

## **Proposta de Trabalho - Projeto Aplicado Multiplataforma Etapa 2 (N708)**

### **Implementação do Sistema Multiplataforma**

#### **Conhecimentos necessários para o desenvolvimento da atividade**

Para o desenvolvimento desta atividade, os alunos deverão possuir ou desenvolver os seguintes conhecimentos:

- Desenvolvimento web frontend (HTML, CSS, JavaScript, frameworks)
- Desenvolvimento backend (APIs, serviços web)
- Desenvolvimento mobile (frameworks multiplataforma)
- Implementação de banco de dados
- Testes de software
- Técnicas de integração entre sistemas
- Versionamento de código com Git e GitHub
- Deploy e publicação de aplicações

#### **Objetivo da proposta de atividade**

Esta atividade tem como objetivo implementar o sistema multiplataforma que foi planejado e documentado na disciplina Projeto Aplicado Multiplataforma Etapa 1 (N705). Os alunos deverão transformar toda a documentação, protótipos e especificações técnicas em um produto de software funcional, desenvolvendo as interfaces, APIs, funcionalidades e integrações previamente definidas.

**Esta etapa foca exclusivamente na implementação do código e entrega do produto final (assim como as mudanças arquiteturais ocorridas no período), baseando-se nos artefatos de planejamento produzidos na Etapa 1 (disciplina N705).**

#### **Descrição da atividade**

Os alunos deverão implementar o sistema multiplataforma completo conforme as especificações e documentação elaboradas na Etapa 1 (N705). O projeto deve concretizar a solução tecnológica para o problema real identificado,

atendendo aos requisitos funcionais e não-funcionais estabelecidos na etapa anterior.

### **Requisitos técnicos obrigatórios:**

1. Implementar o frontend conforme os protótipos da Etapa 1
2. Desenvolver o backend especificado na documentação anterior
3. Implementar o banco de dados de acordo com o modelo definido
4. Garantir a integração funcional entre todos os componentes do sistema
5. Implementar tratamento de erros e validações
6. Desenvolver testes automatizados (unitários e/ou integração)
7. Documentar o código com comentários relevantes
8. Disponibilizar o sistema em ambiente de teste ou produção

### **Relação com a Etapa 1 (N705)**

Esta atividade é a continuação direta da disciplina Projeto Aplicado Multiplataforma Etapa 1 (N705). É fundamental que os alunos:

1. Utilizem os mesmos grupos formados na Etapa 1
2. Implementem o sistema com base na documentação e planejamento realizados anteriormente
3. Justifiquem eventuais mudanças em relação ao planejamento original
4. Mantenham a temática e vinculação ao ODS 11 (Cidades e Comunidades Sustentáveis)

### **Objetivo de Desenvolvimento Sustentável (ODS 11)**

O projeto deve continuar vinculado ao **ODS 11: Cidades e Comunidades Sustentáveis**, conforme estabelecido na Etapa 1. A implementação deve concretizar as contribuições planejadas para este objetivo, demonstrando como a tecnologia desenvolvida pode efetivamente contribuir para cidades mais inteligentes e sustentáveis.

A avaliação considerará como a implementação realiza o potencial de impacto social que foi identificado na fase de planejamento.

Para mais informações sobre o ODS 11, consulte: <https://brasil.un.org/pt-br/sdgs/11>

## **Validação com Público-Alvo Específico**

**Esta é uma exigência fundamental da atividade.** O projeto deve incluir comprovação de interface, apresentação, contato e validação com um público-alvo específico e **BEM DEFINIDO**.

## **Definição do Público-Alvo**

O público-alvo deve ser **específico e identificável**, não genérico. Exemplos:

### **Incorreto (muito genérico):**

- "Comerciantes de modo geral"
- "Moradores de bairros"
- "Usuários de transporte público"

### **Correto (específico e identificável):**

- "Seu José do Mercadinho Dois Irmãos, do bairro Bom Jardim"
- "Associação de Moradores do Conjunto Ceará"
- "Cooperativa de Catadores do Jangurussu"

## **Comprovação Obrigatória**

O grupo deve apresentar evidências de:

### **1. Identificação específica do público-alvo**

- Nome da pessoa/entidade
- Localização exata
- Contexto e necessidades específicas

### **2. Contato e apresentação do projeto**

- Comprovação de reunião/apresentação
- Registro fotográfico ou de vídeo (com autorização)
- Relatório da apresentação

### 3. Validação das funcionalidades

- Feedback do público-alvo sobre as funcionalidades
- Sugestões de melhorias recebidas
- Ajustes implementados baseados no feedback

### 4. Documentação da validação

- Relatório detalhado do processo de validação
- Evidências do contato realizado
- Resultados e aprendizados obtidos

## Entrega da Validação

A comprovação da validação com público-alvo deve ser entregue através do repositório GitHub na pasta `validation/`:

