



INSTITUTO FEDERAL Brasília

Instituto Federal de Brasília

Campus Brasília

Curso Superior de Tecnologia em Sistemas para Internet

SISTEMA DE ATENDIMENTO PARA A COORDENAÇÃO GERAL DE ASSISTÊNCIA ESTUDANTIL

Por

JOSÉ NETO LIMA NASCIMENTO

Tecnólogo

BRASÍLIA

2019

José Neto Lima Nascimento

**SISTEMA DE ATENDIMENTO PARA A COORDENAÇÃO GERAL DE
ASSISTÊNCIA ESTUDANTIL**

*Trabalho apresentado ao Programa de Curso Superior de
Tecnologia em Sistemas para Internet da Instituto Federal
de Brasília como requisito parcial para obtenção do grau
de Tecnólogo em Sistemas de Internet .*

Orientador: Alex Helder Cordeiro de Oliveira

BRASÍLIA
2019

José Neto Lima Nascimento

Sistema de atendimento para a Coordenação Geral de Assistência Estudantil/ José
Neto Lima Nascimento. – BRASÍLIA, 2019-
53 p. : il. (algumas color.) ; 30 cm.

Orientador Alex Helder Cordeiro de Oliveira

Tecnólogo – Instituto Federal de Brasília, 2019.

1. Palavra-chave1. 2. Palavra-chave2. I. Orientador. II. Universidade xxx. III.
Faculdade de xxx. IV. Título

CDU 004

José Neto Lima Nascimento

Sistema de atendimento para a Coordenação Geral de Assistência Estudantil

Trabalho de conclusão de curso de graduação apresentado a Coordenação do Curso Superior de Tecnologia em Sistemas de Internet do Instituto Federal de Brasília – Campus Brasília, como requisito parcial para a obtenção do título de Tecnólogo em Sistemas de Internet.

Aprovado em: ____ de _____ de ____.

BANCA EXAMINADORA

Prof. Alex Helder Cordeiro de Oliveira
Computação/IFB

Prof.^a Dr.^a Primeira Membro da Banca
Computação/IFB

Prof. Dr. Segundo Membro da Banca
Computação/IFB

Prof.^a Dr.^a Terceira Membro da Banca
Computação/IFB

BRASÍLIA
2019

Dedico este trabalho à minha família.

Agradecimentos

Agradeço ao meu orientador Prof. Dr. Nome do Orientador, pela sabedoria com que me guiou nesta trajetória.

Aos meus colegas de sala.

A Secretaria do Curso, pela cooperação.

Gostaria de deixar registrado também, o meu reconhecimento à minha família, pois acredito que sem o apoio deles seria muito difícil vencer esse desafio.

Enfim, a todos os que por algum motivo contribuíram para a realização desta pesquisa.

Elemento opcional.
Espaço destinado à epígrafe (elemento opcional). Nesta folha, o autor usa
uma citação, seguida de indicação de autoria e ano, relacionada com a
matéria tratada no corpo do trabalho.
—NOME DO AUTOR

Resumo

NASCIMENTO, José Neto Lima. SISTEMA DE ATENDIMENTO PARA A COORDENAÇÃO GERAL DE ASSISTÊNCIA ESTUDANTIL. 2021. 52 f. Trabalho de Conclusão de Curso (Graduação) – Tecnólogo em Sistemas para Internet. Instituto Federal de Brasília – Campus Brasília. Brasília/DF, 2021.

O presente trabalho tem como objetivo o desenvolvimento de um sistema para fazer o controle do fluxo de atendimentos que são feitos no decorrer do expediente da Coordenação Geral de Assistência Estudantil do Instituto Federal de Brasília - Campus Brasília. Após algumas reuniões com o demandante, foram notadas diversas necessidades que o modelo de trabalho atual não supria, com todos os problemas levantados após as reuniões, foram feitas diversas etapas para levantar todas as funcionalidades que o sistema deveria ter para que os problemas fossem resolvidos e automatizados. Chegou a fase de desenvolvimento, que foi usada a linguagem de programação JAVA, utilizando-se o padrão MVC (model, view e controller). Antes da implementação foram feitos vários testes para comprovar que todas as necessidades estavam sendo atendidas. Quanto a implementação, devido aos tempos de pandemia, ainda não foi possível implementar o sistema no Instituto Federal, utilizou-se uma rede com dois computadores para testar a compatibilidade de sistema e eventuais erros que pudessem estar ocorrendo. Após a conclusão de todas as etapas da implantação do sistema, ficará destinado ao Coordenação Geral de Assistência e Estudantil do IFB, para manutenção e futuras atualizações.

Palavras-chave: Java. Padrão MVC. Instituto Federal de Brasília. Atendimentos. Sistema Desktop.

Abstract

NASCIMENTO, José Neto Lima. SISTEMA DE ATENDIMENTO PARA A COORDENAÇÃO GERAL DE ASSISTÊNCIA ESTUDANTIL. 2021. 52 f. Trabalho de Conclusão de Curso (Graduação) – Tecnólogo em Sistemas para Internet. Instituto Federal de Brasília – Campus Brasília. Brasília/DF, 2021.

The present work has as objective the development of a system to control the flow of attendances that are made during the working hours of the General Coordenação Geral de Assistência Estudantil of the Federal Institute of Brasília - Brasília Campus. After a few meetings with the plaintiff, several needs were noted that the current work model did not meet, with all the problems raised after the meetings, several steps were taken to raise all the functionality that the system should have in order for the problems to be resolved and automated. The development phase has arrived, using the JAVA programming language, using the MVC standard (model, view and controller). Before implementation, several tests were carried out to prove that all needs were being met. As for the implementation, due to the pandemic times, it was not yet possible to implement the system at the Federal Institute, a network with two computers was used to test the system compatibility and any errors that could be occurring. After the completion of all stages of the implementation of the system, it will be assigned to the IFB's Coordenação Geral de Assistência Estudantil, for maintenance and future updates.

Keywords: Java. MVC standard. Instituto Federal de Brasília.

Lista de Figuras

2.1	Página principal do Sistema	18
2.2	Página principal do Sistema	19
2.3	Tela do sistema após login do usuário	20
3.1	Diagrama de Entidade e Relacionamento do sistema	24
3.2	Modelo lógico relacional do banco de dados do sistema	24
3.3	Diagrama de casos de uso do sistema	25
3.4	Diagrama de sequência do sistema	26
3.5	Diagrama de atividades do sistema	27
3.6	Diagrama de classes do sistema	28
3.7	Representação do padrão MVC	29
3.8	Código de um programa em Java	30
3.9	Código para conexão do JDBC	31
3.10	Código e tela criada em JavaFX	32
3.11	Tela do DB Browser	33
3.12	Tela do brModelo	34
3.13	Tela do Astah UML	35
3.14	Tela do JavaFX Scene Builder	36
4.1	Banco de dados criado para o sistema	38
4.2	Estrutura do código com o padrão MVC	39
4.3	Página de solicitação de senhas	40
4.4	Página de login	41
4.5	Lista de atendimentos na fila	42
4.6	Tela de configurações do sistema	43
4.7	Tela de cadastro de servidores	44

Lista de Algoritmos

Lista de Tabelas

2.1	Cronograma de Atividades	21
A.1	List of conferences on which the searches were performed.	51
A.2	List of journals in which the searches were performed.	52
A.3	Search string per Search Engine.	53

Lista de Acrônimos

CGAE - *Coordenação Geral de Assistência Estudantil*

FURB - *Fundação Universitária Regional de Blumenau*

JBPM - *Java Business ProcessManagement*

IFB - *Instituto Federal de Brasília*

SGBD - *Sistema de Gerenciamento de Banco de Dados*

UML - *Unified Modeling Language (Linguagem de Modelagem Unificada)*

JDBC - *Java Database Connectivity*

SQL - *Structured Query Language*

CRUD - *Create, Read, Update e Delete*

IDE - *Ambiente Integrado de Desenvolvimento*

Sumário

1	Introdução	14
1.1	Tema	14
1.2	Problema	14
1.2.1	Objetivo geral	15
1.2.2	Objetivos específicos	15
1.3	Estrutura do TCC	15
1.3.1	Classificação de pesquisa	16
2	Conceitos gerais e referencial teórico	17
2.1	Conceitos gerais	17
2.2	Referencial Teórico	18
2.2.1	SISTEMA DE HELP DESK E CONTROLE DE CHAMADOS BASE- ADO EM WORKFLOW	18
2.2.2	MV Sistemas	19
2.3	Comparação entre trabalhos correlatos	20
2.4	Cronograma	21
3	Metodologia	22
3.1	Metodologia Ágil	22
3.1.1	Scrum	23
3.2	Tecnologias	23
3.2.1	Banco de dados	23
3.2.2	UML	24
3.2.2.1	Diagrama de casos de uso	25
3.2.2.2	Diagrama de sequência	25
3.2.2.3	Diagrama de atividades	26
3.2.2.4	Diagrama de classes	27
3.2.3	Padrão MVC	28
3.2.4	Linguagem de programação	29
3.2.4.1	Java	29
3.2.4.2	JDBC	30
3.2.4.3	JavaFX	31
3.3	Ferramentas utilizadas	32
3.3.1	NetBeans 8.2 RC	32
3.3.2	DB Browser for SQLite	32
3.3.3	SQLite	33
3.3.4	br Modelo 3.3	33

3.3.5	Astah UML	34
3.3.6	JavaFx Scene Builder	35
4	Apresentação e Análise dos Resultados	37
4.1	Desenvolvimento	37
4.2	Fase de testes	44
5	Conclusões e Trabalhos Futuros	45
	Referências	46
	Apêndice	49
A	Mapping Study's Instruments	51

1

Introdução

Em meio ao progresso e evoluções tecnológicas, a automatização de uma tarefa manual, torna-se essencial, isso gera benefícios para os setores de uma organização. Os processos são realizados de uma maneira mais rápida e efetiva, economizando tempo e recursos que seriam gastos com o trabalho sendo feito manualmente, existem diversas ferramentas para facilitar e agilizar tarefas.

A gestão de processos pode ser implantada em uma organização e funcionar como ferramenta chave de melhoria contínua dos processos produtivos, aumentando a eficiência e consequentemente minimizando perdas e maximizando lucros (LOPES, 2008).

O processo de automatização consiste em transformar rotinas de tarefas repetitivas feitas manualmente, para uma atuação automática, padronizada e eficiente, minimizando o risco de falhas e inseguranças sobre procedimentos, garantindo a sua funcionalidade e melhores resultados. Para Gonsalves (2000), “o processo obedece a uma sequência estrita de atividades, ditada pela sua tecnologia característica ou pela própria lógica do trabalho. A fabricação de produtos como bicicletas, camisas e livros se dá por meio de processos industriais cujas atividades devem ser realizadas em sequência estrita”.

Esse trabalho propõe um sistema para automatizar processos da Coordenação de Assistência Estudantil (CGAE) que são feitos de forma manual, visando a economia de recursos, que são gastos durante esses processos.

1.1 Tema

Desenvolvimento de um sistema que facilite o registro de atendimento da Coordenação Geral de Assistência Estudantil do Instituto Federal de Brasília (IFB), de forma que auxilie a gestão de diversos procedimentos envolvendo a coordenação.

1.2 Problema

Após ser feito um conselho de classe com os professores e servidores do Instituto Federal de Brasília, foram levantadas algumas demandas com relação a Coordenação Geral de Assistência

Estudantil. Dentre algumas demandas, foi verificado que atendimento que é feito atualmente no setor, deixa a desejar quanto ao registro de pessoas que são atendidas pelos servidores.

A Coordenação Geral de Assistência Estudantil do Instituto Federal de Brasília, necessita de um sistema para organizar o atendimento aos alunos e servidores. Após o conselho de classe e levantamento de demandas, foi percebido alguns problemas com relação ao atendimento dos alunos e servidores no CGAE, o sistema atual deixa a desejar no quesito de registrar dados do atendimento sem gastar recursos físicos. Os procedimentos descritos são realizados de forma manual, onde as anotações são feitas no papel, e uma das preocupações é o uso excessivo e desnecessário para qualquer atendimento que é feito.

Além de economizar tempo e perda de dados do público que é atendido no setor, uma automatização do sistema, traria alguns benefícios como economia de tempo, controle do fluxo de atendimento, direcionamento do atendimento para o lugar certo e evitar sobrecarga dos funcionários do setor.

