



Faculdade de Informática e Administração Paulista

Domain Driven Design

Challenge – Salesforce

Grupo Tokito Techs

RM553472 – Claudio Silva Bispo

RM552981 – Patricia Naomi Yamagishi

RM554021 – Sara Ingrid da Silva Pereira

Sumário

1.	Introdução.....	3
2.	Definição do Projeto	3
3.	Objetivo.....	4
4.	Entidades	5
5.	Diagrama Entidade Relacionamento (DER).....	7
6.	Modelo Lógico Relacional.....	7
7.	Sobre o Projeto	8
8.	Diagrama de Classes	9
9.	Classe Model	11
10.	Classe View	12
11.	Classe Controller	13
12.	Protótipos	14
13.	Utilizando a Aplicação	15

1. Introdução

Em um mundo impulsionado pela tecnologia, nosso projeto nasce com o propósito de transformar a interação digital em uma experiência inclusiva e igualitária. Estamos comprometidos em redefinir como as informações no site da Salesforce são acessadas, assegurando que todas as pessoas, com ou sem habilidades técnicas, naveguem com facilidade e sem complicações.

Identificamos que o atual acesso ao site apresenta desafios, e é nossa meta criar um portal que organiza e simplifica o fluxo de informações, tornando cada interação uma oportunidade de conexão e não um empecilho. Os formulários, longe de serem barreiras, serão vias de comunicação eficiente e intuitiva, facilitando cada passo do usuário no site da Salesforce.

2. Definição do Projeto

Em nosso projeto, a digitalização se encontra com a acessibilidade e a simplicidade para fomentar a inclusão. Adotamos um método de autenticação descomplicado, utilizando plataformas populares como Gmail, LinkedIn e Facebook, buscando remover barreiras técnicas e simplificar a experiência do usuário. O apoio fornecido por chatbots interativos é uma ferramenta fundamental para esclarecer dúvidas iniciais, proporcionando uma assistência eficiente e acessível.

Recursos como a busca por voz, opções de idioma variadas e personalização de temas são projetados para que todos tenham acesso, sem exceção. Formulários adaptáveis aceitam tanto a digitação quanto comandos de voz, refletindo o nosso compromisso com uma interatividade sem barreiras. Além

disso, o acompanhamento da jornada dos usuários através do site nos permite coletar informações valiosas sobre suas interações, preferências e tempo de engajamento. Esses dados são essenciais para o aprimoramento contínuo da interface, visando uma experiência cada vez mais personalizada, intuitiva e satisfatória. A cada passo, nosso projeto não se contenta em apenas construir um site - buscamos desenvolver um espaço digital inclusivo, onde cada interação é uma etapa para uma experiência digital enriquecedora.

3. Objetivo

Nesta Sprint, daremos um grande passo no desenvolvimento do nosso sistema, com o início da integração do backend utilizando a linguagem Java. As operações críticas de gerenciamento de dados e as medidas de segurança serão construídas com Java, garantindo uma fundação sólida para o nosso site. Nosso foco atual para esta entrega se direciona para duas estruturas essenciais: a entidade Visitante e a entidade Questionário. Na entidade Visitante, vamos construir uma lógica detalhada para a captação e processamento de informações fundamentais, desde o instante da chegada do visitante ao site até o momento de sua partida.

Paralelamente, a entidade Questionário será desenvolvida para capturar dados cruciais que ajudarão a alinhar as necessidades dos usuários às soluções adequadas, de forma precisa e customizada. Implementar essas classes nos permitirá monitorar e interpretar o comportamento e as preferências dos nossos usuários, abrindo portas para a criação de novas oportunidades de mercado e para o refinamento contínuo dos nossos serviços.

4. Entidades

Plataforma Login: Essa entidade será essencial para rastrear e analisar as atividades de login em várias plataformas. Ao armazenar dados como ID, nome e status de atividade das plataformas, teremos um registro preciso e atualizado da integridade desses sistemas. A criação dessa opção será para simplificar o cadastro e direcionando a todos a aplicarem antes de ver qualquer conteúdo.

