UNIFEOB
ESCOLA DE NEGÓCIOS

2024

PROJETO INTEGRADO

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO
OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADO

SISTEMA DE GESTÃO E INTELIGÊNCIA DE NEGÓCIOS PARA ORGANIZAÇÕES SOCIAIS

ABACO

SÃO JOÃO DA BOA VISTA, SP

NOVEMBRO 2024

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADO

SISTEMA DE GESTÃO E INTELIGÊNCIA DE NEGÓCIOS PARA ORGANIZAÇÕES SOCIAIS

ABACO

MÓDULO COMPUTAÇÃO EM NUVEM

Estrutura de Dados – Prof. Marcelo Ciacco Almeida

Linguagem e Técnicas de Programação – Prof. Nivaldo de Andrade

Tópicos Avançados de Banco de Dados – Prof. Max Streicher Vallim

Computação em Nuvem – Prof. Rodrigo Marudi de Oliveira

Projeto de Computação em Nuvem – Profª. Mariângela Martimbianco Santos

Estudantes:

Everton Henrique Lopes, RA 23000201
Fábio Koiti Konda, RA 23000897
Lucas Pizol Ferreira, RA 23000019
Mariana Franceschi Tessarini, RA 23001518
Yan Gomes Aguiar de Almeida, RA 23000825

SÃO JOÃO DA BOA VISTA, SP
NOVEMBRO 2024

SUMÁRIO

1. INTRODUÇÃO

Seguindo a continuidade do tema do semestre anterior, “Sistema de Gestão e Inteligência de Negócios para Organizações Sociais” onde um software foi desenvolvido juntamente com relatórios personalizados, uso de Business Intelligence e UML, o projeto neste semestre está em conformidade com o tema “Computação em Nuvem”.

O objetivo do projeto é a continuação do desenvolvimento de software para a Organização ABACO, onde sua funcionalidade é de controlar alunos em seus respectivos cursos, que são disponibilizados pela instituição na intenção de ajudar essas pessoas a se formar profissionalmente e serem independentes financeiramente.

O programa será construído totalmente online a partir da arquitetura em nuvem, incluindo gerenciamento de banco de dados e outras ferramentas. Além disso, inclui-se conhecimentos em computação em nuvem, estrutura de dados e linguagem técnicas de programação.

Em suma, o objetivo central da equipe é fornecer à instituição parceira um sistema de gestão que promova a eficiência operacional e a capacidade de análise estratégica, contribuindo positivamente nas organizações sociais.

2. DESCRIÇÃO DA EMPRESA

A organização ABACO, Associação Beneficente de Apoio à Comunidade de Poços de Caldas, é uma entidade sem fins lucrativo, sob o CNPJ nº 02.727.331/0001-53, e com endereço na Rua Soldado Altivo Ferreira da Costa, 60, Bairro Conjunto Habitacional Pedro Afonso Junqueira, na cidade de Poços de Caldas, no estado de Minas Gerais.

Com funcionamento desde 1998, a instituição se dedica no campo de assistência social, principalmente a famílias com situação de vulnerabilidade social, atendendo suas necessidades, oferecendo oficinas e cestas básicas. Um dos seus serviços é a de oficinas, onde ajudam a formar profissionais a partir de diversos cursos gratuitos com a finalidade de emancipação e financeira dos usuários.

Para finalizar, a ABACO segue parceria com a Secretaria de Promoção Social para fortalecimento de vínculos com familiares de encaminhamento do CRAS Sul, sendo este um projeto para reforçar o convívio comunitário social, desenvolvendo experiências familiares e comunitárias.

3. PROJETO INTEGRADO

3.1 TÓPICOS AVANÇADOS DE BANCO DE DADOS

O banco de dados continua tendo grande importância ao longo da evolução da tecnologia, sendo elas a forma de armazenar grandes volumes de dados e onde guardá-las. Como o software precisará de um banco suficiente para armazenar informações pessoais e diversos históricos, como notas, aulas e frequências, a inserção de um banco de dados não relacional (mysql) se torna uma solução viável para o projeto.

Para que a modelagem de dados funcione e esteja de acordo com a necessidade do cliente, Machado (2014) menciona cinco aspectos importantes para planejar um banco de dados:

> "O processo de modelagem consiste em cinco aspectos importantes: Observação, a partir de entrevistas e reuniões, entendimento dos conceitos, identificando e assimilando o objeto observado, representação dos objetos, aplicando técnicas de modelagem de dados Entidade-Relacionamento (MER), verificação de fidelidade e carências, detectando falhas ou verificando aplicações erradas da técnica de representação e validações, buscando aprovação formal do modelo, junto com a participação do usuário final".

Seguindo o passo de observação, o objetivo do software, a partir de reuniões com o cliente, foi de poder armazenar informações pessoais dos alunos e docentes, assim como aplicar faltas e notas, postagem de aulas e cursos, podendo também visualizá-las a partir de listas. A partir disso, o modelo lógico foi desenvolvido, que será abrangido no próximo tópico.

3.1.1 MODELO LÓGICO

Como menciona Machado (2014) o modelo lógico "descreve em formato as estruturas que estarão no banco de dados de acordo com as possibilidades permitidas pela sua abordagem". O modelo foi desenvolvido pelo MySql Workbench, onde pode criar, editar e visualizar as tabelas, seus atributos e suas ligações (relacionamentos).

O modelo teve sua primeira tabela a de logs para referenciar mudanças que foram feitas, incluindo descrição e data que foi realizada. Após isso, seguiu de acordo com os principais dados a serem armazenados: alunos e docentes. Ambos possuem atributos similares como nome, data de nascimento, documentos, contatos e endereço. Contudo a tabela aluno inclui dados como a escolaridade e a situação em relação ao curso inscrito (cadastrado, pendente, cancelado).

Para melhor organização de dados, a tabela endereços foi criada separadamente, já que 2 tabelas (aluno e docente) precisam desses dados e, ao invés de colocar os mesmos atributos nas duas tabelas, foi preferível criar a tabela e colocar o “idaluno” e “iddocente” como chaves estrangeiras. Em endereços inclui os dados de logradouro, número, bairro, zoneamento, complemento (opcional), CEP, UF e município.

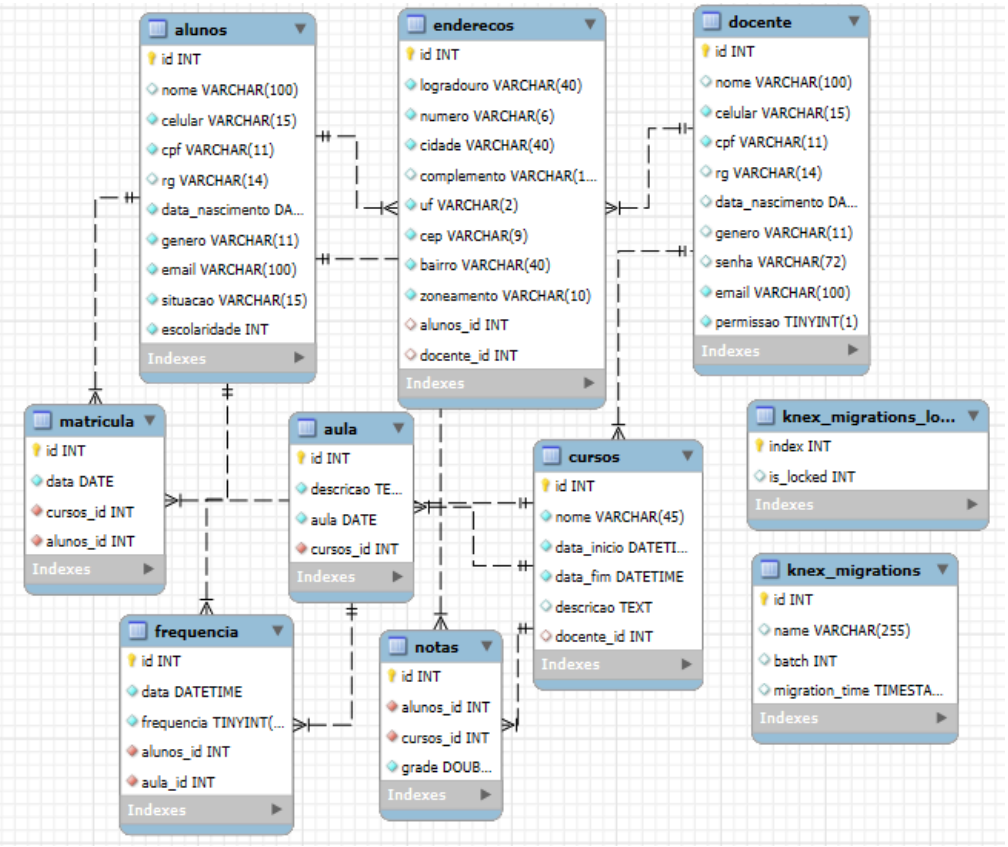
O raciocínio seguinte foi a inserção da tabela cursos, onde o usuário final poderá cadastrá-los, colocando informações de data de início, período do curso, nome e sua descrição, além de uma chave estrangeira que é quem realizará o curso - “iddocente”-.

Outra tabela foi a de matrícula onde, ao cadastrar o aluno, será finalizado com a confirmação da matrícula do curso escolhido. Portanto o id, a data da matrícula e as chaves estrangeiras do aluno e do curso são os atributos aplicados para essa tabela.

Na visão do docente e do cliente, as operações de postar aulas, lançar notas e faltas são essenciais para a organização e um dos principais objetivos do software, com isso uma tabela foi criada para cada funcionalidade. A de aulas, o docente poderá nomeá-la, descrevê-la e indicar a data que foi postada a aula, além de interligar a aula com o curso necessitado. A de notas, poderá nomeá-la para entender de qual avaliação ou trabalho está sendo tratado, além da nota em si, e as ligações - a partir das chaves estrangeiras - , do aluno e do curso. Por último a de frequências, onde o professor colocará a data atual e preencherá caso o aluno não tenha participado da aula, com isso as chaves estrangeiras do “idaluno” e “idaulas” vão ser necessárias nessa tabela para maior controle.

O resultado final, assim como seus relacionamentos estão retratados na imagem abaixo.

Imagem 1: Modelo Lógico



Fonte: Autores (2024).

3.1.2 MODELO FÍSICO

O modelo físico representa a parte final de um banco de dados, onde são definidos os detalhes necessários para sua implementação. Segundo Silberschatz (2011), "as tabelas normalizadas evitam redundâncias e garantem a integridade dos dados". Em relação às tabelas, vale ressaltar que nem todas as datas estão com a expressão “CURRENT_TIMESTAMP”, pois as tabelas que possuem tal expressão conseguem registrar a data e hora que os dados foram inseridos de maneira automática, ao contrário das tabelas sem a expressão, que dão implementadas manualmente. Tal resultado pode ser encontrado no arquivo do projeto de banco de dados completo no Anexo 1.

