SERVIÇO NACIONAL DE APRENDIZAGEM COMERCIAL  
  
SENAC

CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROJETO INTEGRADOR – 2ª ETAPA  
CONSULTA FÁCIL

Integrantes do grupo:  
Gustavo Pereira de Carvalho  
Matheus Ferreira da Cunha Fonseca  
Ruan Rodrigues Maciel da Cruz  
Sérgio Antônio Luiz Pessoa Christofo  
Wesley Souza Xavier

Professor: Gustavo Calixto

EAD - ENSINO À DISTÂNCIA – 2025

# SUMÁRIO

1. 1. Introdução
2. 2. Prova de Conceito Definida
3. 3. Ambiente de Desenvolvimento
4. 4. Desenvolvimento do Frontend
5. 5. Desenvolvimento do Backend
6. 6. Banco de Dados
7. 7. GitHub (repositório e colaboração)
8. 8. Vídeo (roteiro e checklist)
9. 9. Conclusão
10. Referências Bibliográficas

# 1. Introdução

Este documento apresenta a Segunda Etapa do Projeto Integrador do curso de TADS/TSI, dando continuidade à proposta do sistema ‘Consulta Fácil’. Na Primeira Etapa, foram definidos: a visão de produto, perfis de personas (Ana, Pedro e Dona Lúcia), jornadas do usuário e o fluxograma macro do serviço. A partir desses insumos, esta etapa tem como objetivo transformar o conceito em uma prova de conceito funcional, cobrindo frontend, backend e persistência de dados.  
  
Objetivos específicos desta etapa:  
• Selecionar uma jornada representativa e condensá-la em um recorte implementável (MVP/PoC).  
• Especificar o ambiente tecnológico (stack, ferramentas, dependências) e práticas de engenharia.  
• Implementar um frontend responsivo e acessível para as principais interações (login, busca e agendamento).  
• Implementar um backend em PHP com endpoints REST e repositório de dados relacional.  
• Fornecer roteiros de execução, testes básicos e diretrizes de colaboração (Git/GitHub).  
  
Critérios de sucesso:  
• Fluxo de ‘buscar médico’ → ‘ver perfil’ → ‘agendar’ funcionando ponta a ponta.  
• Código executável localmente com instruções claras.  
• Base de dados inicial com registros simulados de médicos e um usuário de teste.  
• Vídeo de até 1 minuto demonstrando o fluxo completo.

# 2. Prova de Conceito Definida

A prova de conceito foca na jornada da persona Ana (mãe ocupada), validando os pontos críticos de valor: rapidez, praticidade e confirmação do agendamento. O recorte selecionado inclui:  
  
Escopo funcional do MVP:  
• Autenticação simples (login/cadastro) do paciente.  
• Busca de médicos por especialidade e localização (campo livre + opções populares).  
• Exibição de um cartão de perfil do médico (nome, especialidade, região, avaliação média simulada).  
• Seleção de um horário disponível e confirmação de agendamento com número de protocolo.  
  
Fora de escopo (para versões futuras):  
• Pagamento online, integração com convênios e mensageria transacional (e-mail/SMS em produção).  
• Avaliações reais, upload de documentos e prontuário eletrônico.  
  
Métricas de validação qualitativas:  
• Número de passos até concluir um agendamento (redução de burocracia).  
• Clareza das mensagens e feedbacks de erro/sucesso.  
• Acessibilidade: contraste, tamanho de fonte e legibilidade em telas menores.

# 3. Ambiente de Desenvolvimento

Stack selecionada e justificativas:  
Frontend:  
• HTML5, CSS3 e JavaScript Vanilla para base e didática.  
• Bootstrap 5 para responsividade rápida e componentes acessíveis.  
• jQuery para interações simples (requisições AJAX e manipulação do DOM).  
  
Backend:  
• PHP 8.x – aderente à disciplina e amplamente suportado.  
• Slim Framework (ou Laravel como alternativa) para organizar rotas REST e middlewares.  
• PDO para acesso seguro ao MySQL, com prepared statements.  
  
Banco de Dados:  
• MySQL 8 – relacional, gratuito e com excelente integração ao PHP.  
  
Ferramentas e execução local:  
• XAMPP/Laragon/Docker (opcional) para stack LAMP pronta.  
• Postman/Insomnia para testar endpoints da API.  
• Git + GitHub para versionamento e colaboração.  
  
Padrões e boas práticas adotadas:  
• Organização por camadas (API → serviço → repositório → DB).  
• Arquivo .env (ou config.php) com variáveis de ambiente (host, usuário, senha, banco).  
• Hash de senha com password\_hash() (bcrypt/argon2).  
• Validação e sanitização de entrada no backend.  
• CORS configurado apenas para o frontend do projeto.

# 4. Desenvolvimento do Frontend

Páginas principais e requisitos de UX:  
• index.html – Login e cadastro rápido.  
• buscar.html – Filtros por especialidade e localização, lista de médicos.  
• agendar.html – Perfil do médico e seleção de horários.  
  
Componentes e padrões:  
• Navbar com logo e acesso ao perfil (placeholder).  
• Cards de médicos com dados essenciais e botão ‘Agendar’.  
• Formulários com validação de campos e mensagens de erro claras.  
  
Acessibilidade e responsividade:  
• Tipografia mínima 16px, botões grandes e contraste suficiente (WCAG AA).  
• Layout fluido (grid/bootstrap) para smartphones e desktops.  
  
