Trabalho (Parte 1): Definição do contexto e dos requisitos.

- **Contexto do sistema**: Livre escolha
- Definir os requisitos do sistema. Mínimo: 10 requisitos
    - Ex1: O sistema não deve permitir cadastrar um cpf inválido;
    - Ex2: O sistema não deve permitir cadastrar a venda de um item que não existe em estoque;
    - Ex3: O sistema não deve permitir o cadastro de notas negativas ou maior 10
- **Pontos**: 2,0 (P1)
- **Data**: 18/10

Trabalho (Parte 2): Dados e modelos

- **Mínimo**: 5 tabelas (07/06)
- Criar as classes de modelos associados às tabelas
    - SQLAlchemy
- **Pontos**: 3,0 (P1)
- **Data**: 26/10

Trabalho (Parte 3): Criação da **API** usando Flask-restfull

- Endpoint para cadastro, edição, seleção e exclusão de cada tabela.
- Uso de arquitetura MVC.
- Estrutura de pacotes divididos por camada do MVC.
- **Pontos**: 5,0 (P1)
- **Data**: 08/11
- OBS: A API deve ser construída segundo o paradigma de Orientação a objetos: classes, herança, polimorfismo, construtor e encapsulamento.
  Além disso, deve possuir no mínimo uma vez o uso de: estruturas de condição, repetição, listas ou dicionários.

Trabalho (Parte 4): Criação da **API** usando Flask-restfull

- Implementação de um **jupyter notebook** explicando cada endpoint e demonstrando o seu funcionamento.
- **Pontos**: 3,0 (P2)
- **Data**: 22/11

Trabalho (Parte 5): Criação da Interface gráfica usando Qt ou Tkinter

- **Mínimo**: Telas de Cadastro, Edição, Seleção e Exclusão para cada domínio (tabela implementada na parte 2)
- Integração com a API
- **Pontos**: 7,0 (P2)
- **Data**: 29/11

- Individual
- Dois ou mais alunos não podem fazer o trabalho no mesmo contexto.
- Em caso de plágio da internet ou de algum outro trabalho a nota de ambos será desconsiderada;
- Pode-se usar o banco de dados que quiser;
- Todo o código deve ser feito em Python;

- Todas as partes do trabalho devem ser enviadas por email, mas precisam comparecer na aula para caso ocorram dúvidas ou problemas as mesmas serem solucionadas.