Para facilitar a inserção de dados foram criadas as stored procedures, sendo elas definidas por Silberschatz (2011) como “conjuntos de instruções SQL que podem ser armazenadas e executadas no banco de dados”. Sua principal função no projeto é encapsular os detalhes dos valores a serem inseridos, garantindo que apenas dados válidos sejam armazenados. Como apresenta o exemplo:

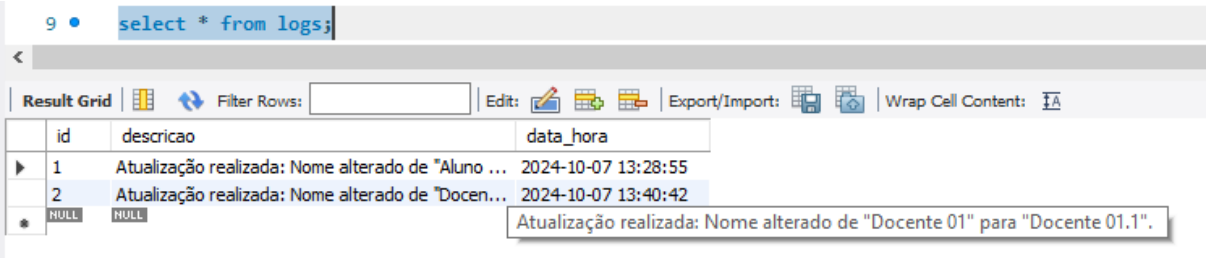
Imagem 2 - Stored Procedure para registrar endereço de aluno

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `registrarEnderecoAluno`(IN parNumero varchar(6), IN parZoneamento varchar(10),
 IN parLogradouro varchar(40), IN parBairro varchar(40), IN parCidade varchar(40),
IN parUf varchar(2), IN parCep varchar(9), IN parComplemento varchar(30), IN parAlunosId int)
BEGIN
insert into enderecos(numero, zoneamento, logradouro, bairro, cidade, uf, cep, complemento, alunos_id)
values (parNumero, parZoneamento, parLogradouro, parBairro, parCidade, parUf, parCep, parComplemento, parAlunosId);
END
```

Fonte: Autores (2024).

Também fez-se uso de triggers, estabelecidos por Hernandez (2003) como "procedimentos que são automaticamente executados em resposta a eventos específicos em uma tabela, como inserções, atualizações ou exclusões". Uma das aplicações teve como objetivo notificar atualizações realizadas nas tabelas "alunos" e "docentes" e, em seguida, registrar na tabela "logs" as mudanças ocorridas, incluindo a data e o horário.

Imagem 3 - Trigger para atualização de informações



Fonte: Autores (2024).

Em relação ao Sistema de Gerenciamento de Banco de Dados (SGDB), o escolhido foi o modelo relacional, por ser melhor "para armazenar e gerenciar dados, onde as informações são organizadas em tabelas, permitindo consultas complexas e consistência através de chaves primárias e estrangeiras", como descreve Silberschatz (2019), visto que as tabelas do projeto possuem muitas chaves estrangeiras, acaba tornando-se inconveniente a implementação de um modelo no qual seja necessário unir os dados manualmente.

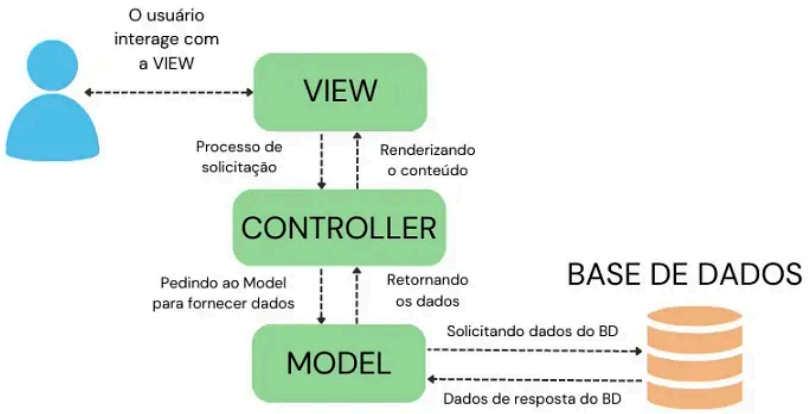
3.2 LINGUAGEM E TÉCNICAS DE PROGRAMAÇÃO

Na criação de um software, é necessário realizar o levantamento de requisitos dos tópicos necessários para a criação do projeto. Nessa fase, é de suma importância entender e escolher a arquitetura a ser utilizada e a metodologia a ser implementada. Nesse caso, temos a

programação orientada a objetos, programação funcional e programação concorrente como exemplo.

Segundo Kamienski (1996), “Programação orientada a objetos (POO) é uma metodologia de programação adequada ao desenvolvimento de sistemas de grande porte”. Este padrão aproxima os objetos do mundo real com a programação. Com exemplos de arquitetura, temos MVC e arquitetura hexagonal. A arquitetura MVC (Model, View e Controller) foi a selecionada para nosso sistema, pois sua abordagem simples comporta o software da ABACO.

Imagem 4: Conceito de MVC



Fonte: Medium.

O sistema foi desenvolvido obedecendo padrões da orientação a objetos, buscando a melhoria contínua ofertando escalabilidade e legibilidade do código. Além disso, foram utilizadas as bibliotecas Express para gerenciar as requisições HTTPs, Jwt para autenticação e Bcrypt para criptografia de senhas. Foi utilizado o Knex como ORM (Object Relational Mapper), que é uma ferramenta que aproxima a linguagem de programação utilizada (JavaScript) com o banco de dados, podendo ter portabilidade para outros provedores como MySQL e PostgreSQL.

3.2.1 APPLICATION PROGRAMMING INTERFACE (API) - BACK-END

Uma API é desenvolvida para realizar a conexão do servidor com a parte visual do sistema. Essa pode ser realizada seguindo alguns conceitos de arquitetura, como GraphQL, RESTful, SOAP, entre outras. A abordagem escolhida para o desenvolvimento do software foi a RESTful, pois é simples e recomendada para a criação de um sistema como o da ABACO.

Com isso, será utilizado os protocolos HTTP para requisições entre o servidor e o cliente. Como menciona Saudate (2013), “O protocolo HTTP trata-se de um protocolo de camada de aplicação (segundo o modelo OSI) e de facilidade de manipulação em aplicações”.

Para o desenvolvimento efetivo de uma API RESTful, há processos que devem ser seguidos:

URL como recurso: Sua conexão e transferência de dados deverá ser feita através de URLs. Por exemplo, na rota /estudantes, será enviada uma requisição do tipo GET para listar todos os estudantes cadastrados.

Imagem 5: Exemplo de GET na rota “/estudantes”

```
export const listarEstudantes = async () => {
  return await api.get("/estudantes");
};

// Resultado
export const estudantes = [
  {
    id: 1,
    nome: "Estudante 01",
    situacao: "Formado",
    celular: "35999999999",
    cpf: "00000000000",
    data_nascimento: new Date(),
    escolaridade: "1º Ano do Ensino Médio",
    genero: "Masculino",
    email: "estudante@email.com",
  },
];
```

Fonte: Autores (2024).

Métodos HTTPs: Em uma arquitetura RESTful, deve-se utilizar os métodos HTTPs corretamente. Para isso, as requisições são realizadas através de verbos e cada um tem sua particularidade de utilização:

- GET: Listagem e consulta de informações
- POST: Envio de dados e criação de registros no banco de dados
- PUT: Atualização de dados no banco de dados
- PATCH: Atualização parcial dos dados no banco de dados
- DELETE: Exclusão de informações

Sem estado: O servidor não deve armazenar estados sobre o cliente entre as requisições.

Representação de recursos: Os dados enviados do servidor para o sistema será em formato JSON

Códigos HTTP: Todas as requisições enviarão o código correto para o sistema, como por exemplo 200 para sucesso, 201 para criado, 204 sem conteúdo e 404 não encontrado.

Considerando a abordagem RESTful, é possível acessar os dados do sistema através de softwares de BI, como Power BI e Google Looker Studio, liberando endpoints específicos para a utilização deles, visualizando os dados necessários sem colocar em risco a integridade dos dados.

A parte de segurança é fundamental na construção do sistema, pois previne que pessoas maliciosas acessem o sistema podendo excluir ou manipular dados sensíveis. Para isso, foram implementados tokens de autenticação e permissão para os utilizadores do sistema, sendo possível realizar o acesso ao software como Coordenador ou Professor, ambos tendo suas permissões de utilização. Foi utilizado o token JWT, pois é fácil de utilizar, flexível, tem suporte e segurança.

Imagem 6: Exemplo de imagem de autenticação

```
export const authenticateUser = async (
  req: AuthenticatedRequest,
  res: Response,
  next: NextFunction
) => {
  try {
    const authorization = req.headers["authorization"];
    if (!authorization)
      return sendUnauthorizeResponse(res, "Token não informado");

    const [, token] = authorization.split(" ");
    if (!token) return sendUnauthorizeResponse(res, "Token não informado");

    const verifyUser = JWTService.verifyToken<{ id: number }>(token);
    if (!verifyUser) return sendUnauthorizeResponse(res, "Token inválido");

    const user = await Docente.findBy(["*"], { id: verifyUser.id });
    if (!user) return sendUnauthorizeResponse(res, "Usuário não encontrado");

