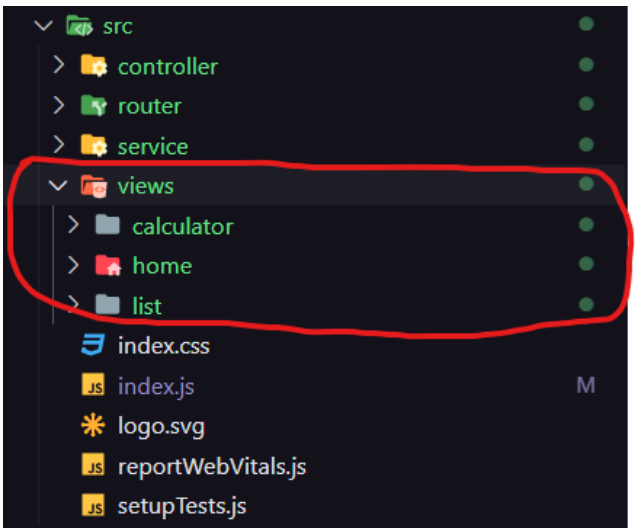
Nome: Estevão Soares da Silva – **RA:** 2417596 – **Arquitetura de Software – ES46A**

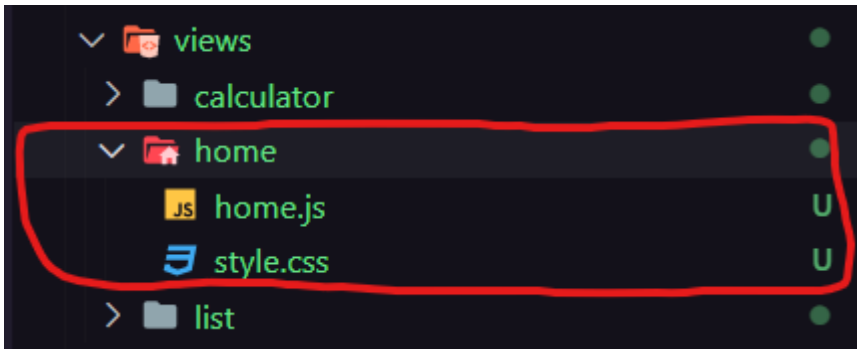
Documentação Exercício Prático

Camada 1 - Cliente

A primeira camada está dentro no caminho **src/views** na onde está localizado toda a parte toda a parte do lado do cliente, nessas pastas está armazenado os códigos de renderização das telas feitas nas tecnologias HTML, CSS, JS.

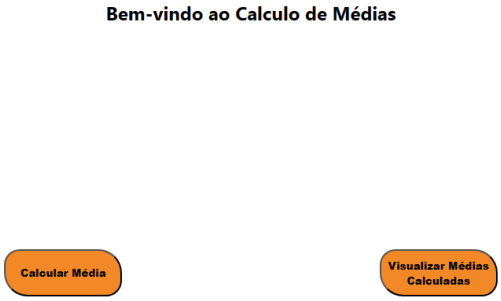


A **primeira tela** é a HOME na onde está localizado em **src/views/home**, na pasta contém dois arquivos sendo um de estilização chamado **style.css** e o outro é o **home.js** na onde faz toda a parte de montagem da tela

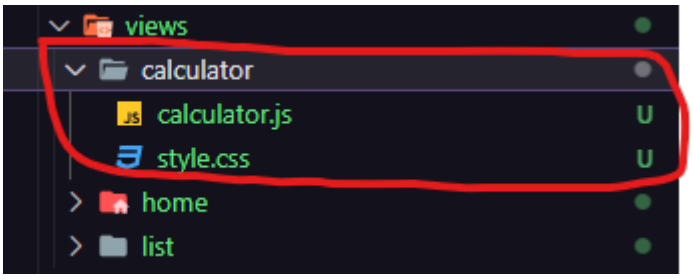


A tela home é bem simples, ela contém um texto e dois botões na qual é o botão de Calcular média onde irá navegar para a tela de Calcular a Média (calculator) e o outro botão é de visualizar as médias calculadas(list).

