Requisitos Funcionais (RF)

1. **RF1 - Criando a conta (Cadastro Inicial):**
    a. O sistema deve exibir uma explicação inicial sobre o que é o SuperID (pode ser um tour, vídeo ou texto simples, a critério da equipe).
    b. O sistema deve permitir que o usuário aceite os termos de uso (termos a serem definidos pela equipe).
    c. O sistema deve possibilitar o cadastro do usuário com Nome, Email e Senha Mestre.
    d. O sistema deve validar o email do usuário utilizando o Firebase Authentication.
    e. O sistema deve salvar o UID (identificador único do usuário) e o IMEI do dispositivo no Firebase Firestore após a criação da conta.
2. **RF2 - Manter banco de dados de senhas de acesso (Gerenciamento de Senha):**
    a. O sistema deve permitir que o usuário cadastre, altere e exclua senhas pessoais após o login.
    b. O sistema deve organizar as senhas em categorias, incluindo as padrão (Sites Web, Aplicativos, Teclados de Acesso Físico), sendo "Sites Web" obrigatória e não excluível, enquanto o usuário pode criar categorias adicionais.
    c. O sistema deve gerar um accessToken de 256 caracteres em Base64 para cada senha cadastrada.
    d. O sistema deve criptografar as senhas antes de armazená-las no Firebase Firestore (o método de criptografia é responsabilidade da equipe).
3. **RF3 - Usuário acessa um site parceiro para fazer o Login sem Senha:**
    a. O sistema deve permitir que sites parceiros ofereçam a opção de login via SuperID.
    b. O sistema deve possibilitar que o site parceiro faça uma requisição à Firebase Function performAuth, enviando sua URL e API Key, e receba um QRCode contendo um loginToken de 256 caracteres em Base64.
    c. O sistema deve permitir que o usuário escaneie o QRCode com o aplicativo SuperID, confirmando a autenticação após inserir a Senha Mestre (se o app não estiver aberto).
    d. O sistema deve atualizar o documento na coleção login do Firebase Firestore com o UID do usuário e a data/hora da autenticação.
    e. O sistema deve permitir que o site consulte o status do login via Firebase Function getLoginStatus, informando o loginToken, com limite de até 3 consultas em 1 minuto, após o qual o token expira.
4. **RF4 - Recuperação de Senha Mestre:**
    a. O sistema deve permitir que o usuário redefina a Senha Mestre via email, utilizando os recursos do Firebase Authentication.
    b. O sistema deve garantir que a troca de senha só seja possível se o email do usuário tiver sido previamente validado.
    c. O sistema deve exibir uma indicação visível no aplicativo caso o email ainda não tenha sido validado.

Requisitos Não Funcionais (RNF)

Os requisitos não funcionais foram extraídos das especificações técnicas, regras de implementação e expectativas implícitas do documento.

1. **RNF1 - Segurança:**
    a. O sistema deve criptografar todas as senhas antes de armazená-las no Firebase Firestore (o algoritmo de criptografia deve ser escolhido pela equipe).
    b. O sistema deve garantir que a recuperação ou troca da Senha Mestre só ocorra com um email validado.
    c. O sistema deve restringir o uso do loginToken a 3 consultas em até 1 minuto, expirando-o após esse período.
2. **RNF2 - Usabilidade:**
    a. O sistema deve apresentar uma explicação inicial clara e útil sobre o SuperID (tour, vídeo ou texto).
    b. O sistema deve oferecer uma interface intuitiva para aceitação dos termos de uso.
    c. O sistema deve exibir continuamente um aviso visível caso o email do usuário não esteja validado.
3. **RNF3 - Desempenho:**
    a. O sistema deve gerar tokens (accessToken e loginToken) de forma eficiente.
    b. O sistema deve processar a autenticação via QRCode e responder às consultas de status de login (getLoginStatus) em poucos segundos.
4. **RNF4 - Compatibilidade:**
    a. O sistema deve ser desenvolvido nativamente para Android utilizando a linguagem Kotlin no Android Studio.
    b. O sistema deve integrar-se obrigatoriamente com Firebase Authentication (gerenciamento de contas), Firebase Firestore (banco de dados) e Firebase Functions (backend).
5. **RNF5 - Documentação:**
    a. O sistema deve incluir um arquivo README.md no repositório GitHub com instruções detalhadas de instalação e uso.
    b. O código-fonte deve conter comentários explicativos para facilitar a compreensão por terceiros.
6. **RNF6 - Gerenciamento de Projeto:**
    a. O código-fonte deve ser armazenado em um repositório GitHub com o nome igual ao do grupo sorteado no Canvas.
    b. O repositório deve seguir a estrutura de branches: main, develop e feature/nome-da-funcionalidade.
    c. O release final deve ser entregue com uma TAG no GitHub, nomeada como 1.0-Final, até 01 de junho de 2025.
    d. As tarefas do projeto devem ser organizadas utilizando a ferramenta GitHub Projects, incluindo registro de horas trabalhadas.
    e. O time deve realizar reuniões periódicas com o professor responsável pelo acompanhamento.
7. **RNF7 - Acesso:**
    a. O repositório GitHub deve conceder acesso aos professores Mateus Dias (mateus.dias@puc-campinas.edu.br), Renata Arantes (renata.arantes@puc-campinas.edu.br) e Luã Marcelo (lua.marcelo@puccampinas.edu.br) para avaliação.

8. **RNF8 - Avaliação:**
    a. A apresentação do projeto em uma banca avaliadora é obrigatória para todos os membros da equipe, com reprovação imediata em caso de ausência injustificada (conforme regimento interno da instituição).