INSTITUTO FEDERAL DO PIAUÍ
Curso de Superior de Tecnologia em Análise e Desenvolvimento de Sistemas
Programação para Internet II, Prof. Rogério Silva, ADS IV – 2025.2

Atividade Diagnóstica - Aprendizagem Ativa por Projetos

Projeto de Diagnóstico: "Controle de Domótica"

Objetivo: Este projeto tem como meta diagnosticar o nível de conhecimento técnico dos alunos em desenvolvimento para internet e a capacidade de organização da equipe e gestão de recursos em projeto com prazos curtos, tomada de decisão.

Os grupos deverão projetar e implementar uma aplicação web que simule o controle de dispositivos de uma casa inteligente.

A ser verificado:

- **Habilidades de Engenharia de Software:** Elaboração de diagramas e documentação.
- **Habilidades Técnicas em Backend:** Uso de frameworks, conexão e manipulação de dados em bancos de dados relacionais.
- **Habilidades Técnicas em Frontend:** Consumo de APIs, manipulação de interface de usuário e gerenciamento de estado.
- **Capacidade de Organização:** Versionamento de código (Git/GitHub), divisão de tarefas e trabalho em equipe.

Artefatos de Projeto

Antes de iniciar a codificação, cada grupo deve entregar e apresentar os seguintes artefatos de documentação, baseando-se no problema proposto:

1. **Lista de Casos de Uso:** Descrever textualmente as principais interações do usuário com o sistema.
    - *Exemplo:* "O usuário deve ser capaz de ligar/desligar um dispositivo."
2. **Diagrama de Caso de Uso:** Representar graficamente as interações listadas, mostrando os atores (usuário) e os casos de uso.
3. **Diagrama de Classes:** Modelar as entidades do sistema (Dispositivo, Cômodo, Cena) e seus relacionamentos, atributos e métodos.
4. **Diagrama de Entidade e Relacionamento (DER):** Modelar as entidades do banco de dados relacional (tabelas) e suas chaves primárias e estrangeiras, refletindo as relações entre elas.
5. **Documentação de API (Endpoints):** Listar e descrever cada endpoint da API REST que será desenvolvido no backend. Para cada endpoint, devem ser especificados:
    - **Method:** (GET, POST, PUT, DELETE)
    - **URL:** A rota do endpoint (ex: /api/dispositivos)
    - **Parâmetros (Query ou URL):** Quaisquer parâmetros necessários para a requisição.
    - **Corpo da Requisição (Body):** O formato JSON esperado para requisições POST ou PUT.
    - **Resposta (Response):** O formato JSON da resposta esperada para sucesso ou falha.

    - **Status Codes:** Possíveis códigos de status HTTP (200 OK, 201 Created, 404 Not Found, etc.).

Descrição Contexto

A casa é composta por diversos **Cômodos**, como a "Sala de Estar", a "Cozinha" ou o "Quarto". Dentro de cada um desses cômodos, existem vários **Dispositivos** que podem ser controlados. Pense em lâmpadas, ventiladores ou tomadas inteligentes. O sistema deve ser capaz de gerenciar cada um desses dispositivos individualmente, alternando seu estado entre Ligado e Desligado a qualquer momento.

O próximo desafio é criar o conceito de **Cena**. Uma cena é uma sequência de ações pré-definidas. Por exemplo, a cena "Chegar em Casa" pode ser programada para ligar a luz da sala, em seguida a luz da cozinha e, por fim, o ventilador do quarto, mas com um detalhe importante: um **Intervalo** de alguns segundos entre cada uma dessas ações. Esse intervalo permite uma transição suave e controlada. Outra cena, como "Modo Cinema", poderia desligar a luz principal, ligar uma luminária específica, tudo em uma ordem e com um ritmo definido com eventuais intervalos. Cenas podem estar desativadas temporariamente. É desejável previsão de comunicação com dispositivos físicos, de fato.

A Solução

A solução deve ser dividida em duas partes que se comunicam via API, se dominar, caso contrário fazer como conseguir, porém Web:

- **Backend:**
    - Deve ser implementado usando uma linguagem e tecnologias de preferência, mas utilizando o conhecimento prévio dos alunos. Evitar estudar conhecimentos novos.
    - Deve expor uma **API REST** para gerenciar todas as entidades (Dispositivo, Cômodo, Cena).
    - **Banco de Dados:** Usar **PostgreSQL** para persistir os dados.
- **Frontend:**
    - Deve ser uma aplicação web, desenvolvida com HTML, CSS e JavaScript puro (ou com uma biblioteca/framework que os alunos já dominam. Não estudar novo.).
    - Deve consumir a API do backend para realizar as operações de criar, ler, atualizar e deletar (CRUD).
- **Deploy:**
    - Se dominar, realizar
- **Mapa de Habilidades**:
    - Elaborar Mapa de Nivelamento de cada Membro com a seguinte escala:
        - A Dominar, Bom Domínio, Fluente* - para cada “conhecimento” utilizado pela equipe.
        - Não será publicado na turma, servirá de base para o Professor conduzir os próximos projetos e aulas.

Cronograma Detalhado

- **Aula 1:** Leitura da documentação, formação dos grupos e elaboração dos artefatos de projeto. Apresentação desses artefatos ao final da aula em aprox. 60 segundos, pode ser 100% verbal.

- **Aula 2 (Em casa/IFPI e Próxima aula):** Desenvolvimento e implementação do projeto. A duração da aula será dedicada a tirar dúvidas e fazer um acompanhamento de perto do andamento dos grupos.
- **Aula 3 (Terceira Aula):** Apresentação final das soluções. Cada grupo terá entre 05 e 10 minutos para demonstrar a aplicação funcionando e explicar as escolhas técnicas feitas.

OBS.: Não é obrigatório ou esperado a conclusão da atividade, visto ser diagnóstica. Entretanto já é atividade avaliativa. A não conclusão por deliberação terá nota zerada. Nota máxima 100.