Schoolis: Um Projeto Open Source para Informatizar o Gerenciamento de Escolas

Lineker Aleksander Guete Calseverini, Djalma Domingos da Silva

e-mail:
lineker.calseverini@gmail.com, djalma.silva15@fatec.sp.gov.br

Faculdade de Tecnologia de São José do Rio Preto

Resumo: Este artigo tem como finalidade elaborar uma aplicação gratuita e com código fonte aberto para que instituições de ensino, principalmente as públicas, possam ter um meio de gerenciar o âmbito escolar, como na matrícula de alunos, na escala de funcionários, na organização de horários das matérias, e na alocação de professores substitutos, além de envio de comunicados para os pais e responsáveis com base nos telefones cadastrados na base de dados. Essa aplicação poderá ser embarcada em soluções de hardware de baixo custo, tornando-o acessível para todas as escolas públicas, que muitas vezes são barradas por burocracias e falta de financiamento público. Será apresentada a arquitetura da aplicação e o ferramental utilizado.

Palavras-chave: escola; informatização; gerenciamento; open-source.

Abstract: *This article aims to develop a free application with open source code so that educational institutions, especially public ones, can have a way of managing the school environment, such as student enrollment, staffing schedules, organization of subjects schedule, and the allocation of substitute teachers, in addition to sending notices to parents and guardians based on the telephone numbers registered in the database. This application can be embedded in low-cost hardware solutions, making it accessible to all public schools, which are often hampered by bureaucracy and lack of public funding. The application architecture and the tooling used will be presented.*

Keywords: *school; computerization; management; open-source.*

INTRODUÇÃO

Na realidade da educação brasileira, existem muitas barreiras para a gestão escolar eficiente, no quesito tecnologia. Como demonstrado por pesquisa realizada pelo INEP (BRASIL, 2020), a disponibilidade de equipamentos de tecnologia da informação e comunicação nas escolas de educação básica brasileira é limitada. Por exemplo, apenas 52,7% das escolas de educação infantil possuem acesso à internet banda larga, e esse percentual cai para 52,0% nas escolas de ensino fundamental. Ao final de 2022, ainda existiam 6,8% de escolas que não possuíam nenhum tipo de conexão com a Internet (BRASIL, 2023).

Essa barreira não se estende apenas à Internet. A falta de investimento público nas escolas também impede que aquisição de novos servidores e novos softwares de gestão ocorra. De acordo com a OCDE (2023), entre 42 países, o Brasil é um dos que menos investem na educação, um resultado de US$4.306 por aluno por ano. Para efeitos de comparação, a média da OCDE é de US$11.560. O mesmo relatório também mostra que houve uma redução de 10,5% no investimento governamental para educação.

Com recursos limitados e incerteza sobre a disponibilidade da internet para acessar sistemas online, escolas confrontam obstáculos substanciais em suas atividades cotidianas. Lidar manualmente com registros acadêmicos, financeiros e de recursos humanos é suscetível a imprecisões, pouco eficiente e consome bastante tempo. Essa situação pode resultar em falhas na comunicação, registros desiguais e atrasos.

Com base em todos esses fatos, muitos gestores de escola resolvem utilizar planilhas de Excel para auxiliar no gerenciamento de suas escolas. Porém, esse tipo de solução também vem com suas desvantagens. A primeira delas é que, além de demandarem tempo e esforço na

manipulação dos dados, também apresentam o risco de inconsistências devido ao seu processo manual e fragmentado. A integração dessas informações é fundamental para evitar equívocos custosos e garantir que as decisões sejam embasadas em dados confiáveis e atualizados.

Mesmo implementando senhas de segurança, a vulnerabilidade dos dados persiste devido à falta de flexibilidade na definição de hierarquia de uso. Isso facilita o acesso não autorizado e a possibilidade de envio indevido dos arquivos. Com a importância da segurança de dados em mente, especialmente diante da LGPD (Lei Geral de Proteção de Dados), é essencial adotar medidas rigorosas para evitar vazamentos ou perda de informações sensíveis, principalmente em instituições públicas.

Em suma, o uso de planilhas pode ser desafiador devido à necessidade de conhecimento avançado para programar fórmulas complexas, afetando a produtividade e o tempo gasto na gestão escolar. A dependência de inserção manual de dados pelos gestores ou funcionários de confiança contribui para uma maior perda de eficiência. Além disso, a limitação das funções pré-programadas dificulta a execução de tarefas específicas, exigindo habilidades adicionais dos usuários para contornar essas restrições.

Por isso, a solução proposta para esses problemas por este artigo é uma ferramenta gratuita, de código fonte aberto, com compatibilidade multiplataforma, chamada Schoolis. A decisão por manter o código fonte aberto se dá por vários motivos. O primeiro deles é que, pelo fato de o código estar público, ele pode ser revisado por pares e ter constantes aprimoramentos, além de haver uma transparência para a comunidade.

