Universidade Federal de Goiás
Instituto de Informática
Engenharia de Software

MindBox
Definição da Arquitetura

Alan Brito Barros
Amanda Lobo Gomes

Goiânia
28 de abril de 2020

Histórico de Versões

Data	Versão	Descrição	Autor	Revisor
28/04/2021	1.0	Primeira versão do documento	Alan Brito Barros, Amanda Lobo Gomes	Alan Brito Barros, Amanda Lobo Gomes

Sumário

1. Introdução

1.1. Finalidade

A finalidade do documento em questão é definir, especificar e elaborar a arquitetura do projeto MindBox.

1.2. Escopo

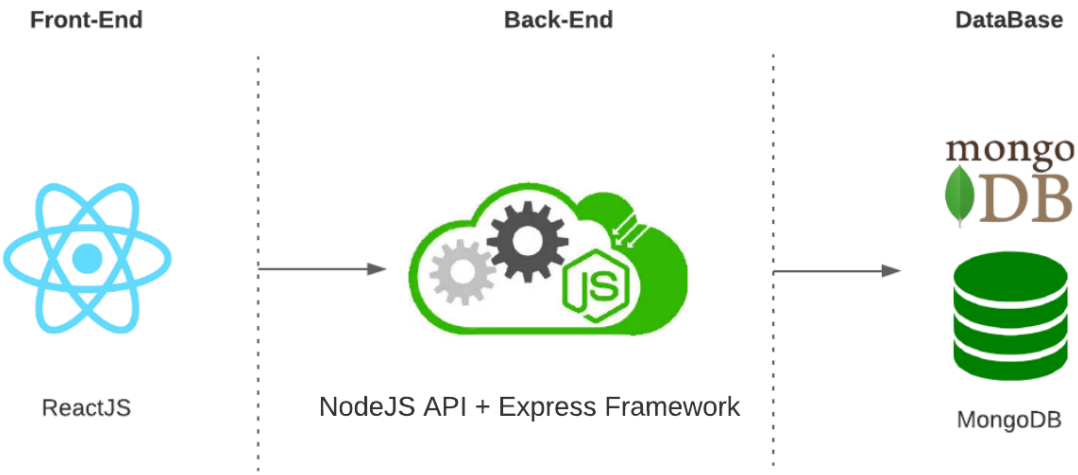
O escopo do documento envolve definir os elementos suficientes e necessários para a elaboração da arquitetura da aplicação. Isso envolve a elaboração e um detalhamento de uma representação arquitetural, evidenciando os componentes e camadas que comporão a aplicação, além de diferentes visões arquiteturais, como a visão de casos de uso, visão lógica, visão de processos, visão de implantação, visão de dados e visão de implementação.

1.3. Visão Geral

O MindBox é uma aplicação web que viabiliza o processo de brainstorming de forma virtual, proporciona uma comunicação entre seus usuários dentro de reuniões e dispõe de ferramentas para interagir e auxiliar times em seus processos criativos. A partir desse documento serão descritos os componentes e camadas que compõem o projeto MindBox.

2. Representação Arquitetural

A arquitetura da aplicação pode ser dividida em três diferentes partes. O front-end utiliza a biblioteca JavaScript ReactJS para a construção da interface da aplicação. O back-end utiliza o NodeJS, juntamente com o Express Web Framework e Mongoose, que fará a modelagem e o preparo dos dados que serão armazenados no banco de dados MongoDB. As imagens abaixo representam a forma com que as tecnologias se relacionam dentro da aplicação.



3. Metas e Restrições da Arquitetura

A aplicação MindBox foi definida como sendo uma aplicação web. As restrições na arquitetura e na tecnologia que a compõe foi estabelecida baseada no propósito da aplicação, além de se basear na competência dos envolvidos no projeto.

Com isso, definiu-se estrategicamente que o desenvolvimento do projeto será utilizado JavaScript por ser extremamente forte no ambiente web, disponibilizando bibliotecas e dependências que fornecem serviços prontos para a composição do projeto. A biblioteca React proporciona aplicações extremamente responsivas e rápidas, algo extremamente necessário dentro de uma aplicação que trabalha com atividades em tempo real.