    req.user = user;
    next();
  } catch (error) {
    return sendUnauthorizeResponse(res, "Token inválido");
  }
};
```

Fonte: Autores (2024).

3.2.2 FRONT-END

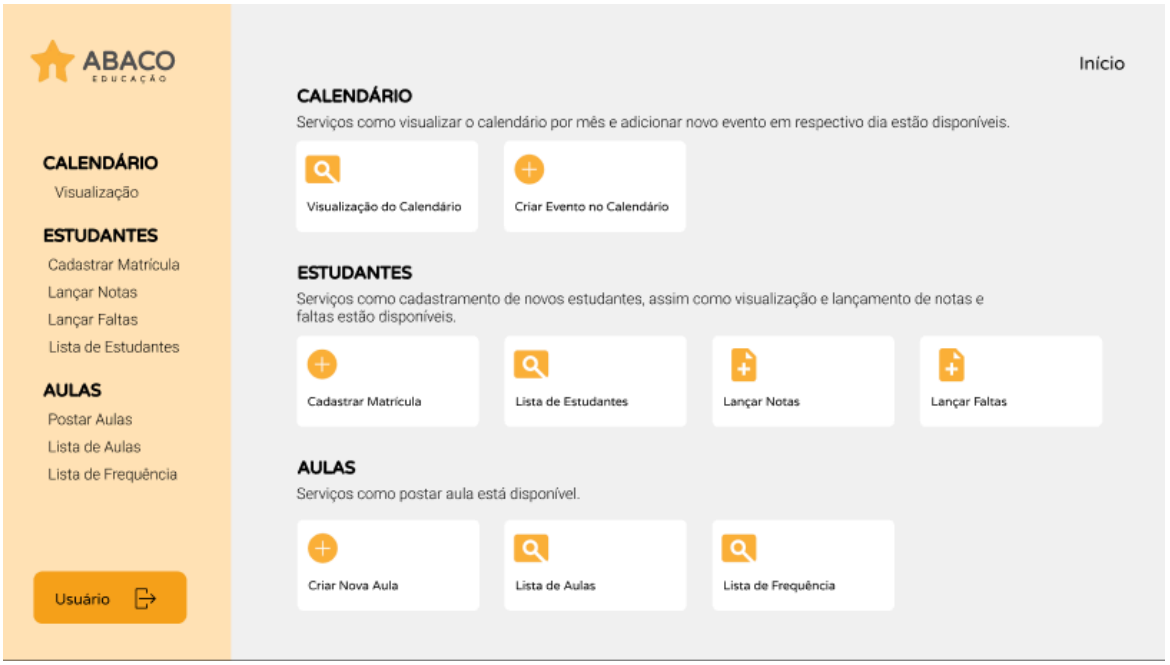
O desenvolvimento de um sistema de gestão para Organizações Sociais, como o software criado para a ABACO, depende da criação de um front-end que alie eficiência e praticidade. Este trabalho explora abordagens teóricas e práticas para a criação do sistema, buscando um design responsivo para ser utilizado em todas as telas e interface de usuário. O desenvolvimento de todo o layout e feedbacks foi realizado com base nas heurísticas de Nielsen.

A interface da plataforma foi totalmente centrada nos usuários, buscando entender quais são as necessidades e os pontos de melhoria. Segundo Nielsen (2001) “Para projetar a melhor UX, preste atenção ao que os usuários fazem, não ao que dizem.”, deixando em evidência a necessidade de entender o fluxo de utilização realizado pelo usuário. Muitas vezes, eles não sabem o que querem, mas sabem que precisam. A experiência e a interface do usuário (UX e UI) são igualmente vitais para a criação de uma interface funcional e atraente, particularmente em uma organização como a ABACO, que conta com colaboradores de diferentes perfis e níveis de conhecimento em tecnologia. A experiência une as intenções do projeto com as necessidades e expectativas do usuário e a simplicidade na interface promove uma interação mais natural. Assim, o design prioriza uma navegação clara, uma paleta de cores com base nas cores da instituição e elementos intuitivos.

A integração com BI é essencial para ajudar na busca por novos leads para a instituição. A ABACO busca ter o contato direto com as pessoas, auxiliando-as nos cursos, buscando entender caso uma pessoa tenha desistido, acompanhar os estudantes em suas trajetórias e colher feedbacks de recém formados. Dashboards e gráficos tornam-se ferramentas fundamentais para visualização de dados relevantes, como a alocação de recursos e o desempenho de programas sociais.

Para a inicialização da prototipagem do software, a ferramenta online Figma foi utilizada para melhor entendimento de como serão o fluxograma e organograma de uma tela para outra. No caso da ABACO, foi solicitado telas que ajudem a virtualizar os dados de alunos e docentes, assim como lançamento de notas e faltas. Na prática, foi realizado a tela login, que direciona ao início com um visual de dashboard com cards para selecionar as diversas opções como cadastro de alunos e docentes, visualização de calendário, criação de evento no calendário, lançamento de faltas e notas e lista de notas, faltas, frequências assim como de alunos e docentes. A visualização completa do design está disponível no site Figma (Anexo 2).

Imagem 7: Design da Tela Inicial



Fonte: Autores (2024).

Após a realização do design, o desenvolvimento do software foi utilizado o moderno framework Angular, onde a aplicação é uma Single Page Application (SPA), o que melhora a performance e fluidez na experiência do usuário, sem necessidade de ficar carregando cada página a cada clique. Isso permite que o sistema seja modular e escalável.

3.3 COMPUTAÇÃO EM NUVEM

A computação em nuvem transformou a forma como empresas gerenciam suas operações, trazendo soluções como armazenamento de dados, análise de grandes volumes de informações e backup, além de permitir a execução de aplicativos e sistemas sem a necessidade de hardware dedicado. Essa tecnologia tem oferecido uma evolução para a área tecnológica, tornando-a essencial no mundo moderno.

A necessidade de utilizar possíveis serviços, assim como um armazenamento mais elástico que não pesasse o computador, fez com que surgissem as aplicações em nuvem. Forrester Research explica que a computação em nuvem é “[...] uma capacidade padronizada de TI (serviços, software ou infraestrutura) entregue via tecnologias da Internet sob um modelo de pagamento por uso e autosserviço”. Portanto, aplicado ao projeto, será utilizado a

máquina virtual da AWS para verificação de orçamentos dos serviços que serão utilizados para o software.

3.3.1 OBJETIVOS DO PROJETO DE CLOUD COMPUTING

A verificação dos serviços disponíveis em nuvem fez com que atingisse os objetivos para o projeto e para a organização, como melhoria da eficiência operacional, automatização do gerenciamento do banco de dados, agendamento de backups automáticos e atualizações do sistema. Para isso, a Amazon RDS (Relational Database Service) foi um dos pilares para solucionar os problemas ao implementar online.

O serviço, segundo a AWS (Amazon Web Service), traz foco em inovação, migração sem rearquitetar aplicativos, redução de tempo gerenciando banco de dados, melhoria de infraestrutura de forma eficiente e diminuição de capital e valores de expansão operacional.

Imagem 8: Conceito da Amazon RDS

Fonte: AWS (Amazon Web Service).

Para reduzir os custos, o Amazon RDS usa um modelo de pagamento conforme o uso, sendo assim a ong só pagará o quanto for utilizado, ao retirar o uso de hardware reduz-se a carga de trabalho e o RDS também oferece opções de instâncias reservadas que ainda pode reduzir mais ainda os custos.

Em relação aos dados, com a ferramenta “Multi-AZ”, é possível a implementação de bancos de dados como MySQL ou PostgreSQL, tolerância a falhas com menos de 35 segundos e latência de confirmação de transação mais rápida.

Com a ajuda do Amazon RDS permite-se aumentar a escalabilidade aumentando o tamanho da instância e adicionando réplicas de leitura com facilidade, sendo assim pode ajustar a capacidade do banco de dados conforme o necessário.

3.3.2 APLICABILIDADE E BENEFÍCIOS DA CLOUD COMPUTING NO PROJETO

A computação em nuvem é usada de maneira estratégica, destacando casos de uso relevantes e alinhados aos objetivos do projeto, além dos benefícios específicos que oferece.

Maior flexibilidade e escalabilidade permite que a computação em nuvem ajuste rapidamente a capacidade de uso do banco de dados conforme for usado, sendo assim reduz a necessidade de investimento em hardware e infraestrutura, permitindo que a ONG faça mais rápido as mudanças na demanda.

A agilidade no desenvolvimento torna o projeto mais responsivo, abordando com mais eficiência a escalabilidade, disponibilidade de dados, automação e maior uptime. Os desenvolvedores podem criar e implantar instâncias de banco de dados rapidamente, acelerando o lançamento de novas aplicações e funcionalidades e promovendo inovações e adaptabilidade. Dessa forma, a computação em nuvem consegue diminuir os fardos associados ao desenvolvimento, implantação e manutenção.

Reduzir custos, principalmente investimento inicial como hardware, software e custos operacionais é um dos principais passos para que os projetos e empresas se adequem a computação em nuvem. Os principais benefícios incluem o acesso sob demanda dos recursos conforme o uso, adicionar ou remover recursos específicos, e abstração da infraestrutura, podendo os aplicativos serem transportados se necessário.

A Amazon RDS traz maior segurança que oferece criptografia de dados, controle de acesso avançado e integração com serviços de gerenciamento de identidade, sendo assim melhorando a proteção de dados e segurança reduzindo riscos associados à segurança de dados.

A aplicação concreta permite migrar os bancos de dados para a Amazon RDS sendo assim a ONG aproveita as capacidades de escalabilidade e gerenciamento simplificado oferecidas pela nuvem.

3.3.3 VANTAGENS DA CLOUD COMPUTING

