

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
RIO GRANDE DO SUL
CAMPUS CANOAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO
DE SISTEMAS

PEDRO ROLIN SCHMITZ

**Ocean - Study Flow Sync: MVP web para
organização de estudos com integração ao Moodle.**

Canoas, 2025.

PEDRO ROLIN SCHMITZ

**Ocean - Study Flow Sync: MVP web para
organização de estudos com integração ao Moodle.**

Proposta de Trabalho de Conclusão de
Curso apresentada como requisito parcial
para obtenção do grau de Tecnólogo em
Análise e Desenvolvimento de Sistemas
pelo Instituto Federal de Educação,
Ciência e Tecnologia do Rio Grande do Sul
– Campus Canoas.

Prof(a). Dr(a). Não definido

Prof(a). Dr(a). Não definido

Canoas, 2025.

SUMÁRIO

1	INTRODUÇÃO	4
1.1	MOTIVAÇÃO	5
1.2	OBJETIVOS	6
1.2.1	Objetivo Geral	6
1.2.2	Objetivos Específicos	6
2	METODOLOGIA	7
3	CRONOGRAMA	8
4	REFERENCIAL TEÓRICO	9
4.1	Ambientes Virtuais de Aprendizagem (AVA)	9
4.2	Moodle	9
4.3	Arquitetura Cliente-Servidor	9
4.4	Tecnologias de Desenvolvimento utilizadas	10
4.4.1	HTML e CSS	10
4.4.2	Bootstrap	10
4.4.3	PHP	10
4.4.4	MySQL	11
4.5	Progressive Web App	11
4.6	APIs REST e Comunicação com Servidores	11
5	ESTADO DA ARTE	12
5.1	Agenda escolar	12
5.2	Estudaqui	13
5.3	Notion	14
5.4	Conclusão	15
	REFERÊNCIAS	16

1 INTRODUÇÃO

O projeto teve início a partir da identificação de um problema recorrente: a falta de organização, centralização de informações e dificuldade de gerenciamento das tarefas/atividades relacionadas ao contexto proposto. Observou-se que os usuários realizavam seus processos de forma manual ou fragmentada, ocasionando atrasos, inconsistências e dificuldades na visualização geral do fluxo de trabalho.

Diante disso, definiu-se como objetivo principal desenvolver um **MVP (Minimum Viable Product)** capaz de validar a necessidade e utilidade de uma aplicação simples, funcional e objetiva, permitindo testar suas principais funcionalidades antes da construção de uma versão completa. O MVP foi planejado para entregar o essencial: cadastro, manipulação e organização das informações principais do sistema, além de disponibilizar uma interface acessível e lógica.

2 METODOLOGIA

Durante o desenvolvimento, diferentes metodologias e processos foram utilizados para garantir organização e clareza na execução das tarefas:

2.1 Metodologia MVP

Optou-se pela abordagem MVP para validar rapidamente a ideia, lançando primeiro uma versão mínima com as funcionalidades essenciais. A partir disso, a coleta de feedbacks orientou melhorias e ajustes.

2.2 Modelagem UML

Foram produzidos diagramas de classe e sequência para representar a estrutura do sistema, suas entidades, relacionamentos e o fluxo de interações. Isso permitiu visualizar a lógica antes da implementação.

2.3 Mapeamento Objeto-Relacional

A modelagem do banco de dados foi realizada considerando o alinhamento entre classes (no código) e tabelas (no banco), garantindo um design coerente com o paradigma orientado a objetos.

2.4 Controle de Versão

Utilizou-se Git/GitHub para registrar versões, organizar tarefas, armazenar código e permitir revisões. Isso facilitou o acompanhamento da evolução e evitou perdas de informação.

2.5 Processo Iterativo

O desenvolvimento ocorreu em ciclos curtos, adicionando e testando funcionalidades progressivamente para reduzir erros e melhorar a estabilidade.

3 DEFINIÇÃO DO ESCOPO DO PROJETO

O escopo do MVP incluiu somente o essencial para validar a proposta inicial. Entre as funcionalidades planejadas, estavam:

- Cadastro e gerenciamento dos principais dados do sistema.
- Interface simples para navegação e execução das tarefas.
- Banco de dados estruturado para armazenar as informações essenciais.
- Fluxos básicos de operação (criar, editar, excluir e visualizar).
- Estrutura modular para expansão futura.

Funcionalidades mais complexas — como automações, relatórios avançados, dashboards e módulos extras — foram deliberadamente deixadas fora do MVP, conforme previsto na abordagem enxuta.