1.2.1 Objetivo geral

O objetivo central é desenvolver um sistema de apoio à Gerencia das diversas atividades desempenhadas pelo CGAE, o sistema busca diminuir o uso de papel, automatizando o atendimento, para auxiliar o controle de pessoas que são atendidas durante o expediente e os servidores que fizeram o atendimento.

1.2.2 Objetivos específicos

- Registrar todos os atendimentos que serão feitos no CGAE;
- Especificar os tipos de atendimentos;
- Registrar o servidor que fez o atendimento e a pessoa que foi atendida;
- Incluir a data, hora, o tipo de atendimento;
- Organizar de modo que agilize o atendimento, evitando filas, com emissão de senhas, direcionando o atendido ao atendente que o atenderá;
- Utilizar os registros de atendimentos para categorizar os tipos atendimentos mais frequentes e em qual época do ano ocorrem com mais frequência.

1.3 Estrutura do TCC

No capítulo 1, é feita uma breve descrição da introdução do projeto, especificando os objetivos principais para o sistema. No capítulo 2, é abordando os trabalhos correlatos, trabalhos semelhantes ao propósito deste projeto, foi realizada uma pesquisa levando em conta funcionalidades semelhantes que poderiam atender as necessidades do CGAE.

1.3.1 Classificação de pesquisa

Com os objetivos desse projeto descritos, está sendo feita uma pesquisa de caráter exploratório, com procedimentos de um estudo de caso e com métodos qualitativos.

2

Conceitos gerais e referencial teórico

2.1 Conceitos gerais

O setor de atendimento ao cliente pode ser encontrado em todos os tipos de organização, sejam elas produtoras de bens ou serviços, não importando o porte e o ramo de atuação. Com o alto crescimento da tecnologia, existe uma necessidade de automatizar todos os serviços, as instituições necessitam de uma forma de executar atividades que antes eram demoradas, para uma forma bem rápida e eficaz. Isso implica na qualidade desde atendimento, segundo Kotler (1998), “Qualidade é, a totalidade de aspectos e características de um produto ou serviço, que proporcionam a satisfação das necessidades declaradas e implícitas”.

O sistema atual do CGAE do Instituto Federal de Brasília, conta com um serviço de atendimento ao público, que tem uma demanda de atendimento, onde são atendidos alunos, servidores e o público em geral, sejam eles com vínculo com a instituição ou não. O atendimento é feito por ordem de chegada, sem emissão de senha ou algum aviso prévio, para fazer o registro é utilizado o uso de papel, onde fica estipulado o tipo de atendimento e o tempo de atendimento que foi feito.

Podemos dividir o objetivo deste trabalho em duas partes, um catálogo de serviços que são prestados pelos CGAE e automatizar o atendimento para o tipo de público, são eles alunos, servidores e pessoas não vinculadas ao Instituto.

Para o catálogo de serviços será usado como base para definir uma regra de funcionamento, um gerenciamento de catálogos de serviços. Segundo Freitas, "O Gerenciamento do Catálogo de Serviços tem como objetivo desenvolver e manter o catálogo de serviços, de forma a garantir sua atualização quanto aos detalhes, status e dependências de todos os serviços atuais ou em desenvolvimento e garantir a sua visibilidade e disponibilidade para os interessados".

A automotização do atendimento, consiste em ofertar uma maneira de que a pessoa que queira atendimento, possa se identificar previamente, informando o tipo de atendimento que deseja, assim sendo direcionado para o servidor que fará o atendimento.

2.2 Referencial Teórico

2.2.1 SISTEMA DE HELP DESK E CONTROLE DE CHAMADOS BASEADO EM WORKFLOW

O sistema¹ tem como objetivo, melhorar o gerenciamento Seção de Apoio ao Usuário da Fundação Universitária Regional de Blumenau (FURB).

O sistema possibilita que, o gerenciamento de todos os chamados técnicos, além de possibilitar ao gestor de TI um acompanhamento detalhado de todos chamados feitos.

O sistema de Help Desk foi desenvolvido utilizando o framework Java Business Process Management (JBPM).

Além disso, é informado no chamado as observações sobre o problema apresentado pelo equipamento em questão. O atendente também informa o tipo de chamado que ele está criando, para que ele seja direcionado automaticamente ao tipo correspondente de técnico.

A Figura 2.1, mostra a tela principal do sistema.



Figura 2.1 Página principal do Sistema

O sistema apresentado nesse tópico, possui algumas funcionalidade que se adequam ao propósito do projeto, entretanto não atende totalmente as necessidades, o sistema não conta

¹Disponível em: <http://dsc.inf.furb.br/arquivos/tccs/monografias/2007-1cristianpauloprigo1vf.pdf>> Acesso em: 05 Nov. 2019

com um registro de dados completos das solicitações do chamados, e também não consta uma funcionalidade que gera um relatório de atividades feitas.

2.2.2 MV Sistemas

A empresa MV Sistemas² possui mais de 20 anos no mercado, o sistema é voltado para atendimento em hospitais em todo Brasil, tem como finalidade automatizar o fluxo de atendimento e controle de dados dos usuários, foi desenvolvido inicialmente na linguagem de programação Clipper, sendo um dos primeiros do País a atender a maioria das funções de um hospital, o sistema distribui uma senha para os pacientes, no cadastrado da senha é feito a especificação e classificação de risco para cada paciente. Após a senha ser gerada o paciente deve aguardar ser chamado pela recepção. Os funcionários da recepção tem acesso as senhas solicitadas, os recepcionistas têm a função de controlar e encaminhar os pacientes para o determinado tipo de atendimento.

O sistema possibilita que o enfermeiro tenha acesso aos dados dos pacientes no ato do atendimento, onde estará especificado o grau de risco e o tipo de atendimento que deverá ser prestado.

A Figura 2.2, mostra a tela principal do sistema.



Figura 2.2 Página principal do Sistema

²Disponível em: <http://www.mv.com.br/pt/>> Acesso em: 05 Nov. 2019

O sistema possui algumas funções de prontuário que informa os pacientes internados no hospital, pacientes que estão ainda no hospital ou que já receberam alta.

Na figura 2.2, temos uma tela de acesso, que é disponibilizado aos recepcionistas e enfermeiros, nesta tela é possível ver o fluxo de atendimento, estoque de produtos, pacientes internados e alguns registros feitos pelos médicos ou enfermeiros após atendimentos.

Produtos

Código: 733 | Descrição: ADOCANTE LIQUIDO 110ML | Descrição Resumida: ADOCANTE LIQUIDO 110 ML

KIT: Mestre | Consignado: Produto Mestre (Genérico) | Não | Não | Não

Unidade: UND | Unidade: UND | Sexo: Ambos | Cód. Sist:

Espécie: 10 | ALIMENTO | Classe: 1 | GÊNEROS NÃO PERECÍVEIS | Sub Classe: 1 | ENLATADOS

Curva ABC: Sem | Lote: Não | Validade: Não | Etiqueta: Não Utilizar | Medicam: Não | Lista: | Código D.C.B: | Controlado Moviment: Não | Bloqueia: Sim | Padronizado: Não

Atividade: 5 | GÊNEROS ALIMENTÍCIOS | Procedimento de Faturamento: | Divisor Fabur:

Unidade: | **Estoque** | Especificação | Substituição | Portaria | Fabricantes | Empresas | Adicionais

Estoque dos Produtos

Localização: | Classificação ABC: Moderado

Qtde Última Mvta	Qtde Estoque Atual	Qtde Est. Doado	Qtde Est. Reservado	Qtde Ordem	Qtde Solicitação	Qtde Est. Máximo	Qtde Est. Mínimo
05/01/2003 15:40	0.000	0.000	0.000	84.000	168.000	10.000	6.000
Qtde Ponto de Pedido	Qtde Cons. Méd. do Cálculo	Qtde Cons. Mensal	Qtde Cons. Trimestral	Qtde Cons. Semestral	Qtde Cons. Anual	Qtde Média Cons.	
7.000	6.000	0.0000	0.000	0.000	0.000	0.0000	0.0000

Digite o Código do Estoque - lista de valores disponível

Record: 1/1 | <OSC> <DBG>

Figura 2.3 Tela do sistema após login do usuário

O sistema apresentado nesse tópico, possui várias funcionalidades que não atendem ao propósito desse projeto, algumas delas são desnecessárias para a CGAE, e não teriam utilidade nenhuma, o sistema também não conta com uma funcionalidade de relatório de atividades feitas, que se adequariam a necessidade do CGAE.

2.3 Comparação entre trabalhos correlatos

O sistema proposto, tem como finalidade ser exclusivo para o Coordenação Geral de Assistência Estudantil, nesse tópico foram apresentados dois sistemas semelhantes, que não atendem exclusivamente as necessidades do setor, e tem uma sobrecarga de serviços que não seriam usados pelo CGAE, o projeto tem o intuito de atender especificamente as necessidades do setor, visando a praticidade e menor custo.

O sistema proposto nesse projeto, será um sistema desktop, feito na linguagem de programação JAVA, utilizando um banco de dados PostgreSQL para armazenar dados que serão salvos quando o registro do atendimento for feito.

2.4 Cronograma

Tabela 2.1 Cronograma de Atividades

Atividades	Ago	Set	Out
Estudar Banco de dados/JDBC e Redes	X		
Criar Banco de Dados	X	X	
Fazer interface do público		X	
Criar interface de relatório de atendimentos		X	X
Fazer interface do público		X	X
Escrever Monografia	X	X	X

3

Metodologia

A presente pesquisa feita nesse projeto, é classificada como exploratória, segundo Gil (2002), este tipo de pesquisa tem como objetivo maior familiaridade com o problema, pode-se dizer que o objetivo principal é o aprimoramento de ideias ou a descobertas de intuições. Partindo de uma demanda, temos uma necessidade de solução de um problema real, em que as pessoas que querem o atendimento passam a utilizar o computador para que possam ser atendidas e os servidores do setor conseguem ter um melhor controle dos dados do atendimento. Antes de partir para a fase de desenvolvimento, foram realizadas entrevistas para o levantamento de funcionalidades que são necessárias para atingir o objetivo desde projeto. Para Sommerville (2003), processo genérico de levantamento e análise contém as seguintes atividades:

- Compreensão do domínio, que determina qual o problema a ser resolvido;
- A classificação, as atividades são separadas pelo nível de complexidade de execução;
- Resolução de conflitos, se muitos stakeholders estão envolvidos, os requisitos apresentarão conflitos;
- Definição das prioridades, definir quais requisitos são prioridades sobre outros;
- Verificação de requisitos, para descobrir se estão completos e consistentes;

Após o levantamento é hora de empregar as tecnologias necessárias para o desenvolvimento do sistema, para que seja o mais fiel possível do levantamento feito na etapa anterior.

3.1 Metodologia Ágil

Segundo Filho (2008, p22), durante a evolução dos processos de Engenharia de Software, a indústria se baseou nos métodos tradicionais de desenvolvimento de software, que definiram por muitos anos os padrões para criação de software nos meios acadêmico e empresarial. Porém, percebendo que a indústria apresentava um grande número de casos de fracasso, alguns líderes experientes adotaram modos de trabalho que se opunham aos principais conceitos das metodologias tradicionais.

De acordo com Beck (2001), no ano de 2001 um encontro entre 17 líderes que operavam no contra fluxo dos padrões da indústria de software, foi discutido formas de trabalho, objetivando chegar a uma nova metodologia de fabricação de software, que pudesse ser usada de forma

padronizada por todos eles e em outras empresas, substituindo os modelos tradicionais de desenvolvimento. O fim desse encontro ficou conhecido como o manifesto ágil.

O Manifesto Ágil, segundo Filho (2008), ressalta o que mais tem valor para as metodologias ágeis, a importância de como saber lidar com pessoas, assim como ter o cliente colaborando para encontrar a melhor solução, entregar o software com qualidade e do que se adaptar às mudanças.

3.1.1 Scrum

O Scrum não define uma técnica específica para o desenvolvimento de software durante a etapa de implementação, é concentrado em descrever como os membros da equipe devem trabalhar para produzir um sistema, que pode ser em um ambiente de mudanças constantes ou não.