Visitante: Ao implementar a entidade Visitante, poderemos centralizar e administrar de forma eficiente as informações pertinentes, como nome, origem, interesses e outras características relevantes. Essa abordagem facilitará a personalização das interações, fornecendo uma experiência mais envolvente e direcionada para cada visitante. Além disso, possibilitará a análise de tendências de comportamento e preferências, e mapear a experiência de cada visitante no site.

Leads: A entidade Leads será o ponto central da estratégia de geração de leads. Aqui, poderemos armazenar informações cruciais sobre os visitantes do site, como nome, e-mail, plataforma de login utilizada e outras variáveis relevantes para entender e atender às necessidades desses potenciais clientes.

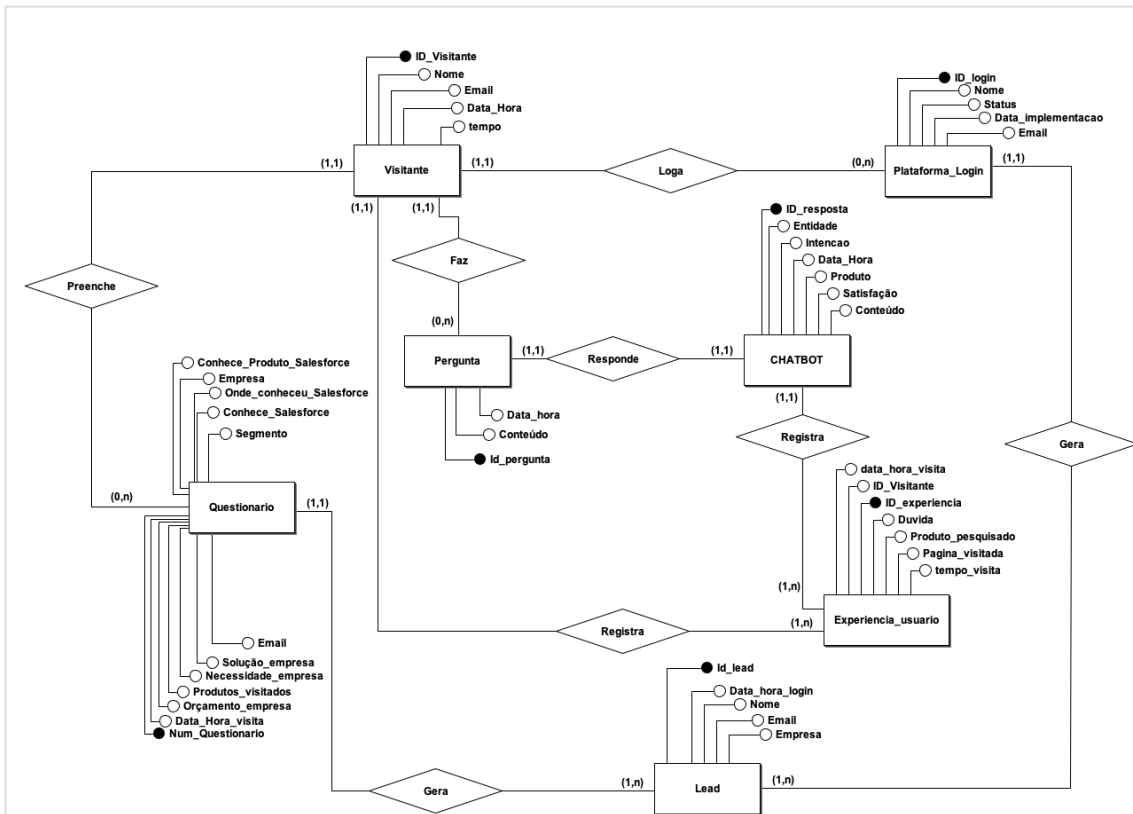
Chatbot: O Chatbot possui a habilidade de integrar e consultar bancos de dados, o que simplifica e torna mais dinâmico o processo de esclarecer dúvidas sobre os produtos, por exemplo. Para isso, utilizaremos a funcionalidade de integração com serviços externos no Watson Assistant. Essa funcionalidade permite a conexão do chat a serviços web, incluindo bancos de dados, para recuperar e apresentar informações aos usuários com base em suas consultas. Essa funcionalidade proporcionará facilidade de acesso para todos os usuários, incluindo aqueles com deficiência visual ou auditiva, pois poderemos oferecer opções de busca por texto ou até mesmo a opção de gravação de áudio.