Prosseguindo, sua flexibilidade permite adaptar soluções para atender às necessidades específicas da escola ou comunidade. Além disso, seu custo reduzido, sendo o código em si gratuito e o pagamento direcionado apenas para suporte e segurança, torna-o acessível. A ausência de dependência de fornecedor possibilita o uso do código em diferentes contextos, sem restrições de localização, finalidade ou horário, conferindo liberdade e autonomia aos usuários.

Para complementar a gratuidade da ferramenta, a infraestrutura necessária será de baixo custo, e por isso, ela terá compatibilidade com a arquitetura ARM64, na forma do Raspberry Pi. Inicialmente criado para fins educacionais no intuito de criar uma geração de novos programadores pelo mínimo valor possível, devido a sua flexibilidade, modularidade, baixo consumo e baixo preço, conseguiu muitos outros usos popularizados por seus entusiastas, e isso inclui a capacidade de servir aplicações.

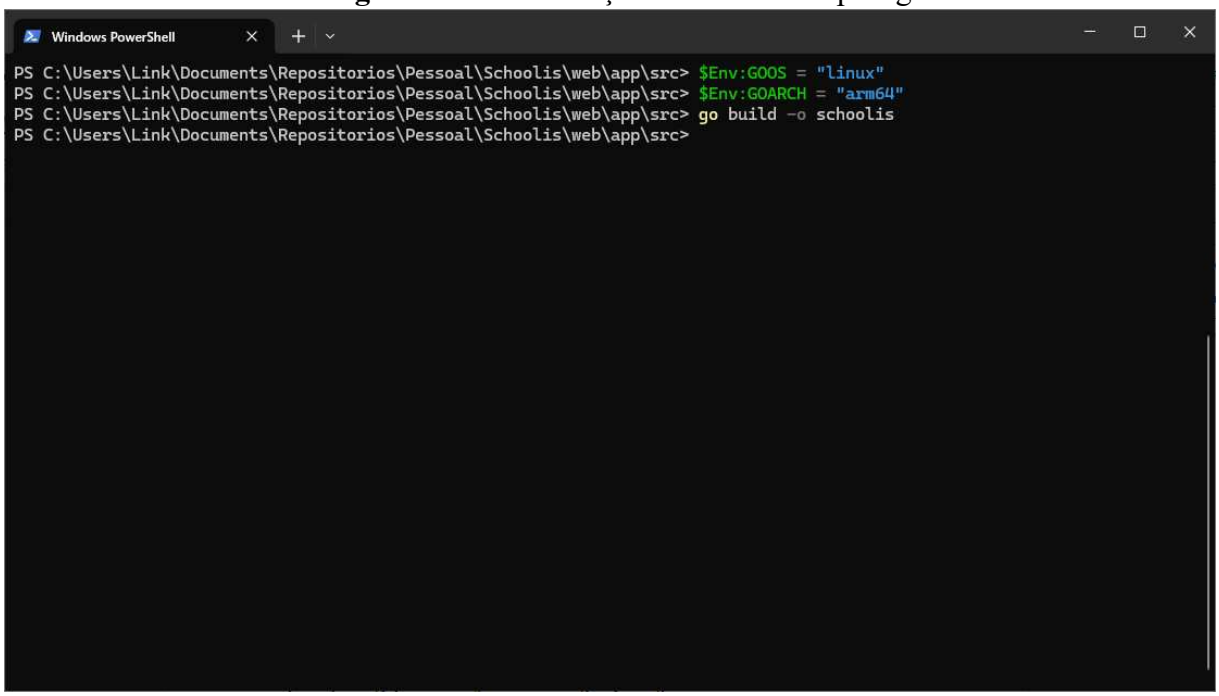
Figura 1: Raspberry Pi Model 3b

Fonte: Alecrim (2018).

E para garantir essa compatibilidade multiplataforma, desempenho para sistemas portáteis como o Raspberry Pi, e facilidade de leitura de código, Schoolis é desenvolvido em Go. Criada por Robert Griesemer, Rob Pike e Ken Thompson, um dos desenvolvedores do C, com o propósito de trabalhar com uma linguagem simples, tipagem segura, computação paralela e compilações rápidas (PIKE, 2012). De acordo com Pike (2012), a compilação em Go pode ser até 50 vezes mais rápida que a de C++.

Ela também permite o *cross-compiling,* uma *feature* que permite-nos programar no ambiente da nossa escolha e compilar para outro sistema operacional e até mesmo outra arquitetura, o que é conveniente para um sistema que rodará em um sistema operacional baseado em Linux e em uma arquitetura ARM64, no caso do Raspberry Pi.

Figura 2: Demonstração do Cross Compiling em Go



Fonte: Próprio Autor.