A metodologia proposta pelo modelo Scrum aplica um sistema de entregas contínuas. O Scrum é uma metodologia destinada a pequenas equipes com menos de dez pessoas. Schwaber e Beedle (2001) sugerem que a equipe seja composta de cinco a nove integrantes, se mais pessoas estiverem envolvidas no projeto, devem-se formar múltiplas Equipes Scrum. Para este projeto utiliza-se a presente metodologia, adequando-a ao cenário, estipulando prazos para entrega dos sprints, chegando ao final dos sprints realizando uma nova reunião e estipulando um novo prazo.

3.2 Tecnologias

Nesta seção é apresentada todas as tecnologias utilizadas durante a fase de desenvolvimento, como linguagem de programação, banco de dados, metodologias de desenvolvimento e etc.

3.2.1 Banco de dados

Antes do desenvolvimento do banco de dados, foi necessário a definição do Diagrama de Entidade e Relacionamento, para melhor abstração do futuro desenvolvimento do banco de dados, é imprescindível que essa etapa seja feita de forma correta, fornecendo informações sobre os aspectos relacionados ao domínio do projeto em questão. Em geral, este modelo representa de forma abstrata a estrutura que possuirá o banco de dados da aplicação.

O modelo representa as entidades do banco de dados, os retângulos representam as entidades, que tentam se aproximar de um objeto que existe no mundo real com uma identificação distinta, os círculos representam seus atributos. Os losangos representam seus relacionamentos, que podem ser uma associação representado por um losango dentro do retângulo, que também possui seus atributos.

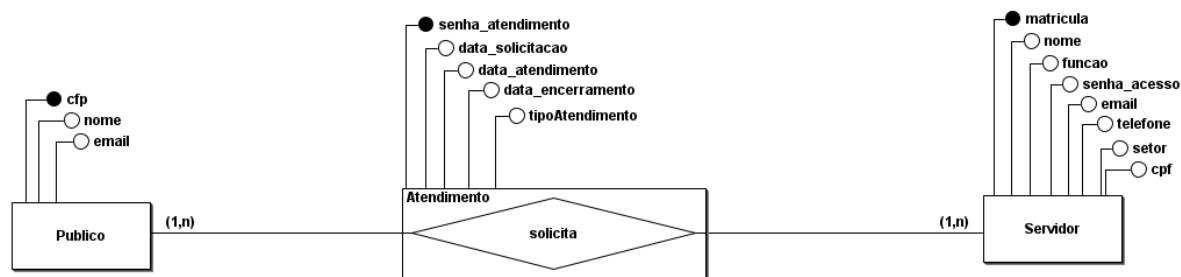


Figura 3.1 Diagrama de Entidade e Relacionamento do sistema

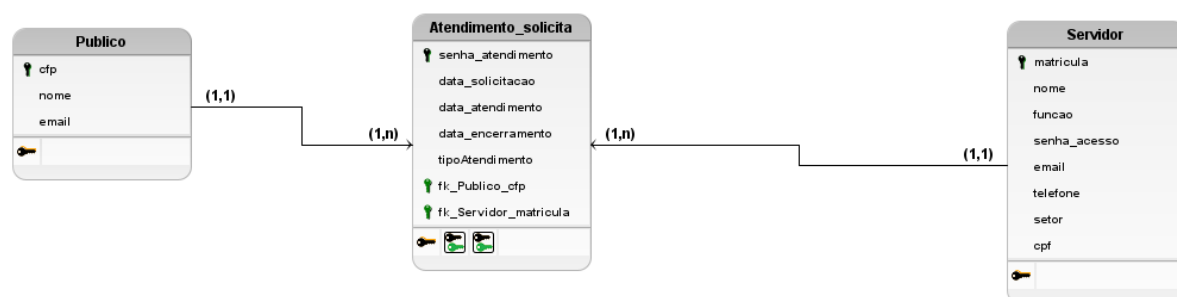


Figura 3.2 Modelo lógico relacional do banco de dados do sistema

Esta é uma representação mais técnica das entidades que o banco de dados deve possuir, especificando cada atributo da entidade. Neste modelo também é apresentado a estrutura de chaves primária e estrangeira, sendo a chave primária, representada pela cor preta, um atributo único e identificador da linha da tabela e a chave estrangeira, representada pela cor verde.

3.2.2 UML

A UML (Linguagem Unificada de Modelagem) é uma linguagem gráfica de modelagem para visualização, especificação, construção e documentação para desenvolver sistemas orientados a objeto, a modelagem é representada por diagramas. É utilizada para uma melhor compreensão do sistema que será desenvolvido, especificando como é o seu funcionamento através das variáveis que interagem com o sistema (VERGÍLIO, 2011).

De acordo com BEZERRA (2006), um caso de uso é a especificação de uma sequência de interações entre um sistema e os agentes externos que utilizam este sistema. A descrição de um caso de uso não se preocupa em definir o funcionamento interno de uma funcionalidade.

A UML possui um total de treze diagramas. Eles são divididos em dois grupos: Diagramas Estruturais e Diagramas Comportamentais, sendo que os comportamentais possuem uma subdivisão chamada de Diagramas de Interação [Martinez, 2015].

3.2.2.1 Diagrama de casos de uso

Segundo BOOCH (2000), um diagrama de caso de uso mostra um conjunto de casos de uso e atores (um tipo especial de classe) e seus relacionamentos. Os diagramas de casos de uso têm um papel fundamental para a modelagem do comportamento de um sistema. Objetivo é a compreensão do comportamento externo do sistema por qualquer stakeholder, o sistema é apresentado através da perspectiva do usuário.



Figura 3.3 Diagrama de casos de uso do sistema

3.2.2.2 Diagrama de sequência

Um diagrama de sequência é um diagrama de interação que dá ênfase à ordenação temporal de mensagens. Um diagrama de sequência mostra conjunto de objetos e as mensagens enviadas e recebidas por esses objetos. Tipicamente os objetos são instâncias nomeadas ou anônimas de classes, mas também podem representar instâncias de outros itens, como colaborações, componentes e nós. (BOOCH; JACOBSON; RUMBAUGH, 2000, p 96).

De acordo com Larman (2012), quando a finalidade é determinar funcionalidades de um sistema, utiliza-se diagramas de sequência com relação à notação da especificação técnica, para qual assim se crie o entendimento de como uma funcionalidade desempenha seu papel no sistema.

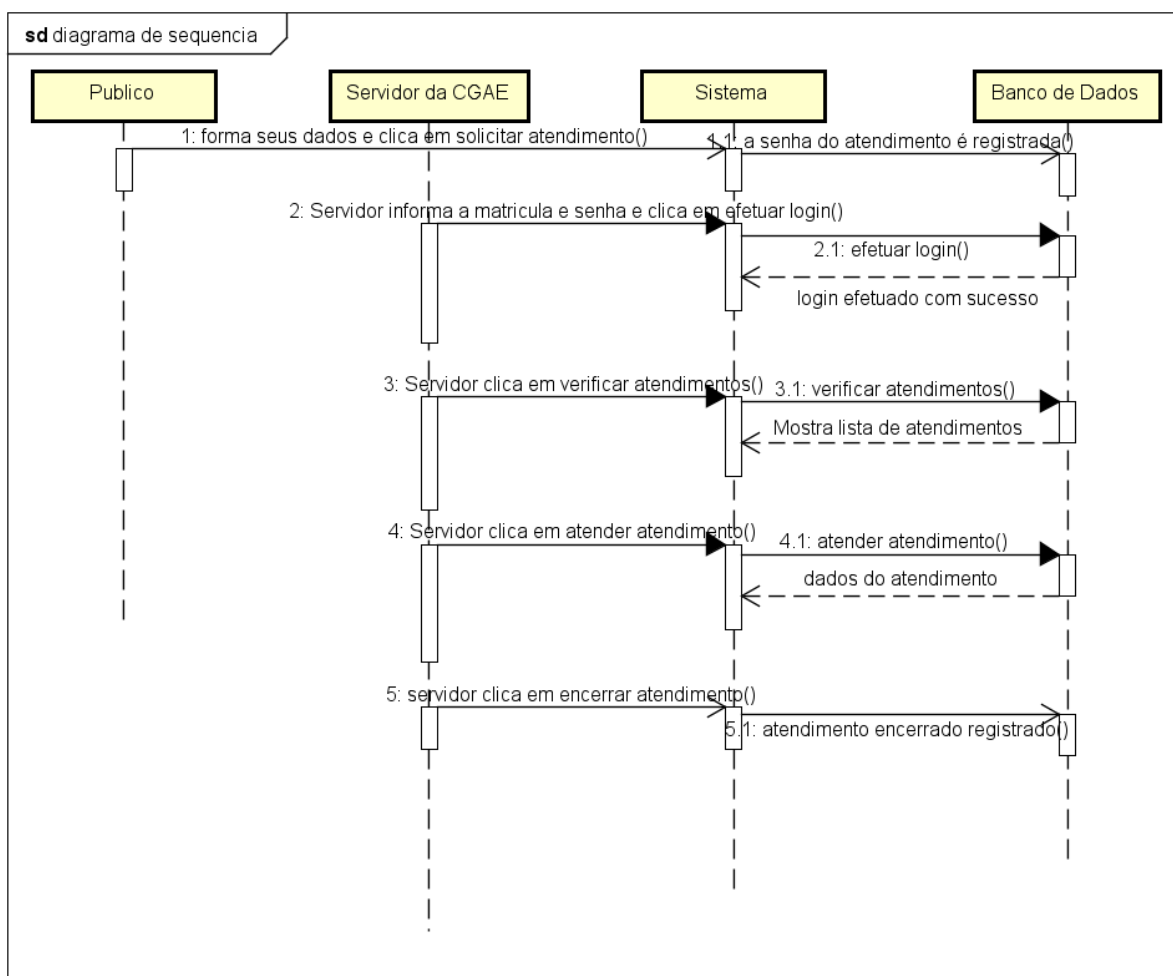


Figura 3.4 Diagrama de sequência do sistema

Como podemos ver na figura 3.4 temos um diagrama de sequência especificando uma ordem de tarefas as serem executadas durante o funcionamento do sistema.

3.2.2.3 Diagrama de atividades

O diagrama de atividades, como definido por seus criadores Booch, Rumbaugh e Jacobson (2006), envolve a modelagem das etapas sequenciais em um processo empresarial dando ênfase ao fluxo de controle de uma atividade para outra. Larman (2012) afirma que com a análise do sistema terminada, a modelagem é caracterizada com maior ênfase no projeto/conceito do sistema, busca-se um refinamento destas representações, a nível dos objetos que farão parte do sistema.

O diagrama de atividades ilustra graficamente como será o funcionamento do software, como será a atuação do sistema na realidade de negócio na qual ele está inserido, objetivo principal a especificação do comportamento do software e como qualquer outro modelo de notação UML, tenta mostrar o funcionamento posteriormente projetado, facilitando o entendimento do que tem que ser feito pelo programador. Na figura 3.5 é exibido o diagrama de atividades

do sistema, os retângulos representam atividades, losangos apresentam um fluxo de opções dependendo da escolha o fluxo pode mudar ou até mesmo voltar para atividade anterior.