Pergunta: A entidade pergunta irá armazenar as perguntas feitas pelo visitante ao Chatbot. Servirá para geração da experiência do usuário e para direcionar melhor a Salesforce quanto ao material disponibilizado no site para consulta do usuário, gerando insights para torná-lo mais assertivo e eficaz.

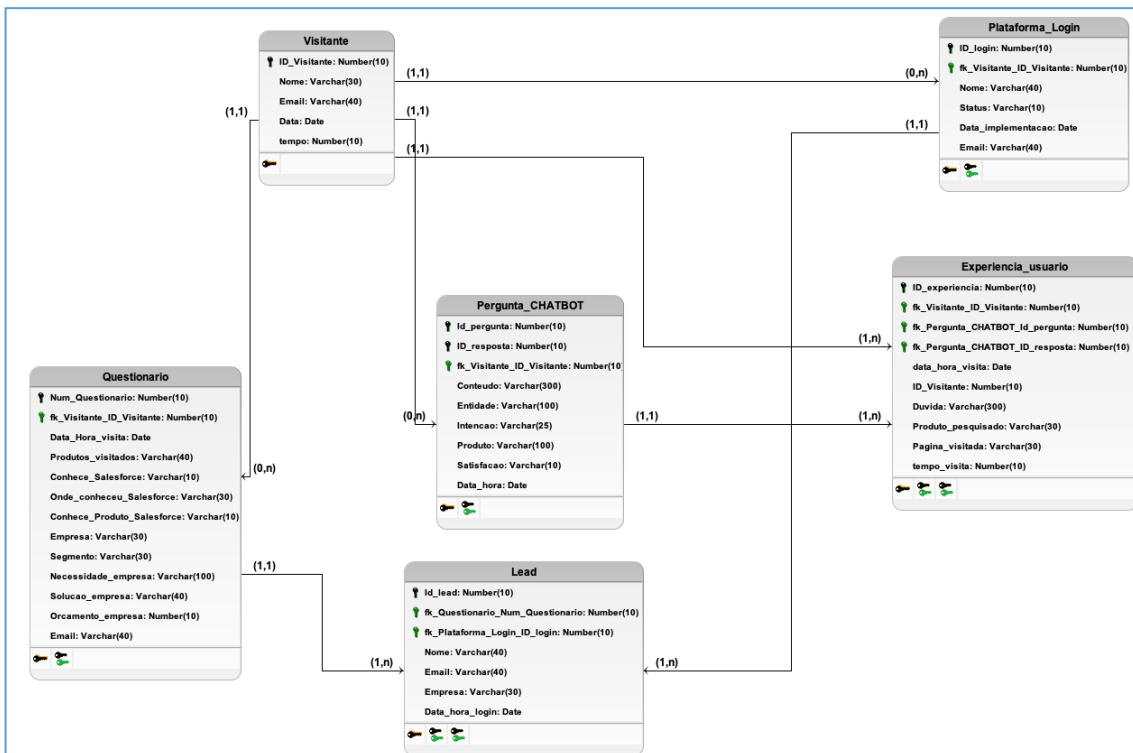
Questionário: A entidade Questionário armazenará as informações do visitante que tenha interesse em entender qual ou quais produtos da Salesforce melhor se encaixam para resolver a dor que a sua empresa possui naquele momento. Esses dados serão armazenados de maneira eficaz. Assim, a Salesforce poderá realizar uma análise mais precisa com as informações fornecidas e ter um overview do problema que um possível cliente deseja solucionar, facilitando a tomada de decisões e a personalização das interações com os clientes. Vai nos dar base para entender o que o usuário deseja já no seu primeiro contato.

