UNIFACS

Ciência da Computação (Unidade Professor Barros)

ARIANE SANTOS GOMES

GABRIEL SANTOS DE SILVA

JOÃO PEDRO DE O. DOS SANTOS SILVA

PEDRO AZEVEDO DE QUEIRÓZ

RELATÓRIO DA ATIVIDADE FORMATIVA A3 DA UC DE USABILIDADE E DESENVOLVIMENTO WEB E MOBILE

Salvador
2022

ARIANE SANTOS GOMES

RA:1272121526

GABRIEL SANTOS DE SILVA

RA:1272121748

JOÃO PEDRO DE O. DOS SANTOS SILVA

RA:1272117418

PEDRO AZEVEDO DE QUEIRÓZ

RA:1272123228

RELATÓRIO DA ATIVIDADE FORMATIVA A3 DA UC DE USABILIDADE E DESENVOLVIMENTO WEB E MOBILE

Relatório apresentado a Unidade Curricular de Usabilidade Web, Mobile do curso de Ciência e Computação, da instituição de Ensino UNIFACS-Professor-Barros, como requisito parcial para a obtenção da Avaliação 3 (A3).

Orientador: Adailton de Jesus Cerqueira Junior

Salvador
2022

SUMÁRIO

INTRODUÇÃO

Esse trabalho mostra a aplicação por meio da linguagem de Modelagem unificada (UML), aplicada para apoiar o desenvolvimento de sistemas de informação orientados a objetos. Os processos baseados em diagrama de classes são usados para dar suporte ao desenvolvimento da visualização de dados dentro desse modelo de referência.

Na etapa de análise, um diagrama de classe pode ajudá-lo a compreender os requisitos do domínio do problema e a identificar seus componentes. As aplicações orientadas a objetos, possuem elementos que geralmente são convertidas em classes e objetos de software reais quando você grava o código. Logo, permite a uma análise e aos modelos exibirem as partes específicas do sistema, interfaces com o usuário, implementações lógicas e assim por diante. Os diagramas de classe são, em suma, uma versão modelo de como o sistema funciona, os relacionamentos entre os componentes do sistema em vários níveis e o plano de implementação desses componentes.

Sendo parte do processo em desenvolver aplicações, a modelagem de banco de dados, igualmente, é a etapa de levantamento, análise, categorização e exploração de todos os dados e tipos de informações que irão sustentar uma aplicação.

