

Miniteste 03 - Princípios de Desenvolvimento WEB

Assunto:

Definição de Requisitos
Modelo Entidade Relacionamento
Métodos HTTP e códigos de resposta.

levi.pereira.junior@ccc.ufcg.edu.br [Mudar de conta](#)



Não compartilhado



Salvando...

* Indica uma pergunta obrigatória

Explique a importância da definição de requisitos em um projeto Web/API. Além disso, discuta como a falta de uma boa definição de requisitos pode impactar negativamente o projeto. *

É importante saber o que o nosso software vai fazer, saber quais são as entidades envolvidas, quais as tecnologias mais adequadas e recursos disponíveis para que consigamos criar um software.

Devemos nos perguntar o que é aquele software, o que faz e para quem faz. E caso não consigamos responder uma ou mais dessas três perguntas, devemos levantar os requisitos para saber quais são as entidades envolvidas e como elas se relacionam.

Isso é importante para que os desenvolvedores envolvidos entendam como o software vai funcionar e que possa remover ambiguidade entre os desenvolvedores em relação a várias partes do sistema.

A falta do levantamento de requisitos pode causar problemas de ambiguidade. Pode haver problemas de escalabilidade, uma vez que não escolheu os padrões e tecnologias adequadas.



Uma empresa precisa desenvolver uma API para gerenciamento de tarefas (Task * Manager API), permitindo que usuários criem, editem e organizem suas tarefas. Para que o projeto tenha um bom planejamento, é necessário definir seus requisitos e modelar suas entidades e relacionamentos.

Com base nesse cenário:

- 1) Defina os principais **requisitos funcionais** dessa API.
- 2) Identifique as principais **entidades** do sistema e descreva seus atributos.
- 3) Explique como essas entidades se relacionam entre si e justifique sua modelagem.

1) Requisitos Funcionais são:

Usuário: Cadastro, login, recuperação de senha;

Tarefa: Criar, editar, excluir, listar;

Organização: Categorizar tarefas, definir prazos e prioridades;

Colaboração: Atribuir tarefas a outros usuários.

2) Entidades do sistema e seus atributos são:

Usuário: id, nome, email e senha;

Tarefa: id, título, descrição, status, prioridade, prazo e usuario_id;

Categoria: id, nome e usuario_id;

Comentário: id, texto, data, usuario_id e tarefa_id.

3) Relacionamentos:

Usuário - Tarefa (1 a N): Um usuário pode ter várias tarefas;

Usuário - Categoria (1 a N): Um usuário organiza tarefas em categorias;

Tarefa - Comentário (1 a N): Cada tarefa pode ter vários comentários.



Considerando a API de gerenciamento de tarefas (Task Manager API), é necessário definir as rotas e os principais retornos HTTP para garantir uma comunicação eficiente entre cliente e servidor. *

Com base nesse contexto:

- 1) Liste as principais **rotas** da API seguindo o padrão RESTful, indicando os métodos HTTP adequados para cada uma.
- 2) Descreva os **códigos de status HTTP** esperados para cada rota.
- 3) Apresente um exemplo de **corpo de resposta (JSON)** para uma requisição bem-sucedida de criação de uma tarefa.

Usuário: /users

POST /users/register: Criar usuário [201 Created, 400 Bad Request]

POST /users/login: Autenticar usuário [200 OK, 401 Unauthorized]

GET /users/me: Obter perfil do usuário [200 OK, 404 Not Found]

PUT /users/me: Atualizar dados do usuário [200 OK, 404 Not Found]

Tarefas /tasks

POST /tasks: Criar nova tarefa [201 Created, 400 Bad Request]

GET /tasks: Listar todas as tarefas do usuário [200 OK, 404 Not Found]

GET /tasks/{id}: Obter detalhes de uma tarefa [200 OK, 404 Not Found]

PUT /tasks/{id}: Atualizar tarefa [200 OK, 404 Not Found]

DELETE /tasks/{id}: Remover tarefa [200 OK, 404 Not Found]

Categorias /categories

POST /categories: Criar categoria [201 Created, 400 Bad Request]

GET /categories: Listar categorias [200 OK, 404 Not Found]

DELETE /categories/{id}: Remover categoria [200 OK, 404 Not Found]

Comentários /tasks/{id}/comments

POST /tasks/{id}/comments: Adicionar comentário [201 Created, 404 Not Found]

GET /tasks/{id}/comments: Listar comentários [200 OK, 404 Not Found]



Em uma API RESTful, os métodos HTTP possuem funções específicas e devem ser utilizados corretamente para garantir a padronização e eficiência das requisições. Considerando as boas práticas no uso dos métodos HTTP, analise as seguintes afirmativas: *

- 1) O método **GET** deve ser utilizado para buscar recursos e não deve modificar dados no servidor.
- 2) O método **POST** é idempotente, ou seja, enviar a mesma requisição várias vezes sempre resultará no mesmo efeito no servidor.
- 3) O método **PUT** pode ser utilizado para atualizar um recurso existente, mas também pode ser usado para criar um recurso se o cliente fornecer um identificador específico.
- 4) O método **DELETE** deve sempre remover um recurso, e seu retorno deve ser obrigatoriamente um código **204 No Content**.
- 5) O método **PATCH** deve ser utilizado para atualizações parciais de um recurso.

Com base nas afirmações acima, marque a alternativa correta:

- ☒ Apenas as afirmativas 1, 3 e 5 estão corretas.
- ☐ Apenas as afirmativas 2 e 4 estão corretas.
- ☐ Apenas as afirmativas 1, 2 e 5 estão corretas.
- ☐ Apenas as afirmativas 3 e 4 estão corretas.
- ☐ Todas as afirmativas estão corretas.



Observe a seguinte URL:

*

[http://localhost:8084/server/CadastrarAluno?
txtNome=Gabriel&txtIdade=14&txtCurso=Engenharia](http://localhost:8084/server/CadastrarAluno?txtNome=Gabriel&txtIdade=14&txtCurso=Engenharia)

Nesta URL:

1 - localhost:8084 refere-se a um servidor web rodando localmente, enquanto **server/CadastrarAluno** é o caminho para um recurso. Além disso, a query string `txtNome=Gabriel&txtIdade=14&txtCurso=Computacao` contém parâmetros passados para o recurso.

Com base nesse cenário, analise as afirmações abaixo:

I. O **localhost:8084** refere-se a um servidor **HTTP**, enquanto a porta **8084** está configurada para atender requisições de serviços web específicos.

II. O **Caminho /server/CadastrarAluno** pode ser associado a um servidor que, ao receber os parâmetros de consulta, executa a lógica de manipulação de dados (como cadastrar um aluno em um banco de dados), podendo gerar um retorno para o usuário em formato HTML ou JSON.

III. A query string `txtNome=Gabriel&txtIdade=14&txtCurso=Computacao` contém os parâmetros que são enviados via método **GET**, e esses dados são visíveis na URL, o que pode implicar riscos de segurança, como a exposição de informações sensíveis.

Assinale a alternativa correta:

- ☐ Somente as afirmações I e II estão corretas.
- ☐ Somente as afirmações I e III estão corretas.
- ☐ Somente as afirmações II e III estão corretas.
- ☐ Todas as afirmações estão corretas.
- ☐ Nenhuma das afirmações está correta.



URL laboratório BD.

Sua resposta

Enviar

Limpar formulário

Nunca envie senhas pelo Formulários Google.

Este formulário foi criado em Computação UFCG.
Does this form look suspicious? [Relatório](#)

Google Formulários