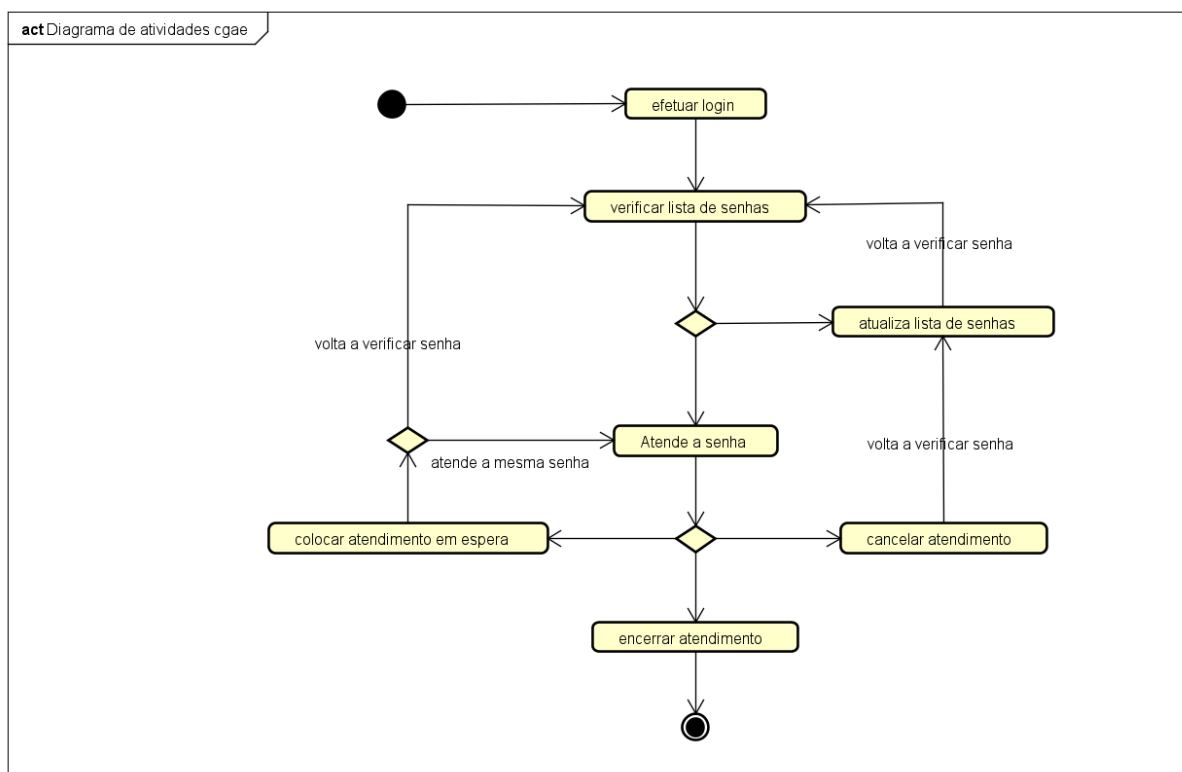


Figura 3.5 Diagrama de atividades do sistema

3.2.2.4 Diagrama de classes

O diagrama de classes apresenta uma visão estática de como as classes estão organizadas. A classe segundo a OMG (2015), é um elemento abstrato que representa um conjunto de objetos com seus atributos e métodos. Tratando dos relacionamentos, no diagrama de classes eles descrevem como as classes interagem umas com as outras e podem também definir responsabilidades (OMG, 2015). Os tipos de relacionamentos possíveis são: dependência, associação, agregação, composição e herança.

Como é possível ver na Figura 3.6, antes do nome dos atributos e métodos existe uma notação, estas notações representam a visibilidade daquele elemento. As visibilidades podem ser as seguintes:

- + pública: outras classes podem ter acesso ao elemento;
- - privada: acessado somente direto pela própria classe;

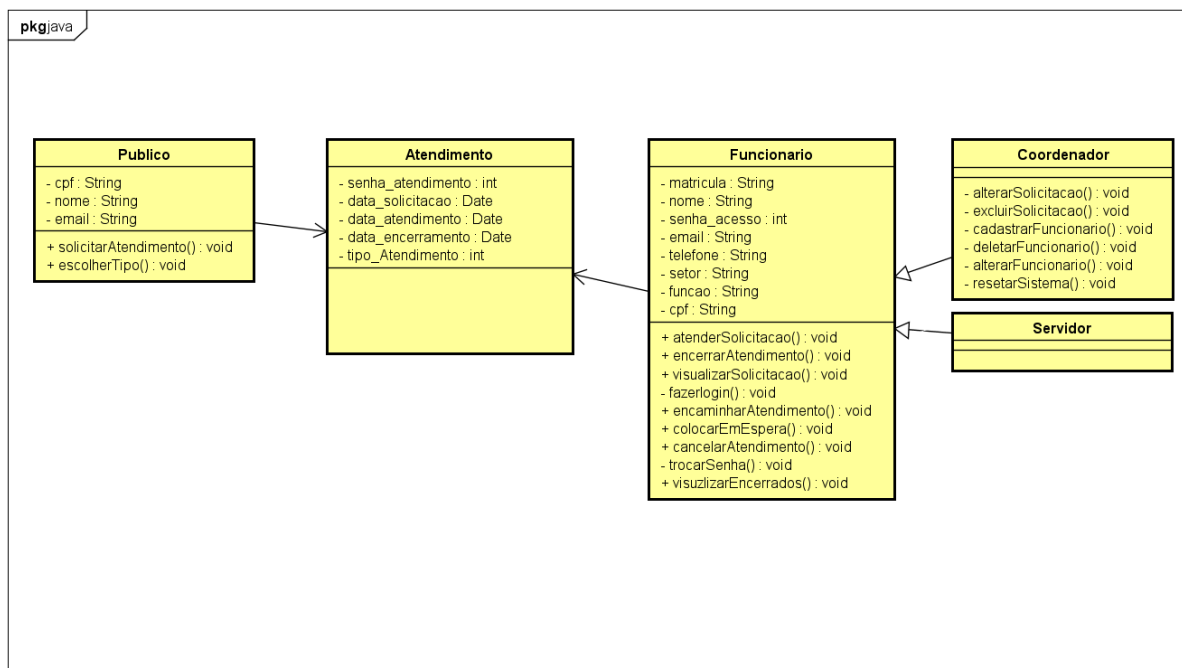


Figura 3.6 Diagrama de classes do sistema

3.2.3 Padrão MVC

Nos primeiros anos da Computação, o software era projetado sob medida para cada aplicação e desenvolvido pelo próprio programador ou organização. O programador que desenvolvia era o mesmo que colocava em funcionamento e consertava os erros quando ocorriam (PRESSMAN, 1995).

Um dos principais objetivos do padrão MVC é auxiliar na organização do código de uma aplicação em camadas, Relações do diagrama de classes. Segundo Macoratti, a fundamentação das divisões das funcionalidades de um sistema em camadas surgiu como alternativa para solucionar alguns problemas existentes nas aplicações monolíticas³, que eram aplicações de difícil manutenção. A necessidade de compartilhar a lógica de acesso aos dados entre vários usuários, impulsionou para o desenvolvimento de aplicações em duas camadas. Para Macoratti, um problema dessa abordagem é o gerenciamento de versões, sempre que há alteração em alguma das regras, força uma atualização em todas as máquinas clientes.

Entretanto com a evolução da internet, que trouxe consigo novos usuários e desenvolvedores, surgiu um novo paradigma na qual a aplicação é disponibilizada por um servidor web onde apenas são realizadas requisições por parte do usuário, e essas requisições são processadas pelo servidor web e enviadas novamente para o usuário. Entre estas novas arquiteturas, têm-se a arquitetura em camadas e a arquitetura MVC (Model, View e Controller). É muito utilizado para casos em que podem existir múltiplas camadas de apresentação para diversos usuários. Na figura 3.7⁴ temos uma representação do padrão MVC.

³São desenvolvidas para serem instaladas em um só lugar.

⁴Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-laravel/>>Acesso em: 06 Fev. 2021.

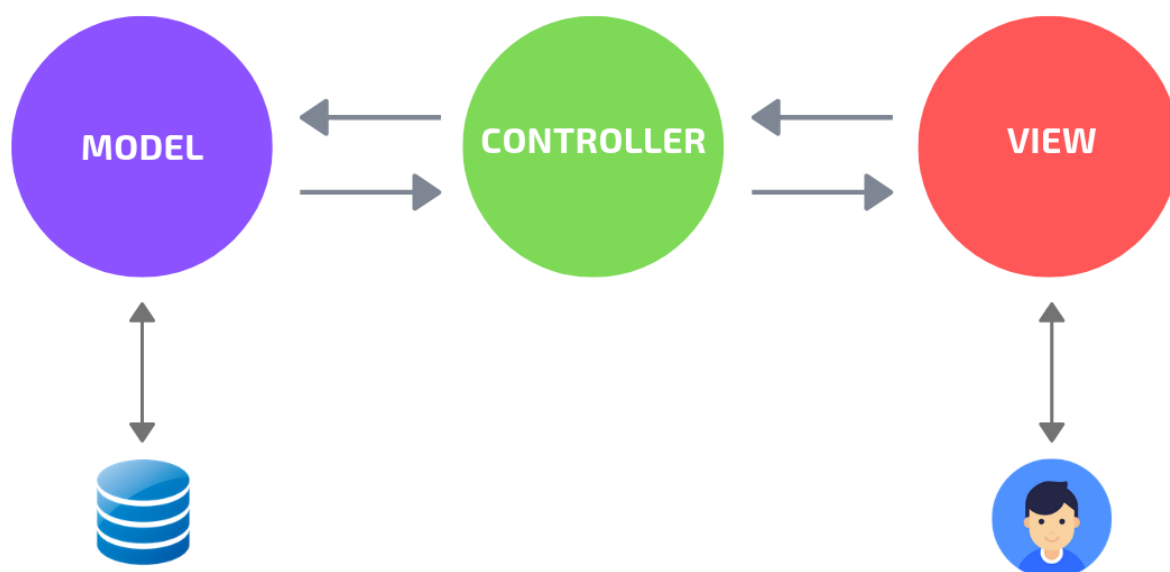


Figura 3.7 Representação do padrão MVC

3.2.4 Linguagem de programação

Linguagem de Programação é uma linguagem escrita e formal que especifica um conjunto de instruções e regras usadas para gerar programas. Um software pode ser desenvolvido para rodar em um computador, dispositivo móvel ou em qualquer equipamento que permita sua execução (Sebasta, 2015).

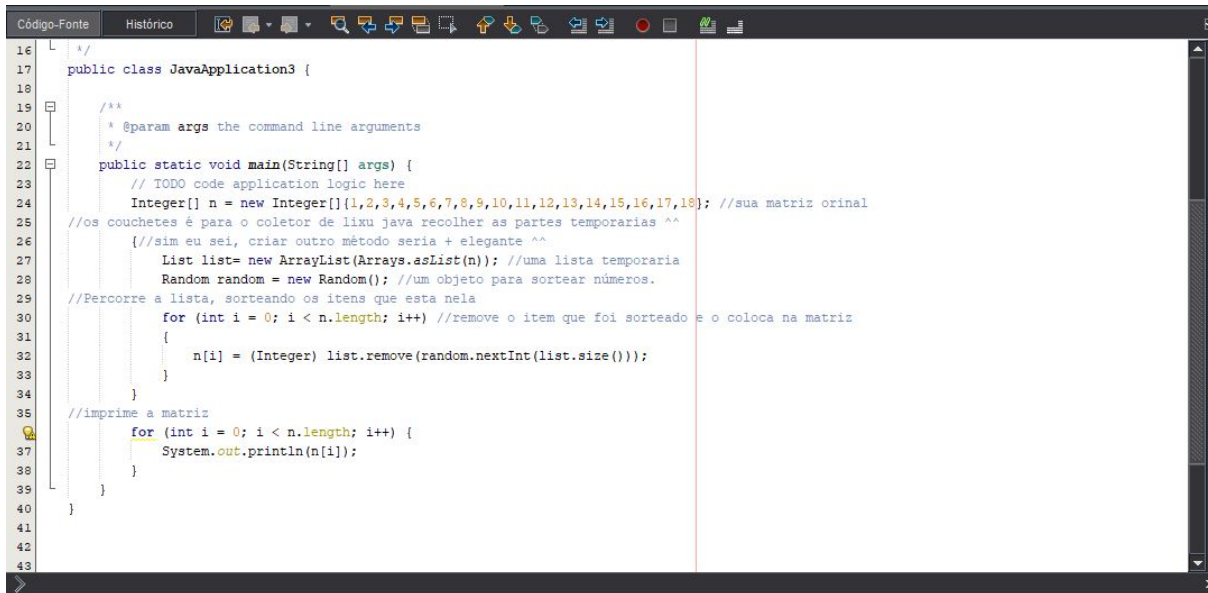
3.2.4.1 Java

O presente projeto foi desenvolvido na linguagem de programação Java, para Jandl Junior (2016), Java é uma plataforma de programação apresentada pelo mercado em 1995 pela Sun Microsystems que ainda provoca entusiasmos em programadores, analistas e projetista de software, pois é o resultado de um enorme trabalho de pesquisa científica e tecnológica.