Está função da tela Home é chamado no App.js



A **segunda tela** é a CALCULATOR na onde está localizado em **src/views/calculator**, na pasta contém dois arquivos sendo um de estilização chamado **style.css** e o outro é o **calculator.js** na onde faz toda a parte de montagem da tela



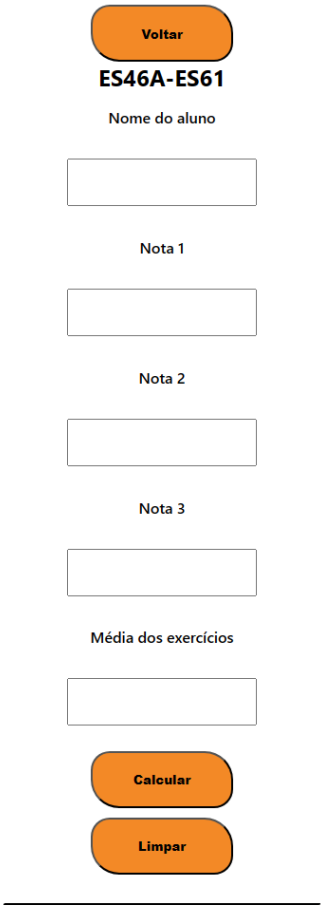
A tela **CALCULATOR** é um pouco mais complexa, ela contém quatros inputs para inserir as notas e média dos exercícios, e três botões que serve para chamar as funções de calcular, de limpar os campos e voltar para a tela.

O botão calcular irá chamar a função onTapCalculate na qual está função, irá calcular a média final passando os valores para a getAverageFinal que retorna a média calculada, salvando em uma variável, para assim chamar o Servidor (Camada 2), na função setAverage passando os valores que irá mandar para o servidor.

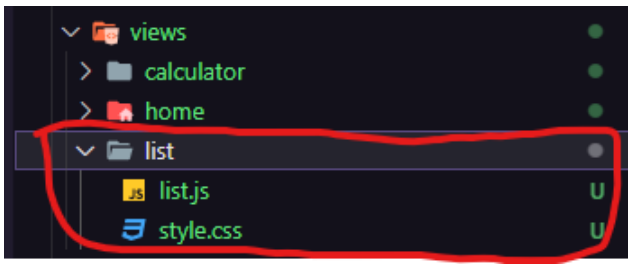
O botão limpar irá chamar a função onTapClear na qual irá limpar todos os campos da tela.

```
const getAverageFinal = (n1, n2, n3, m) => {
  let averagesTotal = (n1 + n2 * 2 + n3 * 3 + m) / 7;

  return averagesTotal.toFixed(2);
};

const onTapCalculate = () => {
  let mediaFinal = getAverageFinal(
    Number(n1),
    Number(n2),
    Number(n3),
    Number(mediaExercise)
  );

  setMediaFinal(mediaFinal);

  setAverage(name, n1, n2, n3, mediaExercise, mediaFinal);
};

const onTapClear = () => {
  setName("");
  setN1("");
  setN2("");
  setN3("");
  setMediaExercise("");
  setMediaFinal("");
};
```

Está função da tela Calculator é chamado no App.js



A **terceira tela** é a LIST na onde está localizado em **src/views/list**, na pasta contém dois arquivos sendo um de estilização chamado **style.css** e o outro é o **list.js** na onde faz toda a parte de montagem da tela



A tela **LIST** é uma tela mais simples, ela contém uma tabela com os campos de nome, nota 1, nota 2, nota 3, média exercício e a média final. Contendo também um botão de Voltar, na qual vai ser redirecionado para a HOME. Ela irá mostrar as informações de todas as médias salvas no Servidor (camada 2).

Quando o usuário adentrar a página, irá ser feita uma chamada para o servidor (camada 2) com a função getAllAverages(), na qual ela irá retornar todos os dados salvos das médias. Para assim ele salvar na variável averages e redenrizar com um map todos os dados na tabela,

```
export default function List() {
  const navigate = useNavigate();

  const [averages, setAverages] = useState([]);

  useEffect(() => {
    let allAlveragesSaved = getAllAverages();

    setAverages(allAlveragesSaved);

    console.log(allAlveragesSaved);
  }, []);

  return (
    <div className="container">
      <h2>Listagem de Médias</h2>
      <table id="table">
        <tr>
          <th>Nome</th>
          <th>Nota 1</th>
          <th>Nota 2</th>
          <th>Nota 3</th>
          <th>Média Exercicio</th>
          <th>Média Final</th>
        </tr>
        {averages.map((average) => (
          <tr>
            <td>{average.name}</td>
            <td>{average.n1}</td>
            <td>{average.n2}</td>
            <td>{average.n3}</td>
            <td>{average.m}</td>
            <td>{average.mf}</td>
          </tr>
        ))}
      </table>
```