MODELOS DO PROJETO

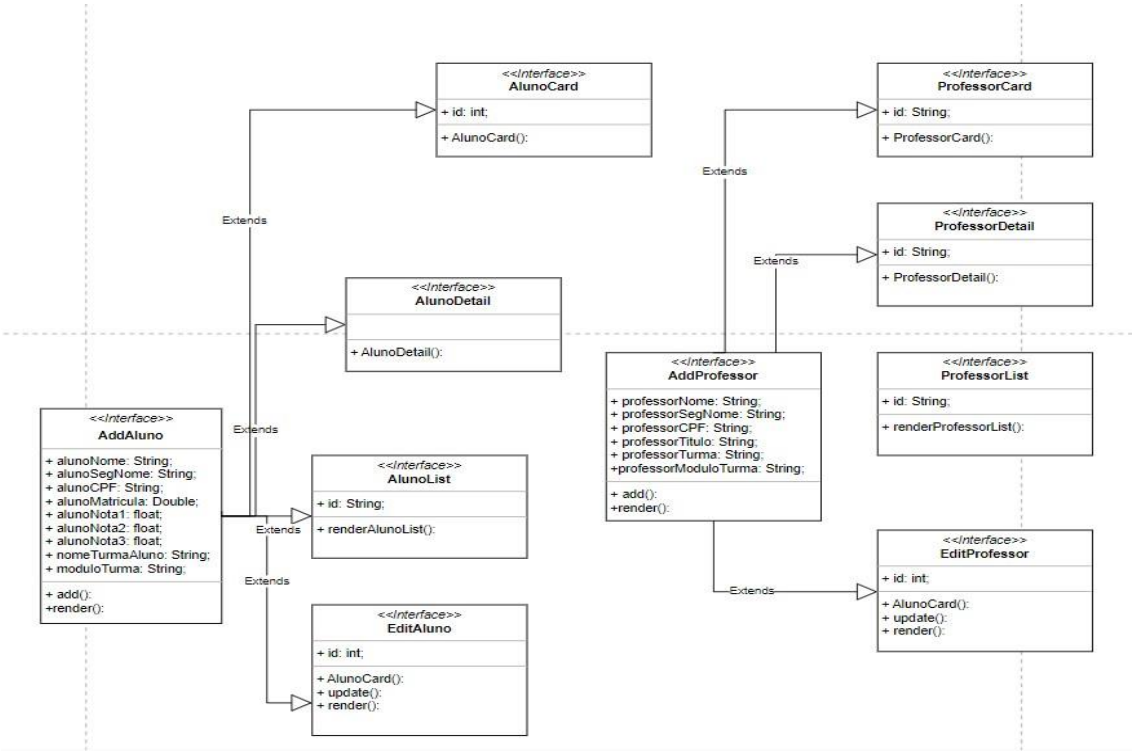


Figura 1 Diagrama de Classe da Aplicação

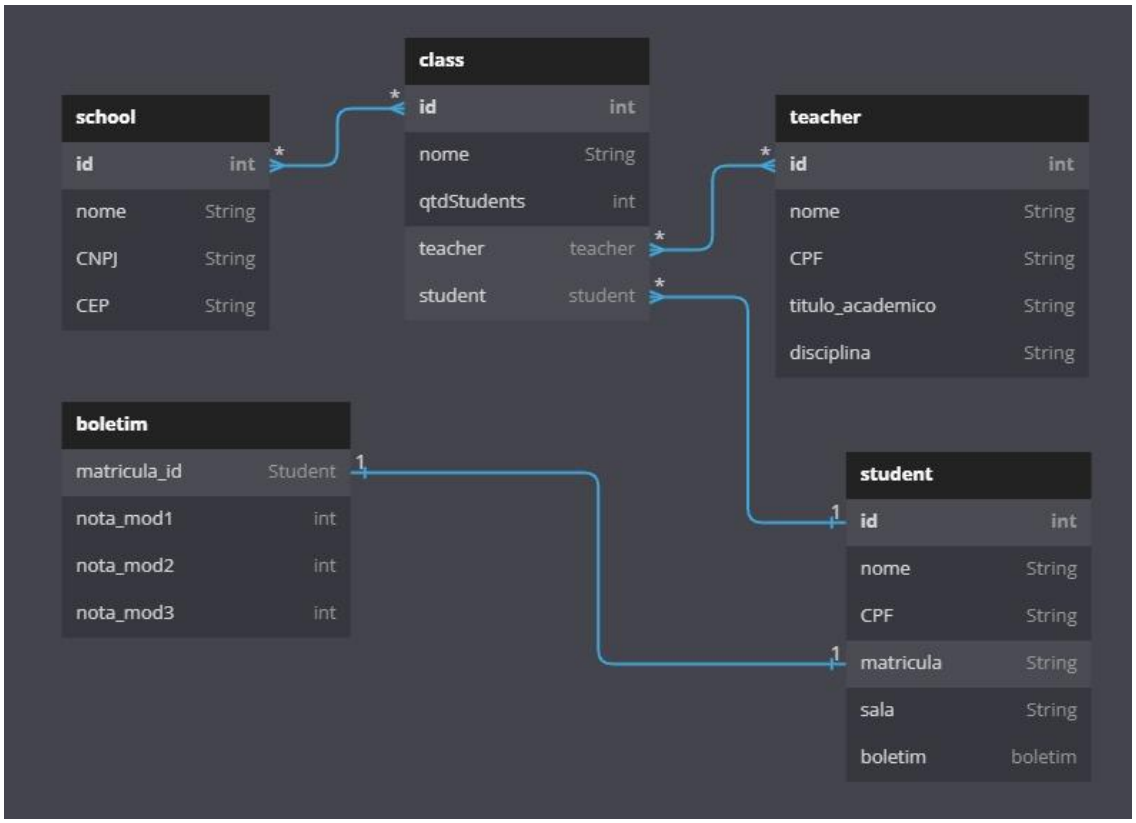


Figura 2 Diagrama de Banco de Dados da Aplicação

DESENVOLVIMENTO E DISCUSSÃO ACERCA DO *SOURCE-CODE*

RESULTADOS

O conceito das regras do negócio passado para nossa equipe foram os de criar um Web-App, utilizando-se de REACT.js em plataforma Node.js e devendo realizar operações de CRUD. Sendo que deveria estar direcionado ao uso de empresas com fins de ensino e educação.

Em nossas reuniões inicial, ocorrida em 14 de setembro do corrente ano, ainda não tínhamos o conceito geral ou direcionamento do projeto como no parágrafo acima foi citado, mas, decidimos quais frameworks usaríamos, como o BootStrap, Json-server, para efetivar o trabalho.

Após a informação das regras do negócio, ficou confusa a situação em questão, pois exigia o uso de um Banco de Dados extremamente seguro e complexo como o ***PostgreeSQL*** para apenas uma entidade escolar, que em nosso caso, seria uma empresa de venda e aplicação de cursos de reforço de matérias escolares.

