ESCOLA SENAI "DR. CELSO CHARURI" - UNIDADE SUMARÉ (CFP 512)
ESCOLA SESI-SP CHALIL ZABANI (CE 436)

TÉCNICO EM DESENVOLVIMENO DE SISTEMAS / LINGUAGENS E CÓDIGOS

GUILHERME VEIGA PEDROMILO
JOÃO PEDRO ARAÚJO TEIXEIRA
ISABELLA TEXEIRA DOS SANTOS
CAIO FELIPE VIEIRA FERREIRA

VENTURO

SISTEMA DE GESTÃO DE RECURSOS EMPRESARIAIS

SUMARÉ
2024

GUILHERME VEIGA PEDROMILO
JOÃO PEDRO ARAÚJO TEIXEIRA
ISABELLA TEXEIRA DOS SANTOS
CAIO FELIPE VIEIRA FERREIRA

VENTURO

SISTEMA DE GESTÃO DE RECURSOS EMPRESARIAIS

Projeto desenvolvido pela equipe Venturo's Dev Team apresentado como requisito de final de atividade integrada entre o curso de "Desenvolvimento de Sistemas" da Escola Senai "DR. Celso Charuri" – unidade Sumaré, com o ensino regular oferecido pela unidade SESI-SP "Chalil Zabani" (CE 436) - unidade Nova Odessa.

Orientadores (professor/a):

MARCIO DENADAI - SENAI
MATHEUS LUIZ OLIVEIRA DE CAMARGO - SENAI
EDINA CRISTINA DE SOUSA IGNACIO - SESI
ERICA FABIANA ALVES DE LIMA LOPES - SESI

SUMARÉ
2024

SUMÁRIO

RESUMO

O projeto Venturo tem como objetivo o desenvolvimento de um *software para* a solução de desafios empresariais, com foco na gestão, auditoria e operação de negócios. Com o propósito de fornecer um ambiente seguro, produtivo e eficiente, o sistema foi idealizado a partir da análise das necessidades do setor varejista. O *software* é estruturado em quatro módulos principais: força de vendas, estoque, financeiro e auditoria. Este trabalho abrange o estudo detalhado da arquitetura do sistema, as tecnologias utilizadas no desenvolvimento, as funcionalidades principais de cada módulo e a elaboração de um manual do usuário. Este trabalho inclui desde a definição do escopo e desenvolvimento até a documentação e elaboração do manual do usuário, assegurando a compreensão e a aplicação prática do *software*.

PALAVRAS-CHAVE: Venturos's Dev Team, ERP, SAAS.

ABSTRACT

The Venturo project aims to develop software to solve business challenges, focusing on management, auditing and business operations. With the purpose of providing a safe, productive and efficient environment, the system was designed based on the analysis of the needs of the retail sector. The software is structured into four main modules: sales force, inventory, finance and audit. This work covers the detailed study of the system architecture, the technologies used in development, the main functionalities of each module and the preparation of a user manual. This work includes everything from defining the scope and development to documentation and preparation of the user manual, ensuring understanding and practical application of the software.

KEY TAGS: Venturos's Dev Team, ERP, SAAS.

INTRODUÇÃO

O sistema Venturo, é um projeto *SAAS* (*Software* como serviço) *Multi-Tenant* com *Multi-Schemas* desenvolvido com base nas funcionalidades de um ERP (*Enterprise Resource Planning*), tendo como um objetivo prover serviços de rotina de estoque, vendas, contabilidade e gestão de funcionários, tendo seu público-alvo micro-empreendedores e empresas comerciais.