Está função da tela List é chamado no App.js

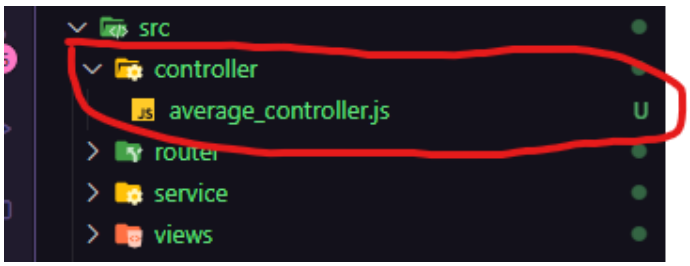
Listagem de Médias

Nome	Nota 1	Nota 2	Nota 3	Média Exercicio	Média Final
TEVAS	10	8	6	6	7.14
tevas	1	2	3	4	2.57
tevas	12	12	1	4	6.14
joao	4	7	2	10	4.86

Voltar

Camada 2 – Servidor

A segunda camada está dentro no caminho **src/controller** na onde está localizado toda a parte toda a parte do lado do servidor, nessa pasta está o arquivo average_controller.js na qual está todas as funções utilizadas na camada.



O arquivo average_controller contém três funções básicas utilizadas no sistema, a primeira função é a *getAllAverages* na qual ela irá chamar a função que faz a busca de todas as averages salvas no banco e assim salvar na variável allAlverages, na qual está retornando essa variável para quem consumir esta função(end-point).

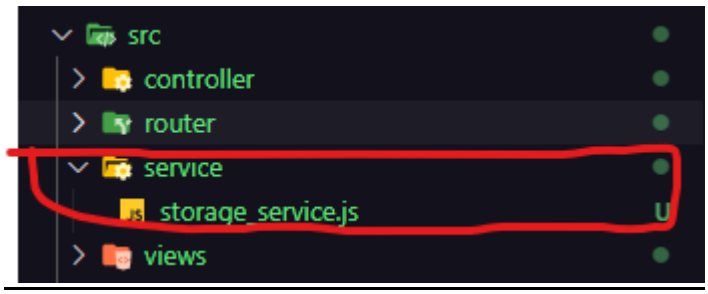
A segunda função é a *getAverageByName* na qual ela irá chamar a função que faz a busca no banco de dados, de todas as averages salvas, para assim fazer uma filtragem de qual average contém o mesmo nome, da passada via parâmetro(query), para assim retornar o valor encontrado para quem consumir está função (end-point).

A terceira função é a *setAverage* na qual ela irá salvar os dados recebidos via parâmetro(query), chamando a função setDataStorage na qual é a função de salvamento no banco de dados passando um objeto para salvamento.

```
import { getAllDataStorage, setDataStorage } from "../service/storage_service";

const getAllAverages = () => {
  let allAverages = getAllDataStorage() ?? [];

  return allAverages;
};

const getAverageByName = (name) => {
  let allAverages = getAllDataStorage() ?? [];

  let index = allAverages.findIndex((average) => average.name === name);

  return index !== -1 ? allAverages[index] : null;
};

const setAverage = (name, n1, n2, n3, m, mf) => {
  let documentToSave = {
    name,
    n1,
    n2,
    n3,
    m,
    mf,
  };

  setDataStorage(documentToSave);
};

export { getAllAverages, setAverage, getAverageByName };
```

Camada 3 – Banco de dados

A terceira camada está dentro no caminho **src/service** na onde está localizado toda a parte toda a parte do lado de manipulação do banco de dados, nessa pasta está o arquivo storage_service.js na qual está todas as funções utilizadas na camada. O banco de dados escolhido foi o LocalStorage na qual é um banco de dados local do próprio navegador.