Não obstante, criamos um sistema de front-end basicamente independente do Banco de Dados do ***pSQL*** (*PostgreeSQL*) onde na nossa própria aplicação do sistema, criamos dois servidores para manuseio de dados para o CRUD utilizando-se do ***Json-Server***.

Como fora requisitado, porém, disponibilizamos os meios para a criação do servidor e Data Base no ***pSQL*** com as *querys* de sua concepção **(Anexo 2, assim como na pasta "back-end" do repositório)**.

Inicialmente, nos preocupamos em trazer toda a funcionalidade do CRUD no sistema, separando os Alunos, dos Professores, cada um em um arquivo independente ***.JSON*** onde seria inicializado, de maneira separadas e em portas separadas no *localhost*, pelo ***json-server*.** Neste projeto, utilizou-se para a o servidor dos Alunos a porta **3006** e para o servidor dos Professores a porta **3007**, já configuradas no ***package-json*** para quando sofrerem o "***npm start***" rodarem nestas portas e de maneira correta.

O ***Axios*** foi a escolha, ao invés do ***Fetch API***, pois traz melhores resultados com arquivos .JSON, já que ele possui dentro de si o *parse* automático do arquivo ***.JSON*** para ***data***.

Para a criação de novos Professores e Alunos colocamos em uso o ***uuidv4,*** capaz de gerar ID`s únicas para cada vez que for adicionado no CRUD, gerar Id`s

fáceis para a **futura implementação de sistema de Login e Tokens, caso o cliente desejar**.

Optamos por fazer uma base de sistema onde demonstramos a simplicidade deste (o **ReforçA+**) e sua ampla aplicação e customização de acordo com desejos da empresa de ensino cliente.

Optamos pelo ***BootStrap5*** por ser uma *Lib Framework* extremamente versátil e ao mesmo tempo responsiva.

Por fim, decidimos manter a linguagem principal e total do projeto como a ***React JavaScript***, evitando ao máximo o uso de *TypeScript.*

Usamos os ***Hooks*** do **React** para utilizar as duas *apis* (**api** e **api2**, respectivamente no código) criadas pelo ***Axios*** para dar o ***get*** no *json-server* e retornar as *promises* em forma de ***data*** dentro do código.

Utilizando-se do ***useState*** e ***useEffect*** e o método de ***props***, criamos o sistema de CRUD e de interpretação dos componentes criados.

Utilizou-se ambos os conceitos de Componentes Classes e Componentes Functors do **React.JS**.

RESULTADOS

Inicialmente, reconhecemos o nível de conhecimento e competências dos integrantes do grupo para contribuir na aplicação. Então, concordamos em Javascript, como a linguagem de programação básica do projeto, junto a TML e CSS. A seguir, estão as tecnologias e bibliotecas também utilizadas:

- *Node.js -v 16.17.1*.
- *React Javascript*, biblioteca padrão para criação de *Apps* e *Pages*;
- A *Axios Lib*, para efetuar o *Get*, *Post*, *Push*, *Update*, *Delete* das informações no *Json-server*;
- Decidiu-se pela criação de duas apis para abrir em *localhost* as informações de Profesores e Alunos em formato *.JSON* usando o *Json-server Lib*;
- *Bootstrap - Lib Framework do Bootstrap*;
- *React-Bootstrap - Lib Framework* de *React Components do Bootstrap*;
- A *Lib uuIDv4* – para a criação de ID's únicas para os usuários caso necessite criar um sistema de login com tokens;
- A *Lib cdbreact - Components Lib de React Bootstrap5*;
- As paletas de cores usadas neste projeto se encontram em: https://color.adobe.com/My-Color-Theme-color-theme-21055343/ (Figura. 01);
- A tabela de *querys* necessárias para criar o *database* (Figura. 2 Diagrama de Banco de Dados da Aplicação) dentro do ***PostgreeSQL*** caso o cliente deseja-se o usar está inclusa, apenas necessário sua inserção no *query-selector* do ***PgAdmin4***, ou por meio do *cmd.exe*, *Powershell*, ou outro terminal de execução compatível.

O projeto tem como objetivo prover uma simples solução com poucos custos e poucas complicações de programações para os usuários leigos e novas empresas que buscam entrar no mercado de ensino de reforço escolar privado.

A aplicação do ReforçA+ é um sistema onde o aluno é vinculado a uma disciplina da escola e, os professores vinculados a essas matérias, conforme o módulo

em que o aluno está. Dessa forma, o aluno cadastrado terá suas notas conforme o módulo que ele está cadastrado e um professor da matéria.