A Cloud Computing foi uma grande mudança para o mercado de trabalho, principalmente na área tecnológica, por trazer melhor eficiência e produtividade na execução de seus projetos. Isso inclui diversas vantagens como:

Elasticidade dos Recursos: Adaptabilidade de acordo com a necessidade, permitindo ajustar recursos com facilidade em períodos de picos e vales.

Pagamento por Uso: Modelo adaptável que dispensa grandes investimentos iniciais; as companhias pagam somente pelo uso que fazem.

Alta Disponibilidade: Assegurando altas taxas de uptime, garantido acesso constante a dados e aplicações.

Segurança Melhorada: Os provedores têm feito investimentos consideráveis em segurança, proporcionando proteção de alto nível.

Acesso à Distância e Colaboração: Facilita o acesso a informações de qualquer local, aumentando a eficiência produtiva.

Recuperação de Catástrofes: Funcionalidades de cópia de segurança automática, assegurando a salvaguarda e a rápida recuperação de informações.

Infraestrutura barata: Os custos de infraestrutura estão sendo reduzidos ao eliminar a necessidade de hardware físico.

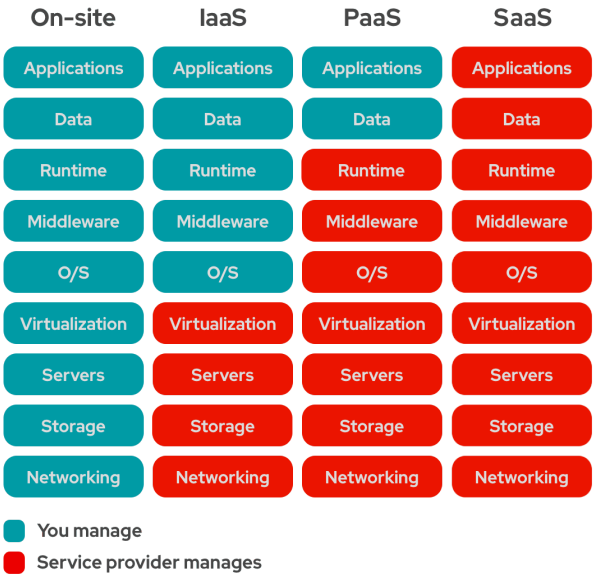
Recursos Humanos: Utilização eficaz de recursos e prevenção.

Capital Economia: Torna possível a aplicação de seus recursos em inovação, e não apenas sua perpetuação em operação.Custos operacionais reduzidos: Eficaz é o processo de gestão de TI porque se concentra em atividades estratégicas.Modelos de preços adaptativos: pacotes que atendem aos requisitos não padronizados de uma empresa.

3.3.4 DESENVOLVIMENTO EM CLOUD COMPUTING

Os modelos de aplicação em nuvem facilitam a empresa por trazer uma combinação de recursos de TI, que são oferecidos por provedores de nuvem. Com elas, as empresas conseguem desenvolver e gerenciar suas aplicações, trazendo flexibilidade, escalabilidade e eficiência. Existem três principais modelos sendo elas o SaaS (Software como Serviço), PaaS (Plataforma como Serviço) e IaaS (Infraestrutura como Serviço). Cada modelo possui sua funcionalidade e níveis de controle diferentes. Na imagem abaixo, mostra os serviços que o próprio usuário consegue controlar (azul) e os que os provedores conseguem gerenciar (vermelho).

Imagem 9: Diferenças de Gerenciamento de modelos aplicados à Nuvem.



Fonte: RedHat.

O primeiro, a Infraestrutura como Serviço oferece justamente conjuntos de serviços voltados para a infraestrutura, incluindo hardware, sistemas operacionais e conectividade. Esse modelo fornece controles em relação a servidores, virtualização assim como configuração e utilização. É utilizado por consumidores que necessitam de um alto controle sobre o ambiente da nuvem. Como exemplo, os provedores de nuvem que são IaaS são Azure, Google Cloud e AWS.

O segundo, a Plataforma como Serviço, além de oferecer conjuntos de Iaas, fornece uma plataforma onde consegue realizar testes e implantar aplicativos. Composto por recursos já configurados e de produtos e ferramentas, o ambiente fica disponível para uso para que os usuários não se preocupem em administrar os recursos de infraestrutura. Dessa forma, o gerenciamento realizado pelo próprio usuário se torna mais baixo, tornando disponível controlar apenas as aplicações e dados. Exemplo de PaaS é o AWS Elastic Beanstalk.

Por último temos o Software como Serviço, onde as empresas dependem totalmente de provedores da nuvem, desde as infraestruturas até aplicações e dados. Essa solução é a que traz mais segurança, pois após o usuário conectar o aplicativo por meio de um API, é o próprio provedor que cuidará das seguranças, escalabilidade e atualizações. Para empresas que não possuem investimentos para instalações e atualizações de software ou que será utilizada em tempos regulares, o Saas é uma ótima opção. Exemplos de Saas são Salesforce, Google Apps e Dropbox.

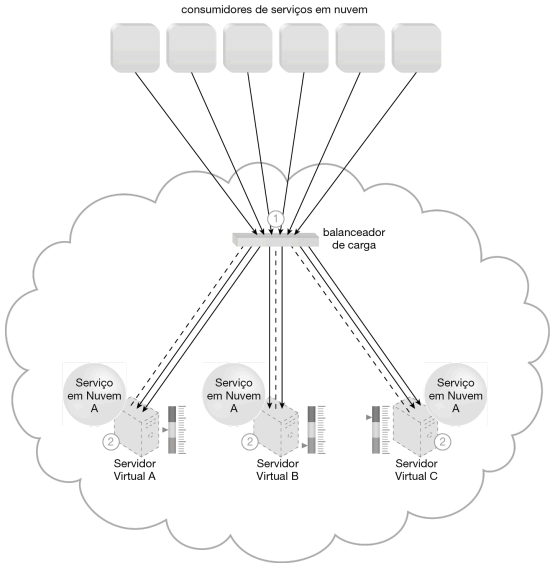
Imagem 10: Diferenças de Funções entre os modelos aplicados à Nuvem.

Modelo de entrega de nuvem	Atividades comuns de consumidores de nuvem	Atividades comuns de provedores de nuvem
SaaS	utiliza e configura serviço em nuvem	implementa, gerencia e mantém serviço em nuvem monitora o uso pelos consumidores de nuvem
PaaS	desenvolve, testa e gerencia serviços em nuvem e soluções baseadas em nuvem	pré-configura a plataforma e provisões subjacentes a infraestrutura, *middleware* e outros recursos de TI requisitados, conforme necessário monitora o uso pelos consumidores de nuvem
IaaS	prepara e configura o esqueleto de infraestrutura, e instala, gerencia e monitora qualquer *software* necessário	provisiona e gerencia o processamento físico, o armazenamento, a cabeação de rede e a hospedagem necessários monitora o uso pelos consumidores de nuvem

Fonte: Thomas Erl e Eric Barceló Monroy.

Apesar de cada modelo de aplicação ter seus diferentes níveis de controle, todos precisam de um mesmo elemento para que possam funcionar nas aplicações em nuvem que é o balanceamento de carga, onde ele ajuda a distribuir as cargas em múltiplas nuvens, podendo melhorar desempenho, escalabilidade, disponibilidade e otimização dos recursos. O balanceador além de melhorar resposta e tempo de atividade, traz escalabilidade adicionando novos servidores dependendo da demanda, tolerância a falhas caso falha um servidor e otimiza recursos maximizando o uso de cada instância.

Imagem 11: Conceito de Balanceamento de Carga



Fonte: Thomas Erl e Eric Barceló Monroy.

Aplicado ao projeto, que é um gerenciamento de cursos onde inclui cadastros e listas, os componentes relevantes da arquitetura de computação em nuvem incluem:

Instâncias virtuais: Onde será hospedado a aplicação e gerenciar cadastros, listas e lançamento de notas e faltas.

Banco de Dados: Utilização de banco de dados relacional, utilizando o MySQL como o sistema de gerenciamento de dados.

Gateway de API: Garantindo segurança e autenticação quando os docentes e administradores precisarem acessar o software.

Balanceador de Carga: Como dito acima, para dividir as cargas em outros serviços, garantindo consistência e disponibilidade do sistema.

Armazenamento de Arquivos: Guardar possíveis documentos - vídeos e imagens - ao postarem as aulas, tornando-as acessíveis para visualização e download.

Monitoramento: Como serviços de Logs para analisar as atividades e detectar problemas de forma rápida e eficiente.

Sistema de Notificações: Aplicado para docentes que precisarem receber um novo token de autenticação para alterarem suas senhas, ou alertar eventos importantes.

Esses componentes, de qualquer forma, conseguem se relacionar entre si para a melhoria do funcionamento do sistema. Como exemplo, o monitoramento e as notificações para que possam receber e identificar os problemas e solucioná-las, banco de dados e Gateway de API para que apenas os autorizados conseguem acessar os dados como notas, faltas e aulas e as instâncias virtuais com o balanceador que, ao ter um fluxo de pessoas utilizando o mesmo serviço hospedado, possa dividir essa carga e evitar congestionamento. Juntos, de forma modular e escalável, fazem com que o software alcance as demandas necessárias.

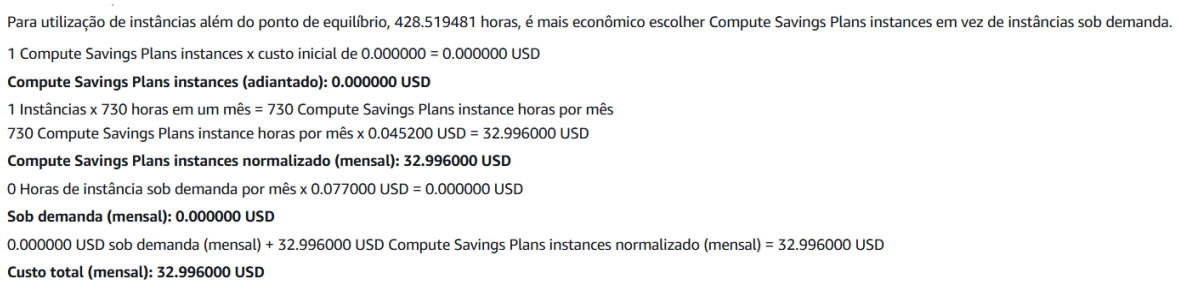
3.3.5 ESCOLHA DO PROVEDOR DE NUVEM (GOOGLE CLOUD OU AWS)

A computação em nuvem permite acessar recursos de TI de forma escalável e sob demanda, sem a necessidade de infraestrutura física. Entre os principais provedores de nuvem estão a AWS (Amazon Web Services) e o Google Cloud. A AWS, pioneira no setor, oferece uma vasta gama de serviços como computação, armazenamento e bancos de dados, sendo ideal para empresas de todos os tamanhos devido à sua flexibilidade e escalabilidade. Já o Google Cloud se destaca pela forte integração com ferramentas de dados e inteligência artificial, além de sua infraestrutura de rede global, sendo muito apreciado por

