**Mapeamento de Funcionalidades (44%)**

Aplicação: Sistema de recomendação de livros (Quiz)

Cliente: Biblioteca Fátima Costa

| **REQUISITOS** | **CODIFICAÇÃO** |
| --- | --- |
| **Entrada:** tela inicial destinada ao usuário; apresentação da aplicação; introdução do usuário na mecânica do quiz interativo auxiliada pela mascote. | **A tela inicial mostrada ao usuário é o arquivo de nome index.js, o nome padrão para a tela inicial em aplicações Next.js, que se encontra dentro da pasta pages, que automaticamente cria rotas no Next.**  **Não utiliza de muitas funções além das já intrínsecas ao react, possuindo diversos elementos com classes distintas, como um container geral, um header, uma parte main principal e um footer.** |
| **Iniciação:** botão de início do  quiz; acesso à tela com a primeira pergunta para dar início ao quiz. | **Se encontra na página inicial, corresponde ao botão:**  **Possui classes que indicam sua função e aparência, linkando para a parte de questões.**  **<div className={styles.action}>**  **<button className={styles.bigButton}>**  **<Link href="/questions">**  **<a>Iniciar teste!</a>**  **</Link>**  **</button>**  **</div>** |
| **Avanço:** funcionalidade destinada ao progresso do usuário no quiz, permitindo o mesmo avançar entre as telas (perguntas). | **A funcionalidade de avanço se encontra intrínseca ao selecionar uma das alternativas da pergunta, uma função onClick que altera a questão mostrada:**  **Uma função que lida com o click da alternativa, recebendo o evento e o valor, o texto, do botão, mudando o estado da variável currentQuestion, alterando a pergunta mostrada. Além disso verifica se as questões já acabaram, guarda as respostas escolhidas e manda o usuário à próxima página caso as perguntas tenham sido respondidas.**  **const handleAnswerButtonClick = (e, value) => {**  **//mudando a pergunta atual**  **const nextQuestion = currentQuestion + 1;**  **//verificando se as questões terminaram**  **if(nextQuestion < question.length){**  **setCurrentQuestion(nextQuestion);**  **//armazenando a resposta**  **setChoices( (arr) => [...arr, value])**  **console.log(choices);**  **} else {**  **console.log('Escolhas: ' + choices)**  **Router.push('/books')**  **}** |
| **Recuo:** opção de retorno para o usuário, permitindo o mesmo recuar entre as telas, caso preciso mudar a resposta. | **Ainda não implementada.** |
| **Encerramento:** opção de encerramento para o usuário; finalização da aplicação, retornando à tela inicial. | **Se encontra na última página, a mesma de amostragem do resultado final, corresponde ao botão:**  **Um botão simples que envia o usuário para a página inicial.**  **<button className={styles.bigButton}>**  **<Link href="/">**  **<a>Refazer teste!</a>**  **</Link>**  **</button>** |
| **Banco de dados:** armazenamento de dados para os títulos (livros) que serão retornados ao usuário. | **Ainda não diretamente implementado, estrutura feita em MongoDB.** |
| **Sistema de recomendação e escolha:** codificação destinada à escolha de conteúdo baseado em filtragem, através de tags correspondente à títulos, para retornar uma resposta (livro recomendado) ao usuário de acordo suas preferências. | **Ainda não diretamente implementado, o planejado é usar um sistema de pesquisa por querry de array, ou seja, unir as respostas escolhidas em um array e realizar uma pesquisa por query no banco de dados.** |
| **Recomendação:** retorno como resposta ao usuário; recomendação com base na filtragem de dados e nos títulos cadastrados no banco de dados da aplicação. Funcionalidade que retorna um livro em destaque mais dois secundários (semelhantes) para o usuário. | **Retorno será enviado pela pesquisa na base de dados, ainda não implementado.** |

| **Adicionais:** acesso às informações sobre o livro e sua localização dentro da biblioteca (endereço, sessão e prateleira). | **Não implementado.** |
| --- | --- |