



Evento: XXVI Jornada de Extensão ▾

PROJETO APPGO: ARQUITETURA BACKEND DO PHYSICGO¹

**Giordano Bruno Biasi Berwig², Cássia Regina Fracaro Polleto³, Barbara Gündel⁴,
Edson Luiz Padoin⁵, Patrícia Carolina Pedrali⁶, Taíse Neves Possani⁷**

¹ Projeto de Extensão AppGO - Desenvolvimento e Implementação de Softwares Educacionais (Unijui).

² Giordano Bruno Biasi Berwig - Bolsista PIBEX; acadêmico de Ciência da Computação; giordano.bruno@sou.unijui.edu.br

³ Cássia Regina Fracaro Polleto - Bolsista PIBEX; acadêmica de Engenharia de Software; cassia.polleto@sou.unijui.edu.br

⁴ Professor Extensionista - Curso de Design - barbara.gundel@unijui.edu.br

⁵ Professor Extensionista - Curso de Ciência da Computação - padoin@unijui.edu.br

⁶ Professor Extensionista - Curso de Engenharia Mecânica - patricia.pedrali@unijui.edu.br

⁷ Professor Extensionista - taíse.possani@unijui.edu.br

INTRODUÇÃO

Este trabalho tem como propósito apresentar uma análise detalhada do progresso e das soluções tecnológicas desenvolvidas no âmbito do Projeto AppGO, com foco nos aplicativos PortGo, PhysicGo e MathGo. Com o intuito de proporcionar uma organização metodológica e clareza na exposição das informações, o conteúdo foi cuidadosamente estruturado em eixos centrais que abordam os pilares tecnológicos do *backend*: 1. Fundamentos da Arquitetura Laravel: MVC e Estrutura de Código, 2. Gestão de Dados e Persistência: MySQL e Eloquent ORM, e 3. Segurança, Otimização e Interatividade Assíncrona no *Backend*. Essa abordagem permitirá compreender a engenharia por trás desses aplicativos, essenciais para sua performance e funcionalidade.

METODOLOGIA

A metodologia empregada neste estudo adota uma abordagem qualitativa, direcionada à análise aprofundada dos elementos que compõem os aplicativos desenvolvidos no escopo do Projeto AppGO. Caracteriza-se, igualmente, como uma pesquisa descritiva, buscando detalhar os aspectos técnicos e os fundamentos conceituais envolvidos na idealização e implementação do PhysicGo. Adicionalmente, este trabalho fundamenta-se em uma pesquisa bibliográfica, uma vez que todas as etapas de sua construção foram embasadas em fontes previamente publicadas e reconhecidas, incluindo literatura especializada, artigos científicos e documentos relevantes nas áreas de educação e tecnologia da informação, garantindo a solidez e a credibilidade das informações apresentadas.



RESULTADOS E DISCUSSÃO

1. Fundamentos da Arquitetura Laravel: MVC e Estrutura de Código

A base tecnológica do *backend* do PhysicGo (app para o estudo de Física) reside no Laravel (2025), um proeminente framework PHP, escolhido por sua elegância e robustez. Ele se estrutura firmemente no padrão de arquitetura MVC (*Model-View-Controller*), para garantir uma clara separação de responsabilidades. Essa abordagem metodológica é fundamental para o desenvolvimento de aplicações escaláveis e de fácil manutenção, permitindo que componentes distintos do sistema sejam desenvolvidos e gerenciados de forma independente. Ao adotar o MVC, o PhysicGo assegura que a lógica de negócio, a gestão de dados e a apresentação ao usuário sejam tratadas em camadas isoladas, promovendo um código limpo, organizado e sustentável a longo prazo, otimizando o fluxo de trabalho da equipe de desenvolvimento.