desenvolvedores e startups. Este projeto busca comparar essas duas plataformas, analisando aspectos como custos, escalabilidade, segurança e facilidade de uso, para entender qual delas é mais vantajosa em diferentes contextos.

O primeiro serviço a ser comparado é a de hospedagem para a aplicação central do projeto. Os dois orçamentos foram verificados com período de 1 ano de serviço e com base em uma instância micro na região de São Paulo, com 1vCPU (processador virtual) e 1GiB (gibibyte) e sistema operacional com SQL Server. A configuração foi realizada para um software onde o tráfego de usuários e o armazenamento são baixos. Enquanto a AWS oferece o EC2 (Amazon Elastic Compute Cloud) mensalmente por 33 USD (média de 190 reais), o Google Cloud oferece o Compute Engine mensalmente por 84.55 USD (média de 490 reais). A diferença de valor é voltado aos motivos de licenciamento da Microsoft e infraestrutura de rede em certas regiões pode ser mais cara.

Imagem 12: Orçamento do EC2 aplicado ao projeto.



Para utilização de instâncias além do ponto de equilíbrio, 428.519481 horas, é mais econômico escolher Compute Savings Plans instances em vez de instâncias sob demanda.

1 Compute Savings Plans instances x custo inicial de 0.000000 = 0.000000 USD

Compute Savings Plans instances (adiantado): 0.000000 USD

1 Instâncias x 730 horas em um mês = 730 Compute Savings Plans instance horas por mês

730 Compute Savings Plans instance horas por mês x 0.045200 USD = 32.996000 USD

Compute Savings Plans instances normalizado (mensal): 32.996000 USD

0 Horas de instância sob demanda por mês x 0.077000 USD = 0.000000 USD

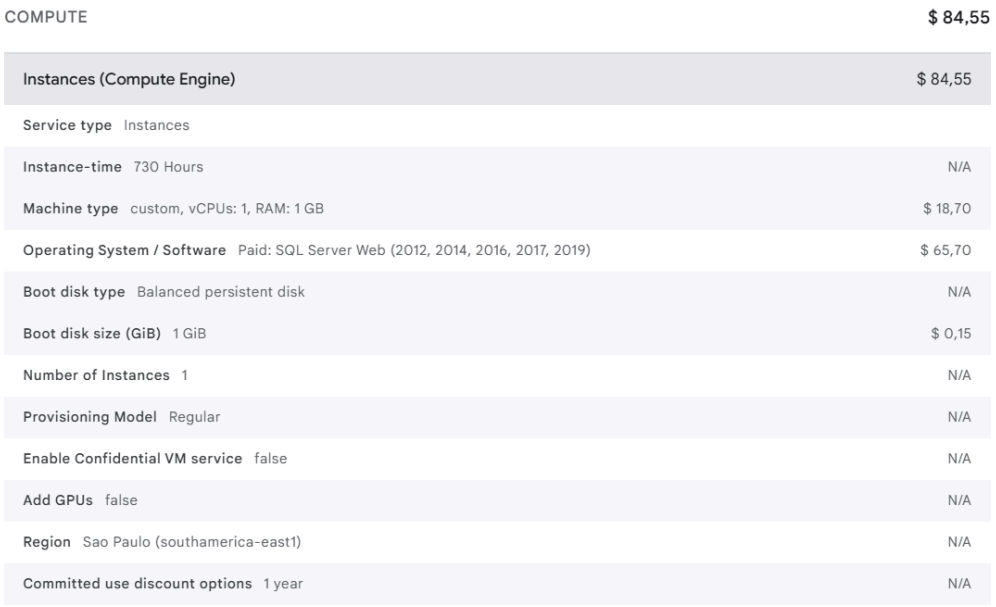
Sob demanda (mensal): 0.000000 USD

0.000000 USD sob demanda (mensal) + 32.996000 USD Compute Savings Plans instances normalizado (mensal) = 32.996000 USD

Custo total (mensal): 32.996000 USD

Fonte: Autores (2024).

Imagem 13: Orçamento do Compute Engine aplicado ao projeto.



COMPUTE	$ 84,55
Instances (Compute Engine)	$ 84,55
Service type Instances	
Instance-time 730 Hours	N/A
Machine type custom, vCPUs: 1, RAM: 1 GB	$ 18,70
Operating System / Software Paid: SQL Server Web (2012, 2014, 2016, 2017, 2019)	$ 65,70
Boot disk type Balanced persistent disk	N/A
Boot disk size (GiB) 1 GiB	$ 0,15
Number of Instances 1	N/A
Provisioning Model Regular	N/A
Enable Confidential VM service false	N/A
Add GPUs false	N/A
Region Sao Paulo (southamerica-east1)	N/A
Committed use discount options 1 year	N/A

Fonte: Autores (2024).

O segundo serviço foi em questão de armazenamento de objetos e dados, como dados pessoais de alunos e docentes, assim como de aulas, notas e faltas. Ambos orçamentos foram incluídos 8 GB (Gigabyte) de armazenamento, de forma Standard. No caso da AWS foi oferecido o S3 (Simple Storage Service) onde acrescenta outras informações como quantidade de solicitações HTTP (PUT ou GET) e quantos GB serão retornos e verificados pelo serviço, com valor mensal de 0.39 USD (média de 2,25 reais). Já o do Google Cloud oferece o Cloud Storage, com apenas um campo diferencial de quantos GB serão transferidos entre as regiões (ambas na América Latina), com valor mensal de 0.52 USD (média de 3 reais). A diferença de valor é em relação a transferência de dados.

Imagem 14: Orçamento da S3 aplicado ao projeto.

Tiered price for: 8 GB
8 GB x 0,0405 USD = 0,32 USD
Custo total do nível = 0,324 USD (custo do armazenamento do S3 Standard)
5.000 Solicitações PUT para S3 Standard Storage x 0,000007 USD por solicitação = 0,035 USD (custo de solicitações PUT do S3 Standard)
5.000 solicitações GET em um mês x 0,00000056 USD por solicitação = 0,0028 USD (custo de solicitações GET padrão do S3)
5 GB x 0,0014 USD = 0,007 USD (custo retornado do S3 Select)
5 GB x 0,004 USD = 0,02 USD (custo verificado do S3 Select)
0,324 USD + 0,0028 USD + 0,035 USD + 0,007 USD + 0,02 USD = 0,39 USD (Total de armazenamento do S3 Standard, solicitações de dados, custo da seleção do S3)
Custo do S3 Standard (mensal): 0.39 USD

Fonte: Autores (2024).

Imagem 15: Orçamento da Cloud Storage aplicado ao projeto.

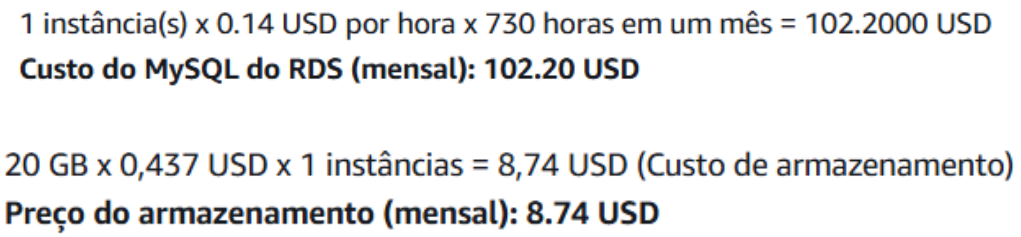
STORAGE	$ 0,52
Cloud Storage	$ 0,52
Service type Cloud Storage	
Data Transfer within Google Cloud 2 GB	$ 0,26
Total amount of storage 8 GB	$ 0,26
Location type Region	N/A
Location Sao Paulo (southamerica-east1)	N/A
Storage class Standard Storage	N/A
Source region Latin America	N/A
Destination region Latin America	N/A

Fonte: Autores (2024).

O terceiro serviço comparado é o gerenciamento do banco de dados (MySQL) para configuração, operação e escalabilidade de implantação do banco na nuvem. Ambos foram aplicados em uma instância na região de São Paulo com 1 vCPU, 1.7 GiB e 20 GB de armazenamento total. A AWS oferece o RDS para MySQL (Amazon Relational Database Service) com valor mensal de 110.94 USD (média de 640 reais) enquanto o Google Cloud

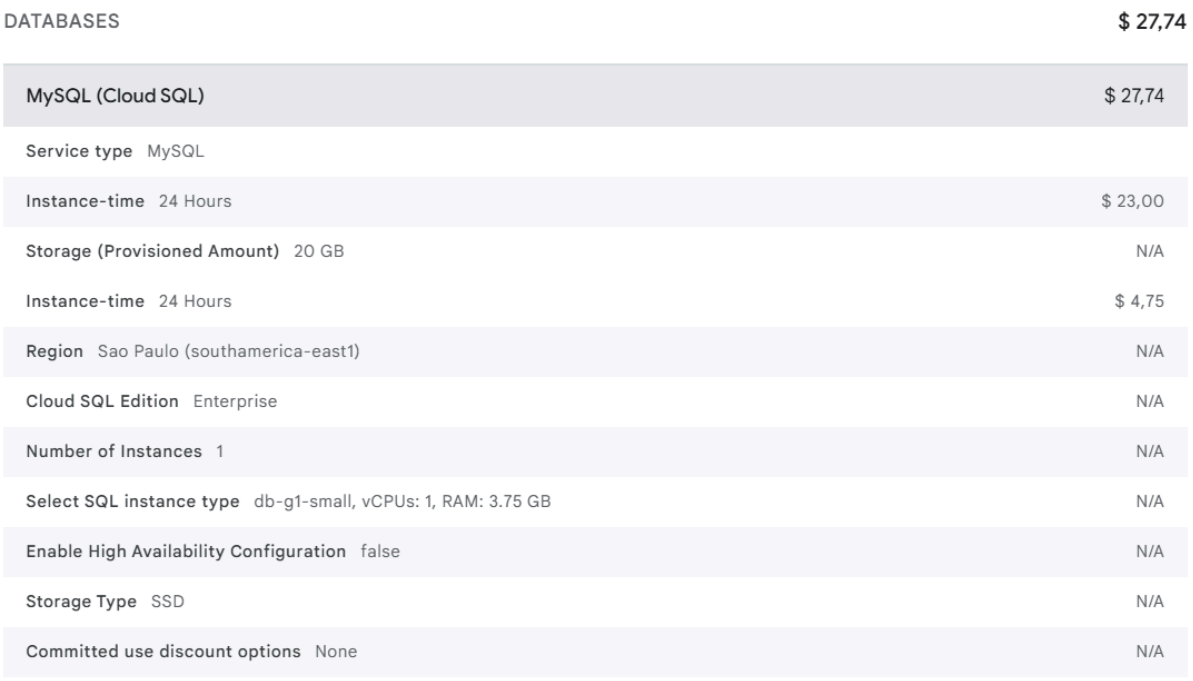
oferece o Cloud SQL com valor mensal de 27.74 USD (média de 160 reais). A diferença de valor traz motivos como modelagem de preço (cobrança em segundo da Google Cloud e em hora da AWS), custo de rede e escalabilidade, além de que o vCPU é compartilhado.

Imagem 16: Orçamento do RDS para MySQL aplicado ao projeto.



Fonte: Autores (2024).

Imagem 17: Orçamento do Cloud SQL aplicado ao projeto.

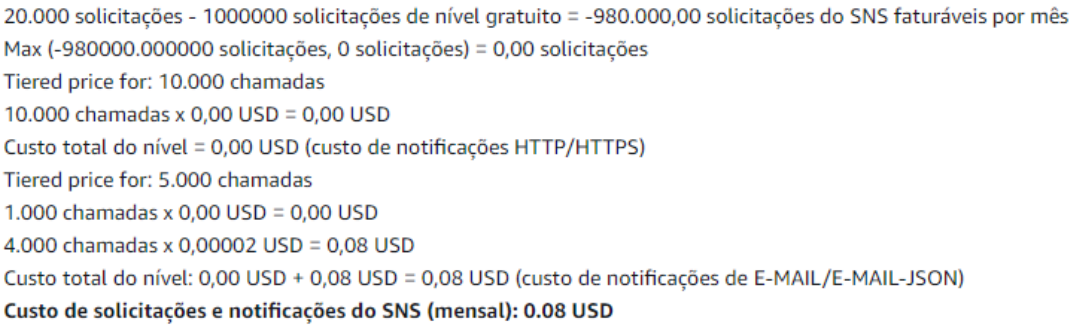


DATABASES	$ 27,74
MySQL (Cloud SQL)	$ 27,74
Service type MySQL	
Instance-time 24 Hours	$ 23,00
Storage (Provisioned Amount) 20 GB	N/A
Instance-time 24 Hours	$ 4,75
Region Sao Paulo (southamerica-east1)	N/A
Cloud SQL Edition Enterprise	N/A
Number of Instances 1	N/A
Select SQL instance type db-g1-small, vCPUs: 1, RAM: 3.75 GB	N/A
Enable High Availability Configuration false	N/A
Storage Type SSD	N/A
Committed use discount options None	N/A

Fonte: Autores (2024).

A última comparação se retrata de um serviço de notificações para que docentes e administradores possam receber notificações de possíveis eventos ou redefinição de senha. Enquanto o AWS oferece o SNS (Amazon Simples Notification Service) por 0.08 USD mensais (média de 50 centavos), a Google Cloud oferece o Firebase Cloud Messaging, que não possui custo, pois já está incluso no provedor.

Imagem 18: Orçamento do SNS aplicado ao projeto.



Fonte: Autores (2024).

Comparando os dois provedores com os serviços que serão utilizados no processo, a diferença de valores totais ficou de 112.81 USD (média de 650 reais) para a Google Cloud e 144.41 USD (média de 832 reais) para a AWS. Apesar do Google Cloud resultar em um valor mais baixo, a AWS foi escolhida pois oferece uma ampla variedade de serviços, desde computação e armazenamento até inteligência artificial, permitindo encontrar soluções para diversas necessidades, sendo a escalabilidade um grande destaque, já que pode aumentar ou diminuir recursos rapidamente conforme sua demanda.

Além disso, a confiabilidade da infraestrutura da AWS garante alta disponibilidade, com múltiplas regiões e zonas de disponibilidade. A segurança é outra prioridade, com recursos avançados que protegem dados e conformidade com várias normas de segurança.

3.3.6 DESENVOLVIMENTO EM CLOUD COMPUTING

Como mencionado, os modelos de aplicação de computação em nuvem que serão utilizados no projeto são os modelos IaaS e SaaS. O serviço IaaS tem como funcionalidade fornecer recursos de infraestrutura, armazenamento, servidores e redes, podendo os usuários gerenciar e configurar de forma que mais deseja. Já o SaaS, proporciona um ambiente diretamente pela web, sendo a base do site ideal para empresas que querem reduzir os custos com o TI e simplificar a gestão do programa.