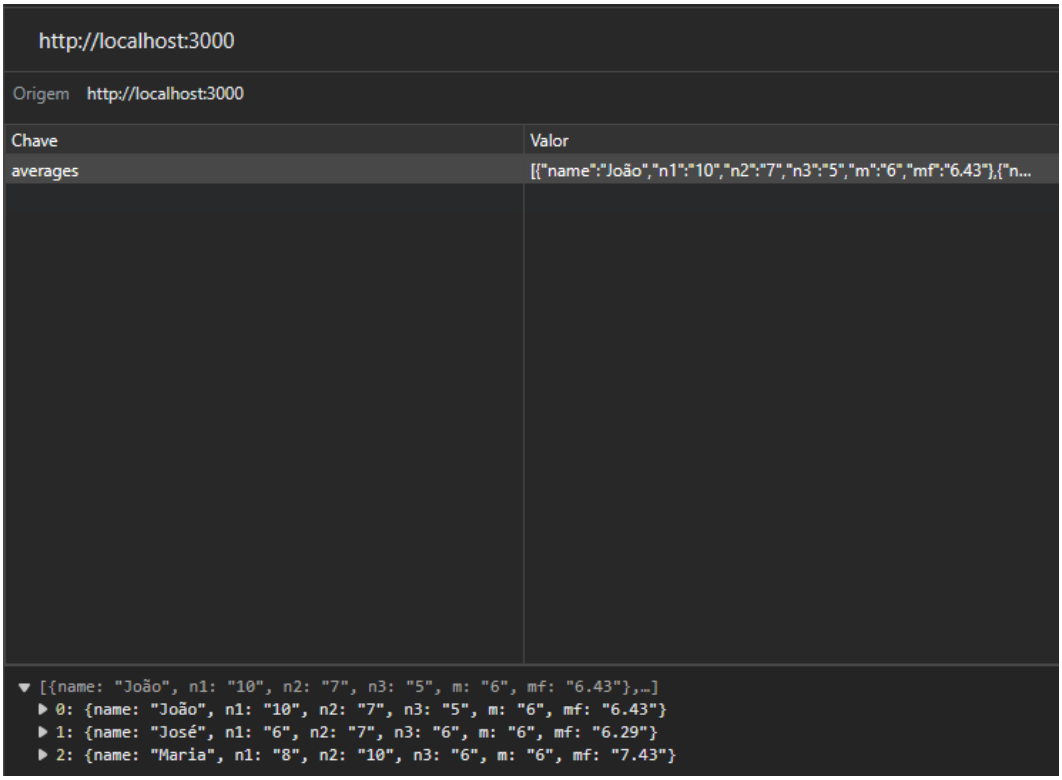
O arquivo storage_service contém três funções básicas utilizadas para manipulação do banco de dados, a primeira função é a *getAllDataStorage* na qual ela irá fazer a busca no banco de dados, de todas as averages salvas no banco e assim salvar na variável allAverages na qual depois será transformada em um objeto para assim retornar na função.

A segunda função é a *setDataStorage* na qual ela irá receber no parâmetro os dados para salvar no banco de dados.

A terceira função é a *deleteAllDataStorage* na qual irá apagar todas as averages do banco de dados.

```
const key = "averages";

const getAllDataStorage = () => {
  let allAverages = localStorage.getItem(key);

  let stringToJson = JSON.parse(allAverages);

  return stringToJson ?? [];
};

const setDataStorage = (data) => {
  let allAverages = getAllDataStorage() ?? [];

  let documentToSave = [...allAverages, data];

  let stringifyDocument = JSON.stringify(documentToSave);

  localStorage.setItem(key, stringifyDocument);
};

const deleteAllDataStorage = () => localStorage.clear();

export { getAllDataStorage, setDataStorage, deleteAllDataStorage };
```

Os dados salvos no banco de dados **LocalStorage**



Outros - Camada 4

Nesta camada contém algum outros arquivos do projeto sendo a node_modules na qual irá guardar os packages utilizados, a router na qual irá fazer o roteamento das páginas, o package-json e package-lock na qual faz o controle dos pacotes entre outros arquivos como .gitignore, readme.md, reportWebvitals, setupTests.