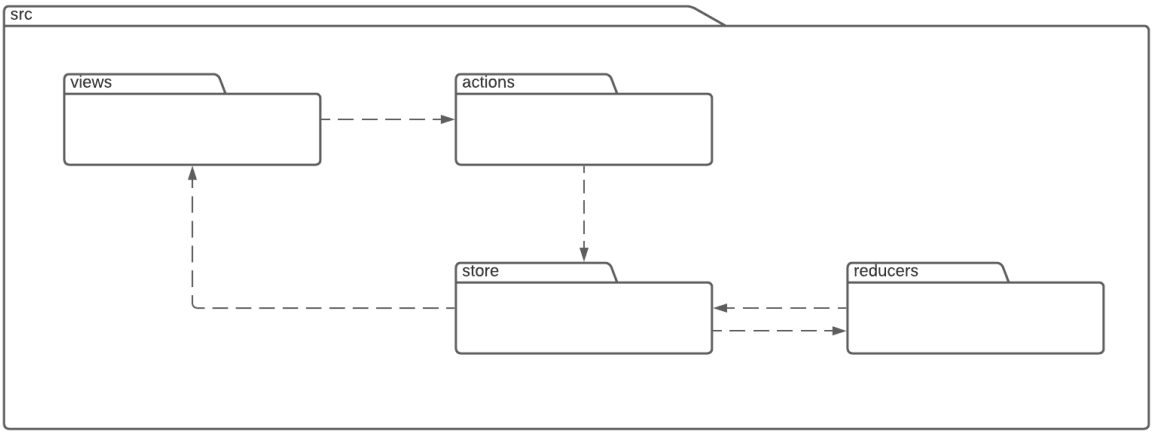
O back-end em Node.js, além de ter uma ótima sinergia com o ambiente React, permite uma certa facilidade uma futura expansão para a adição de uma aplicação mobile com a utilização de React Native.

4. Visão Lógica

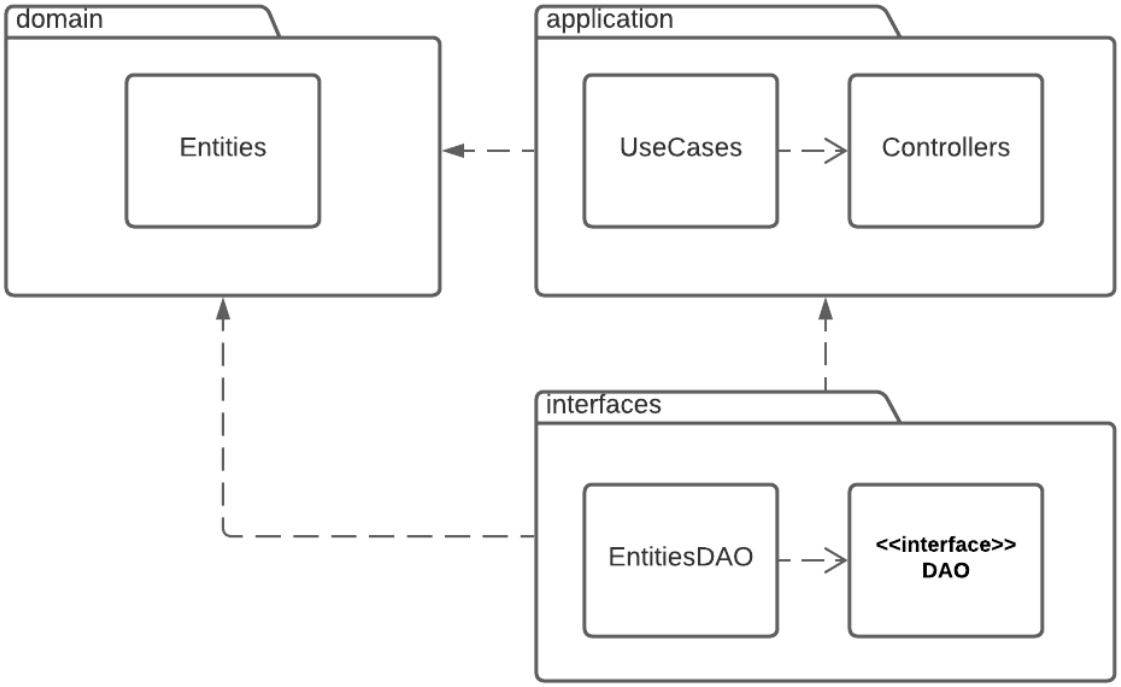
4.1. Visão Geral

A visão lógica mostra um subconjunto do modelo de design significativo em termos de arquitetura, ou seja, um subconjunto das classes, subsistemas, pacotes e realizações de caso de uso.