4 DESENVOLVIMENTO DO PROJETO

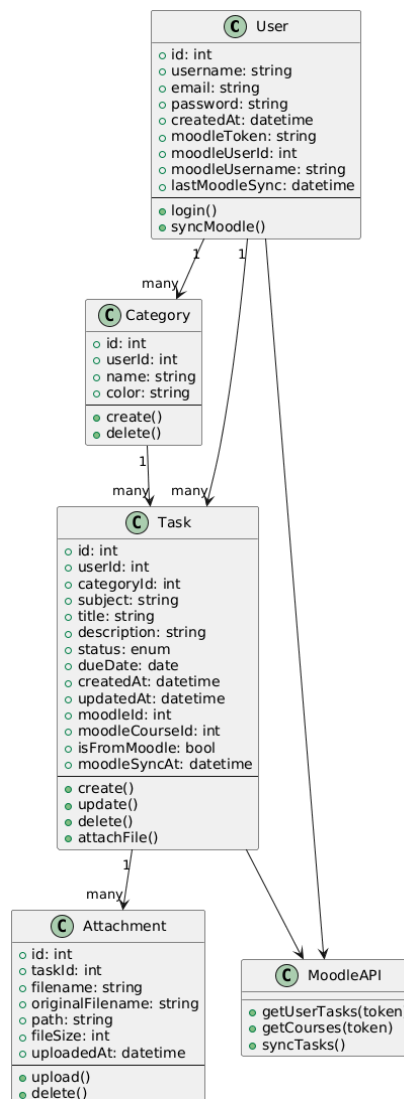
4.1 Análise e Projeto Orientado a Objetos

O processo iniciou com a identificação das entidades do sistema, suas responsabilidades e interações. A partir disso, foram definidas classes, atributos, métodos e relacionamentos, seguindo princípios da orientação a objetos, como coesão, encapsulamento e responsabilidade única.

4.2 Diagramas de Classe e de Sequência

Diagrama de Classes: Representou as entidades principais, seus atributos e relações (associação, agregação ou composição).

Figura 1.1: Diagrama de Classes



Fonte: autor.

Diagramas de Sequência: Ilustram o comportamento do sistema ao longo do tempo, mostrando como objetos se comunicam para executar operações essenciais.

Esses diagramas foram fundamentais para estruturar o código e prever possíveis problemas lógicos.

4.3 Projeto do Banco de Dados

O banco foi projetado com foco em consistência, normalização e clareza. As entidades possuem chaves primárias, relacionamentos bem definidos e tipos de dados adequados para as operações previstas no MVP.

4.4 Diagrama Entidade-Relacionamento (ER)

O DER apresentou visualmente as entidades, suas chaves primárias, chaves estrangeiras e cardinalidades. Isso orientou a implementação correta das tabelas e evitou conflitos de integridade.

4.5 Seleção e Configuração de Frameworks e Tecnologias

As tecnologias foram escolhidas considerando simplicidade, documentação e desempenho:

- XAMPP para servidor local
- PHP como linguagem principal
- MySQL como SGBD
- HTML, CSS e JS para interface
- ORM/Modelos PHP para abstração do banco
- Git para versionamento

Após definidos, todos os frameworks foram configurados e integrados ao ambiente de desenvolvimento.

4.6 Desenvolvimento da Interface do Usuário

A interface foi criada com foco em simplicidade, usabilidade e navegação clara. Os elementos foram posicionados para garantir:

- Fluxo intuitivo
- Design responsivo básico
- Facilidade no cadastro e manipulação de dados
- Redução de erros por parte do usuário

5 RESULTADOS

5.1 Validação da Hipótese do MVP

A hipótese inicial — de que seria possível criar um sistema simples, funcional e organizado, que atendesse às necessidades básicas dos usuários — foi validada.

O MVP demonstrou que a arquitetura proposta funciona, o banco de dados responde adequadamente e a interface permite o uso sem dificuldades.

5.2 Resultados Obtidos

- CRUD implementado com sucesso
- Navegação funcional
- Banco de dados estruturado e consistente
- Estrutura de pastas organizada
- Testes realizados com sucesso
- Melhorias aplicadas com base em problemas encontrados (ex.: validações, ajustes visuais, correções de fluxo)

6 CONCLUSÃO

O que você aprendeu com o trabalho

- A importância de planejar antes de programar.
- Como a modelagem UML facilita o desenvolvimento.
- Como separar o essencial do opcional em um MVP.
- A utilidade do controle de versão e testes iterativos.
- Práticas fundamentais de orientação a objetos e banco de dados.

O que teria feito diferente

- Refinar mais cedo a interface do usuário.
- Dedicar mais tempo ao planejamento do banco de dados.
- Implementar testes automatizados desde o início.
- Dividir as entregas em sprints menores para facilitar acompanhamento.

Próximos Passos

- Expandir funcionalidades que ficaram fora do escopo.
- Implementar novos módulos e relatórios.
- Melhorar a UI/UX com base nos feedbacks.
- Adicionar autenticação avançada e camadas extras de segurança.
- Criar uma versão mais robusta e completa após validação total do MVP.