Experiência do Usuário: Ao manter no banco de dados uma entidade que registra a experiência dos usuários em nosso sistema, seremos capazes de compreender melhor suas interações, preferências e desafios. Isso nos permitirá aprimorar continuamente a interface e os processos, proporcionando uma experiência mais intuitiva e satisfatória. Além disso, teremos dados valiosos para orientar a criação de conteúdos e funcionalidades que atendam às necessidades e expectativas dos usuários, promovendo assim um ambiente mais engajador e eficiente. Além de gerar leads valiosos e podendo desenvolver argumentos fortes para as vendas.

5. Diagrama Entidade Relacionamento (DER)



6. Modelo Lógico Relacional



7. Sobre o Projeto

Para a entrega desse projeto, as classes Visitante e Questionário estão no coração do sistema, construídas para otimizar a experiência do usuário e coletar dados com precisão. A classe Visitante é responsável por capturar e organizar dados essenciais, como nome, e-mail e tempo de visita, fornecendo uma base sólida para a construção de relações duradouras com os clientes e visitantes do site. Através dela, é possível acompanhar a jornada do usuário, entender seus interesses e preferências, facilitando a segmentação e a personalização da comunicação.

Já a classe Questionário, é uma ferramenta poderosa para a compreensão profunda das necessidades e expectativas dos usuários. As respostas coletadas são essenciais para direcionar as estratégias de produto e marketing, permitindo que a empresa responda de forma mais efetiva aos desafios do mercado e às dores dos clientes.

Este projeto foi desenvolvido seguindo o padrão de arquitetura de software MVC (Model-View-Controller). Esse padrão estrutural permite uma clara separação de preocupações, dividindo o projeto em três camadas interconectadas: o Modelo, que gerencia os dados e a lógica de negócios; a Visão, que define a representação visual e a interação do usuário; e o Controlador, que atua como um intermediário entre o Modelo e a Visão, controlando o fluxo de dados e as respostas às interações do usuário. A adoção desse padrão possibilitou uma organização mais eficiente do código, facilitando tanto a manutenção quanto a expansão futura do projeto, além de promover uma maior reusabilidade dos componentes desenvolvidos.