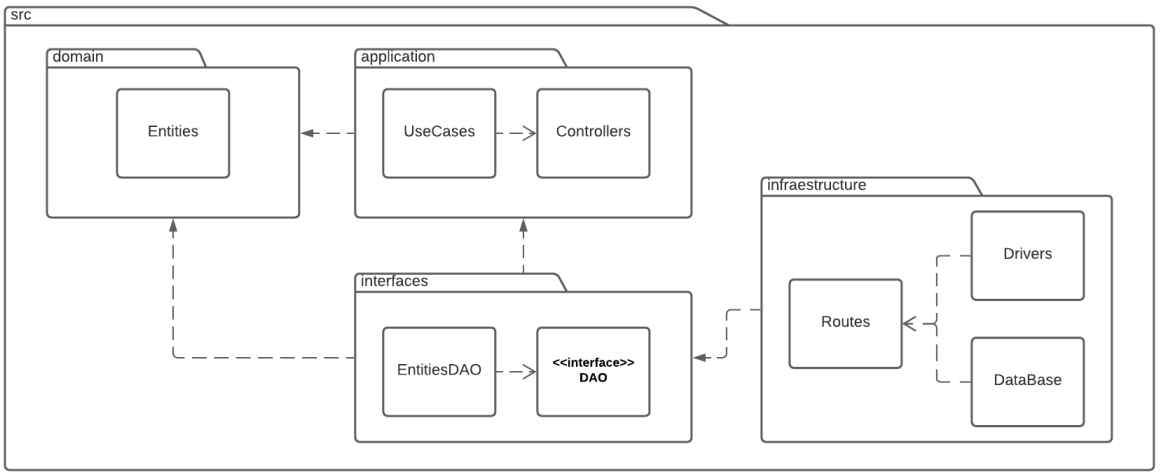
4.2. Camada de Visão e Controle



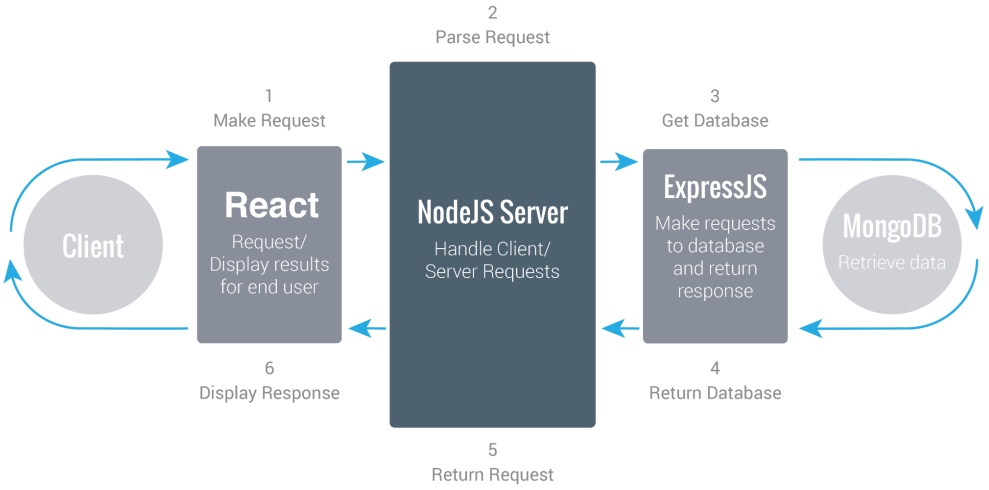
4.3. Camada de Negócio



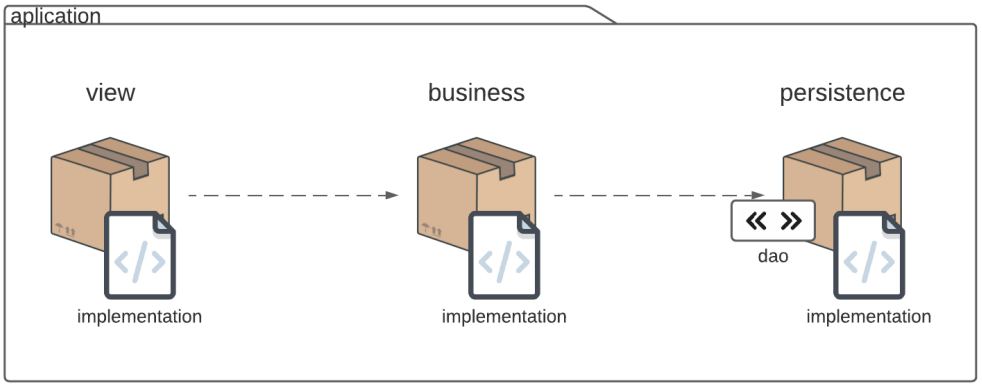
4.4. Camada de Persistência



5. Visão de Implantação



6. Visão da Implementação



7. Tamanho e Desempenho

A aplicação MindBox, por se tratar de uma aplicação web e rodar dentro do browser de um usuário, não possui um tamanho especificado. O desempenho da aplicação, por sua vez, deve ir de acordo com todos os requisitos previamente documentados. Para tal, o único pré-requisito é que o usuário que esteja utilizando o MindBox tenha uma conexão de internet estável, permitindo o carregamento de áudio e vídeo, sendo que a largura da banda necessária está diretamente vinculada à quantidade de participantes presentes na sala.

8. Qualidade

O MindBox, como aplicação, deve entregar qualidade em seu produto. Isso envolve uma série de características a serem satisfeitas, de modo com que a aplicação atenda às necessidades de seus usuários, devidamente documentadas através dos requisitos funcionais e não funcionais.

São várias as características que envolvem a qualidade de um software, e durante todo o projeto do MindBox tais pontos são checados para garantir a qualidade do produto final:

8.1. Funcionalidade

8.1.1. Adequação

A aplicação deve fazer o que foi proposto.

8.1.2. Acurácia

A aplicação deve gerar resultados corretos ou conforme acordados.

8.1.3. Interoperabilidade

A aplicação deve ser capaz de interagir com os demais sistemas especificados.

8.1.4. Segurança de Acesso

A aplicação deve evitar o acesso não autorizado/autenticado.

8.2. Confiabilidade

8.2.1. Maturidade

A aplicação apresenta falhas com baixa frequência.

8.2.2. Tolerância a falhas

A aplicação tem reação esperada às possíveis falhas.

8.2.3. Recuperabilidade

