Universidade Federal do Agreste de Pernambuco – UFAPE
Curso de Bacharelado em Ciência da Computação
Disciplina: redes de computadores - 2022.2

Projeto de desenvolvimento utilizando conceitos de Redes de Computadores

Geisianny Bernardo
Gison Villaça
Rodrigo Leandro

Garanhuns-PE
2023

Nome da equipe: Protocol Pilots

Tema: Um servidor de chat em tempo real utilizando a tecnologia Docker, Node.js e o WebSocket.

Descrição do projeto de programação:

Este projeto tem como objetivo criar um servidor de chat em tempo real usando as tecnologias Docker, Node.js e WebSocket. O servidor permitirá que vários participantes se conectem e conversem em tempo real, compartilhando mensagens instantaneamente. A utilização do Docker garantirá a facilidade de implantação e escalabilidade, enquanto o Node.js e o WebSocket possibilitarão a comunicação bidirecional assíncrona em tempo real.

Benefícios:

- Isolamento: O uso do Docker isola o ambiente de execução do servidor, tornando-o consistente e independente do ambiente de hospedagem.
- Escalabilidade: A arquitetura baseada em contêineres facilita a escalabilidade horizontal, permitindo a adição de mais instâncias do servidor conforme necessário.
- Comunicação em Tempo Real: O WebSocket oferece comunicação assíncrona em tempo real, tornando-o ideal para aplicativos de chat e notificações em tempo real.
- Facilidade de Implantação: Através do Docker, implantar o servidor de chat em diferentes ambientes se torna mais simples e consistente.
- Aprendizado de Tecnologias: Este projeto oferece uma oportunidade de aprender sobre Docker, Node.js e WebSocket.

Objetivos:

- Demonstrar Comunicação em Tempo Real: Construir um servidor de chat que permita a comunicação instantânea entre múltiplos participantes, ilustrando o uso eficaz da tecnologia WebSocket.
- Explorar Tecnologias Modernas: Familiarizar-se com tecnologias relevantes, incluindo Docker para virtualização, Node.js para o desenvolvimento do servidor e WebSocket para comunicação assíncrona em tempo real.
- Compreender o Docker: Aprender a construir imagens Docker, executar contêineres e entender os princípios de isolamento e portabilidade que o Docker proporciona.
- Aplicar Conceitos de Rede: Ganhar experiência na configuração e gerenciamento de conexões de rede dentro de contêineres, incluindo mapeamento de portas e comunicação entre contêineres.

- Desenvolver uma interface de usuário Simples: Criar uma interface de chat em HTML para interação dos usuários, permitindo o envio e recebimento de mensagens em tempo real.
- Implementar Comunicação Bidirecional: Utilizar o WebSocket para estabelecer a comunicação bidirecional entre o servidor e os clientes, permitindo a entrega instantânea de mensagens.
- Testar Cenários de Uso: Experimentar o servidor de chat com vários usuários, observando como as mensagens são entregues a todos os participantes em tempo real.
- Praticar Melhores Práticas: Aplicar boas práticas de desenvolvimento, como modularização do código, tratamento de erros e segurança, para criar um aplicativo robusto.
- Aprender Implantação de Aplicativos: Compreender os conceitos de construção de imagens Docker e implantação de contêineres em um ambiente de produção ou teste.
- Aprimorar Habilidades de Solução de Problemas: Enfrentar desafios técnicos que podem surgir durante a construção e implantação do servidor de chat e aprender a solucioná-los de maneira eficaz.
- Promover a Colaboração: melhorar as habilidades de colaboração, incluindo compartilhamento de código, revisão de código e resolução conjunta de problemas.
- Preparar para Projetos Mais Complexos: Usar este projeto como base para compreender como criar aplicativos mais complexos em um ambiente Dockerizado, aprendendo a coordenar vários componentes.

Ao atingir esses objetivos, os participantes do projeto estarão mais bem preparados para desenvolver aplicativos de comunicação em tempo real, compreender os benefícios do Docker e aplicar seus conhecimentos em projetos futuros que envolvam tecnologias semelhantes.

Metodologia para o Projeto

Definição do Escopo:

- Identificar os requisitos do projeto, como funcionalidades essenciais do chat em tempo real, tecnologias a serem usadas (Docker, Node.js, WebSocket) e metas específicas.

Planejamento:

- Criar um plano detalhado, incluindo etapas do projeto, alocação de tarefas (caso seja uma equipe), cronograma e recursos necessários.

Configuração do Ambiente:

- Instalar e configurar as ferramentas necessárias, como Docker e Node.js, no ambiente de desenvolvimento.

Desenvolvimento do Servidor:

- Escrever o código do servidor Node.js que inclua a configuração do WebSocket para estabelecer comunicação em tempo real.

Criação da Interface de Chat:

- Desenvolver a interface HTML para o chat, incluindo a área de mensagens, caixa de entrada e botão de envio.

Teste Unitário:

- Realizar testes no servidor e na interface de chat para verificar se as mensagens são entregues corretamente e se a comunicação em tempo real está funcionando.

Construção do Dockerfile:

- Criar um Dockerfile para construir a imagem do servidor Node.js, incluindo todas as dependências necessárias.

Teste de Imagem Docker:

- Construir a imagem Docker e testar sua execução, certificando-se de que o servidor de chat funcione corretamente dentro do contêiner.

Implantação do Contêiner:

- Executar um contêiner Docker com base na imagem criada, garantindo que a comunicação em tempo real seja mantida.

Teste de Integração:

- Realizar testes de integração para verificar se a interface de chat é acessível e funcional a partir de um navegador.

Teste Multiusuário:

- Simular vários usuários em diferentes navegadores ou dispositivos para testar a funcionalidade de comunicação em tempo real.

Otimização e Refinamento:

- Avaliar a eficiência e o desempenho do projeto, otimizando-o conforme necessário.

Documentação:

- Criar documentação clara e concisa que inclua instruções para construção da imagem Docker, execução do contêiner e uso da interface de chat.

Encerramento:

- Realizar uma revisão final do projeto, garantindo que todas as funcionalidades estejam operando conforme o planejado.

Apresentação:

- Uma demonstração prática da aplicação de chat em tempo real em um vídeo com duração de até 10 minutos, protagonizado pelos membros do grupo.

Limpeza e Encerramento:

- Encerrar os contêineres e remover os recursos do projeto, como imagens e contêineres, do ambiente de desenvolvimento.

Ao seguir essa metodologia, temos o objetivo de desenvolver um servidor de chat em tempo real utilizando Docker, Node.js e WebSocket de maneira estruturada e eficiente. Isso permitirá a compreensão de conceitos-chave e a aplicação prática das tecnologias envolvidas.