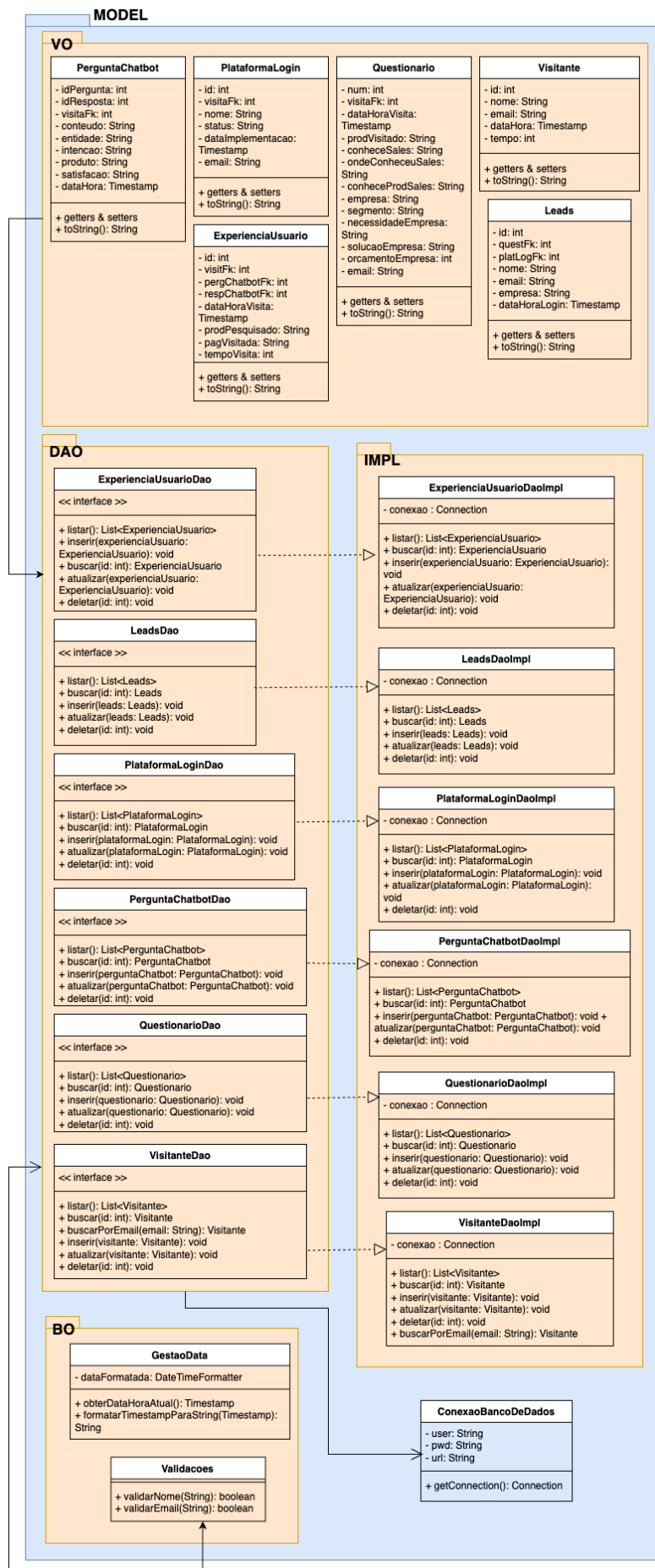
8. Diagrama de Classes

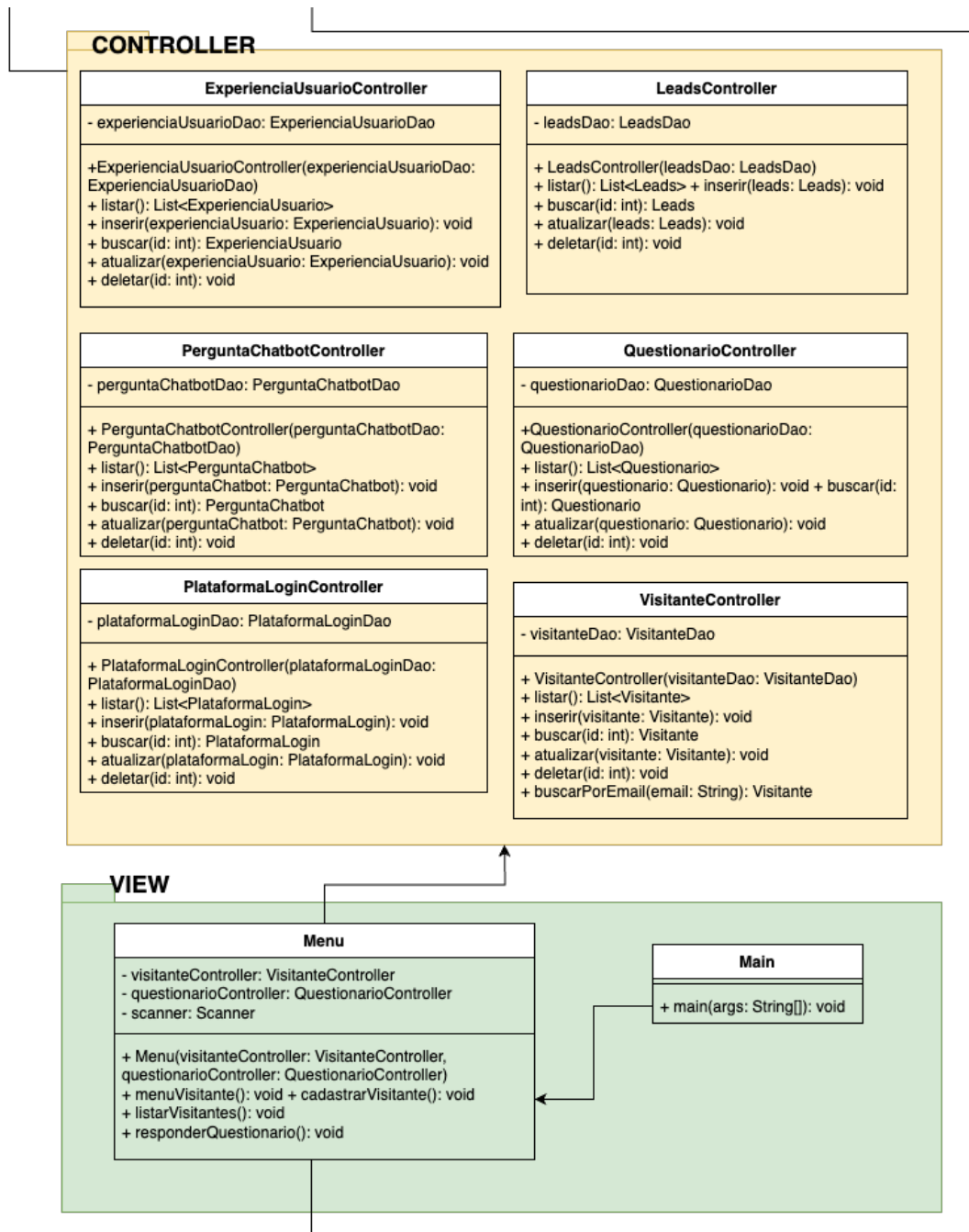
O diagrama de classes UML do projeto fornece uma visão estruturada da arquitetura de software, destacando a organização das classes e suas interações. Ele é fundamental para entender como o sistema foi construído, facilitando a manutenção e expansão futuras do projeto. Este diagrama reflete a aplicação do padrão MVC (Model-View-Controller), que separa a lógica de negócio (model), a interface de usuário (view) e o controle de entrada (controller), promovendo assim uma estrutura clara e modular.