A importância e funcionamento do balanceamento de carga em cloud, é uma forma de balancear as cargas que é essencial para a distribuição do tráfego das cargas entre os outros servidores. A escalabilidade e desempenho garantem um tráfego promote que o tráfego seja distribuído de uma forma igual, evitando sobrecarga do servidor. Redundância e alta disponibilidade quando distribui as cargas de trabalho, o balanceamento de carga aumenta a

confiabilidade, em casos de falha é direcionado a outros servidores. O balanceamento de recursos em ambiente de nuvem otimiza a utilização dos recursos que permite aos provedores usarem a infraestrutura de forma eficaz.

Outros aspectos da anatomia da computação em nuvem ajudam a garantir desempenho, disponibilidade e escalabilidade nos serviços. Os aspectos incluem servidores físicos e virtuais para fornecer várias instâncias virtuais para serviços e suportar escalabilidade, data centers para segurança e armazenamentos, redes e conectividade. Por fim, a virtualização permite dividir um único servidor físico em várias máquinas virtuais. Cada máquina virtual pode rodar um sistema operacional e aplicações como se fosse um servidor independente, o que aumenta a flexibilidade e o aproveitamento dos recursos.

Além disso, a computação em nuvem é impulsionada por paradigmas tecnológicos subjacente**s** que formam a base para flexibilidade e escalabilidade. Esses paradigmas demonstram os princípios fundamentais que permitem que a nuvem entregue serviços sob demanda, através da internet, de maneira eficiente e com custos controlados, o que inclui:

Elasticidade e Escalabilidade: Estes conceitos permitem que os recursos de cloud aumentem ou diminuam conforme a necessidade. Escalabilidade garante que a infraestrutura suporta grandes quantidades de dados e usuários; elasticidade permite que isso aconteça de forma automática.

Computação Distribuída: Em vez de rodar uma aplicação em um único servidor, a cloud utiliza uma rede de servidores interconectados para distribuir as tarefas e dados. Isso melhora a disponibilidade e a rapidez do sistema, pois se um servidor falhar, outro pode assumir a carga.

Automação e Gerenciamento: Ferramentas automatizadas são essenciais na cloud, pois facilitam o gerenciamento dos recursos, ajustes de configurações e até monitoramento de segurança.

Combinando esses aspectos, a computação em nuvem oferece uma plataforma poderosa que atende a diversas necessidades, permitindo que as empresas inovem, cresçam e se adaptem rapidamente às mudanças do mercado.

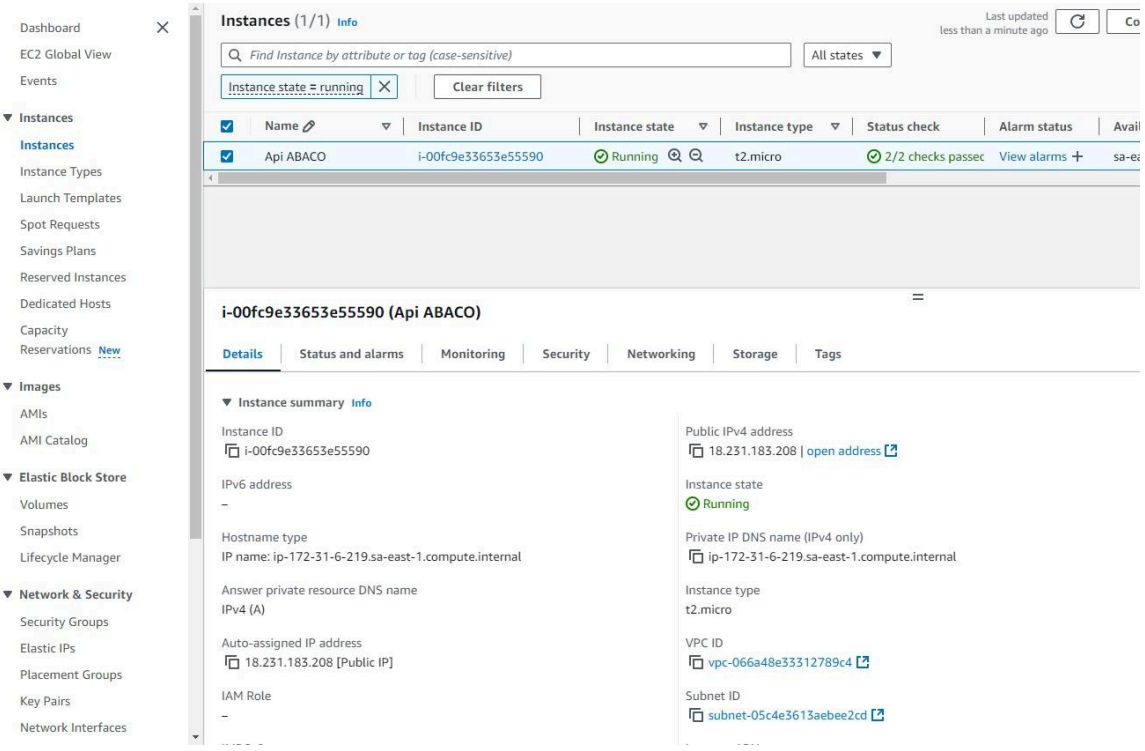
3.3.7 GOOGLE CLOUD OU AWS

Para montar um cenário de implantação na AWS com um orçamento bem baixo, o foco principal deve ser usar os serviços que oferecem flexibilidade, escalabilidade e baixo

custo. Com isso, cada serviço teve seu próprio cenário para a verificação e adequação ao projeto.

O Amazon EC2 foi escolhido para hospedar a aplicação central do sistema. Esta aplicação permite à equipe da ABACO realizar o cadastro e gerenciamento de cursos, controlar matrículas e monitorar o progresso dos alunos. Um dos principais benefícios do EC2 é a capacidade de escalabilidade automática, que permite que a aplicação aumente ou reduza seus recursos conforme a demanda. Isso é especialmente útil durante períodos de alta atividade, como a abertura de novos cursos. A segurança do EC2 é reforçada pela configuração de grupos de segurança que restringem o acesso aos dados sensíveis, garantindo que apenas usuários autorizados possam acessá-los. Para o projeto, o cenário do EC2 foi implementado com uma instância t2.micro com 1 vCPU e 1 GiB.

Imagem 19: Cenário de Implantação de EC2

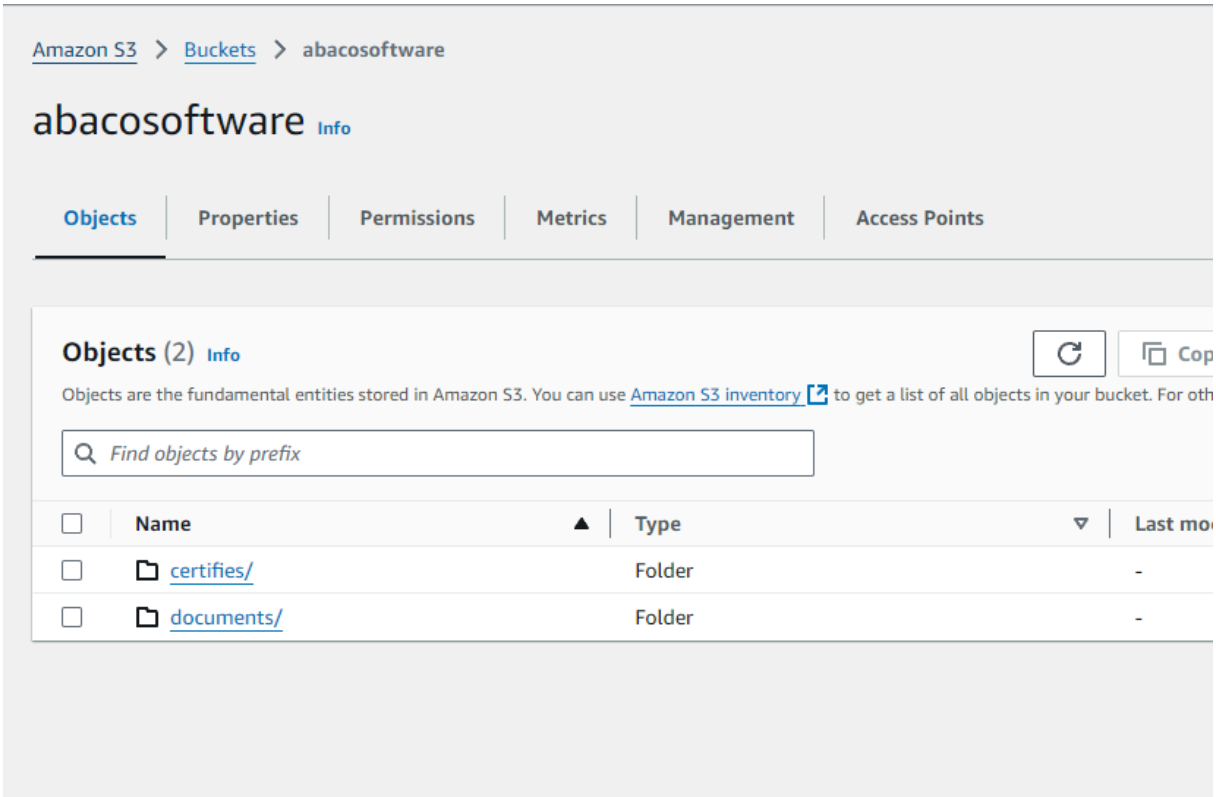


Fonte: Autores (2024).

O Amazon S3 foi designado como o repositório de armazenamento para todos os materiais dos cursos, como documentos, vídeos e certificados de conclusão. Este serviço oferece alta durabilidade e disponibilidade, assegurando que o conteúdo esteja sempre acessível aos alunos. Para otimizar os custos de armazenamento, o S3 conta com políticas de ciclo de vida que permitem a transferência de arquivos inativos para classes de armazenamento de menor custo. A estruturação dos dados em buckets específicos por curso

facilita a organização e o acesso aos materiais, tornando o processo de gerenciamento mais eficiente.

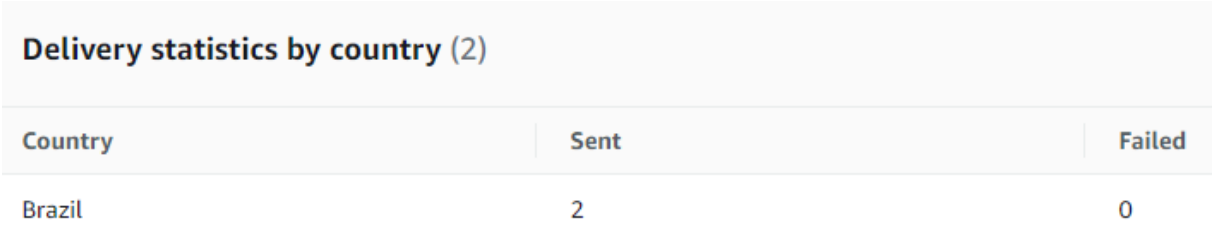
Imagem 20: Cenário de Implantação de S3



Fonte: Autores (2024).

O Amazon SNS será usado como ferramenta de comunicação, desempenhando um papel específico na segurança das contas. O envio de SMS (Short Message Service) para redefinir a senha é uma forma simples e segura de ajudar os usuários a recuperarem o acesso às suas contas. Quando um usuário esquecer a sua senha, será possível enviar um código SMS com um código padrão OTP (One-Time Password) que é um código de 6 dígitos para o usuário validar, juntamente com um token de acesso com expiração para 5 minutos. O código então será inserido em um padrão OTP disponibilizado no software, possibilitando-o de criar uma nova senha. Essa abordagem oferece uma camada adicional de segurança e oferece uma melhor experiência ao usuário.

Imagem 21: Cenário de Implantação da Amazon SNS



Delivery statistics by country (2)

Country	Sent	Failed
Brazil	2	0

Fonte: Autores (2024).

Em conjunto, esses três serviços da AWS proporcionam uma infraestrutura completa e eficiente para o sistema de controle de alunos da Organização ABACO, oferecendo não só robustez e segurança, mas também flexibilidade e otimização de recursos.

3.4 ESTRUTURA DE DADOS

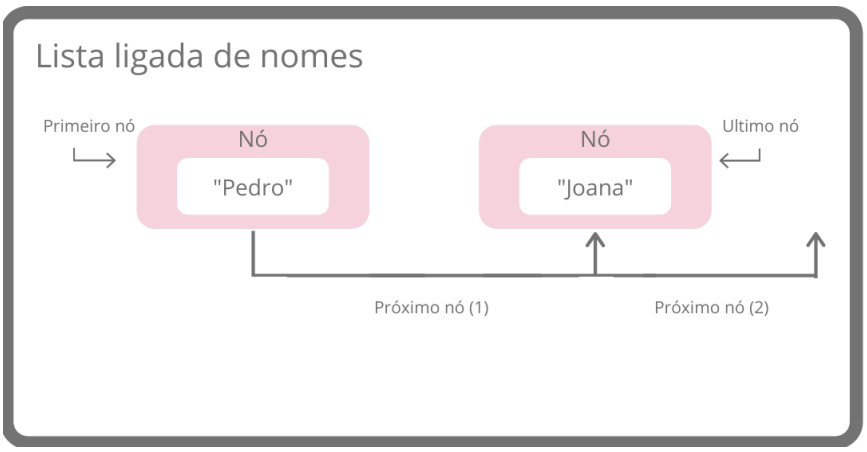
Os algoritmos são construídos a partir das estruturas de dados, sendo ela essencial para a eficiência do programa (Finkel e Bentley, 2005). As estruturas são formas organizadas de armazenar e manipular dados em um sistema, permitindo que informações sejam gerenciadas de maneira eficiente.

Diferentes tipos de estruturas de dados como listas, pilhas, filas e árvores, oferecem diversas formas de acesso e manipulação, dependendo das necessidades do projeto. Como o projeto foca na funcionalidade de chamada de alunos, o sistema utilizará a linguagem "TypeScript" e a estrutura de dados "Lista" para permitir a gestão eficiente de informações sobre alunos e docentes, registro de aulas, notas e cursos e controle de presença.

3.4.1 LEVANTAMENTO DE REQUISITOS

Uma das estruturas mais utilizadas e a única que será aplicada ao projeto são as listas. As listas são baseadas em uma sequência encadeada de elementos, de maneira que estejam dispostas uma do lado da outra de forma horizontal. É utilizada principalmente em casos com conjuntos pequenos, onde as principais funções dentro delas é a de remover, adicionar e visualizar elementos. Os elementos são denominados como os "nós" da lista, onde possuem o conjunto "Estudante" com as informações pessoais e um ponteiro para o próximo conjunto (outro estudante).

Imagem 22: Exemplo de Estrutura de Lista



Fonte: TreinaWeb.

A definição da estrutura de dados do projeto como várias listas foi resultado da organização e decisão final do que o projeto consistiria. Como já mencionado, o objetivo do software é o gerenciamento de estudantes, notas, faltas e aulas. Com isso, os requisitos funcionais que serão implementados são:

