Documentação do Projeto: Aplicativo de Rastreamento de Hábitos

1. Introdução

Este documento serve como a documentação oficial do projeto para o aplicativo de rastreamento de hábitos. Ele visa fornecer uma visão abrangente do aplicativo, desde sua concepção e funcionalidades até as diretrizes de design, aspectos técnicos e estratégias de teste. O objetivo é servir como um recurso central para desenvolvedores, designers e qualquer pessoa envolvida no projeto, garantindo clareza, consistência e facilidade de manutenção.

2. Visão Geral do Aplicativo

a) Nome do Aplicativo

Habit Tracker (Nome provisório)

b) Propósito e Objetivo

O aplicativo Habit Tracker tem como propósito auxiliar os usuários na criação, manutenção e monitoramento de hábitos saudáveis e produtivos. Seu objetivo principal é fornecer uma ferramenta simples e intuitiva que motive os usuários a alcançar seus objetivos pessoais através da consistência diária.

c) Público-Alvo

Indivíduos que desejam:

- Desenvolver novos hábitos (ex: beber mais água, ler diariamente).
- Manter hábitos existentes (ex: exercitar-se regularmente, meditar).
- Monitorar seu progresso e identificar padrões em suas rotinas.
- Pessoas que buscam melhorar a produtividade e o bem-estar pessoal.

d) Funcionalidades do MVP (Produto Mínimo Viável)

A primeira versão do aplicativo (MVP) incluirá as seguintes funcionalidades essenciais:

- **Criação e Personalização de Hábitos:** Permite ao usuário adicionar novos hábitos, nomeá-los e definir sua frequência.
- **Definição de Frequência:** Suporte para hábitos diários, em dias específicos da semana ou X vezes por semana.
- **Marcação de Conclusão (Check-in):** Interface simples para marcar um hábito como concluído no dia.
- **Visualização do Progresso:** Exibição clara dos hábitos e seu status de conclusão para o dia atual, incluindo a contagem de sequências (streaks).
- **Lembretes/Notificações:** Capacidade de configurar lembretes para os hábitos em horários específicos.
- **Configurações Básicas:** Opções para gerenciar notificações.

3. Diretrizes de Design (UI/UX)

As diretrizes de design visam garantir uma experiência de usuário consistente, intuitiva e visualmente agradável. Elas abrangem:

a) Princípios de Design

- **Simplicidade e Clareza:** Interface limpa, minimalista e fácil de entender.
- **Motivação e Feedback Positivo:** Feedback visual imediato, celebração de conquistas e progresso visível.
- **Acessibilidade:** Contraste adequado, tamanhos de toque apropriados e suporte a leitores de tela.

b) Paleta de Cores

- **Cores Principais:** Verde (#4CAF50) para sucesso, Azul (#2196F3) para ações primárias, Cinza claro (#F5F5F5) para fundos, Cinza escuro (#424242) para textos.
- **Cores de Apoio:** Laranja (#FF9800) para lembretes, Vermelho suave (#F44336) para alertas, Roxo (#9C27B0) para gamificação.

c) Tipografia

- **Fonte Principal:** Roboto (Android) ou SF Pro (iOS).
- **Hierarquia de Textos:** Definida por tamanhos e pesos para guiar a leitura.

d) Estrutura de Navegação

- **Navegação Principal:** Bottom Tab Bar com as seções: Home/Hoje, Hábitos, Progresso e Configurações.

e) Layouts de Tela (Mockups)

- **Tela Principal (Hoje):** Exibição dos hábitos diários com status de conclusão e sequências.
- **Tela de Adicionar Hábito:** Formulário para criação e personalização de novos hábitos.
- **Tela de Progresso:** Visualização de estatísticas e desempenho dos hábitos.

f) Componentes de Interface, Animações e Responsividade

- **Componentes:** Cartões de hábito, botões, indicadores de progresso e ícones padronizados.
- **Animações:** Feedback de conclusão, transições de tela suaves e estados de loading.
- **Responsividade:** Adaptação a diferentes tamanhos de tela e orientações (retrato/paisagem).