Fluxo do usuário (Ana):  
1) Login → 2) Busca ‘Pediatria’ em ‘Goiânia’ → 3) Visualiza médicos → 4) Seleciona horário → 5) Confirma agendamento.  
  
Interações com a API:  
• POST /auth/login, POST /auth/register  
• GET /medicos?especialidade=&localizacao=  
• POST /consultas (id\_medico, data\_hora)

# 5. Desenvolvimento do Backend

Arquitetura da API (exemplo com Slim):  
• /auth/login – autenticação e emissão de sessão simples.  
• /auth/register – cadastro com hashing de senha.  
• /medicos – listagem/consulta por filtros.  
• /consultas – criação e listagem por usuário autenticado.  
  
Segurança e validação:  
• Sanitização de entradas (filter\_input, regex e validações de domínio).  
• Prepared statements via PDO.  
• Rate limit básico por IP (middleware simples) e logs de erro.  
  
Testes básicos:  
• Status HTTP e payloads esperados (200/201/400/401/404).  
• Fluxos felizes e cenários de erro (usuário inexistente, credenciais inválidas, horário indisponível).

# 6. Banco de Dados

Modelo Entidade-Relacionamento (simplificado):  
• usuarios(id, nome, email, senha\_hash)  
• medicos(id, nome, especialidade, localizacao, avaliacao\_media)  
• consultas(id, id\_usuario, id\_medico, data\_hora, protocolo)  
  
Regras principais:  
• Um usuário pode ter N consultas; um médico pode atender N consultas.  
• Índices em (especialidade, localizacao) para busca eficiente.  
• Constrangimentos de integridade referencial (FK) e unicidade de email.  
  
Script SQL (trecho):  
CREATE TABLE usuarios (  
 id INT AUTO\_INCREMENT PRIMARY KEY,  
 nome VARCHAR(120) NOT NULL,  
 email VARCHAR(160) NOT NULL UNIQUE,  
 senha\_hash VARCHAR(255) NOT NULL  
);  
  
CREATE TABLE medicos (  
 id INT AUTO\_INCREMENT PRIMARY KEY,  
 nome VARCHAR(120) NOT NULL,  
 especialidade VARCHAR(80) NOT NULL,  
 localizacao VARCHAR(120) NOT NULL,  
 avaliacao\_media DECIMAL(2,1) DEFAULT 4.5  
);  
  
CREATE TABLE consultas (  
 id INT AUTO\_INCREMENT PRIMARY KEY,  
 id\_usuario INT NOT NULL,  
 id\_medico INT NOT NULL,  
 data\_hora DATETIME NOT NULL,  
 protocolo VARCHAR(20) NOT NULL,  
 FOREIGN KEY (id\_usuario) REFERENCES usuarios(id),  
 FOREIGN KEY (id\_medico) REFERENCES medicos(id)  
);

# 7. GitHub (repositório e colaboração)

Estratégia de versionamento e workflow:  
• Branches: main (estável), dev (integração), feature/\* (funcionalidades).  
• Pull Requests com revisão cruzada entre membros e checklist de testes.  
• Commits semânticos (feat:, fix:, docs:, chore:, refactor:).  
  
Conteúdo mínimo do repositório:  
• /frontend, /backend, /db, README.md, LICENSE, .gitignore, docs/.  
• README.md com: visão do projeto, setup local (XAMPP/Laragon), variáveis de ambiente, scripts SQL, rotas da API, créditos.  
  
Checklist para entrega:  
• Todos os membros adicionados como colaboradores.  
• Histórico de contribuições distribuído.  
• Link do vídeo incluído no README.

# 8. Vídeo (roteiro e checklist)

Roteiro (até 60 segundos):  
0–5s: Tela inicial + logo.  
5–15s: Login do usuário de teste.  
15–35s: Busca por ‘Pediatria’ em ‘Goiânia’ e visualização dos médicos.  
35–55s: Seleção de horário e confirmação de agendamento (mostrar protocolo).  
55–60s: Tela de sucesso + mensagem final.  
  
Checklist de gravação:  
• Usar resolução 1280x720 (720p) para leveza.  
• Narrar ou incluir legendas curtas.  
• Evitar dados sensíveis; usar base simulada.  
• Exportar em MP4 (H.264) e anexar ao GitHub ou subir no YouTube como ‘Não listado’.

# 9. Conclusão

A PoC desenvolvida para o ‘Consulta Fácil’ confirma a viabilidade do fluxo de agendamento online, atacando as principais dores identificadas nas personas: tempo, clareza e simplicidade. A escolha da stack (PHP + MySQL + Bootstrap) prioriza curva de aprendizado, suporte e velocidade de entrega, sem abrir mão de boas práticas de segurança, organização e testes. Como próximos passos, recomendam-se: integração com provedores de e-mail/SMS, autenticação JWT, agenda médica com bloqueios por conflito, relatórios de comparecimento e módulo de avaliação real pelos pacientes.

# Referências Bibliográficas

BÜRDEK, B. E. Design: História, Teoria e Prática do Design de Produtos. São Paulo: Editora Blucher, 2006.  
  
DAMAZIO, F.; FREIRE, K. Design de serviços: conceitos e reflexões sobre o futuro da disciplina. In: Congresso Brasileiro de P&D em Design, 9., 2010, São Paulo. Anais... 2010.  
  
Documentação oficial:  
• PHP Manual – php.net  
• MySQL Reference Manual – dev.mysql.com  
• Bootstrap – getbootstrap.com  
• Slim Framework – slimframework.com