Dentro dessa arquitetura, o Model atua como a camada de dados e lógica de negócio do PhysicGo, sendo o elo direto com o banco de dados. Cada Model corresponde a uma entidade do sistema (como usuários, perguntas de física ou opções de resposta), gerenciando operações básicas de persistência de dados e encapsulando regras de negócio específicas e relacionamentos entre entidades, utilizando o poderoso Eloquent ORM. Paralelamente, a View constitui a camada de apresentação, responsável por tudo que o usuário visualiza e interage. No Laravel, as Views são construídas com o sistema de *templates* Blade, que oferece uma sintaxe intuitiva para incorporar lógica de exibição de dados dinamicamente, garantindo fidelidade ao protótipo visual do PhysicGo, inicialmente planejado no Figma.

O Controller atua como o intermediário dinâmico e orquestrador central entre o Model e a View, processando todas as requisições que chegam ao backend do PhysicGo. Quando uma interação do usuário é detectada, o sistema de rotas do Laravel direciona essa requisição HTTP para o método apropriado dentro de um Controller. Este coordena as ações necessárias: interage com os Models para buscar ou manipular dados relevantes, aplica qualquer lógica de processamento específica e prepara os dados para serem exibidos pela View correspondente. Essa coordenação eficiente é vital para que o PhysicGo responda dinamicamente às interações, integrando a lógica de *backend* com a experiência visual do usuário e, por fim, garantindo a manutenção e expansão contínua do sistema. A segregação de responsabilidades



permite que as equipes trabalhem de forma independente, acelerando o processo e reduzindo o risco de *bugs*, essencial para a adaptação a novas demandas educacionais e futuras funcionalidades.

2. Gestão de Dados e Persistência: MySQL e Eloquent ORM

A gestão eficiente dos dados é um pilar insubstituível para qualquer aplicação interativa moderna, e o PhysicGo não é exceção. A escolha estratégica do MySQL como Sistema de Gerenciamento de Banco de Dados (SGBD) para o backend do projeto fundamenta-se em sua reconhecida robustez, comprovada performance, escalabilidade intrínseca e na vasta comunidade de suporte que o cerca. Sendo um banco de dados relacional amplamente consolidado no mercado, o MySQL oferece a confiabilidade necessária para armazenar e organizar grandes volumes de informações críticas, como o extenso acervo de perguntas e respostas de física, os dados detalhados dos usuários e seus progressos individuais, além de todas as configurações e lógicas subjacentes que regem a experiência de aprendizado no aplicativo. Sua capacidade de lidar com múltiplas transações simultâneas e garantir a integridade referencial dos dados é vital para sustentar um ambiente de aprendizado dinâmico, onde diversos usuários interagem em tempo real e de forma contínua.

A interação do *backend* do PhysicGo com o banco de dados MySQL (ORACLE, 2025) é, predominantemente, mediada pelo Eloquent ORM (Object-Relational Mapping), uma ferramenta poderosa e inerentemente elegante do Laravel. O Eloquent opera como uma camada de abstração, traduzindo as complexas operações de banco de dados SQL em métodos e propriedades de objetos PHP intuitivos, eliminando a necessidade de escrever consultas SQL diretamente na maioria dos casos. Essa abordagem não apenas eleva a legibilidade do código e a produtividade, mas também reduz significativamente a incidência de erros tipicamente associados à manipulação direta de SQL. Um dos maiores benefícios do Eloquent ORM é sua capacidade de simplificar drasticamente a gestão de relacionamentos entre dados, o que é fundamental para a estrutura interconectada do PhysicGo. Ele permite definir relações complexas (como "um-para-muitos" ou "muitos-para-muitos") diretamente nos Models com poucas linhas de código, possibilitando carregar dados relacionados de forma otimizada (preguiçosa ou ansiosa), essencial para a rede de informações que sustenta o aprendizado interativo e a recuperação rápida de dados.



Para assegurar a evolução controlada e versionada do esquema do banco de dados do PhysicGo, são empregadas as Migrations do Laravel. As Migrations funcionam como um sistema de "controle de versão" para o banco de dados, permitindo que a equipe de desenvolvimento defina e modifique o esquema das tabelas por meio de código PHP, garantindo que as alterações sejam aplicadas de forma consistente em diferentes ambientes. Complementarmente, as Seeders do Laravel são utilizadas para popular o banco de dados com dados iniciais ou de teste após a execução das Migrations. Essa combinação de Migrations e Seeders oferece uma solução robusta e completa para o ciclo de vida do banco de dados, assegurando que o armazenamento e a manipulação de dados no PhysicGo sejam eficientes, seguros e facilmente gerenciáveis ao longo do tempo.