4. Aspectos Técnicos

a) Tecnologia Escolhida

- **Framework:** Flutter
- **Linguagem:** Dart
- **Justificativa:** Facilidade de uso, Hot Reload, UI expressiva, desempenho próximo ao nativo e comunidade crescente, tornando-o ideal para desenvolvedores iniciantes.

b) Estrutura do Projeto

- `lib/`: Código-fonte Dart (`main.dart`, `models/`, `providers/`, `screens/`).
- `android/` e `ios/`: Arquivos de projeto nativos.
- `pubspec.yaml`: Gerenciamento de dependências (`provider`, `uuid`, `shared_preferences`).

c) Gerenciamento de Estado e Persistência de Dados

- **Gerenciamento de Estado:** `Provider` para gerenciar a lista de hábitos e notificar a UI sobre mudanças.
- **Persistência de Dados (MVP):** `shared_preferences` para armazenar os dados dos hábitos localmente no dispositivo. Para futuras versões, pode-se considerar SQLite ou Firebase Firestore.

d) Modelo de Dados

- **Classe `Habit`:** Define a estrutura de um hábito, incluindo nome, frequência, lembretes, status de conclusão e datas de conclusão. Inclui métodos para marcar como concluído, verificar status e calcular sequências.

5. Estratégia de Testes

Uma abordagem de testes em camadas será utilizada para garantir a qualidade do aplicativo.

a) Tipos de Testes

- **Testes de Unidade:** Para validar a lógica de negócios isoladamente (ex: cálculo de sequência, métodos de modelo).
- **Testes de Widget:** Para verificar o comportamento da UI e a interação do usuário com componentes específicos (ex: renderização de hábitos, marcação de conclusão).
- **Testes de Integração:** Para testar fluxos completos do aplicativo em um ambiente real (ex: adicionar hábito, marcar, verificar persistência).

b) Priorização e Ciclo de Testes (MVP)

- **Prioridade:** Alta para testes de unidade da lógica de negócios e testes de widget das telas principais. Média para testes de integração de fluxos críticos.
- **Ciclo:** Desenvolvimento com testes de unidade/widget, testes manuais em emulador/dispositivo, testes de integração e testes de regressão contínuos.

c) Casos de Teste Essenciais

- **Criação de Hábitos:** Adicionar com e sem nome, definir frequência.
- **Marcação de Conclusão:** Marcar/desmarcar hábitos.

- **Visualização de Progresso:** Verificação de sequências e exibição de hábitos diários.
- **Lembretes:** Configuração e recebimento de notificações.

6. Próximos Passos e Considerações Futuras

Após a conclusão do MVP, as seguintes áreas podem ser exploradas para futuras versões do aplicativo:

- **Implementação de Telas Adicionais:** Tela de Hábitos (lista completa), Tela de Progresso (gráficos e estatísticas detalhadas), Tela de Configurações completas.
- **Persistência de Dados Avançada:** Migração para SQLite ou Firebase para maior escalabilidade e sincronização.
- **Notificações Locais Avançadas:** Utilização de `flutter_local_notifications` para agendamento de notificações robustas.
- **Gamificação:** Adição de pontos, níveis, distintivos e desafios para aumentar o engajamento.
- **Recursos Sociais:** Compartilhamento de progresso (com opção de privacidade).
- **Widgets:** Para acesso rápido ao status dos hábitos na tela inicial do dispositivo.
- **Backup e Restauração:** Funcionalidade para salvar e restaurar dados do usuário.

7. Glossário

- **MVP (Minimum Viable Product):** Produto Mínimo Viável. A versão de um novo produto que possui apenas recursos suficientes para satisfazer os primeiros clientes e fornecer feedback para o desenvolvimento futuro do produto.
- **UI (User Interface):** Interface do Usuário. O que o usuário vê e interage no aplicativo (botões, telas, textos).
- **UX (User Experience):** Experiência do Usuário. Como o usuário se sente ao usar o aplicativo (facilidade, satisfação, eficiência).
- **Flutter:** Um kit de desenvolvimento de UI de código aberto criado pelo Google para construir aplicativos compilados nativamente para celular, web e desktop a partir de um único código-base.
- **Dart:** Linguagem de programação otimizada para UI, desenvolvida pelo Google, usada no Flutter.
- **Provider:** Um pacote de gerenciamento de estado para Flutter, simples e escalável.
- **Shared Preferences:** Um pacote Flutter para armazenar dados simples de chave-valor localmente no dispositivo.
- **Streak:** Sequência. O número de dias consecutivos em que um hábito foi realizado.

8. Referências

[1] Flutter Documentation. Disponível em: https://flutter.dev/docs [2] Provider Package. Disponível em: https://pub.dev/packages/provider [3] Shared Preferences Package. Disponível em: https://pub.dev/packages/shared_preferences [4] UUID Package. Disponível em: https://pub.dev/packages/uuid