Gerenciamento de Estudantes: O sistema deve permitir o cadastro de novos estudantes com informações pessoais e acadêmicas (nome, data de nascimento, nível de escolaridade, gênero, RG, CPF, email, telefone e endereço), edição e atualização das informações, listar todos os estudantes cadastrados, possibilitando filtragem por curso e situação de matrícula e exclusão de registros.

Gerenciamento de Cursos: O sistema deve permitir cadastro de novos cursos oferecidos pela organização (nome, período, docente, descrição), associação de estudantes aos cursos matriculados, incluindo detalhes das turmas, controle de situação (ativo, inativo ou em andamento) e exclusão e edição de cursos cadastrados.

Controle de Matrícula: O sistema deve permitir registro e cancelamento de matrículas de estudantes nos cursos e geração de relatórios de matrículas (quantidade por curso, situação e detalhes de progressão).

Controle de Acesso: O sistema deve ter autenticação e autorização dos usuários, garantindo acesso diferenciado para administradores e docentes e registro de ações realizadas pelos usuários para auditoria e segurança.

Da mesma forma, os requisitos não funcionais - características e qualidade do sistema - foram definidos, sendo eles:

Disponibilidade e Escalabilidade: O sistema deve ser acessível, garantindo alta disponibilidade para todos os usuários e escalável para suportar picos de acesso, principalmente em épocas de matrícula.

Usabilidade: Interface do sistema deve ser intuitiva e responsiva, facilitando uso em dispositivos móveis e desktops.

Segurança e Privacidade: O sistema deve utilizar autenticação para segurança e criptografia para proteção de dados.

Para garantir o desempenho adequado do sistema, tempos de respostas e uso de memória foram realizados junto ao funcionamento do software e às operações desejadas da estrutura de lista. O tempo de resposta será diferente de cada lista, pois depende do seu tamanho e do conjunto de dados que a interface deve mostrar.

O tempo de resposta segue em uma constância pelas listas, no momento, serem com uma quantidade de conjuntos de dados parecidas. A escolha pela estrutura de lista traz clareza e organização aos dados que precisam ser visualizados pelos usuários. Assim como a implementação dos requisitos funcionais e não funcionais permite ao usuário praticidade e segurança para realizar seus objetivos dentro do software, atendendo as expectativas da instituição.

3.4.2 VALIDAÇÃO DOS REQUISITOS

O sistema será baseado exclusivamente em listas como estrutura de dados, o que simplifica o modelo de dados e facilita o desenvolvimento e a manutenção do software. Ao utilizar listas para gerenciar informações como alunos, cursos e matrículas, o sistema ganha em flexibilidade, atendendo às demandas da organização de forma direta e eficiente. Esse modelo oferece ainda a vantagem de uma integração mais leve e otimizada com a arquitetura de nuvem, proporcionando um desempenho adequado às operações de leitura e escrita. As vantagens de usar as listas no projeto incluem:

Simplicidade de Implementação e Manutenção: As listas são uma estrutura de dados fundamental, amplamente suportada por diferentes linguagens e ambientes de programação. Por serem intuitivas e de fácil manipulação, o uso de listas permite que a equipe de desenvolvimento implemente rapidamente funcionalidades básicas de cadastro, atualização, listagem e exclusão de registros. Em sistemas que não exigem operações complexas de consulta ou relação entre os dados, como no caso da ABACO, as listas oferecem a simplicidade necessária, sem comprometer a eficiência.

Flexibilidade para Gerenciar Informações Diversas: No sistema da ABACO, as listas serão utilizadas para armazenar informações sobre alunos, cursos e matrículas, atendendo de maneira direta e eficiente as necessidades de armazenamento de dados. Cada lista pode ser dedicada a uma categoria de informações (ex.: listaAlunos, listaCursos, listaMatriculas), o que facilita a estruturação e a organização do código. A flexibilidade das listas permite que dados sejam adicionados, removidos ou atualizados conforme necessário (TANENBAUM; WETHERALL, 2011).

Eficiência em Operações de Leitura e Escrita: Em um ambiente de computação em nuvem, onde o sistema precisa ser escalável e responsivo, a utilização de listas proporciona um modelo de armazenamento leve e eficiente. As listas permitem uma integração simplificada com APIs e serviços de banco de dados em nuvem, otimizando a transmissão de dados e facilitando a escalabilidade horizontal, essencial para suportar picos de acesso, como os que ocorrem em períodos de matrícula (ARMSTRONG, 2014).

O uso de listas como estrutura de dados no sistema oferece uma solução simplificada e prática para gerenciar informações de alunos, cursos e matrículas, alinhada com as necessidades específicas da Organização ABACO. Ao adotar listas como estrutura central, o sistema mantém uma abordagem leve, que facilita a integração com serviços de computação em nuvem e permite um desempenho adequado nas operações de alta demanda. Além disso, a escolha por listas assegura que o sistema será capaz de crescer junto com a organização, mantendo a eficiência e a responsividade em todas as operações essenciais de gerenciamento e controle.

3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: ENFRENTANDO ESTEREÓTIPOS

3.5.1 ENFRENTANDO ESTEREÓTIPOS

A apostila “enfrentando estereótipos” inicia-se a partir da apresentação de diversidades culturais, religiosas e étnicas dentro do contexto universitário. Durante esse tempo, cria-se vínculos sociais, incluindo compartilhamento de experiências e influências nos comportamentos.

Antes de exemplificar, é importante a definição da palavra “estereótipo”, que simplifica e generaliza diversos aspectos da vida, como estilos de vida, aparências, ações e pensamentos de forma categorizada.

Em casos universitários é comum a união em grupos para realizar diversas atividades, transformando aos poucos a individualidade própria da pessoa em coletividade com o grupo. Essas transformações acontecem ao longo do tempo e podem ser positivas, já que fazem parte do processo de formação da personalidade. Apesar da mudança, a identidade própria ainda deve estar presente em si mesmo, mesmo com pressão social.

Para a criação do grupo, pessoas que possuem gostos similares tendem a se unir de forma mais fácil e natural.

Outro tipo de estereótipo forçado ao longo da humanidade é a definição de cursos que se encaixam em cada gênero - masculino e feminino -, Isso se inicia a partir da infância, onde reforçam, como exemplo, na escolha de brinquedos e cores voltados para meninas e meninos, fazendo com que interfira nas escolhas profissionais das pessoas.

Ademais, a sociedade atual junto com a mídia, forçam a seguir modelos de padrões voltados a aparência, assim como conquistas, estudos e relacionamentos. Essa pressão se torna uma das consequências que o estereótipo traz: interpretação rasa do valor de uma pessoa pelo que sua aparência ou escolhas representam diante daquele padrão.

Portanto, essa formação de estereótipos pode trazer influências negativas para a sociedade se feita de forma geral, pois pode gerar preconceitos e fazer com que pessoas sofram com essa pressão ao longo de sua vida.

Um tópico apresentado na apostila se volta ao padrão do que a mulher tem que viver, sempre ao redor de cuidar da família e da sua própria moradia, sendo fundamental dentro da sociedade.