- **Relatório de Validação** (`validation/validation_report.md`)
- **Definição do Público-Alvo** (`validation/target_audience.md`)
- **Evidências fotográficas/vídeo** (`validation/evidence/` - com autorização dos participantes)
- **Feedback coletado** (`validation/feedback/`)
- **Inclusão no README.md** do projeto com resumo da validação

## Passo a passo para o desenvolvimento da atividade

### 1. Revisão da documentação da Etapa 1

- Análise dos requisitos e da arquitetura definida
- Revisão dos protótipos e especificações de APIs
- Identificação de possíveis ajustes necessários

### 2. Configuração do ambiente de desenvolvimento

- Preparação dos repositórios de código
- Configuração das ferramentas de desenvolvimento
- Setup do banco de dados

### 3. Desenvolvimento do backend

- Implementação das APIs conforme especificação
- Desenvolvimento da lógica de negócio
- Integração com o banco de dados

### 4. Desenvolvimento do frontend

- Implementação das interfaces conforme protótipos
- Desenvolvimento da lógica de apresentação
- Integração com as APIs do backend

#### 5. **Validação com Público-Alvo**

- Identificação específica do público-alvo
- Contato e apresentação do projeto
- Coleta de feedback e sugestões
- Implementação de ajustes baseados no feedback
- Documentação do processo de validação

#### 6. **Testes e Validação Técnica**

- Testes unitários e de integração
- Validação dos requisitos funcionais e não-funcionais
- Correção de bugs e ajustes

#### 7. **Finalização e Entrega**

- Deploy do sistema
- Documentação final
- Apresentação do produto

### **Orientações para o envio da atividade**

1. Todos os integrantes do grupo devem estar devidamente cadastrados na equipe via AVA.
2. Todos os representantes do grupo devem submeter no AVA, no local apropriado para envio da atividade:
  - Documento contendo nome completo de todos os integrantes e suas respectivas matrículas
  - No documento deve conter o link para o repositório GitHub contendo o código-fonte do projeto
  - **O Relatório de Validação com Público-Alvo deve estar no repositório GitHub (pasta validation/)**
3. O repositório GitHub deve conter:
  - README.md detalhado na raiz do projeto, contendo:
    - Objetivo do projeto
    - Funcionalidades implementadas
    - Tecnologias utilizadas

- Instruções de instalação e execução
    - Credenciais de teste (se aplicável)
    - Link para versão em produção ou vídeo demonstrativo
  - Código-fonte completo e organizado
  - Documentação técnica atualizada
4. O sistema deve estar funcional e testável pelos avaliadores, com instruções claras para execução.

## Estrutura do repositório (obrigatória)

**IMPORTANTE:** Trabalhos que não seguirem exatamente a estrutura proposta do repositório GitHub sofrerão penalização de 2,0 pontos na avaliação final.

```
/
├── README.md                # Documentação principal do projeto
├── docs/                    # Documentação técnica atualizada
│   ├── requirements/
│   │   └── requirements.md    # Requisitos atualizados
│   ├── architecture/
│   │   └── architecture.md    # Arquitetura final implementada
│   └── api/
│       └── api_documentation.md # Documentação da API implementada
├── validation/              # Validação com público-alvo
│   ├── target_audience.md    # Definição específica do público-alvo
│   ├── validation_report.md   # Relatório detalhado da validação
│   ├── evidence/              # Evidências fotográficas/vídeo
│   └── feedback/              # Feedback coletado e ajustes
├── frontend/                # Código do frontend
│   ├── web/                  # Implementação web (se aplicável)
│   │   ├── src/
│   │   ├── public/
│   │   └── package.json
│   └── mobile/               # Implementação mobile (se aplicável)
│       ├── src/
│       └── package.json
├── backend/                  # Código do backend
│   ├── src/
│   ├── tests/                # Testes automatizados
│   └── package.json
└── database/                 # Scripts e modelos de banco de dados
    └── schema.sql            # Esquema do banco de dados
```

## Orientações sobre a estrutura do repositório:

- **Obrigatória:** A estrutura deve ser seguida exatamente como especificada
- **Pastas opcionais:** Apenas `web/` ou `mobile/` dentro de `frontend/` (conforme o tipo de implementação)
- **Arquivos obrigatórios:** Todos os arquivos `.md` especificados devem estar presentes
- **Organização:** O código deve estar organizado nas pastas correspondentes
- **Penalização:** -2,0 pontos para estruturas que não sigam o padrão estabelecido

## Conteúdo obrigatório do README.md

1. **Título e descrição do projeto**
  - Nome do sistema
  - Breve descrição do propósito
  - Problema solucionado
2. **Funcionalidades implementadas**
  - Lista de funcionalidades principais
  - Status de implementação (completo, parcial)
  - Screenshots das telas principais
3. **Tecnologias utilizadas**
  - Linguagens de programação
  - Frameworks e bibliotecas
  - Banco de dados
  - Ferramentas de desenvolvimento
4. **Arquitetura do sistema**
  - Visão geral da arquitetura implementada
  - Componentes principais
  - Integrações realizadas
5. **Instruções de instalação e execução**
  - Pré-requisitos
  - Passo a passo para instalação