Neste projeto foi utilizada a versão 8 do Java, esta versão foi liberada em 2014, ela trouxe o maior número de características na linguagem desde da versão 5, como expressões lambda, referencias para métodos, métodos default em interfaces, operações em massa nas coleções e uma nova API para data e hora (Jandl Junior, 2016). Características importantes pela qual a linguagem foi escolhida para o projeto:

- Orientada a objetos;
- Independência de plataforma;
- Sem ponteiros;
- Performance compacta e independente;
- Execução concorrente de múltiplas rotinas;
- Linguagem robusta;

- Fortemente tipada;
- Possui mecanismo de reflexão;
- Incentiva controle de erros.



```
16  */
17  public class JavaApplication3 {
18
19      /**
20       * @param args the command line arguments
21       */
22      public static void main(String[] args) {
23          // TODO code application logic here
24          Integer[] n = new Integer[]{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18}; //sua matriz orinal
25          //os couchetes é para o coletor de lixo java recolher as partes temporarias ^^
26          //sim eu sei, criar outro método seria + elegante ^^
27          List list= new ArrayList(Arrays.asList(n)); //uma lista temporaria
28          Random random = new Random(); //um objeto para sortear números.
29          //Percorre a lista, sorteando os itens que esta nela
30          for (int i = 0; i < n.length; i++) //remove o item que foi sorteado e o coloca na matriz
31          {
32              n[i] = (Integer) list.remove(random.nextInt(list.size()));
33          }
34          //imprime a matriz
35          for (int i = 0; i < n.length; i++) {
36              System.out.println(n[i]);
37          }
38      }
39  }
40
41
42
43
```

Figura 3.8 Código de um programa em Java

Na figura 3.8, temos um exemplo de um programa escrito na linguagem Java, a função do programa é sortear números de forma aleatória e retirar os que foram sorteados até que sobre apenas um, assim sorteando os números que estão de dentro de um array. Pode-se notar a criação de dois objetos, um do tipo Random e outro do tipo List, o Random server para sortear de forma aleatória os componentes dentro da lista, o List é a lista que contém todos os elementos, isso é feito através de um loop que começa em 0 e vai até o último elemento existente no array, após o número ser sorteado, o próprio é removido da lista.

3.2.4.2 JDBC

O JDBC (Java Database Connectivity), é uma API que contém elementos para que uma aplicação Java consiga acessar dados de um SGBD locais ou remotos. No JDBC é marcante o uso da SQL, a linguagem padrão do SGBD (Jandl Junior, 2016). A API permite o acesso genérico a qualquer SGBD, fica por conta de o programador construir o código para uma conexão e escolher a forma de interação. O uso do SQL é imprescindível. Para a conexão ser feita, foi utilizado um driver que é uma classe especial que deve implementar a interface Java.

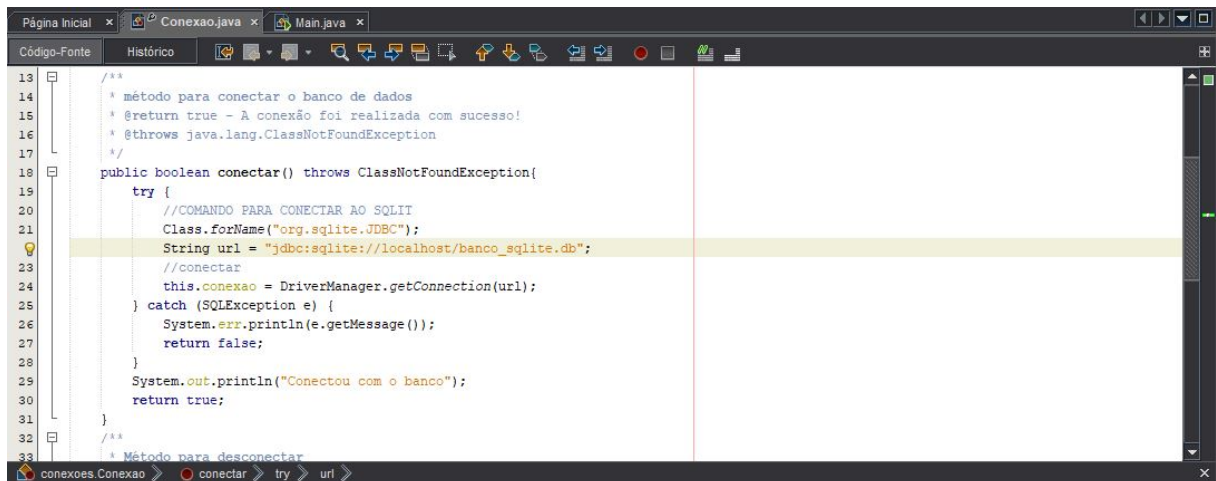


Figura 3.9 Código para conexão do JDBC

Basicamente o código tem que conter uma query em SQL concatenado com uma String na linguagem Java que será traduzida pela API e solicitada ao SGBD. Na figura 3.9 temos um exemplo de conexão usando o uma endereço local na máquina e um arquivo .db onde ficam armazenados os dados da aplicação, com essa conexão é possível realizar todos os CRUD's⁵ necessários para a aplicação.

3.2.4.3 JavaFX

Para a criação de telas e elementos front-end utilizou-se o JavaFX, permite expandir o poder do Java, permitindo que os desenvolvedores usem qualquer biblioteca Java em aplicações JavaFX. Dessa forma, os desenvolvedores podem expandir seus recursos no Java e aproveitar a tecnologia de apresentação que o JavaFX fornece para criar experiências visuais envolventes (Java, 2021).

JavaFX é uma tecnologia de software que, ao ser combinada com Java, permite a criação e implantação de aplicações de aparência moderna e conteúdo rico de áudio e vídeo. Não podendo ser usada sem o Java é uma poderosa ferramenta que possui elementos para interações com usuário.

Na figura 3.10 podemos ver exemplo de projeto criado a partir do código básico que gera uma tela de JavaFX.

⁵São as quatro funções básicas do armazenamento persistente .

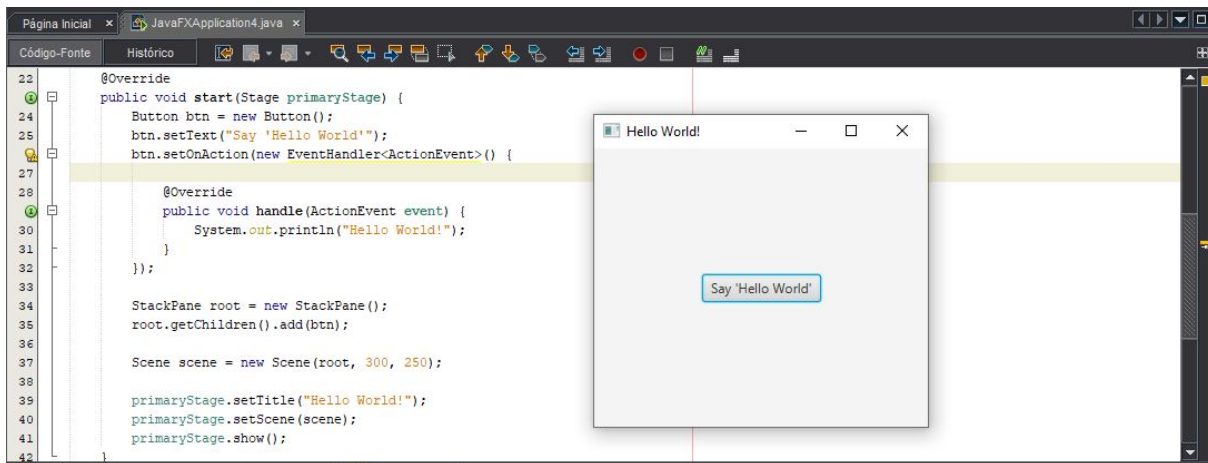


Figura 3.10 Código e tela criada em JavaFX

3.3 Ferramentas utilizadas

Nesta secção serão apresentadas as ferramentas que foram utilizadas para interagir com as tecnologias descritas na secção anterior.

3.3.1 NetBeans 8.2 RC

Para o desenvolvimento do código foi feito o uso de uma IDE⁶, o NetBeans⁷ na sua versão 8.2 é um ambiente de desenvolvimento para melhor criação e depuração do código escrito para desenvolver aplicações. É um programa repleto de recursos que pode ser usado por muitos aspectos do desenvolvimento de software.

O NetBeans é um dos ambientes de desenvolvimento mais utilizados pelos desenvolvedores Java. Mais do que um editor de código, ele possui um conjunto de ferramentas que auxiliam a programação de tarefas comuns relacionadas à implementação de aplicações.

3.3.2 DB Browser for SQLite

DB Browser⁸ for SQLite⁹ é uma ferramenta de código aberto, visual e de alta qualidade para criar, projetar e editar arquivos de banco de dados compatíveis com SQLite. O DB Browser foi utilizado para criar o banco de dados da aplicação, devida a sua interface de fácil uso e entendimento. Conta com uma interface familiar semelhante a uma planilha e possui uma funcionalidade para executar comando sql.

Na figura 3.11 é mostrada um banco de dados criado via DB Browser, também podemos ver que sua interface é amigável e possui diversas funcionalidades.

⁶ Ambiente de desenvolvimento integrado

⁷ Disponível em: <https://netbeans.apache.org/>> Acesso em: 15/02/2021

⁸ Disponível em: <https://sqlitebrowser.org/>> Acesso em: 15/02/2021

⁹ Disponível em: <https://www.sqlite.org/index.html>> Acesso em: 15/02/2021

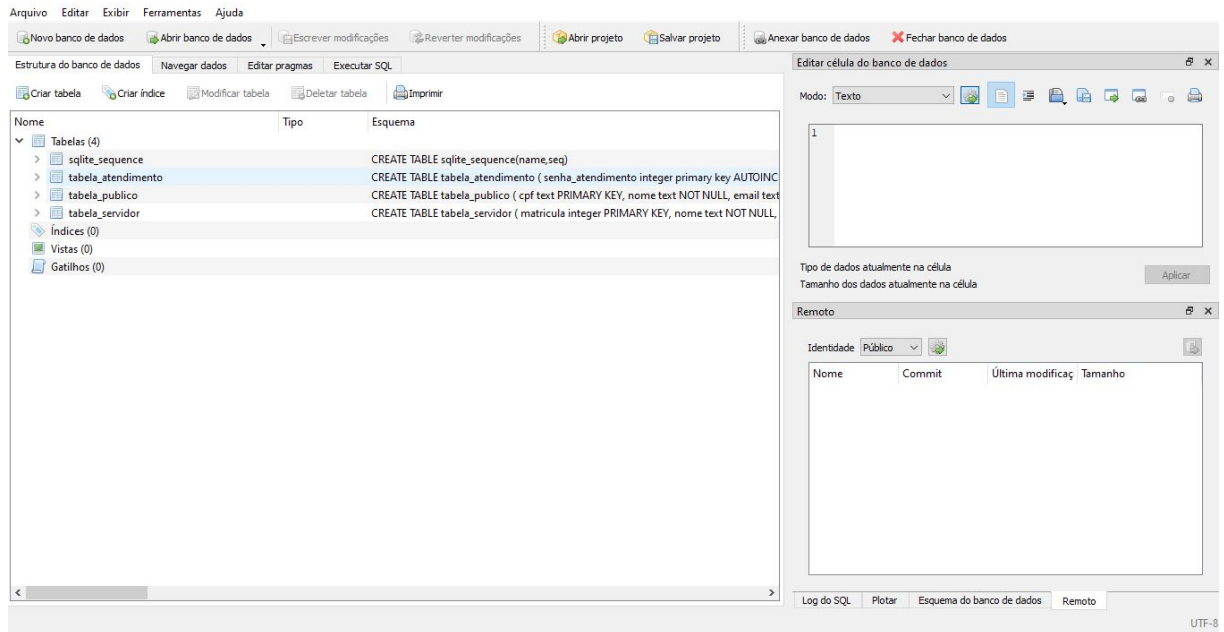


Figura 3.11 Tela do DB Browser

3.3.3 SQLite

SQLite é um mecanismo de banco de dados SQL embutido. Ao contrário da maioria dos outros bancos de dados SQL, o SQLite não tem um processo de servidor separado. O SQLite lê e grava diretamente em arquivos de disco comuns. Um banco de dados SQL completo com várias tabelas, índices, gatilhos e visualizações está contido em um único arquivo de disco (SQLite, 2021). Alguns dos motivos para o uso deste Sistema Gerenciador de Banco de Dados(SGBD):

- Funciona bem para o tamanho de fluxo de dados da aplicação;
- Leve e de fácil uso;
- Fácil manutenção;
- Nenhuma configuração ou administração necessária.