Um dos primeiros componentes apresentados é o “**Header**”, a ***navbar*** simples e totalmente customizável possuindo 3 Links dentro dela: clicando na **Brand**, será redirecionado para a “**Home**”, clicando em “**Home**”, o mesmo, e possuindo um “**Menu**”, ***dropdown*** do **BootStrap5**, onde, no modelo, poderá acessar a área de *Management de Alunos e Professores*, mas podendo ser totalmente customizado de acordo com o desejo da Empresa. O componente “**Home**” do projeto possui um sistema de slide com rolagem automática (*Carousel*) para visualizar notícias ou recados, e trazendo dados sobre o sistema de maneira customizável.

Colocamos dois Botões para fácil acesso, no modelo de apresentação, para o *Management System de Alunos e Professores*.

Trazemos ainda o ***Footer***, onde traz links que direcionam às redes sociais do *ReforçA+* (Ou a Empresa Cliente), como o Facebook, Twitter e Instagram, contatos com o ReforçA+ (Ou a Empresa Cliente) e a parte de ajuda, caso o usuário esteja com algum problema.

O sistema cadastra o aluno com seus dados como Nome e CPF, após o adiciona a uma disciplina e módulo da turma, com sua matrícula e notas, se possuir. O aluno pode ser editado conforme inicie em um novo módulo ou mude algum de seus dados e notas, são ordenados na lista que é possível ser visualizada em “Menu → Seção de Management de Alunos”, além de ser deletados os seus cadastrados, caso deseje.

Da mesma forma o sistema cadastra os professores e os lista pelo caminho “Menu → Seção de Management de Professores”. Nesta parte, é possível visualizar os professores cadastrados no sistema, cadastrar um novo professor com Nome e CPF, definir o título acadêmico específico, seja graduado, mestre, doutor, dentre outros, a turma que ensina e o módulo. Assim como nas funcionalidades relacionadas aos alunos, as informações de professor podem ser alteradas e/ou excluídas.

CONSIDERAÇÕES FINAIS

Por fim, entendemos que a simplicidade de um sistema pode trazer melhor resultados do que toda sua aplicação e construção como um Software.

Não obstante, apenas criamos o ***back-end*** como algo opcional, pois apesar de banco de dados serem algo necessário hoje em dia, nem sempre, para uma empresa de ensino que não possui capital grande para sua manutenção e contratação de mão de obra capacitada, é possível.

O ***postgreeSQL*** é um excelente banco de dados, tanto que empresas que utilizam este são as que geralmente estão intrínsecas com sistemas do governo, como SUS, FIES, etc. Por esta razão, entendemos que seria de demasiado custo promover uma solução com sua aplicação total para todas as empresas que almejamos ofertar nosso serviço do **ReforçA+.** Neste âmbito, optamos pela sua *opcionalidade* de implantação de acordo com a demanda, apenas deixando criado seu esqueleto e seu diagrama para caso seja necessária sua aplicação.

Por isso, tentamos criar uma solução mais simples e direta o possível onde qualquer pessoa com o desejo nobre de ensinar e ajudar, mesmo que de forma gratuita, como em comunidades necessitadas, possam usar.

O projeto foi criado com o viés de suprir qualquer capacidade financeira de sua empresa cliente, podendo ser acessível para todos e todas as comunidades que tanto, por muitas vezes, necessitam.

REFERÊNCIAS

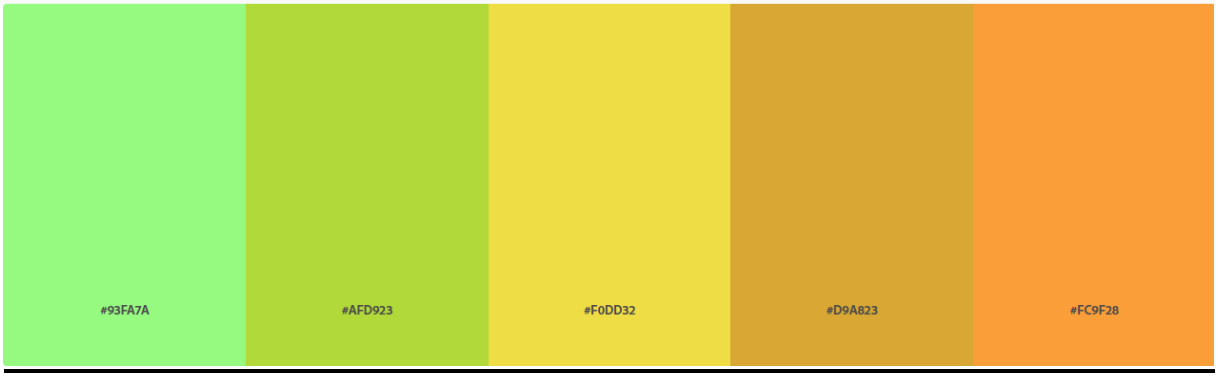
A APLICAÇÃO da Linguagem de Modelagem Unificada (UML) para o suporte ao projeto de sistemas computacionais dentro de um modelo de referência: The application of UML to support computational systems design within a reference model framework. **Carlos Alberto Costa**, Rio Grande do Sul, p. 19-36, 4 abr. 2001. Disponível em: https://www.scielo.br/j/gp/a/RRQQ7mKTFztQXK9Sz7BKtWQ/?lang=pt. Acesso em: 7 dez. 2022.

