílias, colegas de trabalho etc)

Principais funcionalidades:

Visualizar próximos encontros
Receber lembretes sobre encontros próximos
Criar novos encontros (com local, data, hora, comentários, etc), bem como gerenciá-los (apagar ou editar)
Confirmar/cancelar presença em eventos criados

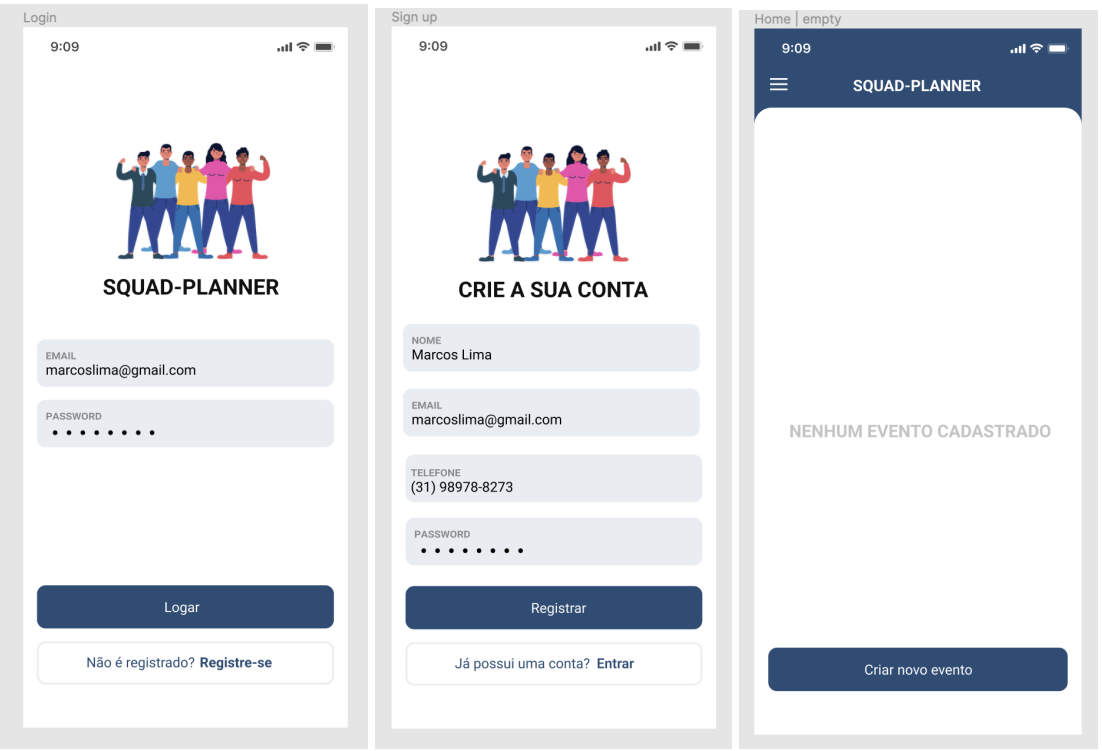
DETALHAMENTO DAS FUNCIONALIDADES

Identificador	Funcionalidade	Categoria
FP-1	Cadastrar usuário no sistema	Crítico
FP-2	Editar perfil de usuário no sistema	Útil
FP-3	Realizar login do usuário no sistema	Crítico
FP-4	Cadastrar evento	Crítico
FP-5	Editar evento	Importante

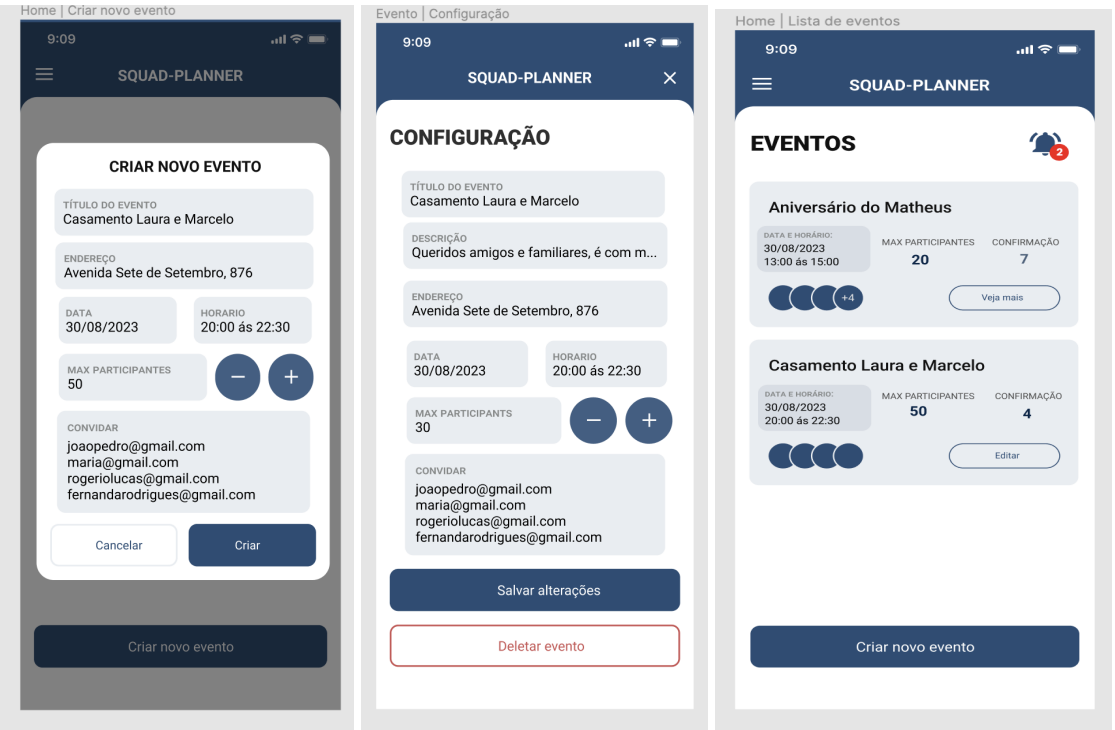
FP-6	Deletar evento	Importante
FP-7	Visualizar lista de eventos	Crítico
FP-8	Confirmar presença no evento	Crítico
FP-9	Cancelar presença no evento	Crítico
FP-10	Receber notificação de evento	Crítico