3.3.4 br Modelo 3.3

O BR Modelo¹⁰ é uma ferramenta de código aberto e totalmente gratuita voltada para ensino de modelagem de banco de dados relacionais com base na metodologia defendida por Carlos A. Heuser no livro “Projeto de Banco de Dados”. A ferramenta foi concebida pelo autor como trabalho de conclusão do curso de especialização em banco de dados pelas universidades UFSC (SC) e UNIVAG (MT), orientado pelo Professor Dr. Ronaldo dos Santos Mello, após se constatar a inexistência de uma ferramenta nacional que pudesse ser utilizada para essa finalidade (sis4, 2021).

Na figura 3.12 temos uma apresentação da tela inicial do programa, é exibido um exemplo básico de um modelo qualquer.

¹⁰Disponível em: <http://www.sis4.com/brModelo/>> Acesso em: 28/02/2021

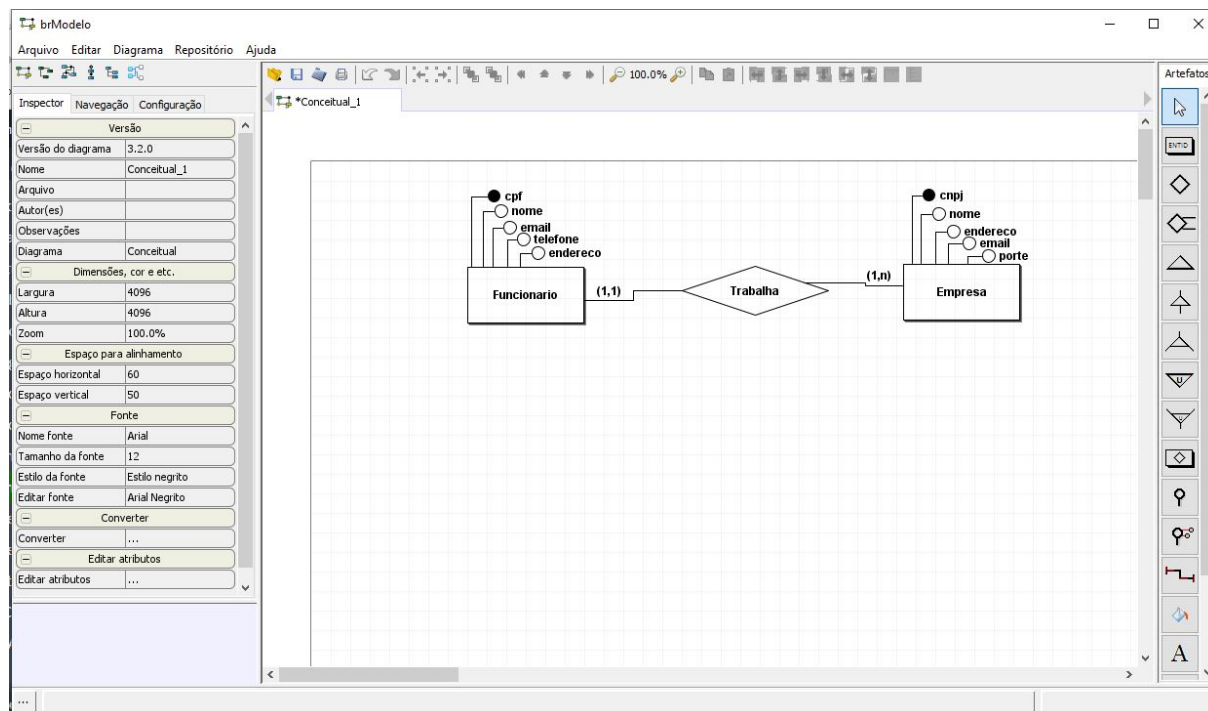


Figura 3.12 Tela do brModelo

O programa foi utilizado para desenvolver o modelo relacional do banco de dados do sistema, possui uma interface amigável e interativa. Na figura 3.1 e 3.2 do tópico 3.2.1 podemos ver o diagrama de entidade e relacionamento e o modelo lógico relacional feito pelo programa.

3.3.5 Astah UML

Quando a modelagem UML direciona seu processo de desenvolvimento, você precisa de uma ferramenta que seja projetada especificamente para UML e fornecerá todos os recursos de que você precisa - sem ser muito complicada. Isso é Astah UML. Uma ferramenta simples de aprender e usar, Astah UML¹¹ permitirá que você crie os diagramas UML de que precisa (Astah, 2021). O software permite fazer diversos diagramas pedidos na UML, segue abaixo a lista de diagramas que podem ser feitos pelo software:

- Diagrama de classes;
- Diagrama de sequência;
- Diagrama de componentes;
- Diagrama de atividades;
- Diagrama de casos de uso;
- Diagrama de implantação;
- Mapas mentais.

Na figura 3.13 temos a tela inicial do software, podemos ver um exemplo de diagrama de casos de uso de um sistema qualquer.

¹¹Disponível em: <https://astah.net/products/astah-uml/>. Acesso em: 26/02/2021

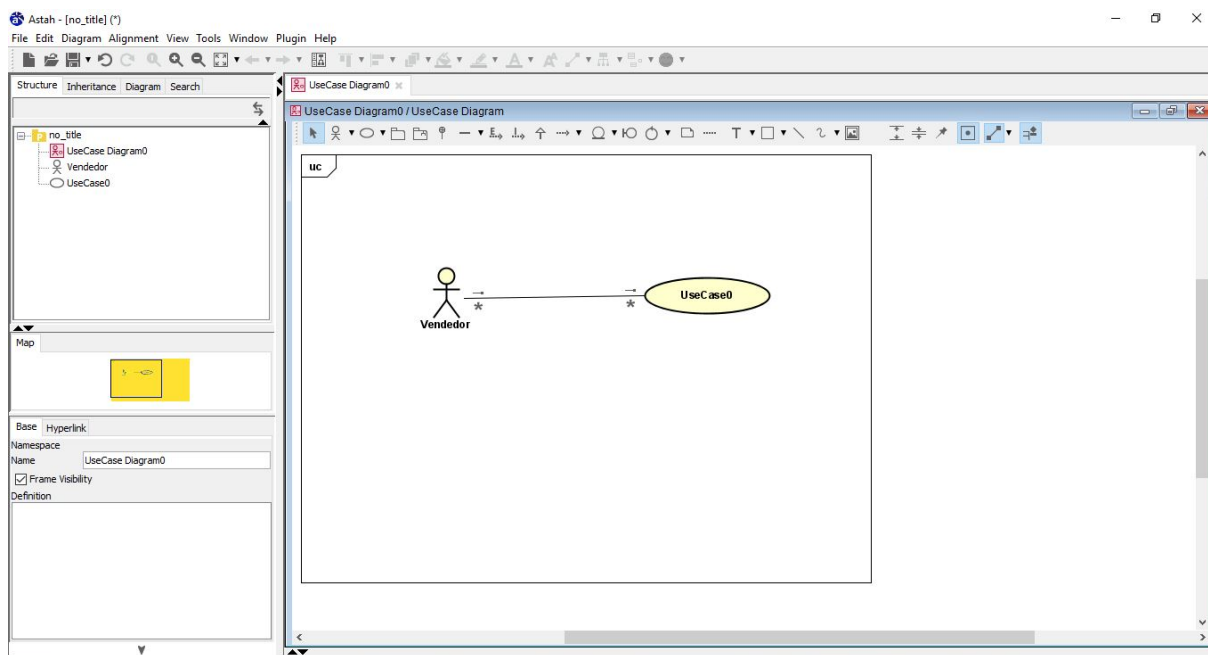


Figura 3.13 Tela do Astah UML

O software foi utilizado para criar os diagramas das figuras 3.3, 3.4, 3.5 e 3.6 do tópico 3.2.2, são mostrados nas figuras respectivamente o diagrama de casos de uso e o diagrama de classes do sistema.

3.3.6 JavaFx Scene Builder

Para melhor exibição do front-end foi utilizada uma ferramenta chamada JavaFX Scene Builder¹², segundo a Oracle (2021), é uma ferramenta de layout visual que permite aos usuários criar interfaces de usuário de aplicativos JavaFX rapidamente, sem codificação. Os usuários podem arrastar e soltar componentes de UI para uma área de trabalho, modificar suas propriedades, aplicar folhas de estilo e o código FXML para o layout que estão criando é gerado automaticamente em segundo plano. O resultado é um arquivo FXML que pode então ser combinado com um projeto Java ligando a IU à lógica do aplicativo.

Após a criação da tela no Scene Builder é gerado um FXML, uma linguagem de marcação baseada em XML que permite aos usuários definir a interface de usuário de um aplicativo, separadamente da lógica do aplicativo.

¹²Disponível em: <https://www.oracle.com/java/technologies/javase/javafxscenebuilder-info.html>. Acesso em: 28/02/2021

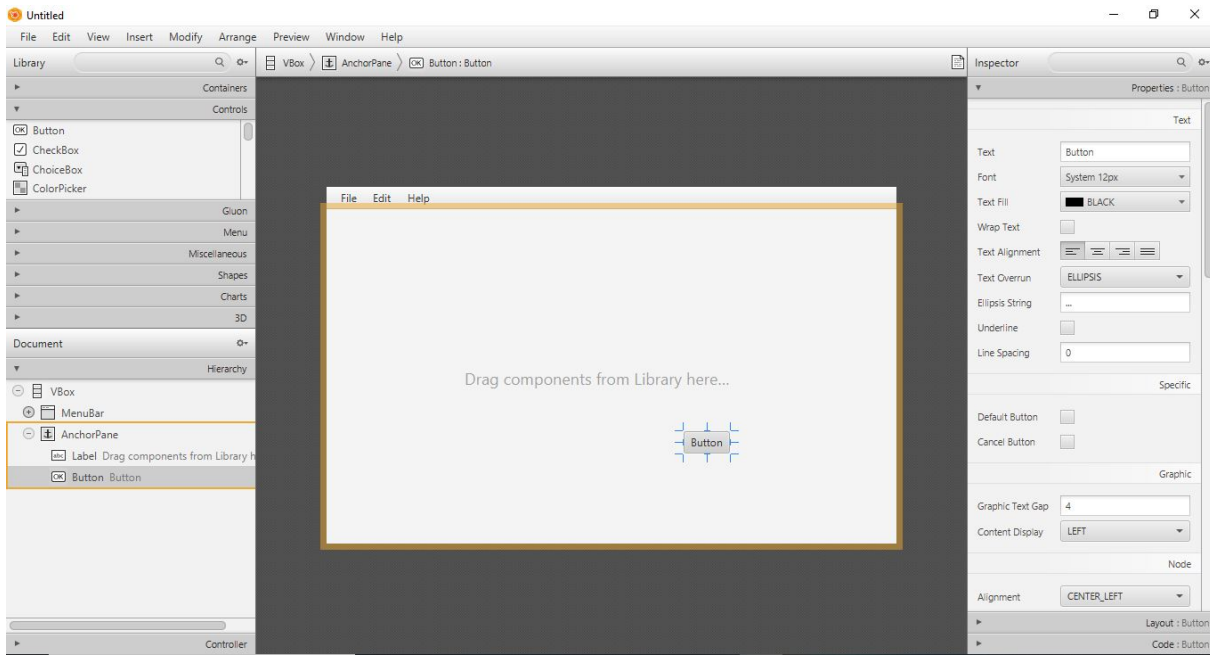


Figura 3.14 Tela do JavaFX Scene Builder

Na figura 3.14 é apresentada a tela do Scene Builder, com uma tela básica sendo criada.

4

Apresentação e Análise dos Resultados

Neste capítulo será abordado o processo de desenvolvimento do sistema descrito ao longo do trabalho e a apresentação dos resultados, cujo o objetivo é desenvolver um sistema desktop de atendimento ao público da Coordenação Geral de Assistência Estudantil do IFB campus Brasília, onde uma pessoa que queria atendimento, deverá se identificar para poder solicitar uma senha e registrar no sistema, assim podendo ser atendido por um dos servidores do setor.

