### ****Proposta de Organização da Configuração do SIGA Mobile****

#### ****1. Estrutura do Sistema****

A estrutura será composta por três camadas principais:

* **Front-end**: Interface do usuário no aplicativo móvel. Responsável por exibir as informações e captar as interações dos usuários.
* **Back-end**: Processamento lógico e comunicação com o banco de dados. Garante o funcionamento das funcionalidades principais e a segurança.
* **Banco de Dados**: Armazena as informações do sistema, como dados dos alunos, histórico acadêmico, solicitações e notificações.

#### ****2. Módulos Funcionais****

O sistema será dividido em módulos correspondentes às funcionalidades principais, garantindo organização e fácil manutenção:

1. **Login e Autenticação**
   * Função: Validar as credenciais dos alunos para acesso ao sistema.
   * Dependência: Back-end para verificação de credenciais.
2. **Calendário de Provas e Atividades**
   * Função: Exibir datas importantes.
   * Dependência: Banco de dados para buscar informações do calendário acadêmico.
3. **Matrícula Remota**
   * Função: Permitir a inscrição ou rematrícula do aluno em disciplinas.
   * Dependência: Back-end para processar matrículas e verificar disponibilidade de vagas.
4. **Solicitações**
   * Função: Registrar pedidos de revisão, documentos, regime domiciliar, entre outros.
   * Dependência: Comunicação com o back-end para registrar solicitações e com a administração para análise e resposta.
5. **Notificações e Avisos**
   * Função: Enviar alertas automáticos e atualizações.
   * Dependência: Complementar ao calendário e às solicitações, sendo gerido pelo back-end para envio programado.
6. **Plano de Ensino**
   * Função: Exibir conteúdos das disciplinas e aulas planejadas.
   * Dependência: Back-end para consultar informações da disciplina.
7. **Gerenciar Segurança da Conta**
   * Função: Alterar e-mail, senha e configurar opções de segurança.
   * Dependência: Back-end para validação e aplicação das alterações.
8. **Visualizar Histórico Escolar**
   * Função: Apresentar as notas e desempenho do aluno.
   * Dependência: Banco de dados para recuperar informações acadêmicas.
9. **Consultar Horários de Aula**
   * Função: Exibir a grade horária atual.
   * Dependência: Banco de dados para buscar horários atribuídos às disciplinas.

#### ****3. Organização da Arquitetura do Sistema****

A arquitetura será baseada no modelo **MVC (Model-View-Controller)** para modularidade e escalabilidade:

* **View**: Interface do aplicativo construída em tecnologias como React Native ou Flutter.
* **Controller**: Regras de negócio implementadas no back-end para processar solicitações e validar dados.
* **Model**: Representação dos dados armazenados no banco de dados, como informações de alunos, disciplinas e notificações.

#### ****4. Fluxo de Dados****

1. O aluno faz login e autentica suas credenciais.
2. Após o login, o sistema libera acesso às funcionalidades baseadas no perfil do usuário.
3. O aluno interage com os módulos (ex.: solicitações, calendário, histórico escolar).
4. O sistema armazena as interações e retorna feedback (ex.: matrícula confirmada ou solicitação em análise).
5. Notificações automáticas são geradas para manter o aluno informado sobre atualizações.

#### ****5. Configurações Gerais****

1. **Controle de Acesso**:
   * Alunos têm acesso restrito às funcionalidades acadêmicas e suas informações pessoais.
   * A administração tem acesso a ferramentas de análise e gerenciamento de solicitações.
2. **Segurança**:
   * Dados sensíveis (ex.: senha) serão criptografados.
   * Implementação de autenticação em duas etapas para segurança adicional.
3. **Escalabilidade**:
   * Banco de dados e back-end configurados para lidar com alta demanda de acessos simultâneos.
4. **Integração**:
   * APIs para comunicação entre o aplicativo e sistemas externos, como servidores do SIGA.

#### ****6. Possíveis Ferramentas a serem utilizadas****

* **Desenvolvimento do Front-end**: React Native ou Flutter.
* **Back-end**: Node.js, Python (Django/Flask) ou Java (Spring Boot).
* **Banco de Dados**: MySQL ou PostgreSQL.
* **APIs**: REST API para comunicação eficiente entre as camadas.
* **Segurança**: JWT (JSON Web Tokens) para autenticação segura.