3. Segurança, Otimização e Interatividade Assíncrona no Backend

A segurança é um pilar inegociável no desenvolvimento do PhysicGo, e fundamental devido à sensibilidade dos dados de usuários e do conteúdo educacional. O Laravel, como framework robusto, integra um conjunto abrangente de recursos que protegem o aplicativo contra as vulnerabilidades *web* mais comuns. Isso inclui defesas contra SQL Injection (com *prepared statements* do Eloquent ORM), CSRF (Cross-Site Request Forgery) (com *tokens* únicos por sessão) e XSS (Cross-Site Scripting) (com *escaping* automático de conteúdo no Blade). Adicionalmente, o sistema de autenticação do Laravel emprega *hashing* de senhas, garantindo que as credenciais nunca sejam armazenadas em texto simples, e a gestão de níveis de acesso controla as funcionalidades por tipo de usuário, reforçando a confidencialidade e a integridade das informações em toda a plataforma.

Para garantir uma experiência de usuário fluida e responsiva, o PhysicGo prioriza a otimização de performance e a interatividade assíncrona. Uma estratégia chave é a entrega de perguntas uma por vez, onde o servidor envia cada questão sob demanda, otimizando a quantidade de dados transferidos e reduzindo drasticamente o tempo de carregamento. A interatividade assíncrona é um diferencial alcançado pela sinergia entre o *backend* do Laravel e tecnologias de *frontend* como AJAX e jQuery. Quando o usuário interage, o *frontend* envia uma requisição AJAX para o *backend*; o Controller processa essa requisição e retorna uma resposta em JSON. O jQuery, então, atualiza dinamicamente o conteúdo da página sem a



necessidade de recarregar, o que melhora significativamente a fluidez da interação, tornando a navegação mais agradável e contínua e otimizando o desempenho geral da aplicação.

CONSIDERAÇÕES FINAIS

Este trabalho detalhou o desenvolvimento das soluções educacionais do Projeto AppGO, com uma análise aprofundada focada no PhysicGo. A criação de uma interface intuitiva, aliada à implementação de um *backend* robusto, foi essencial para a funcionalidade e o impacto da plataforma. A metodologia qualitativa empregada permitiu uma exploração minuciosa dos aspectos técnicos, evidenciando a utilização estratégica de ferramentas modernas como Figma para o design, e Laravel e jQuery para o desenvolvimento, o que sublinha a aplicação de boas práticas de arquitetura de software e design centrado no usuário em todo o processo.

A abordagem de entrega progressiva de conteúdo e a responsividade da interface do PhysicGo contribuíram significativamente para uma experiência de aprendizado mais fluida e acessível aos usuários. Concluimos que a integração sinérgica entre tecnologia de ponta e princípios pedagógicos inovadores possui um imenso potencial para modernizar o ensino. Essa fusão não apenas torna o processo educacional mais dinâmico e eficiente, mas também o alinha de forma mais eficaz às necessidades e expectativas das metodologias educacionais contemporâneas, demonstrando o impacto transformador do Projeto AppGO na educação.

Palavras-chave: PhysicGo. Educação. Inovação. Backend. Laravel. MySQL.

REFERÊNCIAS BIBLIOGRÁFICA

AMAZON AWS. **O que é uma framework em programação e engenharia?** Disponível em: <https://aws.amazon.com/>. Acesso em: 26 jun. 2025.

JQUERY FOUNDATION. **jQuery API Documentation**. [2024]. Disponível em: <https://api.jquery.com/>. Acesso em: 6 ago. 2025.

LARAVEL. **Documentação**. [2025]. Disponível em: <https://laravel.com/docs>. Acesso em: 6 ago. 2025.

ORACLE. **MySQL Documentation**. [2025]. Disponível em: <https://dev.mysql.com/doc/>. Acesso em: 6 ago. 2025.