Para compreender este sistema devemos compreender o que é *Multi-Tenant* com *Multi-Schemas*, e o que torna o sistema Venturo abranger estas características. Um sistema *Multi-Tenant* é um *software* desenvolvido para atender vários clientes/usuários por um único intermediador (API principal ou API's estruturadas que conectam vários clientes diferentes e "conversa" com um único banco de dados ou inúmeros bancos de dados, sejam eles para cada cliente ou não) que se conectam naquele sistema, diferente de um *Single-Tenant* onde cada usuário conecta-se ao um único intermediador e tem seu banco de dados próprio:

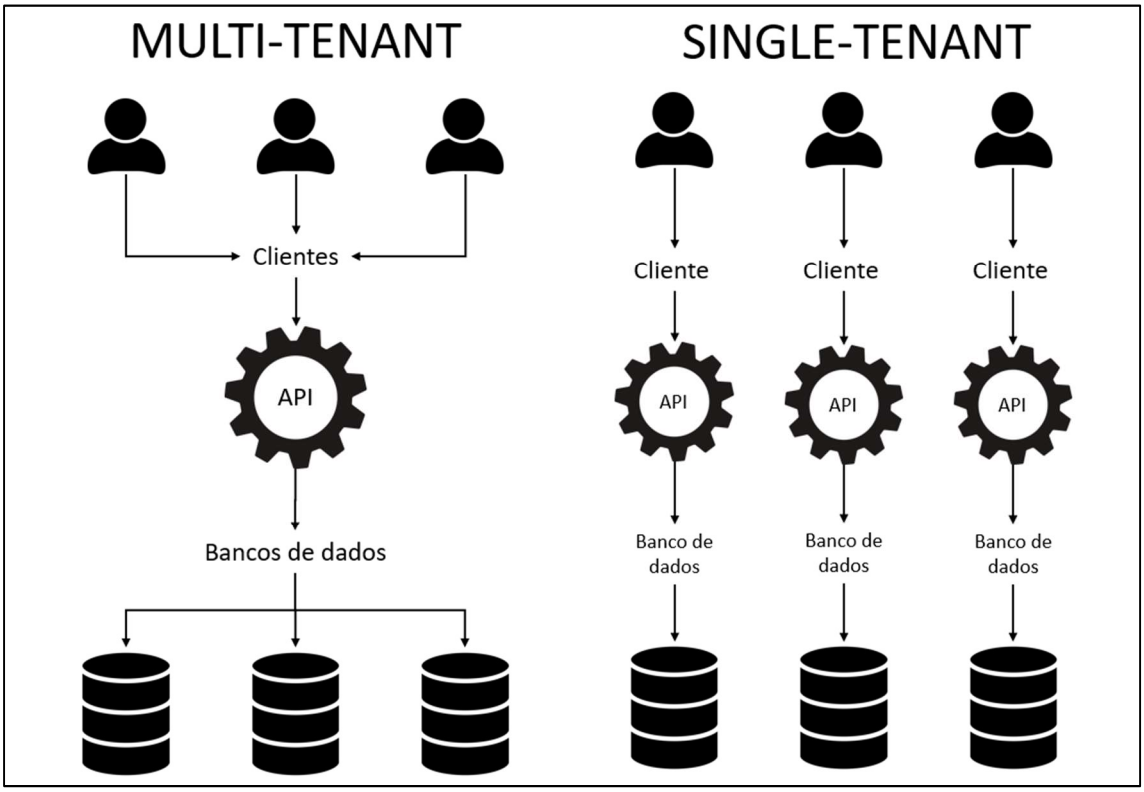

Figura 1 - Esquema de funcionamento

No sistema Venturo, além de prover um ambiente Multi-Tenant com um *database* principal, ele também contém o *Multi-Schemas*, o que abrange a capacidade de para cada empresa tenha um banco de dados com informações privadas.

DESENVOLVIMENTO DO PROJETO

Para o desenvolvimento do sistema Venturo, foi necessário pesquisas e idealizar uma estrutura de funcionamento lógico do sistema, onde o primeiro desafio era estruturar a funcionalidade *Multi-Tenant* com *Multi-Schemas*.