4.1 Desenvolvimento

Quanto ao processo de desenvolvimento utilizou-se as etapas da engenharia de software atendendo ao modelo clássico (PRESSMAN, 1995) e ágil. A seguir encontraremos o detalhamento de cada etapa do processo de desenvolvimento.

Inicialmente foi realizada a etapa de levantamento de requisitos, foram feitas reuniões com os demandantes, onde o problema foi relatado, que se trata da ausência do controle do fluxo de atendimento e dos dados de pessoas atendidas pelo setor durante o decorrer do expediente. Após essas reuniões os seguintes requisitos foram propostos:

- Cadastro de servidores: O sistema teria uma autenticação em que os servidores do setor teriam seu cadastro para poderem acessar o sistema e poder atender ao público interno e externo do IFB.
- Solicitação de senha: O público interno e externo do IFB, deveria antes de ser atendido, solicitar a senha de atendimento pelo sistema, enviando alguns dados para o sistema.
- Registro da solicitação: O sistema registra a data e hora da solicitação feita pelo público.
- Registro do atendimento: O sistema registra o atendimento feito pelo servidor, registra a data e hora do início e do fim do atendimento.

Após a fase de levantamento dos requisitos do sistema, levando em consideração as necessidades estipuladas pelo demandante, chegamos as seguintes especificações:

- O sistema será uma aplicação desktop com rede instalada localmente, seu funcionamento não dependerá de internet, as solicitações só poderão ser feitas indo ao setor;
- Será necessário um computador na entrada do setor para que possam ser feitas as solicitações por parte do público;
- O banco de dados será instalado em uma máquina local no setor;

De acordo com os requisitos apresentados, foi realizado o desenvolvimento do sistema, utilizando as técnicas de engenharia de software (PRESSMAN, 1995). Também foi empregada a metodologia ágil Scrum. Primeiramente foi desenvolvido o banco de dados da aplicação, deixando o código o mais legível possível para uma futura mudança durante do desenvolvimento das outras partes, posteriormente a programação e o design do sistema que foram desenvolvidos em conjunto. Depois de cada etapa revisava-se a possibilidade de novas funcionalidades para o sistema.

Na figura 4.1, temos a apresentação do banco de dados criado para o funcionamento do sistema.

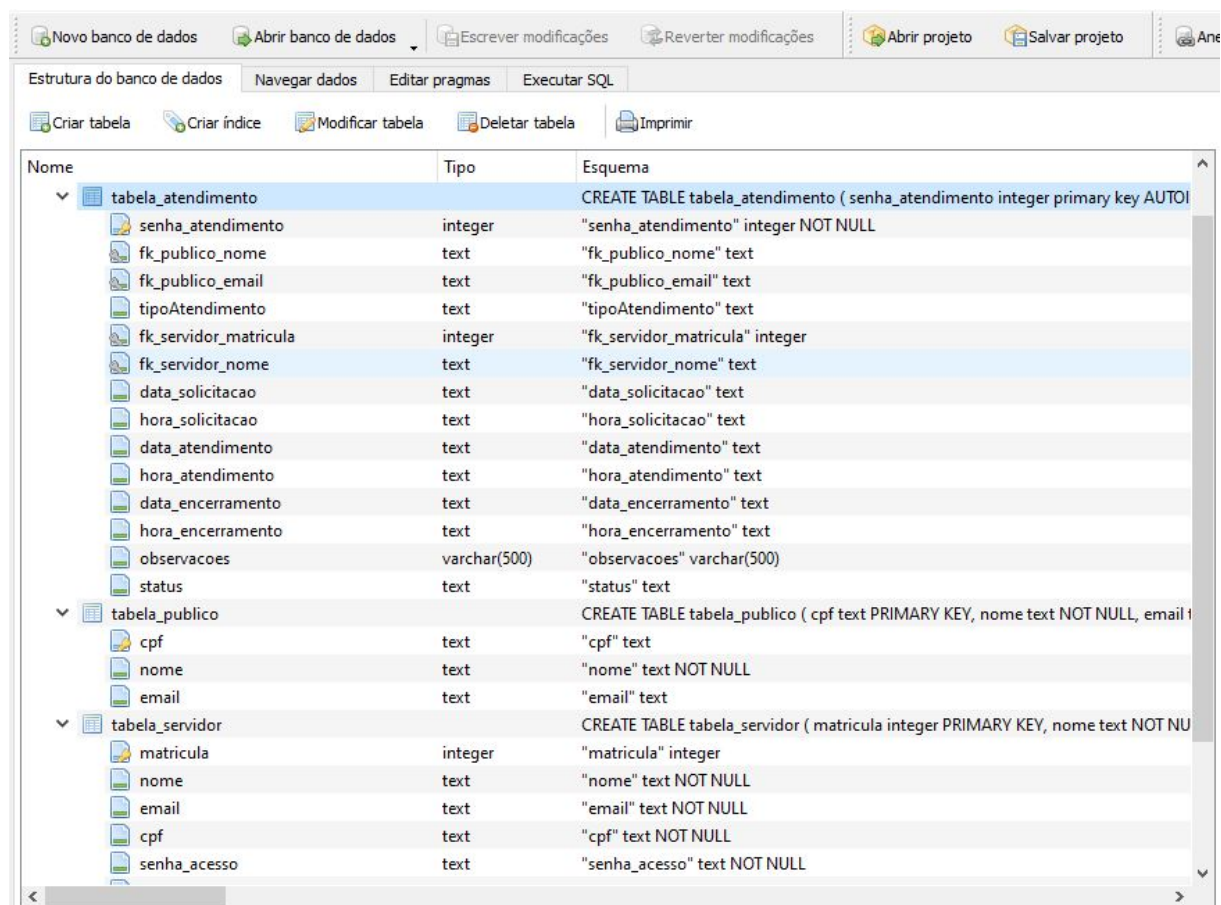


Figura 4.1 Banco de dados criado para o sistema

Projeto é um empreendimento não repetitivo, caracterizado por uma sequência clara e lógica de eventos, com início, meio e fim, que se destina a atingir um objetivo claro e definido, sendo conduzido por pessoas dentro de parâmetros predefinidos de tempo, custo, recursos envolvidos e qualidade. (VARGAS, 2005, P.7)

Para facilitar futuras manutenções que forem necessárias para o sistema, a organização do código seguiu o padrão MVC. Abaixo podemos observar a figura 4.2 com a estrutura do código feito que foi desenvolvido na linguagem de programação Java:

- Pasta conexões – onde se encontra as classes necessárias para a comunicação com o

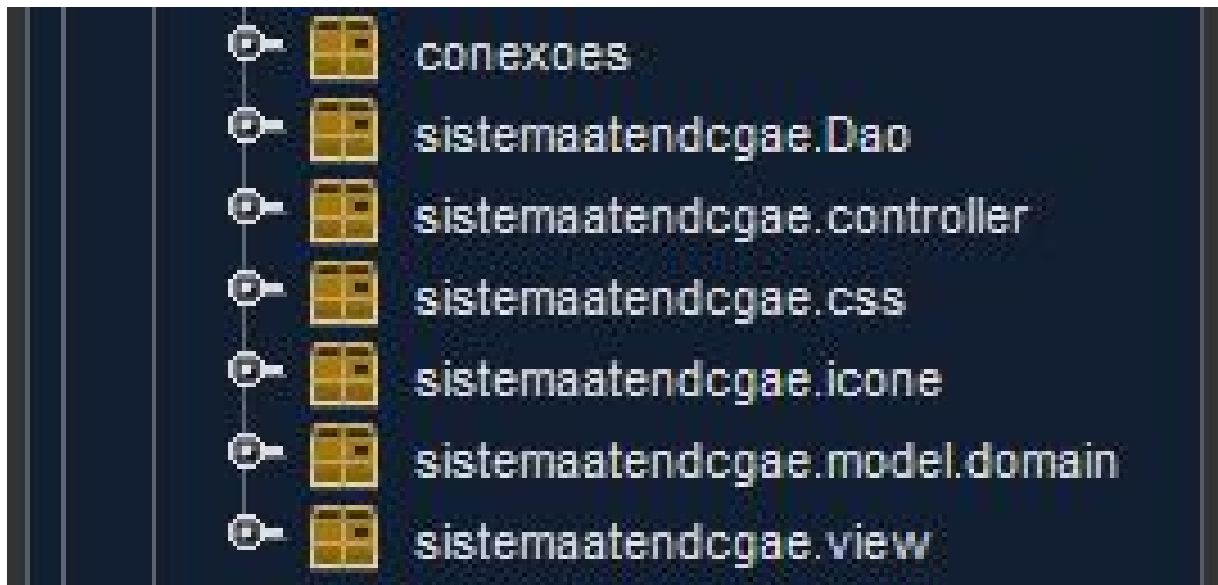


Figura 4.2 Estrutura do código com o padrão MVC

banco de dados;

- Pasta `sistemaatendcgae.Dao` – onde se encontram as classes que fazem a comunicação entre o banco de dados e o controller;
- Pasta `sistemaatendcgae.controller` – onde se encontram as classes de controller que contém o código necessário para a view funcionar de forma correta, além disso torna possível a troca de dados com as classes Dao e as classes Model;
- Pasta `sistemaatendcgae.css` – Pasta que contém as folhas de estilos necessárias para estilizar a view;
- Pasta `sistemaatendcgae.icone` – Pasta onde contém arquivos de imagens e ícones que são usados pela view;
- Pasta `sistemaatendcgae.model.domain` – Pasta onde se encontram as classes Model, que são necessárias para comunicação com as classes Dao;
- Pasta `sistemaatendcgae` – Onde se encontram as páginas do sistema.

Para este módulo uma página foi desenvolvida, contendo o formulário de solicitação, que deverá ser completamente preenchido e com escolha do tipo de atendimento, na presente página ilustrada na figura 4.3 o solicitante deverá colocar alguns dados para poder solicitar uma senha para posteriormente ser atendido pelo setor. O solicitante não precisa fazer login no sistema para ser atendido, apenas se registrar, após o registro deverá aguardar a sua vez. Após a confirmação da solicitação, a mesma passará a existir no banco de dados do sistema, podendo assim ser vista pelo servidor que estiver logado no sistema.

The image shows a web interface for the 'Coordenação Geral de Assistência estudantil'. On the left is a sidebar with the logo of the Instituto Federal de Brasília and a menu with options: 'Home', 'Para o público' (highlighted in red), 'Para o servidor', and 'Sair'. The main content area has a green header with the title 'Coordenação Geral de Assistência estudantil'. Below the header is a light gray box titled 'Solicite aqui seu atendimento'. This box contains a form with the following fields: 'Matrícula/Cpf:', 'Nome:', and 'E-mail:', each followed by a text input field. Below these is a section titled 'Selecione seu atendimento' with a dropdown menu currently showing 'Selecione o tipo'. At the bottom of the form are two buttons: 'Confirmar' and 'Limpar'.

Figura 4.3 Página de solicitação de senhas

Para este módulo foi desenvolvido um conjunto de páginas com diversas funcionalidades, que serão usadas pelos funcionários do setor, seguindo o levantamento de requisitos, existem dois tipos de funcionários, o tipo Admin que tem privilégios para mudar configurações do sistema e o segundo tipo servidor, que não tem privilégios, apenas executa as funções básicas do sistema.

A figura 4.4 a seguir mostra a tela do funcionário do setor, para acessar o sistema. O funcionário deverá efetuar o login para poder ter acesso as demais funcionalidades do sistema.

The image shows a web application interface for the 'Coordenação Geral de Assistência estudantil'. On the left is a sidebar with the logo of the 'INSTITUTO FEDERAL Brasília' and a menu with four items: 'Home', 'Para o público', 'Para o servidor' (highlighted in red), and 'Sair'. The main content area has a green header with the title 'Coordenação Geral de Assistência estudantil'. Below the header is a login form with a green background. The form contains two input fields: 'Matrícula' (with a person icon) and 'Senha' (with a lock icon). To the right of the 'Senha' field is a 'Visualizar senha' button. Below the fields are two buttons: 'Log in' and 'Esqueceu a senha?'.

Figura 4.4 Página de login

A figura 4.5 mostra a lista de atendimentos que estão na fila, apenas os funcionários que estão logados conseguem ver esta lista, nessa tela tem a opção de atender ou atualizar a lista, para poder excluir, a conta logada tem quer ser um administrador do sistema.



