|  |
| --- |
| TESTE DE CONHECIMENTOS |

|  |
| --- |
| 1 – Explique (de forma objetiva) os princípios da orientação a objeto? |
| Os princípios da orientação a objeto são:  Encapsulamento: ocultação dos detalhes internos de uma classe.  Abstração: foco nas características essenciais de um objeto.  Herança: criação de uma nova classe a partir de uma existente.  Polimorfismo: capacidade de um objeto se comportar de diferentes formas dependendo do contexto. |

|  |
| --- |
| 2 – Você conhece a arquitetura em três camadas? Explique com suas palavras o que sabe a respeito? |
| Camada de apresentação: esta camada é responsável por apresentar as informações ao usuário final. É a interface do usuário, com formulários  Camada de lógica de negócios: é onde as regras de negócios e processamento de dados são implementados. Essa camada é responsável por receber as solicitações da camada de apresentação  Camada de acesso a dados: é a camada mais baixa e lida com o armazenamento e acesso aos dados. Ela se comunica com a camada de lógica de negócios e é responsável por interagir com o banco de dados, arquivos, serviços web ou outras fontes de dados. |

|  |
| --- |
| 3 – O que você sabe sobre Design Patterns? Cite 3. |
| Design Patterns, ou Padrões de Projeto, são soluções comuns para problemas recorrentes na programação de software. Eles são soluções testadas e comprovadas que podem ser aplicadas em diferentes projetos  Singleton  Observer  Dependency injection |

|  |
| --- |
| 4 – O que você sabe sobre DDD e TDD? Explique com suas palavras. |
| DDD (Domain-Driven Design)  O DDD é uma abordagem para o desenvolvimento de software que se concentra na modelagem do domínio do negócio e na linguagem utilizada, O objetivo é criar um modelo que capture a complexidade do domínio e que possa ser facilmente atualizado à medida que o sistema cresce.  TDD (Test-Driven Development)  Já o TDD é tem o objetivo de desenvolvimento de software que se concentra na criação de testes automatizados antes do código de produção. Isso faz com que o desenvolvedor defina claramente os requisitos do sistema e desenvolva o código necessário para atender requisitos de forma incremental e orientada por testes. |

|  |
| --- |
| 5 – Qual o seu entendimento sobre ORM? |
| ORM (Object-Relational Mapping) é uma técnica de programação que permite que os objetos de uma aplicação sejam mapeados diretamente para as tabelas de um banco de dados relacional. Isso elimina a necessidade de escrever código SQL manualmente e facilita a interação com o banco de dados. |

|  |
| --- |
| 6 – Quais são as junções utilizadas em um Banco de Dados (Join)? Dê exemplos. |
| Existem quatro tipos principais de junções (joins) em bancos de dados relacionais:  INNER JOIN: Retorna somente as linhas que possuem uma correspondência nas duas tabelas envolvidas na junção. Exemplo:  SELECT \*  FROM tabela1  INNER JOIN tabela2  ON tabela1.coluna = tabela2.coluna;  LEFT JOIN: Retorna todas as linhas da tabela da esquerda (primeira tabela mencionada na cláusula SQL) e as correspondentes da tabela da direita (segunda tabela mencionada), caso haja. Se não houver correspondência na tabela da direita, as colunas serão preenchidas com valores NULL. Exemplo:  SELECT \*  FROM tabela1  LEFT JOIN tabela2  ON tabela1.coluna = tabela2.coluna;  RIGHT JOIN: Retorna todas as linhas da tabela da direita e as correspondentes da tabela da esquerda, caso haja. Se não houver correspondência na tabela da esquerda, as colunas serão preenchidas com valores NULL. Exemplo:  SELECT \*  FROM tabela1  RIGHT JOIN tabela2  ON tabela1.coluna = tabela2.coluna; |

|  |
| --- |
| 7 – Já uso o Gulp? Para que serve? |
| Não, Gulp é uma ferramenta de automatização de tarefas em JavaScript que é executada no ambiente Node.js. Ele permite que os desenvolvedores definam tarefas personalizadas que automatizam processos como compilação de código, minificação de arquivos, otimização de imagens e sincronização do navegador. |

|  |
| --- |
| 8 – Qual a sua familiaridade com o NPM? |
| Pouca familiaridade |

|  |
| --- |
| 9 – De que forma você trataria requisições para um Backend, através do Frontend de uma aplicação? |
| Eu iria padronizar as requisições, como o método HTTP utilizado (GET, POST, PUT, DELETE), a estrutura dos dados enviados e as respostas esperadas do servidor, utilizaria uma biblioteca de requisições, bibliotecas como o Fetch API e o jQuery Ajax podem ajudar a simplificar a lógica de envio e recebimento de requisições, além de oferecer recursos como tratamento de erros e cancelamento de requisições, validar os dados enviados pelo usuário antes de enviá-los para o backend, para evitar problemas como injeção de código malicioso ou erros de formatação. Implementar segurança: Utilizar técnicas de segurança, como autenticação e criptografia, é essencial para proteger a aplicação contra ataques maliciosos.  Em geral, é importante manter uma boa comunicação entre o frontend e o backend da aplicação, utilizando padrões estabelecidos e garantindo a segurança e integridade dos dados. Além disso, seguir as melhores práticas de desenvolvimento de software e testar rigorosamente as funcionalidades da aplicação antes de disponibilizá-las para os usuários. |

|  |
| --- |
| 10 – O que você sabe sobre GIT e TFS? |
| GIT e TFS são sistemas de controle de versão utilizados no desenvolvimento de software.  O GIT é um sistema de controle de versão distribuído, que permite que múltiplos desenvolvedores trabalhem em um mesmo projeto ao mesmo tempo e de forma colaborativa, compartilhando alterações de código entre si.  O TFS, por sua vez, é um sistema de controle de versão centralizado desenvolvido pela Microsoft, que é integrado ao ambiente de desenvolvimento Visual Studio. |

TESTE PRÁTICO

O Teste consiste no desenvolvimento de uma aplicação web para administração de uma locadora de Filmes.

**PRÉ-REQUISITOS:**

* Usar arquitetura em 3 camadas (Apresentação, Services e Data).
* Utilizar a linguagem C#, Asp.Net MVC, e Framework 4.5 ou superior.
* Criação de pelo menos 4 entidades (Usuários, Clientes, Filmes, Locações).
* Criação dos Serviços e Repositórios para todas as entidades.
* Utilizar SQL Server com Entity Framework.
* FrontEnd com Bootstrap.  
    
  **OBS:** Preferência por utilização do .Net Framework, porém pode utilizar o .Net Core caso queira, porém a analise será mais criteriosa.

**ENTREGA:**

* O Projeto deverá ser publicado no GitHub.
* O repositório deverá ser público.

**O QUE SERÁ AVALIADO:**

* Qualidade de código.
* Organização.
* Lógica.
* Quanto do proposto foi desenvolvido.

**PRAZO PARA ENTREGA:**

* 48 horas.

**OBSERVAÇÃO IMPORTANTE:**

* Caso você não tenha tempo hábil para terminar todo o projeto, não se preocupe, faça o máximo possível. Isso poderá bastar para nossa avaliação.