IBM, Documentação. Rational Software Architect Standard Edition. *In*: **Diagrama de Classe**. [*S. l.*], 5 mar. 2021. Disponível em: https://www.ibm.com/docs/pt-br/rsas/7.5.0?topic=structure-class-diagrams. Acesso em: 7 dez. 2022.

ANEXOS

Anexo 01

Figura 01:



Anexo 02:

Querys do pSQL:

```
CREATE DATABASE schooldb
    WITH
    ENCODING = 'UTF8'
    LC_COLLATE = 'Portuguese_Brazil.1252'
    LC_CTYPE = 'Portuguese_Brazil.1252'


CREATE TABLE school (school_id INTEGER NOT NULL UNIQUE,
                                        nome VARCHAR(255),
                                        CNPJ VARCHAR(255),
                                        CEP VARCHAR(255),
                                        PRIMARY KEY(school_id));


CREATE TABLE teachers (teacher_id INTEGER NOT NULL UNIQUE,
                                         nome VARCHAR(255) NOT NULL,
                                         CPF VARCHAR(255) NOT NULL UNIQUE,
                                         titulo_academico VARCHAR(255) NOT NULL,
                                         disciplina VARCHAR(255),
                                         PRIMARY KEY(teacher_id));


CREATE TABLE students (students_id INTEGER NOT NULL UNIQUE,
                                         nome VARCHAR(255) not null,
                                         CPF varchar(255) NOT NULL UNIQUE,
                                         matricula VARCHAR(255) NOT NULL UNIQUE,
                                         sala VARCHAR(255),
                                         PRIMARY KEY(students_id));


CREATE TABLE boletim (boletim_id INTEGER NOT NULL UNIQUE,
                                        matricula VARCHAR(255) references
```

```
                                        students (matricula),
                                        nota_mod1 INTEGER,
                                        nota_mod2 INTEGER,
                                        nota_mod3 INTEGER,
                                        PRIMARY KEY(boletim_id));

CREATE TABLE classes (class_id INTEGER NOT NULL UNIQUE,
                                        nome VARCHAR(255), qtdStudents INTEGER,
                                        PRIMARY KEY(class_id),
                                        teacher INTEGER REFERENCES teachers (teacher_id),
                                        student INTEGER REFERENCES students (students_id));

ALTER TABLE students ADD boletim INTEGER REFERENCES boletim (boletim_id);
ALTER TABLE classes ADD CONSTRAINT PROFESSOR UNIQUE (teacher, class_id );
```

ANEXO 03:

Repositório do GitHub:

https://github.com/Keirz/UsabilidadeA3_Entrega_2022_2

ANEXO 04:

- **DAS INSTRUÇÕES DE USO:**
- Primeiramente irá baixar como arquivo ZIP o código disponibilizado neste repositório.
- Abrir com o Visual Studio Code.
- Em seguida Abrir o terminal na pasta Entrega_2022_2 e digitar:
- npm install
- cd contact-app
- npm install
- npm install axios
- (caso necessário, npm audit fix --force para corrigir problemas de versões)
- cd server-api
- npm install
- cd server-api-profs
- npm install

=> Com isso você terá instalado os recursos necessários para rodar o app, agora só falta fazer acontecer com as seguintes linhas de comando:

- Abrir 1 console de comando no Visual Studio Code e digitar:
- cd server-api
- npm start
- Agora abrir um segundo console de comando no Visual Studio Code e digitar:
- cd server-api-profs
- npm start
- Abrir um terceiro console de comando no Visual Studio code e digitar:
- cd contact-app
- npm start

Pronto! Com isso você estará rodando dois json-servers:

- Na porta 3006 se encontra as informações dos Alunos.
- Na porta 3007 se encontra as informações dos Professores.

E na porta principal 3000 estará rodando o App ReforçA+.