As classes de modelo (VO), representam as estruturas de dados usadas em todo o sistema. Elas são utilizadas pelas classes DAO (Data Access Object) correspondentes, responsáveis pela comunicação direta com o banco de dados, executando operações como inserção, atualização, busca e exclusão de registros.

Os Controllers atuam como intermediários entre a view e o model, processando as requisições do usuário, manipulando o model e decidindo qual view será mostrada. No projeto, temos controllers específicos para cada VO, que gerenciam as operações relacionadas aos seus respectivos modelos.

A classe `ConexaoBancoDeDados` é essencial para o funcionamento do sistema, provendo a conexão com o banco de dados utilizada pelas classes DAO. Este componente é um exemplo de como o projeto se beneficia da separação de responsabilidades, permitindo a reutilização da lógica de conexão em diversas partes do sistema. As classes de validação, como `Validacoes`, oferecem métodos estáticos para garantir a integridade dos dados antes de serem processados ou persistidos, enquanto `GestaoData` fornece utilitários para manipulação de datas e horas. Por fim, a classe `Menu` representa o ponto de entrada para a interação do usuário com o sistema, encadeando a lógica de apresentação e navegação entre diferentes funcionalidades oferecidas pelo sistema.





9. Classe Model

A camada Model atua como o núcleo do sistema de gerenciamento de visitantes e questionários. Ela gerencia a lógica de negócios, as regras de aplicação e o armazenamento dos dados, com foco nas entidades Visitante e Questionário.

Enquanto a entidade Visitante armazena dados dos usuários para personalizar a experiência no site, a entidade Questionário coleta respostas para análise de preferências e necessidades dos visitantes. Essa camada interage com o banco de dados através das classes DAO, garantindo operações de dados seguras e eficientes. Assim, a camada Model é crucial para a integridade dos dados e suporte à lógica de negócios do projeto.

10. Classe View

A classe Menu, parte do pacote view, é a interface interativa do sistema. Aqui, o usuário pode realizar ações como cadastrar-se no sistema, listar todos os usuários cadastrados e responder a um questionário. Cada uma destas ações é mapeada para um método específico:

menuVisitante(): O núcleo interativo que apresenta o menu principal ao usuário, permitindo a seleção de diferentes funcionalidades do programa. Este método lida com a entrada do usuário e direciona para as ações correspondentes, tratando exceções como erros de entrada com cuidado para não interromper a experiência do usuário.

cadastrarVisitante(): Encarrega-se de coletar dados como nome e e-mail do visitante, validando-os com a ajuda da classe Validacoes para garantir que os dados inseridos atendam aos critérios definidos. Uma vez validados, os dados são utilizados para criar um novo objeto Visitante que é então inserido no sistema por meio do VisitanteDao.