METODOLOGIA

Este artigo tem caráter de desenvolvimento aplicado, utilizando conceitos de Engenharia de Software para desenvolvimento da arquitetura da aplicação, e tem como base a documentação específica das bibliotecas padrões do Go (GOOGLE, 2024), principalmente o pacote net/http, além da documentação para o GORM (JINZHU, 2024), uma biblioteca específica que visa facilitar o desenvolvimento de Modelos. Também será utilizada a documentação do Raspberry Pi OS (RASPBERRY PI LTD, 2024), sistema operacional baseado em Debian especificamente para Raspberry Pi.

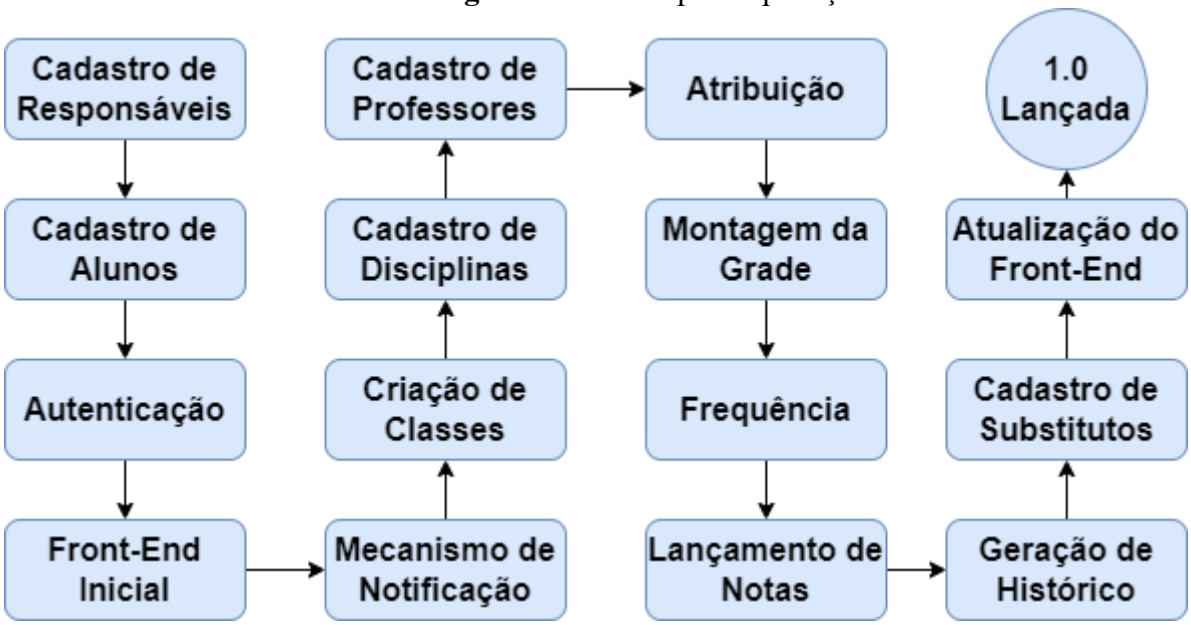
Livros didáticos sobre Go também foram utilizados para fixação do aprendizado. A bibliografia escolhida foi: Arundel (2023), que possui uma didática mais voltada a TDD (*Test-Driven Development*, ou Desenvolvimento Orientado a Testes no Português); Bodner (2021), que passa um conhecimento mais aprofundado do Go idiomático e McGavren (2019), com uma didática mais visual.

O banco de dados será o PostgreSQL, devido seu caráter *open source*, e a possibilidade de utilizar *plugins* e funções embutidas exclusivas que facilitam na apuração de problemas dentro do banco. Com isso, sua documentação (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2024) também é essencial para o desenvolvimento do Schoolis. Para desenho da arquitetura, será utilizada a ferramenta Umbrello. Para modelagem da API, será utilizado o padrão Swagger.

DESENVOLVIMENTO

Devido ao porte da aplicação, foi necessário definir um *roadmap* de como a aplicação será feita. A prioridade é um cadastro de alunos e responsáveis, seguido da autenticação, e após isso a criação de um *front-end* inicial para comunicação com a API. Após isso, prosseguir com a atribuição de disciplinas à classe, o que vai demandar a criação de professores, classes e disciplinas. Na sequência, será feito o lançamento de frequência e notas, possibilitando a geração de históricos e após a atualização do *front-end,* a versão 1.0 será lançada.