a) Rascunho das telas do app:

i) Respectivamente: tela de login, tela de criação de conta e tela inicial:



ii) Respectivamente: tela de criação do evento, de edição de evento e lista de eventos



ARQUITETURA DA APLICAÇÃO

A arquitetura do aplicativo Squad Planner é projetada para proporcionar uma experiência eficiente e colaborativa na organização de eventos entre amigos. A estrutura do aplicativo é baseada em princípios sólidos de desenvolvimento Flutter, seguindo uma abordagem modular e escalável. A arquitetura é composta por diversos componentes inter-relacionados, cada um desempenhando um papel específico para garantir a funcionalidade e desempenho ideais.

Frontend Flutter, Backend Firebase, Serviços de Notificação, SQLite, API.

PLATAFORMA DE HARDWARE

Dispositivos Móveis: A aplicação Squad Planner será destinada a dispositivos móveis. Portanto, os desenvolvedores devem ter acesso a smartphones ou tablets para testar e depurar o aplicativo nas plataformas iOS e Android.
Computadores para Desenvolvimento: Para o desenvolvimento do aplicativo, os desenvolvedores precisarão de computadores pessoais, que podem ser Mac, Windows ou Linux, dependendo das preferências da equipe.

PLATAFORMA DE SOFTWARE

Plataforma de Software:
Flutter SDK: O Flutter SDK é a base para o desenvolvimento do aplicativo. Certifique-se de que todos os membros da equipe tenham o Flutter SDK instalado em seus ambientes de desenvolvimento.
Ambiente de Desenvolvimento Integrado (IDE): Visual Studio e Android Studio.
Github: Para gerenciamento colaborativo do código-fonte. Plataformas como GitHub, GitLab ou Bitbucket podem ser usadas para hospedar repositórios Git.
Plataforma de Gerenciamento de Projeto: Trello para coordenar as tarefas, atribuir responsabilidades e rastrear o progresso.

Banco de Dados (para Desenvolvimento Local): SQLite.
Banco de Dados (para Desenvolvimento em Nuvem): Firebase.
Ferramentas de Design: Figma para criação de telas.

NECESSIDADE API

- Integração com Serviços em Nuvem.
- Segurança e Controle de Acesso.

REQUISITOS FRONT-END

Requisitos Funcionais:

- O usuário deve ser capaz de ver seus próximos eventos.
- O usuário deve ser capaz de criar novos eventos e editar eventos já criados.
- O usuário deve ser capaz de visualizar os grupos nos quais faz parte.
- O usuário deve ser capaz de criar novos grupos e editar grupos já existentes.
- O usuário deve ser capaz de editar suas informações pessoais.
- O usuário deve ser capaz de aceitar convites para eventos.

Requisitos Não Funcionais:

- As informações primárias do sistema devem ser carregadas em não mais do que 3 segundos

REQUISITOS BACK-END

Requisitos Não Funcionais:

- O sistema deve ter disponibilidade de 99,99%
- O sistema deve verificar os usuários a cada 6 meses.

SERVIÇOS LOCAIS

Desenvolvimento Local: ambiente de desenvolvimento Flutter, para criar e testar a interface do usuário e a lógica de frontend.
Banco de Dados Local: SQLite.

PROCESSAMENTO

Frontend (Flutter): O processamento no lado do cliente ocorre principalmente no dispositivo dos usuários, com a execução do código Flutter. Garantir que o código seja otimizado para oferecer uma experiência fluida em uma variedade de dispositivos móveis.
Backend (Firebase Cloud Functions): As Cloud Functions do Firebase realizam processamento no lado do servidor em resposta a eventos específicos, como o cálculo da divisão de despesas. Garantir que o código seja eficiente e dimensionado conforme a demanda do aplicativo.

MEMÓRIA

Frontend (Flutter): Alocar memória de forma eficiente para armazenar dados temporários e otimizar o desempenho da interface do usuário. Gerenciar o ciclo de vida dos objetos para evitar vazamentos de memória.
Backend (Firebase Firestore, Firebase Cloud Functions e SQLite): Durante operações de leitura e gravação de dados no Firestore, é essencial gerenciar a memória de forma eficaz. No contexto das Cloud Functions, garantir que a memória alocada seja suficiente para a execução eficiente das operações. Além disso, ao incorporar SQLite como um banco de dados local, é

crucial otimizar consultas e garantir uma gestão eficiente da memória para melhorar o desempenho do backend.

ARMAZENAMENTO

Firebase Cloud Storage: Gerenciar o armazenamento de mídias, como imagens relacionadas aos eventos. Monitorar e ajustar a capacidade de armazenamento conforme necessário.

SERVIÇOS EM NUVEM

Armazenamento em nuvem: Utiliza armazenamento em nuvem Google Cloud e Firebase.

DIAGRAMA DA ARQUITETURA