

PROJETOS INFRAESTRUTURA DE COMUNICAÇÃO



Estrutura do Projeto

Semana	Carga Horária	Foco Principal (3h em aula)	Marco e Entrega
1	3h	Planejamento e Configuração: Definição da solução, divisão de tarefas, pesquisa de bibliotecas Python, configuração do ambiente de desenvolvimento.	Entrega: Documento de Planejamento Inicial (Tema, Escopo, Tecnologias, Divisão de Tarefas). Marco: Ambiente Python (Bibliotecas) configurado e código inicial (esqueleto) pronto para a funcionalidade principal.
2	3h	Desenvolvimento Básico: Implementação da funcionalidade central do projeto (parte mais básica da rede). Testes unitários iniciais.	Entrega: Protótipo funcional da funcionalidade principal em linha de comando. Marco: Funcionalidade de Rede Core implementada e testada.
3	3h	Refinamento e Interface: Adição de funcionalidades secundárias, tratamento de erros, desenvolvimento de uma interface simples (terminal ou web/GUI básica).	Entrega: Aplicação com interface básica e funcionalidades secundárias. Marco: Projeto funcionalmente completo e utilizável.
4	3h	Documentação e Preparação para Apresentação: Revisão final do código, testes em grupo, finalização da documentação (README, instruções), preparação da apresentação da 5ª semana.	Entrega: Versão final do código, Documentação (README/Manual), Roteiro da Apresentação. Marco: Projeto 100% finalizado e documentado, pronto para apresentação.

Propostas de Projetos (8 Temas)

1. Redes Sem Fio (Wi-Fi)

Tema Principal	Sugestões de Possibilidades (Aplicações)
Monitor de Sinal Wi-Fi	<p>Possibilidade 1: Coletar e exibir em tempo real (ou salvar em arquivo) informações como intensidade de sinal (RSSI) e MAC/SSID de redes Wi-Fi próximas (usando bibliotecas como <code>scapy</code> ou comandos do sistema operacional).</p> <p>Possibilidade 2: Criar uma ferramenta que varre uma rede Wi-Fi e exibe a lista de dispositivos conectados e seus IPs (requer bibliotecas de escaneamento de rede como <code>scapy</code> ou <code>python-nmap</code>).</p>
Entrega 5ª Semana: Aplicação Python em linha de comando ou com interface gráfica simples que varre e exibe dados relevantes de redes Wi-Fi ou dispositivos conectados.	

PROJETOS INFRAESTRUTURA DE COMUNICAÇÃO



2. Redes Ethernet

Tema Principal	Sugestões de Possibilidades (Aplicações)
Analizador/Emulador de Pacotes Ethernet	<p>Possibilidade 1: Criar um script Python que captura pacotes na interface de rede (usando <code>scapy</code>) e filtra/exibe informações específicas (MAC de origem/destino, tipo de protocolo, etc.).</p> <p>Possibilidade 2: Desenvolver um emulador simples que constrói e envia pacotes Ethernet com dados definidos pelo usuário (ex: um "ping" customizado, focando na camada 2).</p>
Entrega 5ª Semana: Aplicação que captura/analisa ou constrói/envia pacotes Ethernet, exibindo o conteúdo relevante na tela.	

3. Roteamento (Conceitos)

Tema Principal	Sugestões de Possibilidades (Aplicações)
Simulador de Tabela de Roteamento Simples	<p>Possibilidade 1: Criar um programa que simula um roteador lendo uma tabela de rotas estáticas (armazenada em um arquivo JSON/CSV) e determina a "melhor rota" para um IP de destino fornecido (simulando a decisão de roteamento).</p> <p>Possibilidade 2: Desenvolver uma ferramenta que usa comandos do sistema (como <code>tracert/traceroute</code>) e formata a saída de forma mais amigável, analisando o caminho do pacote.</p>
Entrega 5ª Semana: Aplicativo Python que simula a lógica de decisão de roteamento com base em uma tabela de rotas fornecida ou que analisa e formata o resultado de um <i>traceroute</i> .	