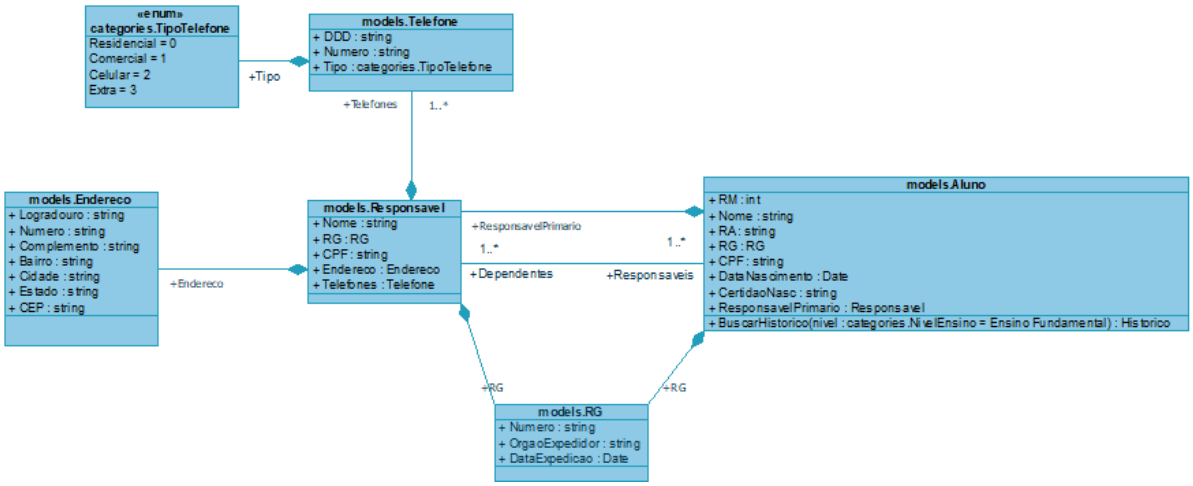
Figura 3: Roadmap da Aplicação



Fonte: Próprio Autor.

Embora o Go não seja necessariamente uma linguagem orientada a objetos, a utilização do Umbrello para demonstrar a arquitetura da aplicação facilitará na visualização e no desenvolvimento. Na arquitetura da aplicação usamos os *models*, uma estrutura que representa um registro no banco de dados. Começamos pelo próprio cadastro de alunos em si, onde foi definido os *models* Aluno, Responsável, RG, Telefone e Endereço. Telefone e Endereço pertencerão exclusivamente ao Responsável, e um Aluno poderá ter múltiplos Responsáveis, porém só terá um Responsável Primário, relacionamento representado pela Figura 4.

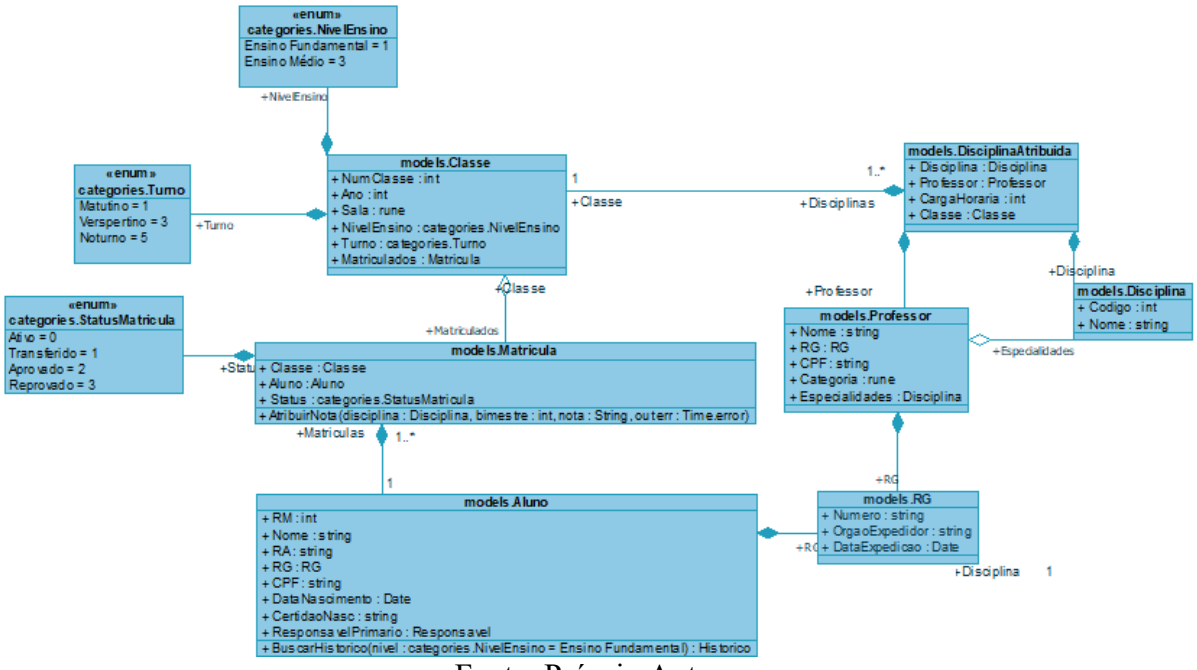
Figura 4: Relacionamento Aluno Responsável



Fonte: Próprio Autor.