listarVisitantes(): Oferece uma visão geral dos usuários cadastrados, recuperando e exibindo informações de todos os visitantes armazenados no banco de dados através do VisitanteDao.

responderQuestionario(): Este método permite que um visitante cadastrado responda a um questionário. Se o e-mail fornecido não estiver associado a nenhum visitante, o sistema oferece a opção de cadastro. Posteriormente, o questionário é apresentado, e as respostas do usuário são coletadas e armazenadas utilizando o QuestionarioDao. A importância desta parte do código reside na capacidade de capturar, validar e armazenar dados do usuário, essenciais para análises de negócios e melhorias do sistema. Ao coletar informações precisas, a empresa pode personalizar experiências, direcionar recursos e desenvolver estratégias de mercado mais eficazes.

11. Classe Controller

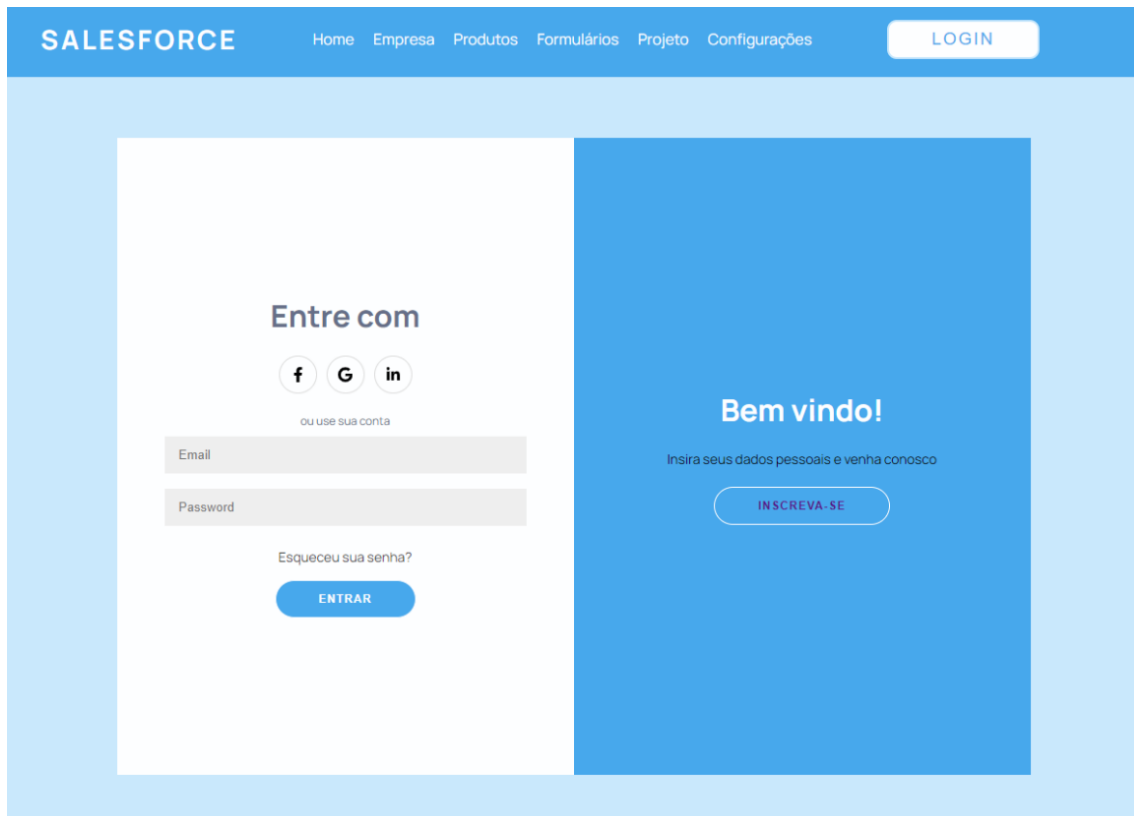
As classes Controller atuam como intermediárias entre a visualização dos dados (View) e o gerenciamento dos mesmos (Model). Elas recebem as entradas do usuário, processam esses dados com a ajuda das classes do Model e determinam os dados que serão exibidos na View.

