**Exercício 1: Setup Inicial**

1. Crie um novo projeto ASP.NET Web API.
2. Implemente um controlador simples com um método para retornar uma mensagem de boas-vindas.

**Exercício 2: Recursos Básicos**

1. Crie um controlador para gerenciar um recurso, como "Produtos".
2. Implemente métodos GET, POST, PUT e DELETE para manipular esse recurso.
3. Use uma lista em memória como fonte de dados para os exercícios iniciais.

**Exercício 3: Validação de Dados**

1. Adicione validação aos campos de entrada para evitar dados inválidos.
2. Utilize atributos de validação, como [Required], [MaxLength], etc.

**Exercício 4: Relacionamentos entre Recursos**

1. Crie dois recursos relacionados, como "Autores" e "Livros".
2. Implemente a lógica para associar livros a autores e buscar livros por autor.

**Exercício 5: Versionamento da API**

1. Explore diferentes abordagens para versionamento de APIs, como por meio do URI ou de cabeçalhos.
2. Implemente versionamento em sua API para garantir compatibilidade com versões futuras.

**Exercício 6: Autenticação e Autorização**

1. Implemente autenticação usando tokens JWT (JSON Web Tokens).
2. Adicione autorização para controlar o acesso a diferentes recursos da API.

**Exercício 7: Paginação e Ordenação**

1. Implemente paginação em listas longas de recursos.
2. Adicione suporte a ordenação por diferentes campos.

**Exercício 8: Upload de Arquivos**

1. Crie um endpoint para fazer upload de arquivos, como imagens.
2. Implemente a lógica para salvar os arquivos no servidor.

**Exercício 9: Documentação da API**

1. Use ferramentas como Swagger para gerar documentação interativa para sua API.
2. Documente os diferentes endpoints, parâmetros e respostas.

**Exercício 10: Tratamento de Erros**

1. Implemente um tratamento de erros robusto na sua API.
2. Retorne códigos de status HTTP apropriados e mensagens de erro detalhadas.

**Exercício 11: Testes de Unidade e Integração**

1. Escreva testes de unidade para os controladores da sua API.
2. Crie testes de integração para garantir que os endpoints estejam funcionando conforme o esperado.

**Exercício 12: Implementação de Cache**

1. Explore diferentes estratégias de cache, como cache de resposta ou cache de banco de dados.
2. Adicione caching apropriado para recursos que não mudam com frequência.

**Exercício 13: Implementação de Filtros e Buscas**

1. Adicione filtros que permitam aos usuários buscar recursos com base em diferentes critérios.
2. Implemente recursos de pesquisa usando parâmetros de consulta.

**Exercício 14: Trabalhando com Formatos de Resposta**

1. Suporte diferentes formatos de resposta, como JSON e XML.
2. Permita que os clientes solicitem o formato desejado.

**Exercício 15: Segurança Avançada**

1. Explore autenticação com OAuth 2.0 para permitir acesso seguro de terceiros.
2. Implemente fluxos de autorização, como o fluxo de concessão de código.