O próximo diagrama representa o relacionamento Aluno – Classe – Professor, um aluno se associa a Classe por meio de um registro chamado matrícula, o qual possuí um *status*, que no código é representado por um *Enumerator*, que pode possuir os seguintes valores: Ativo, Transferido, Aprovado ou Reprovado. A classe possui seu próprio turno, Matutino, Vespertino ou Noturno, além do seu nível de ensino, que é Fundamental ou Médio. Ela se relaciona ao Professor por meio das disciplinas atribuídas. O relacionamento pode ser visualizado na figura 5.

Figura 5: Relacionamento Aluno Classe Professor

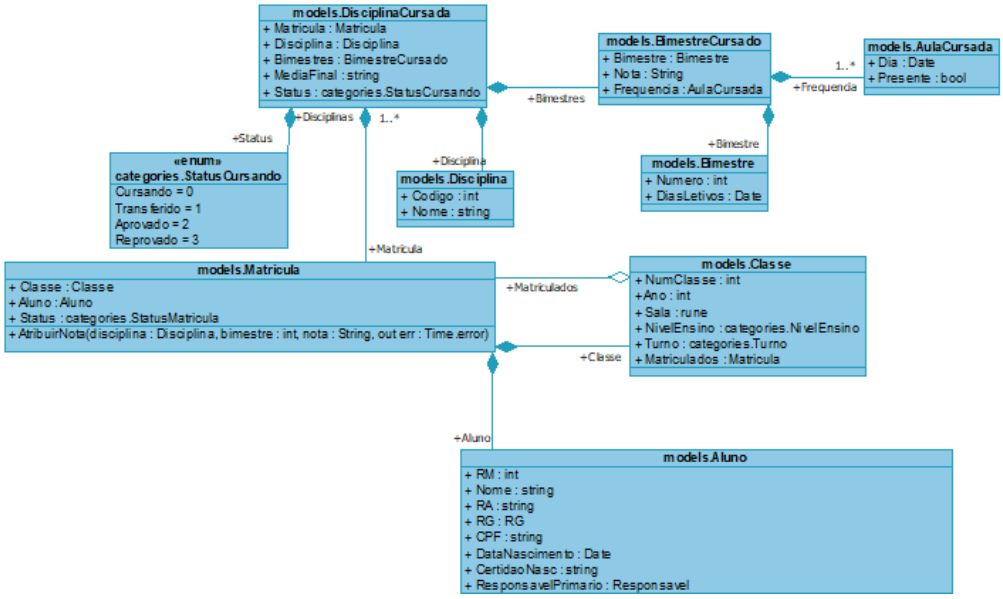


Fonte: Próprio Autor.

Já no relacionamento Aluno Nota, a conexão entre o aluno e seu desempenho escolar é feito por meio da matrícula. Em cada matrícula do aluno estão as disciplinas cursadas naquele ano, elas também possuem seu próprio *status*: Cursando, Transferido, Aprovado ou Reprovado.

Cada disciplina cursada possui seus bimestres, onde estão os registros de frequência e sua nota. Esse é representado na Figura 6.