Por exemplo, a **VisitanteController** gerencia as operações relacionadas aos visitantes, como a inserção de novos registros ou a busca de visitantes existentes. Similarmente, a **QuestionarioController** lida com as ações pertinentes aos questionários, facilitando a coleta e análise de respostas.

Essa camada de controle é essencial para manter a lógica de negócios separada da interface do usuário, permitindo que as operações sejam realizadas de maneira eficiente e organizada. As controllers garantem que o projeto seja fácil de manter e escalar, contribuindo para uma estrutura de código mais limpa e modular.

12. Protótipos

Tela de Login:



O protótipo da tela de login do Salesforce apresenta uma interface limpa e moderna. No topo, uma barra azul contém o logotipo "SALESFORCE" e um menu de navegação com os itens "Home", "Empresa", "Produtos", "Formulários", "Projeto" e "Configurações". Um botão "LOGIN" está posicionado no canto superior direito. A área principal é dividida em duas seções. À esquerda, sobre um fundo branco, há o título "Entre com" seguido por ícones de login com Facebook, Google e LinkedIn. Abaixo, o texto "ou use sua conta" precede campos de entrada para "Email" e "Password". Um link "Esqueceu sua senha?" e um botão "ENTRAR" em azul completam esta seção. À direita, sobre um fundo azul sólido, o texto "Bem vindo!" é exibido, seguido por "Insira seus dados pessoais e venha conosco" e um botão "INSCREVA-SE" em branco com contorno azul.

Página sobre o formulário:



O protótipo da página sobre o formulário do Salesforce possui o mesmo cabeçalho azul com o logotipo, menu de navegação e botão "LOGIN". O conteúdo principal é dividido. À esquerda, o título "Sobre esse formulário" é seguido por um texto explicativo: "Se você não conhece sobre o que é CRM ou as melhores soluções para sua empresa, responda o questionário abaixo para te conhecermos melhor e te direcionar com as melhores soluções. Teremos uma IA e um time comprometido para entender suas necessidades e preocupações." Um botão "Preencha agora" está localizado na base do texto. À direita, uma ilustração mostra uma pessoa interagindo com uma interface digital que apresenta ícones de perfil e documentos, representando o processo de coleta de dados para o CRM.

13. Utilizando a Aplicação

Para rodar o sistema do projeto, siga os passos resumidos abaixo:

Certifique-se de ter o Java instalado: O projeto requer uma JVM (Java Virtual Machine) compatível. Verifique a instalação com `java -version` no terminal.

Inclua o Oracle JDBC Driver: Garanta que o arquivo `.jar` do driver Oracle JDBC esteja na pasta `lib` do projeto para conexão com o banco de dados.

Configuração do Banco de Dados: Certifique-se de que o SGBD Oracle está configurado corretamente e acessível. Utilize as credenciais de conexão (usuário, senha, URL) especificadas no arquivo `ConexaoBancoDeDados.java`.

Compilação do Projeto: Compile o código do projeto. Se estiver usando uma IDE (Eclipse, IntelliJ, etc.), utilize as funcionalidades de build da própria IDE. Para compilação manual, use o comando `javac` no terminal, incluindo o classpath para o JDBC driver se necessário.

Execução do Sistema: Execute o programa principal do seu projeto. Isso pode ser feito diretamente pela IDE ou via linha de comando, ajustando o classpath conforme a localização dos arquivos `.class` e o driver JDBC.

Notas Adicionais:

Verifique a Conexão com o Banco: Antes de rodar o sistema, confirme se as configurações de conexão estão corretas e se o banco de dados está online.

Segurança: Assegure-se de que informações sensíveis, como credenciais de banco de dados, não estejam expostas no código.

Dependências: Certifique-se de que todas as dependências, especialmente o driver JDBC do Oracle, estejam corretamente configuradas e acessíveis pelo projeto. Seguindo esses passos, você poderá compilar e executar o sistema do seu projeto de forma eficiente.