Alunos: Lémersom Fernandes Filho – R.A.: 2410176

Luís Felipe Mori – R.A.: 2266245

Documentação Projeto de calculadora de Médias (MVC)

1. Estrutura do Projeto

O projeto segue uma estrutura organizada que adere ao padrão Model-View-Controller (MVC), proporcionando uma separação clara das responsabilidades.

```
/src
|-- /components
|   |-- InputLabel.js
|-- /models
|   |-- ScoreModel.js
|-- /service
|   |-- HistoryService.js
|   |-- RoutingService.js
|-- /screens
|   |-- Calculator.jsx
|   |-- History.jsx
|   |-- Home.jsx
|-- /controller
|   |-- calculatorController.js
|-- main.jsx
|-- index.jsx
|-- App.jsx
|-- App.css
|-- index.css
|-- routes.js
```

1.1. Pasta src

/src

/components: Contém componentes reutilizáveis.

InputLabel.jsx: Componente para rótulos e entradas de texto.

/models: Armazena classes que representam modelos de dados.

ScoreModel.js: Modelo para armazenar informações sobre notas de alunos.

/screens: Contém componentes relacionados a páginas ou telas específicas.

Calculator.jsx: Componente para a calculadora de notas.

History.jsx: Componente para exibir o histórico de cálculos.

Home.jsx: Componente para a página inicial.

/services: Contém lógica de negócios e integração com serviços externos.

HistoryService.js: Serviço para interagir com o histórico no Firebase.

RoutingService.js: Serviço para criar funções de navegação.

AppRoutes.jsx: Configuração das rotas utilizando react-router-dom.

firebaseConfig.js: Configuração do Firebase com inicialização do aplicativo e referência ao Firestore.

main.js: Ponto de entrada principal da aplicação.

1.2. Componentes MVC

1.2.1. Model (ScoreModel)

A pasta /models contém a classe ScoreModel, responsável por representar a estrutura de dados utilizada para armazenar informações sobre notas de alunos.

1.2.2. View (Componentes em /screens)

As pastas /screens contêm os componentes relacionados a cada tela ou página do aplicativo.

Calculator: Componente que lida com a entrada de notas, realiza cálculos e gerencia a navegação.

History: Componente para exibir o histórico de cálculos anteriores.

Home: Componente para a página inicial.

1.2.3. Controller (Calculator e HistoryService)

Calculator: Atua como o controlador para a página de cálculos.

Manipula o estado local (score).

Calcula médias, limpa entradas e interage com o histórico.

HistoryService: Serviço responsável por interagir com o histórico no Firebase.

1.3. Serviços (/services)

1.3.1. HistoryService

O serviço HistoryService é responsável pela integração com o histórico no Firebase.

Funções:

adicionarDadosAoHistorico: Adiciona dados ao histórico no Firebase.

1.4. Persistência de Dados com Firebase

A persistência de dados no Firebase é realizada pelo serviço HistoryService, que interage com o Firestore.

Arquivo: firebaseConfig.js:

Configuração do Firebase com inicialização do aplicativo e referência ao Firestore.

HistoryService.js:

Utiliza as configurações do Firebase para interagir com o Firestore.

Função adicionarDadosAoHistorico adiciona dados ao histórico no Firestore.

1.5. Camadas

1. Camada de Apresentação (Interface do Usuário):

- Arquivos: Calculator.jsx, History.jsx, Home.jsx (dentro de /screens).

- Descrição: Responsável por representar a interface do usuário, interagindo diretamente com o usuário e utilizando componentes, serviços e modelos para fornecer funcionalidades.

2. Camada de Lógica de Aplicação (ou Controle):

- Arquivos: calculatorController.js (dentro de /controller), RoutingService.js (dentro de /service).

- Descrição: calculatorController.js pode conter lógica específica para a tela de calculadora, enquanto RoutingService.js pode coordenar a navegação entre as telas.

3. Camada de Dados (ou Persistência):

- Arquivos: ScoreModel.js (dentro de /models), HistoryService.js (dentro de /service).

- Descrição: ScoreModel.js define a estrutura dos dados relacionados a pontuações, e HistoryService.js lida com a persistência ou manipulação de dados relacionados a histórico.

4. Camada de Infraestrutura (ou Serviços):

- Arquivos: InputLabel.js (dentro de /components), main.jsx, index.jsx, App.jsx, App.css, index.css, routes.js (na raiz de /src).

- Descrição: InputLabel.js é um componente reutilizável, e os arquivos na raiz fornecem suporte geral para a aplicação.

2. Configuração do Ambiente

Certifique-se de configurar as variáveis de ambiente no arquivo .env para fornecer as credenciais do Firebase.

```
VITE_API_KEY=SEU_API_KEY_AQUI
VITE_AUTH_DOMAIN=SEU_AUTH_DOMAIN_AQUI
VITE_PROJECT_ID=SEU_PROJECT_ID_AQUI
VITE_STORAGE_BUCKET=SEU_STORAGE_BUCKET_AQUI
VITE_MESSAGING_SENDER_ID=SEU_MESSAGING_SENDER_ID_AQUI
VITE_APP_ID=SEU_APP_ID_AQUI
VITE_MEASUREMENT_ID=SEU_MEASUREMENT_ID_AQUI
```