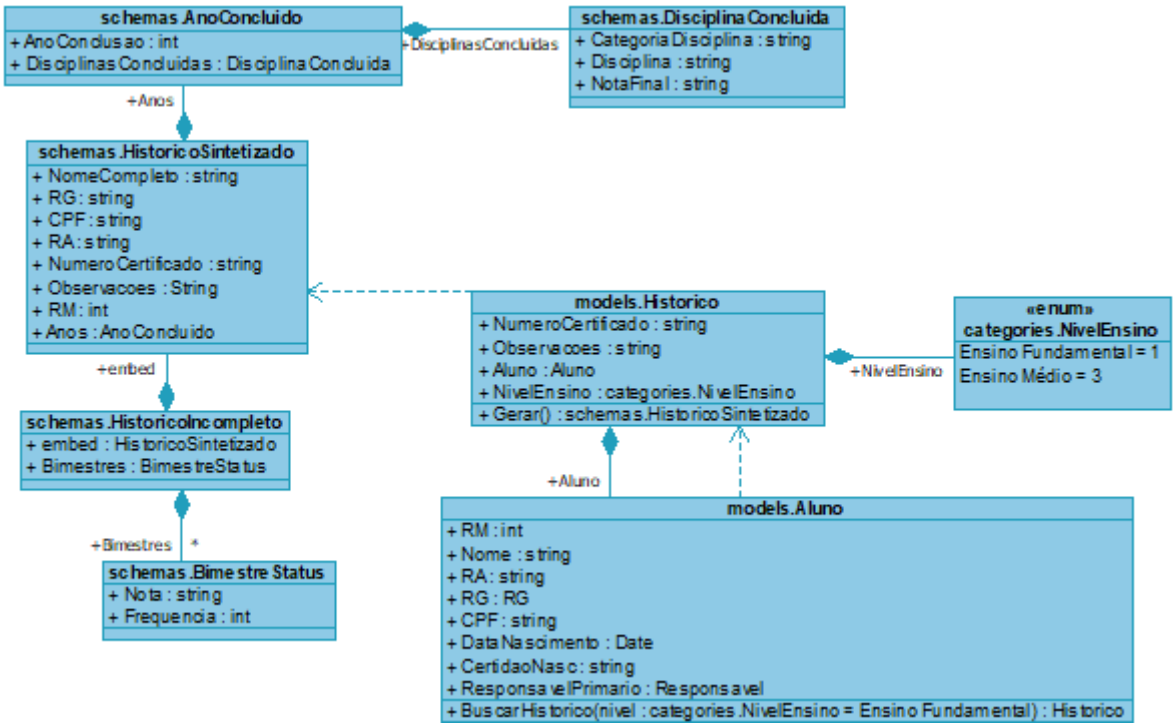
Figura 6: Relacionamento Aluno Nota



Fonte: Próprio Autor.

Na conclusão do ensino, o histórico é gerado, este possui um número de validação. Mas um histórico escolar incompleto também pode ser gerado, este não possui número de validação e somente é usado em casos de transferência. Nesta relação o modelo Aluno possui um método chamado BuscarHistorico, este método pode retornar um Histórico Escolar completo ou incompleto. A relação é representada pela figura 7.

Figura 7: Relacionamento Aluno Histórico

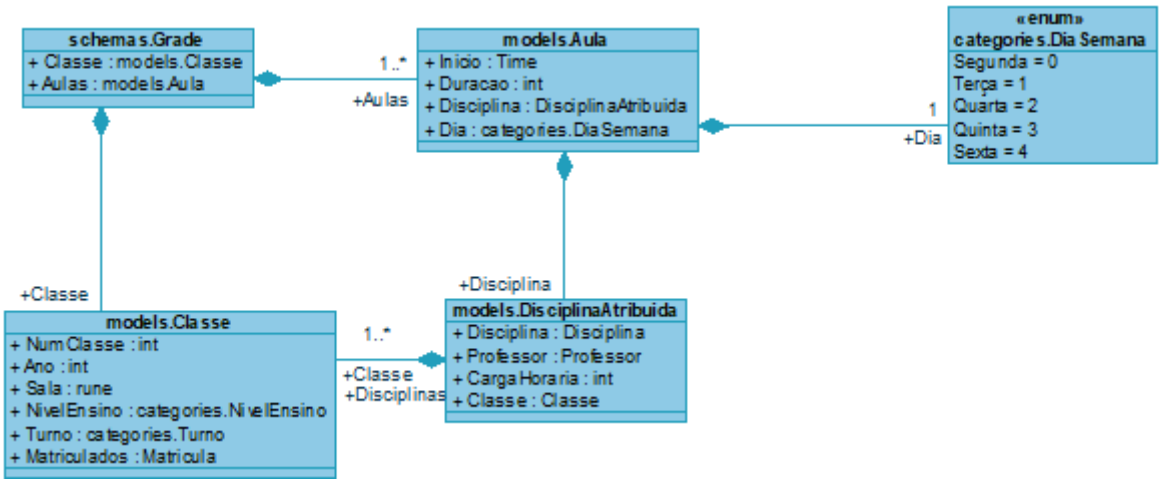


Fonte: Próprio Autor.

Para os Históricos gerados por esse método, foi criado um pacote chamado *schemas*, separado dos *models*. Essa decisão se deve ao fato de que esses *schemas* não são armazenados em banco e na verdade puxam dados de vários *models* para sintetizar dados. Este pacote de *schemas* voltará a ser utilizado para outras estruturas que serão montadas como resposta para a API da aplicação.

O próximo relacionamento desenhado foi entre Classe e a Grade de Horários. A ideia é a de que exista um melhor controle de como a grade é montada por sala. Assim cada classe ativa no sistema possui uma grade própria, com um atributo representando a lista de aulas daquela grade sendo que cada aula tem o identificador da matérias, seu início, sua duração e a qual dia da semana ela pertence. O diagrama na figura 8 é a representação deste relacionamento.