A idealização da beleza está presente, principalmente, nas redes sociais e propagandas de produtos para beleza, tanto para homens quanto para mulheres. Por não terem um corpo que se encaixe no estereótipo de beleza, muitas pessoas buscam chegar nesse patamar através de cirurgias plásticas e/ou dietas radicais que podem acabar com o corpo e o psicológico. Em um ambiente social, como a universidade, possuir um corpo que não esteja próximo ao padrão pode trazer insegurança e baixa autoestima.

Atualmente as séries televisivas possuem influência o bastante para mudar um modo de vida, independente da idade. Isso ocorre sobretudo, pois os telespectadores se identificam com os personagens e suas atitudes, o lado positivo é que elas divertem, educam e informam,

vendo pelo outro lado, a má interpretação pode ter efeitos muito perigosos na mente de uma pessoa conturbada, fazendo-a cometer um crime.

Uma plataforma com poder o suficiente para alterar um estilo de vida é o Youtube, originalmente um site, para assistir e postar vídeos, que graças a remuneração baseada no número de visualizações e seguidores se tornou, de certa forma, profissional. Seu maior problema são as pessoas dispostas a fazer qualquer coisa para ganhar atenção e, consequentemente, dinheiro.

Aqueles com muitos seguidores na internet são chamados de influenciadores digitais, pois conseguem influenciar seus seguidores em alguma instância, seja opinião, atitude, moda, entre outros. Dito isso, transtornos podem ser causados pelo seu público, como por exemplo, um influenciador diz não ter gostado do produto de tal marca e seus seguidores acabam indo xingar a mesma, do mesmo modo, o público pode se sentir inferior por conta de sua aparência. Em resumo, um influenciador deve ter responsabilidade na hora de se comunicar com seus fãs, pois suas palavras têm peso.

Geralmente o preconceito se manifesta em frases comuns e piadas, como por exemplo, “lugar de mulher é na cozinha hahaha”, “acorda, baiano”, seu principal malefício é o preconceito e a exclusão daqueles que não se encaixam nos padrões estabelecidos. Embora presentes no dia a dia, principalmente na internet, também existem campanhas que buscam promover a diversidade. No fim, sua melhor forma de combate é o diálogo, visando compreender e respeitar.

Estando presente até mesmo no mercado de trabalho, onde jovens são vistos como sem experiência e com um ritmo lento, já idosos (no Brasil) são desatualizados e possuem ritmo diferente dos demais. Isso ocorre por conta do choque de gerações, já que cada geração possui habilidades e experiências distintas, mas essa diferença pode enriquecer tanto o ambiente de trabalho quanto às relações pessoais.

3.5.2 ESTUDANTES NA PRÁTICA

O material prático solicitado foi realizado no formato de vídeo, publicado na plataforma online “Youtube”. O vídeo retrata sobre a importância cultural de cada região brasileira, assim como o significado de “estereótipo” e suas consequências na vida atual.

O material está disponível em link no anexo 3.

4. CONCLUSÃO

Com a finalização do projeto, foi adquirida uma visão mais abrangente e sistemática de cada processo, contemplando as demandas necessárias para seu correto funcionamento, bem como as exigências inerentes a cada etapa.

Durante esse percurso, compreendeu-se que o funcionamento de um aplicativo em versão desktop, a partir de códigos e banco de dados, é um processo complexo e multidisciplinar, que envolve diversas fases desde a implementação do projeto em nuvem, assim como finalização e integração do front-end com o back-end até armazenando os futuros dados que serão incluídos pela ABACO.

Um dos desafios encontrados foi a aplicação prática dos conhecimentos teóricos adquiridos, o que demandou aprofundamento em cada tema abordado no semestre, pesquisa adicional em cada área e, assim, a elaboração completa da programação do projeto que pode auxiliar na instituição e em outros imóveis.

Nesse sentido, foi fundamental se colocar no cliente para atingir o objetivo final de entregar um software com as necessidades desejadas, o que permitiu novas experiências e novos aprendizados durante esse semestre.

REFERÊNCIAS

ABACO. **Associação Beneficente de Apoio à Comunidade.** Disponível em: https://abaco.ong.br/. Acesso em: 15 agosto 2024.

ANDRADE, Ana Paula. **O que é e como funciona a estrutura de dados de lista.** Disponível em: https://www.treinaweb.com.br/blog/o-que-e-e-como-funciona-a-estrutura-de-dados-lista. Acesso em: 06 nov. 2024.

ARMSTRONG, T. ***Cloud computing: principles, systems and applications***. Springer, 2014. Acesso em: 07 nov. 2024.

AWS. ***AWS Pricing Calculator.*** Disponível em: https://calculator.aws/. Acesso em: 08 nov. 2024.

CURY, Thiago E.; BARRETO, Jeanine dos S.; SARAIVA, Maurício de O.; et al. **Estrutura de Dados**. Porto Alegre: SAGAH, 2018. *E-book.* p.53. ISBN 9788595024328. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9788595024328/. Acesso em: 06 nov. 2024.

ERL, Thomas; MONROY, Eric B. **Computação em Nuvem: Conceitos, Tecnologia, Segurança e Arquitetura**. 2nd ed. Porto Alegre: Bookman, 2024. E-book. p.23. ISBN 9788582606599. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9788582606599/. Acesso em: 30 out. 2024.

FINKEL, H.; BENTLEY, J. ***Data Structures: An Advanced Approach.*** New York: Academic Press, 2005. Acesso em: 07 nov. 2024.

GOOGLE CLOUD. ***Google Cloud Calculator.*** Disponível em: https://cloud.google.com/products/calculator. Acesso em: 08 nov. 2024.

HERNANDEZ, Michael. ***Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design.*** 3. ed. Berkeley: Addison-Wesley, 2003. Acesso em: 15 out. 2024.

JUNIOR, E. **Ensinamentos de Jakob Nielsen sobre interação do usuário com interfaces.** 14 Set. 2021. Disponível em: https://brasil.uxdesign.cc/ensinamentos-de-jacob-nielsen-sobre-intera%C3%A7%C3%A3o-do-usu%C3%A1rio-com-interfaces-f510ac9b2a73. Acesso em 04 nov. 2024

KAMIENSKI, Carlos. **Introdução ao Paradigma de Orientação a Objetos.** Centro Federal de Educação Tecnológica da Paraíba, 1996. Acesso em: 25 out. 2024.

MACHADO, Felipe Nery R. **Banco de Dados – Projeto e Implementação.** 4ª edição. Rio de Janeiro: Érica, 2020. E-book. pág.28. ISBN 9788536532707. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9788536532707/. Acesso em: 09 out. 2024.

NORMANDO, Celio. **MVC e Princípios de projeto.** Disponível em: https://medium.com/@celionormando/arquitetura-mvc-e-princ%C3%ADpios-de-projeto-3d0b278ef910. Acesso em: 29 out. 2024.

REDHAT. **O que são IaaS, PaaS e Saas?.** Disponível em : https://www.redhat.com/pt-br/topics/cloud-computing/iaas-vs-paas-vs-saas. Acesso em 07 nov. 2024.

SAUDETE, Alexandre. **Ebook - REST: Construa Apis Inteligentes de Maneira Simples**. Casa do Código, 2013. Acesso em: 25 out. 2024

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. ***Database System Concepts.*** 6. ed. New York: McGraw-Hill, 2011. Acesso em: 15 out. 2024.

SILBERSCHATZ, A.; KORTH, Henry F.; SUDARSHAN, S. **Sistemas de banco de dados.** 6. ed. São Paulo: Pearson, 2019. Acesso em: 22 out. 2024.

SILVA, Luiz FC.; RIVA, Aline D.; ROSA, Gabriel A.; e outros. **Banco de Dados Não Relacional .** Porto Alegre: SAGAH, 2021. E-book. pág.2. ISBN 9786556901534. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9786556901534/. Acesso em: 09 out. 2024.

TANENBAUM, A. S.; WETHERALL, D. ***Computer Networks*.** 5. ed. Pearson, 2011. Acesso em: 07 nov. 2024.

ANEXOS

ANEXO 1: Drive contendo o banco de dados do projeto. Disponível em: https://drive.google.com/drive/folders/1bwrQL2UXCI5uPzXOFT8mXGBLkhfHYupz.

ANEXO 2: Protótipo do software realizado a partir do site Figma. Disponível em: https://www.figma.com/design/wUlk6Zuqivzmawq4jRQ74E/ABACO?node-id=0-1&t=6IFOCO1gglleOOQA-1.

ANEXO 3: Vídeo formação para vida. Disponível em: https://youtu.be/hyrX-4UeOPs.