Figura 4.5 Lista de atendimentos na fila

Na figura 4.6 podemos notar a tela de um funcionário que é administrador do sistema, podendo visualizar a lista de servidores e a lista de todos os atendimentos na fila, em andamento, em espera ou encerrados, assim como poderá fazer todas as funções de CRUD. Além das funcionalidades anteriormente descritas, também poderá restaurar o sistema.



Figura 4.6 Tela de configurações do sistema

Na figura 4.7 ilustra a tela de cadastro de um funcionário, clicando no botão adicionar da figura 4.6 é encaminhada para esta tela, que só pode ser acessada por um administrado do sistema.

Cadastro de Servidores do CGAE

Voltar

Matricula _____ Nome _____

Cpf _____ E-mail _____

Telefone _____ Setor _____

Senha _____ Confirmar senha _____

Selecione a função ▼

Salvar **Limpar**

Figura 4.7 Tela de cadastro de servidores

4.2 Fase de testes

A fase de teste durou cerca de duas semanas após a conclusão da fase de desenvolvimento do sistema. Foram necessárias trocas de comunicações com a parte demandante. Devido a pandemia em que o mundo se encontra, encontrou-se dificuldades para concluir essa fase, porém conseguiu-se obter o feedback do demandante.

O sistema teve uma boa aceitação por parte dos funcionários do setor, tendo em vista que o sistema tem a finalidade de ser desenvolvido sobre medida para a CGAE, todas as partes do sistema, funcionalidades, design e registros externos foram devidamente aceitos. Devido a pandemia ainda não foi possível instalar o sistema no setor do IFB, os testes foram possíveis apenas localmente em uma máquina com o software instalado, não foi possível testar a dinâmica de um fluxo de atendimento diário e real que existe no setor.

5

Conclusões e Trabalhos Futuros

Ao final deste trabalho, conclui-se que o sistema foi devidamente desenvolvido, documentado e testado, utilizando a metodologia de projetos Pressman (1995), a metodologia ágil Scrum e o padrão MVC para organização do código. Como dito no capítulo anterior a instalação ainda não foi concluída.

O presente trabalho conclui que metodologias, tecnologia e ferramentas aplicadas em conjunto, faz com que o projeto melhore sua qualidade, tendo em vista que todas os métodos utilizados facilitam a comunicação com o demandante, assim se aproximando do sistema ideal.

Abordando um pouco sobre a experiência de desenvolver o sistema, notou-se a dificuldade de desenvolver algo do zero, porém trouxe a proximidade de desenvolver algo na prática e que pode servir para o mercado de trabalho, trabalhando não só a habilidade de desenvolver código, mas também a comunicação com cliente. Diversas pesquisas foram feitas para entregar um sistema que fosse o mais próximo possível do relato inicial.

Em relação a trabalho futuros, poderão ser implementadas novas funcionalidades, armazenar o sistema em algum servidor online, para que possam ser feitas agendamento de atendimento, dessa forma facilitando o atendimento, tentando minimizar o fluxo de atendimento.

Referências

PRIGOL, Cristian Paulo. SISTEMA DE HELP DESK E CONTROLE DE CHAMADOS BASE-ADO EM WORKFLOW. 2007. Trabalho de Conclusão de Curso (Bacharelado - Sistemas de Informação. UNIVERSIDADE REGIONAL DE BLUMENAU, Santa Catarina, 2007.[Orientado: Prof. Marcel Hugo]

FREITAS, Marcos André dos Santos. Fundamentos do gerenciamento de serviços TI: paratário para a certificação ITIL V3 Foundation. Rio de Janeiro: Brasport, 2010. 376 p.

MV Sistemas. MV - Líder em Software de Gestão de Saúde.Disponível em: <<http://www.mv.com.br/pt/>>Acesso em: 05.nov.2019.

LOPES, Marco Aurélio D.; BEZERRA, Marlene. Gestão de Processos: fatores que influenciam o sucesso na sua implantação. 2008.

GONÇALVES, José Ernesto Lima. As empresas são grandes coleções de processo. 2000.

KOTLER, P. Administração de marketing: Análise, planejamento, implementação e controle. 5ªed. São Paulo: Atlas, 1998.

GIL, Antonio Carlos. Como elaborar projetos de pesquisas, edição nº4. São Paulo: Editora Atlas S.A..2003.

SOMMERVILLE, I. Engenharia de software. 6º ed. Tradução Maurício de Andrade. São Paulo: Ed Addison-Wesley, 2003

VERGILIO, Silva (2011). Introdução a UML. Disponível em:<<http://www.inf.ufpr.br/silvia/ESNovo/UML/pdf/IntroduzUMLAl.pdf>>. Acesso em: 02/02/2021.

MARTINEZ, Marina (2015) UML. Disponível em: <<http://www.infoescola.com/engenhariade-software/uml/>>. Acesso em: 02/02/2021.

BOOCH, G.; JACOBSON, I.; RUMBAUGH, J. UML: Guia do usuário. Rio de Janeiro: Campus, 2000.

OMG, O. M. G. Unified Modeling Language (UML). 2015. <http://www.omg.org/spec/UML/2.5/Beta2/>>. Acesso em: 22/05/2015.

MACORATTI, J. Carlos. Artigos de Tecnologia da Informação. <http://www.macoratti.net>. Acesso em: 06/03/2021.

PRESSMAN, Roger. S. Engenharia de software. 3ª edição São Paulo: Makron Books, 1995.

SEBASTA, Robert W. Concepts of Programming Languages(em inglês), 11ª ed. Pearson, 2015.

JANDL JUNIOR, Peter. Java: guia do programador: atualizado para Java 8. 3. ed., rev. e ampl., 1. reimpr. São Paulo: Novatec, 2016.

SQLITE. SQLite. Disponível em: <https://www.sqlite.org/index.html>>. Acesso em: 22/02/2021.

SIS4. sis4: brModelo 3. Disponível em: <http://www.sis4.com/brModelo/>>. Acesso em: 22/06/2021.

ASTAH. Astah UML. Disponível em: <https://astah.net/products/astah-uml/>>. Acesso em: 26/02/2021.

JAVA. Java: JavaFX. Disponível em: <https://www.java.com/pt-BR/download/help/javafx.html>> Acesso em: 20/02/2021.

ORACLE. Oracle: JavaFX Scene Builder. Disponível em: <https://www.oracle.com/java/technologies/javase/javafxscenebuilder-info.html>>. Acesso em: 28/02/2021.

BEZERRA, E. Princípios de Análise e Projeto de Sistemas Uml: Um guia Prático para Modelagem de Sistemas. Rio de Janeiro: Campus, 2006.

LARMAN, Craig. Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Iterative Development. Pearson Education India, 2012.

LEAN INSTITUTE BRASIL. Disponível vem <<http://www.lean.org.br>>. Acesso em 05/01/2021.

VARGAS, R.V. Gerenciamento de projetos: estabelecendo diferenciais competitivos. 6. ed. Rio de Janeiro: Brasport, 2005.

FILHO, D. L. B.: Experiências com desenvolvimento ágil. Instituto de Matemática e Esta-

tística da Universidade de São Paulo (Dissertação de Mestrado). 2008.

BECK, K., et al.: Manifesto for Agile Software Development. 2001. Disponível em: <<http://www.agilemanifesto.org>>
Acesso em: 12/02/2021.

SCHWABER, K.; BEEDLE, M. Agile Software Development With Scrum. Primeira Edição.
Upper Saddle River: Prentice-Hall. 2001.

Apêndice

A

Mapping Study's Instruments

Tabela A.1 List of conferences on which the searches were performed.

Acronym	Conference
APSEC	Asia Pacific Software Engineering Conference
ASE	IEEE/ACM International Conference on Automated Software Engineering
CSMR	European Conference on Software Maintenance and Reengineering
ESEC	European Software Engineering Conference
ESEM	International Symposium on Empirical Software Management and Measurement
ICSE	International Conference on Software Engineering
ICSM	International Conference on Software Maintenance
ICST	International Conference on Software Testing
InfoVis	IEEE Information Visualization Conference
KDD	ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
MSR	Working Conference on Mining Software Repositories
OOPSLA	Object-Oriented Programming, Systems, Languages and Applications
QSIC	International Conference On Quality Software
SAC	ACM Symposium on Applied Computing
SEAA	EUROMICRO Conference on Software Engineering and Advanced Applications
SEDE	19th International Conference on Software Engineering and Data Engineering
SEKE	International Conference on Software Engineering and Knowledge Engineering

Tabela A.2 List of journals in which the searches were performed.

Journal title
ACM Transactions on Software Engineering and Methodology
Automated Software Engineering
Elsevier Information and Software Technology
Elsevier Journal of Systems and Software
Empirical Software Engineering
IEEE Software
IEEE Computer
IEEE Transactions on Software Engineering
International Journal of Software Engineering and Knowledge Engineering
Journal of Software: Evolution and Process
Software Quality Journal
Journal of Software
Software Practice and Experience Journal

Tabela A.3 Search string per Search Engine.

Search Engine	Search String
Google Scholar	bug report OR track OR triage “change request” issue track OR request OR software OR “modification request” OR “defect track” OR “software issue” repositories maintenance evolution
ACM Portal	Abstract: "bug report"or Abstract:"change request"or Abstract:"bug track"or Abstract:"issue track"or Abstract:"defect track"or Abstract:"bug triage"or Abstract: "software issue"or Abstract: "issue request"or Abstract: "modification request") and (Abstract:software or Abstract:maintenance or Abstract:repositories or Abstract:repository
IEEEExplorer (1)	((((((((((Abstract": "bug report") OR "Abstract": "change request") OR "Abstract": "bug track") OR "Abstract": "software issue") OR "Abstract": "issue request") OR "Abstract": "modification request") OR "Abstract": "issue track") OR "Abstract": "defect track") OR "Abstract": "bug triage") AND "Abstract": software)
IEEEExplorer (2)	((((((((((Abstract": "bug report") OR "Abstract": "change request") OR "Abstract": "bug track") OR "Abstract": "software issue") OR "Abstract": "issue request") OR "Abstract": "modification request") OR "Abstract": "issue track") OR "Abstract": "defect track") OR "Abstract": "bug triage") AND "Abstract": maintenance)
IEEEExplorer (3)	((((((((((Abstract": "bug report") OR "Abstract": "change request") OR "Abstract": "bug track") OR "Abstract": "software issue") OR "Abstract": "issue request") OR "Abstract": "modification request") OR "Abstract": "issue track") OR "Abstract": "defect track") OR "Abstract": "bug triage") AND "Abstract": repositories)
IEEEExplorer	((((((((((Abstract": "bug report") OR "Abstract": "change request") OR "Abstract": "bug track") OR "Abstract": "software issue") OR "Abstract": "issue request") OR "Abstract": "modification request") OR "Abstract": "issue track") OR "Abstract": "defect track") OR "Abstract": "bug triage") AND "Abstract": repository)
Citeseer Library	(abstract: "bug report"OR abstract:"change request"OR abstract:"bug track"OR abstract:"issue track"OR abstract:"defect track"OR abstract:"bug triage"OR abstract: "software issue"OR abstract: "issue request"OR abstract: "modification request") AND (abstract:software OR abstract:maintenance OR abstract:repositories OR abstract:repository)
Elsevier	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "software issue"OR "issue request"OR "modification request") AND (software OR maintenance OR repositories OR repository)
Scirus	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "software issue"OR "issue request"OR "modification request") AND (software maintenance OR repositories OR repository) ANDNOT (medical OR aerospace)
ScienceDirect	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "issue request"OR "modification request") AND LIMIT-TO(topics, "soft ware")
Scopus	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "software issue"OR "issue request"OR "modification request") AND (software maintenance OR repositories OR repository)
Wiley	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "software issue"OR "issue request"OR "modification request") AND (software maintenance OR repositories OR repository)
ISI Web of Knowledge	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "software issue"OR "issue request"OR "modification request") AND (software maintenance OR repositories OR repository) ANDNOT (medical OR aerospace)
SpringerLink	("bug report"OR "change request"OR "bug track"OR "issue track"OR "defect track"OR "bug triage"OR "software issue"OR "issue request"OR "modification request") AND (software maintenance OR repositories OR repository) ANDNOT (medical OR aerospace)