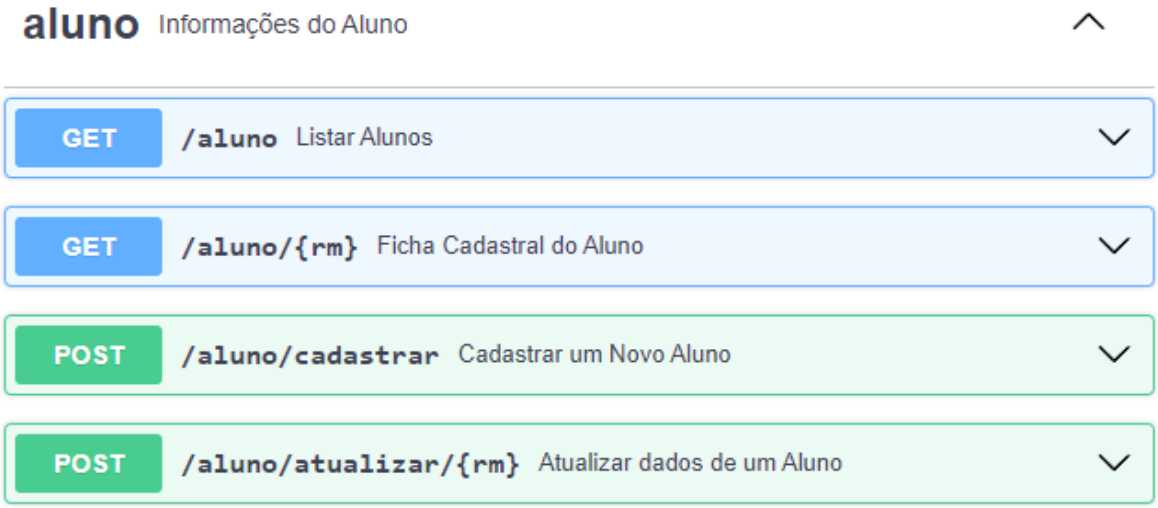
Figura 8: Relacionamento Classe Grade



Fonte: Próprio Autor.

A comunicação com a aplicação vai ser feita por meio de *endpoints* de API HTTP. O *endpoint* /aluno terá opções para leitura, criação e atualização. Por segurança dos dados, não será possível excluir um aluno do sistema, já que alunos tem o direito de consultar dados de sua educação pelo restante da vida, contanto que eles vão até a escola para tirar informações.

Figura 9: Documentação do Endpoint /aluno



Fonte: Próprio Autor.

RESULTADOS E DISCUSSÕES

Para criarmos uma estimativa de quanto esta aplicação valeria no mercado, foi utilizado o método de análise de pontos de função, de acordo com o Manual de Práticas de Contagem de Pontos de Função (IFPUG, 2010) em sua versão 4.3.1, para avaliar a complexidade da mesma e verificar o preço médio de mercado por cada ponto de função. Para a análise de pontos de função, consideramos apenas as relações Aluno Responsável, Aluno Classe Professor e Aluno Nota. Então o valor da aplicação pode ser acima do resultado da análise.

Como todas as consultas são ao banco interno da aplicação, todos os pontos de função vão ter alimentação interna. Foi identificado as funções principais dessas relações seriam os *models* Aluno, Responsável, Classe, Professor, Disciplina e Disciplina Cursada, foi estimado uma quantia de 7 pontos de função para cada uma delas, considerando uma aplicação que implementa o CRUD e Visualização de cada uma delas, a estimativa é de 210 pontos de função no total, utilizando um fator multiplicativo de 5 vezes, chamado de método NESMA, como podemos verificar nas Figuras 10 e 11.

Figura 10: Análise de Pontos de Função

Nome da Função	Tipo	TD	AR/TR	Complex.	PF IFPUG	PF Local do EM
Aluno	ALI	10	2	Baixa	7	7,00
Responsável	ALI	15	1	Baixa	7	7,00
Classe	ALI	7	3	Baixa	7	7,00
Professor	ALI	7	3	Baixa	7	7,00
Disciplina	ALI	3	4	Baixa	7	7,00
Disciplina Cursada	ALI	7	3	Baixa	7	7,00

Fonte: Próprio Autor.

Figura 11: Estimativa Final de Quantidade de Pontos de Função

Total PF não ajustados (contagem detalhada)	42
Total PF não ajustados (contagem estimativa)	42
Total PF não ajustados (contagem indicativa)	210

Fonte: Próprio Autor.

Segundo o Acórdão 1364/2019 do Tribunal de Contas da União (apud Fattocs, s.d.), o preço do ponto de função pode variar entre R$255,00 e R$1000,00. Considerando um corpo de 57 licitações, o preço médio do ponto de função no relatório foi de R$488,00, e este é o preço que vamos usar para a estimativa do valor da aplicação caso ela não fosse gratuita. Com isso, temos que o valor total estimado da aplicação é de R$102.480,00.

A estimativa de entrega da aplicação não foi definida, devido a diversos fatores como tamanho do time e tempo disponível para trabalho. Considerando que esses fatores influenciam no tempo de entrega e que esse tempo pode ser extenso, não é possível fazer uma estimativa precisa com os recursos atuais, que seriam de um desenvolvedor no seu tempo livre.

CONCLUSÃO

Após modelagem e arquitetura inicial da aplicação, além de pesquisas em Mercado, vimos as vantagens que o Go tem a entregar como os fatos de ser uma linguagem própria para programação concorrente, ser compilada, possuir *cross-compiling*, e também, a compilação ser rápida. Além disso, foi visto a importância de criar a arquitetura da aplicação antes de seu desenvolvimento, pois é através desse processo que podemos avaliar o tempo de desenvolvimento, a complexidade da aplicação e seu possível valor de mercado.