A aplicação tem capacidade de se recuperar rapidamente.

8.3. Usabilidade

8.3.1. Inteligibilidade

A aplicação deve ter seus conceitos de forma a serem facilmente entendidos.

8.3.2. Operacionalidade

A aplicação deve ser fácil de controlar e operar.

8.4. Eficiência

8.4.1. Comportamento em relação ao tempo

A aplicação não deve ultrapassar o tempo estipulado de resposta e processamento de dados.

8.4.2. Comportamento em relação aos recursos

A aplicação não deve ultrapassar a quantidade estipulada de recursos.

8.5. Manutenibilidade

8.5.1. Estabilidade

A aplicação deve minimizar os riscos de bugs no momento em que alterações são feitas.

8.5.2. Testabilidade

A aplicação deve proporcionar um ambiente propício para o teste de novas alterações.

9. Referências

- **SWEBOK**, v3.
- **Qualidade de Software - Engenharia de Software**. DevMedia, 2021. Disponível em: https://www.devmedia.com.br/qualidade-de-software-engenharia-de-software-29/18209. Acesso em: 28 abr 2021.
- **React Architecture Best Practices and tips from Community Experts**. Simform, 2018. Disponível em: https://www.simform.com/react-architecture-best-practices/. Acesso em: 28 abr 2021.
- **Uma Breve Introdução a Arquitetura Limpa com Nodejs.** Decom, 2019. Disponível em:

http://www2.decom.ufop.br/terralab/uma-breve-introducao-a-arquitetura-limpa-com-node-js/. Acesso em: 28 abr 2021.

- **Minicurso ReactJS**. VictorVH, 2020. Disponível em: https://victorvhpg.github.io/minicurso-react.js/slides/#/37. Acesso em: 28 abr 2021.
- **Arquitetura NodeJS**. GitHub, 2020. Disponível em: https://fga-eps-mds.github.io/2019.1-MaisMonitoria/docs/doc-arquitetura
- https://blog.geekhunter.com.br/flux/. Acesso em:
- **Utilizando React Redux Firebase**. TecSinapse, 2020. Disponível em: https://blog.tecsinapse.com.br/utilizando-react-redux-firebase-2bf93ea9f422. Acesso em: 28 abr 2021.