|  |  |
| --- | --- |
| Atividades | Match Band |
| Requisitos | - Fazer o que o cliente precisa! Atender suas necessidades!  Requisitos Funcionais:  - Usuario se Cadastra como Músico com Nome, e-mail, senha, nome artistico, foto, descrição, sexo, genero musical,contato  - Usuario Cadastra Banda com Nome da Banda, e-mail, senha, foto, descrição da banda, membros, genero musical, Tipo do musico que esta procurando  - Banda Pesquisa Músico com Filtro por genero musical  - Musico Pesquisa Banda com Filtro por genero musical  - Banda Conversa com Músico por chat  - Musico Conversar com Banda por chat  - Musico edita usuário com nome, senha, e-mail, foto, descrição, sexo, genero musical, contato, nome artistico  - Banda Edita úsuario com edição de nome, senha, e-mail, foto, descrição, sexo, genero musical, contato  - Deletar Senha - Deletar cadastro  Requisitos Não Funcionais:  - Separação de Interesses (Separation of Concerns - SoC): separar corretamente lógica, interação com o usuário e comportamentos na aplicação.  - Utilizar a Arquitetura de Software MVC (Model-View-Controller)  - Portabilidade: desenvolver para diferentes plataformas. A Arquitetura MVC, através de sua separação de interesse, permitirá a criação de diferentes Views para o mesmo Model.  - Plataforma Web  - Desktop  - Notebook  - Tablet  - Celular  - Usabilidade: oferecer uma experiência de uso simples para as pessoas considerando os padrões de interação das plataformas que elas estão usando.  - Usabilidade para plataforma Web  <http://www.w3.org/WAI/WCAG20/quickref/>  - Desempenho: melhorar a performance da aplicação, buscando uma melhor plataforma para o usuário.  - Segurança: garantir que apenas os usuários habilitados acessarão a aplicação e que os dados serão transmitidos de forma segura. |
| Projeto | Match between system and the real world:  O sistema deve falar a língua do usuário, com palavras, frases e imagens familiares, fazendo com que a informação apareça de forma natural e lógica. Isso vai acertar em cheio com nosso projeto, pois temos que ser bem claros em palavras e imagens para que o usuario possa utilizar de forma agradavel e sem complicação.  Error prevention  Utilizar de forma correta mensagens de erro, cores e de forma suave, mostrar erros que o usuario pode ter cometido e/ou o sistema.  - Projeto de Interação com o Usuário (para atender o requisito não funcional de usabilidade)  mob1.png  8 -Estética e Design Minimalista (a interface apresenta apenas as informações essenciais)  mob2.png  6. Reconhecimento ao invés de lembrança (Ao invés de se lembrar do nome da banda, a interface oferece uma forma de reconhecimento)  5. Prevenção de erros (Ajuda o usuário a não entrar com um nome inválido de banda)  mob3.png  2. Conexão entre o sistema e o mundo real (O resultado da busca fala “a língua” do usuário com termos apropriados àquele usuário) |
| Implementação | <https://github.com/giulianobertoti/eXtremeProgramming> |
| Testes | <https://github.com/giulianobertoti/eXtremeProgramming/tree/master/Tablatures/src>    1º Teste: Adicionei 2 objetos Music no atributo musics da classe MusicList e depois testei se de fato haviam 2.  2º Teste: Fiz uma busca passando um objeto Specification e conferi se o resultado era correto. |
| Manutenção | 1ª Testes Automatizados: em toda atualização, principalmente naquelas onde não fui eu que implementei a lógica (foi outro membro da equipe), eu tenho testes automatizados para checar se eu quebrei ou não o código.  2ª Encapsulamento: para acrescentar um novo requisito funcional do cliente, por exemplo comparação por ano da música, eu precisei alterar apenas a classe Specification, pois todas as comparações da busca por músicas estão lá (ou seja, meu projeto está bem encapsulado). A classe MusicList, que é a classe principal de lógica, não precisou ser alterada.  **public** **class** Specification {  **private** String band;  **private** String musicName;  **private** **int** year;  **public** Specification(String band, String musicName, **int** year){  **this**.band = band;  **this**.musicName = musicName;  **this**.year = year;  }    **public** **boolean** matches(Specification spec){  **if**(!band.equals(spec.band)) **return** **false**;  **if**(!musicName.equals(spec.musicName)) **return** **false**;  **if**(year!=spec.year) **return** **false**;  **return** **true**;  }  } |
|  | FIM |