- Comandos para execução
- Configurações necessárias

#### **6. Acesso ao sistema**

- URL de acesso (se hospedado)
- Credenciais de teste

#### **7. Validação com Público-Alvo**

- Definição específica do público-alvo
- Resumo do processo de validação
- Principais feedbacks recebidos
- Ajustes implementados

#### **8. Equipe de desenvolvimento**

- Membros da equipe
- Papéis e contribuições principais

### **Critérios de avaliação (pontuação máxima de 5 pontos)**

#### **1. Funcionalidade e Qualidade - 1,8 pontos**

**1.1 Implementação completa dos requisitos funcionais - 0,6 pontos**

**1.2 Qualidade da interface e experiência do usuário - 0,6 pontos**

**1.3 Tratamento de erros e validações - 0,6 pontos**

#### **2. Arquitetura e Código - 1,2 pontos**

**2.1 Aderência à arquitetura planejada - 0,4 pontos**

**2.2 Qualidade e organização do código - 0,4 pontos**

**2.3 Implementação adequada do banco de dados - 0,4 pontos**

#### **3. Testes e Documentação - 1,2 pontos**

**3.1 Implementação de testes automatizados - 0,5 pontos**

**3.2 Qualidade da documentação técnica - 0,3 pontos**

**3.3 Instruções claras para instalação e execução - 0,4 pontos**

#### 4. Validação com Público-Alvo - 0,8 pontos

**4.1 Definição específica e identificável do público-alvo** - 0,3 pontos

**4.2 Comprovação de contato/apresentação realizada** - 0,3 pontos

**4.3 Documentação do feedback e ajustes implementados** - 0,2 pontos

#### Checklist para submissão

Para garantir que seu projeto atenda aos requisitos para avaliação, verifique se:

##### Implementação Técnica:

- O sistema implementa todos os requisitos funcionais definidos na N705
- O frontend está implementado conforme os protótipos da N705
- O backend e APIs estão funcionando corretamente
- O banco de dados está implementado conforme o modelo definido
- Existem testes automatizados para as principais funcionalidades
- O README.md contém todas as seções obrigatórias
- O sistema está disponível para teste (deploy ou instruções claras)
- O código está comentado e organizado
- **O repositório segue EXATAMENTE a estrutura de pastas e arquivos definida (PENALIZAÇÃO: -2,0 pontos)**
- **Repositório está público**

##### Validação com Público-Alvo:

- O público-alvo está especificamente identificado (nome, localização, contexto)
- Foi realizado contato/apresentação com o público-alvo específico
- Existe comprovação fotográfica/vídeo da apresentação (com autorização)

- Foi coletado feedback do público-alvo sobre as funcionalidades
- Foram implementados ajustes baseados no feedback recebido
- **Relatório detalhado da validação está no repositório GitHub (validation/validation\_report.md)**
- **Definição do público-alvo está no repositório (validation/target\_audience.md)**
- **Evidências fotográficas/vídeo estão na pasta validation/evidence/**
- **Feedback coletado está na pasta validation/feedback/**
- O README.md inclui resumo da validação realizada

## Penalizações

**ATENÇÃO: As seguintes penalizações serão aplicadas automaticamente:**

- **-2,0 pontos:** Repositório que não seguir exatamente a estrutura de pastas e arquivos definida
- **-2,0 ponto:** Ausência do relato de Validação com Público-Alvo no repositório GitHub
- **-0,5 pontos:** README.md incompleto ou sem seções obrigatórias
- **-5,0 pontos:** Sistema não funcional.
- **-5,0 pontos:** repositório inexistente, incorreto ou privado.

## Total de pontos

A atividade tem valor total de 5,0 pontos, que correspondem a 50% da nota do trimestre.

**Nota final = Pontuação obtida nos critérios de avaliação - Penalizações aplicadas**

## Prazo para o envio da atividade

A entrega da atividade deverá ser realizada até o dia **01/12/2025**, às 23h59, via AVA no local apropriado para submissão.

Não serão aceitas entregas por e-mail ou torpedo, assim como após o prazo estabelecido.

## **Ferramentas e Recursos**

- GitHub (versionamento): <https://github.com/>
- Heroku (deploy): <https://www.heroku.com/>
- Vercel (deploy frontend): <https://vercel.com/>
- Netlify (deploy frontend): <https://www.netlify.com/>
- Firebase (backend e hosting): <https://firebase.google.com/>
- Jest (testes): <https://jestjs.io/>
- Postman (testes de API): <https://www.postman.com/>