4. Redes Profinet (Conceito de I/O e Protocolos Industriais Simples)

Tema Principal	Sugestões de Possibilidades (Aplicações)
Emulador de Dispositivo I/O Simples	<p>Possibilidade 1: Criar um servidor TCP/UDP simples (usando <code>socket</code>) em Python que simula um dispositivo I/O (Input/Output) de campo. O "dispositivo" recebe comandos simples (ex: "LER_STATUS") e retorna um estado simulado (ex: "LIGADO", "DESLIGADO").</p> <p>Possibilidade 2: Desenvolver um cliente (usando <code>socket</code>) que se conecta ao servidor emulador da Possibilidade 1 e envia comandos para ler e "escrever" (mudar estado simulado) no dispositivo I/O.</p>
Entrega 5ª Semana: Aplicação Cliente/Servidor (local ou na mesma rede) onde o cliente simula um controlador Profinet enviando comandos simples para um servidor (dispositivo I/O).	

5. Redes de Sensores (Wireless Sensor Networks - WSN)

Tema Principal	Sugestões de Possibilidades (Aplicações)
Coleta e Visualização de Dados de Sensores	<p>Possibilidade 1: Criar um script que simula a geração de dados de sensores (temperatura, umidade) aleatoriamente, salva esses dados em um arquivo CSV/Banco de Dados Simples (SQLite) e exibe um gráfico básico (usando <code>matplotlib</code>).</p> <p>Possibilidade 2: Desenvolver um servidor (usando <code>socket</code> ou um micro-framework como Flask) que recebe dados simulados de "sensores" via requisições HTTP/TCP e armazena.</p>
Entrega 5ª Semana: Aplicação Python que simula, coleta, armazena e exibe (via terminal ou gráfico simples) dados de uma rede de sensores simulada.	

6. FDDI (Conceito de Token Ring e Falha)

Tema Principal	Sugestões de Possibilidades (Aplicações)
Simulador de Transmissão por Token Ring	<p>Possibilidade 1: Criar uma simulação em linha de comando que modela o conceito de <i>Token Ring</i> (ou o anel duplo do FDDI) com 4-5 nós. O programa deve simular a passagem de um "token" (variável/objeto) de nó para nó, garantindo que apenas o nó com o token pode "transmitir" (imprimir uma mensagem).</p> <p>Possibilidade 2: Implementar a lógica de detecção e recuperação de falha de nó no anel simulado, como em FDDI. Quando um nó falha, o token deve pular para o próximo nó, simulando a auto-reparação do anel.</p>
Entrega 5ª Semana: Simulação em linha de comando que demonstra a passagem do token e a exclusividade de transmissão em um anel de nós.	

7. Firewall (Conceito Básico)

Tema Principal	Sugestões de Possibilidades (Aplicações)
Simulador de Filtro de Pacotes (Firewall)	<p>Possibilidade 1: Desenvolver um script que simula um firewall, lendo regras de filtragem (ex: "BLOQUEAR IP X na porta Y") de um arquivo de configuração. O programa recebe como entrada um "pacote" (ex: IP de origem, porta de destino) e decide se ele deve ser PERMITIDO ou BLOQUEADO com base nas regras.</p> <p>Possibilidade 2: Criar uma ferramenta de teste de conectividade simples (usando a biblioteca <code>socket</code>) que verifica se uma porta específica em um IP remoto está aberta ou fechada, simulando o comportamento de um firewall.</p>
Entrega 5ª Semana: Aplicação que simula a filtragem de pacotes com base em um conjunto de regras predefinidas ou que testa o estado de portas (Port Scanner Básico).	

8. Redes na Computação em Nuvem

Tema Principal	Sugestões de Possibilidades (Aplicações)
Script de Gerenciamento de Rede na Nuvem (IaaS)	<p>Possibilidade 1: Usar a API do Google Cloud, AWS ou Azure (ou uma biblioteca de terceiros mais simples, como <code>boto3</code> para AWS) para criar um script Python que lista as Máquinas Virtuais (VMs) ou buckets de armazenamento existentes na conta do aluno (após configuração mínima e segura).</p> <p>Possibilidade 2: Criar uma aplicação web simples (usando Flask) que exibe o status de serviços de rede simulados. O script deve simular o monitoramento de recursos da nuvem (uptime, latência simulada) e exibir em uma página web local.</p>
Entrega 5ª Semana: Script Python que interage minimamente com uma plataforma de nuvem (leitura de recursos) ou uma aplicação web local que simula o monitoramento de rede na nuvem.	