O custo de adquirir um software como este para uma escola é relativamente baixo, considerando apenas o preço de um Raspberry Pi, que atualmente é de cerca de R$600,00, além do valor de implantação, supondo um tempo de implantação de 40 horas, podemos estimar um

preço de implantação de R$6000,00. No entanto, o valor potencial do software, avaliado inicialmente em R$102.480,00 com base na análise de pontos de função, ressalta a sua significativa contribuição para a educação pública. Portanto, é crucial dar continuidade ao desenvolvimento desta aplicação.

REFERÊNCIAS

ALECRIM, Emerson. **Raspberry Pi 3 Model B+ melhora conectividade e desempenho, mas mantém preço baixo**. 2018. Disponível em: https://tecnoblog.net/noticias/raspberry-pi-3-model-b-plus/. Acesso em: 02 jul. 2024.

ARUNDEL, John. **For the Love of Go**. 2. ed. Lostwithiel, Cornwall: Bitfield Consulting, 2023. 209 p.

BARUCHO, Airton. **Saiba porque abolir o uso de planilhas na gestão escolar**. 2024. Disponível em: https://deltasge.com.br/site/saiba-porque-abolir-uso-planilhas-gestao-escolar/. Acesso em: 18 mar. 2024.

BODNER, Jon. **Learning Go**: an idiomatic approach to real-world go programming. Sebastopol: O'Reilly, 2021. 352 p.

BRASIL. Assessoria de Comunicação Social do Inep. Inep. **Pesquisa revela dados sobre tecnologias nas escolas**. Disponível em: https://www.gov.br/inep/pt-br/assuntos/noticias/censo-escolar/pesquisa-revela-dados-sobre-tecnologias-nas-escolas. Acesso em: 17 mar. 2024.

BRASIL. Gape. Anatel. **Em 2022, Brasil registrou 9,5 mil escolas sem acesso à internet**. 2023. Disponível em: https://www.gov.br/anatel/pt-br/assuntos/noticias/em-2022-brasil-registrou-9-5-mil-escolas-sem-acesso-a-internet. Acesso em: 17 mar. 2024.

CELLAN-JONES, Rory. **A 15 pound computer to inspire young programmers**. 2011. Disponível em: https://www.bbc.co.uk/blogs/thereporters/rorycellanjones/2011/05/a_15_computer_to_inspire_young.html. Acesso em: 22 mar. 2024.

EDUBLOG DA ZEHAN | TENDÊNCIAS EDUCACIONAIS. Zehan. **Os Desafios da Gestão Escolar Sem Tecnologia**: por que escolas precisam de sistemas informatizados. Por Que Escolas Precisam de Sistemas Informatizados. 2024. Disponível em: https://www.zehan.com.br/post/os-desafios-da-gest%C3%A3o-escolar-sem-tecnologia-por-que-escolas-precisam-de-sistemas-informatizados. Acesso em: 18 mar. 2024.

FATTOCS. **Qual o preço do ponto de função (ou quanto custa)**. Disponível em: https://www.fattocs.com/blog/qual-o-preco-do-ponto-de-funcao/. Acesso em: 11 jun. 2024.

GOOGLE. **Go Packages**. Disponível em: https://pkg.go.dev/. Acesso em: 03 abr. 2024.

IFPUG. **Manual de Práticas de Contagem de Pontos de Função**: versão 4.3.1. Princeton Junction: Ifpug, 2010. 571 p.

JINZHU. **GORM Guides**. Disponível em: https://gorm.io/docs/. Acesso em: 03 abr. 2024.

MCGAVREN, Jay. **Head First Go**. Sebastopol: O'Reilly, 2019. 556 p. (Head First).

OCDE. **Education at a Glance 2023: OECD Indicators**: country notes - brazil. Country Notes - Brazil. 2023. Disponível em: https://www.oecd-ilibrary.org/sites/7a9958a8-en/index.html?itemId=/content/component/7a9958a8-en#section-d1e4190-12bcad3e46. Acesso em: 17 mar. 2024.

PIKE, Rob. **Go at Google**: language design in the service of software engineering. Language Design in the Service of Software Engineering. 2012. Disponível em: https://go.dev/talks/2012/splash.article. Acesso em: 02 abr. 2024.

PILTCH, Avram. **How to Set Up a Raspberry Pi Web Server**. 2022. Disponível em: https://www.tomshardware.com/news/raspberry-pi-web-server,40174.html. Acesso em: 22 mar. 2024.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. **PostgreSQL: Documentation**. Disponível em: https://www.postgresql.org/docs/. Acesso em: 04 abr. 2024.

RASPBERRY PI LTD. **Raspberry Pi Documentation - Raspberry Pi OS**. Disponível em: https://www.raspberrypi.com/documentation/computers/os.html. Acesso em: 04 abr. 2024.

REDHAT. **O que é open source?** 2024. Disponível em: https://www.redhat.com/pt-br/topics/open-source/what-is-open-source. Acesso em: 21 mar. 2024.