Uma das formas de resolver este problema, foi utilizar a biblioteca Knex.js, a qual tem a funcionalidade de um *SQL query builder*, o que auxilia a configuração dinâmica de bancos de dados, sendo uma configuração para o banco principal, e outra para os bancos de dados das empresas:

Configuração de banco de dados principal

```
require('dotenv').config();
const knex = require('knex');

const knexConfig = {
  client: 'mysql2',
  connection: {
    host: process.env.DB_HOST,
    user: process.env.DB_USER,
    password: process.env.DB_PASSWORD,
    database: process.env.DB_NAME
  // Banco de dados principal
  },
  pool: { min: 0, max: 7 }
};

const mainDb = knex(knexConfig);

module.exports = { knexConfig, mainDb };

```

Configuração de bancos de dados das empresas

```
require('dotenv').config();
const knex = require('knex');
// Função para criar uma conexão Knex.js din
âmica para o banco da empresa específica
const createEmpresaKnexConnection = (
empresaId) => {
  return knex({
    client: 'mysql2',
    connection: {
      host: process.env.DB_HOST,
      user: process.env.DB_USER,
      password: process.env.DB_PASSWORD,
      database: `${empresaId}`
  // Nome do banco da empresa específica
    }
  });
};

module.exports = {
  createEmpresaKnexConnection,
};
```

Figura 2 - Módulo de configuração do Knex.js

Desta forma, toda vez que um usuário da empresa fizer uma requisição para a API, é necessário enviar o “ID” de registro da empresa, para que ela se conecte ao banco de dados e consiga realizar a requisição sem erros.

Após solução deste problema, foi desenvolvido um protótipo em React JS utilizando a *framework* Vite para verificar se a estrutura definida seria executada,

para isso foi criado um sistema de autenticação com *token*, baseado na estrutura desenvolvida e em um fluxo de *refresh token*:

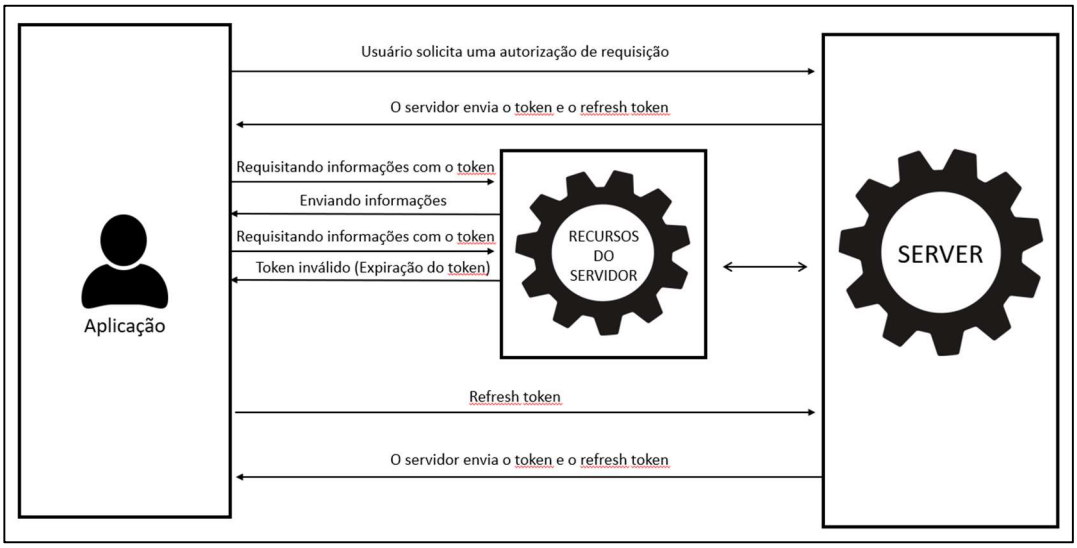

Figura 3 - Esquema de autenticação

Quando um usuário concluí seu login, a aplicação envia uma requisição para o servidor, e o servidor responde com o token e o refresh token com todas as informações necessárias, dando início ao ciclo de requisições feitas pelo usuário que duram 15 minutos, isso porque o token tem um tempo de expiração igual a 15 minutos, e então o refresh token é usado para solicitar as informações e gerar um novo token.

Após resolver essas questões essenciais para o funcionamento básico do projeto, e aplicar outras rotas de funcionalidade no *backend* para implementar todas as funcionalidades desejadas para o sistema, sendo ao mesmo tempo implementando o design ao projeto, utilizando o Gestalt, e outras técnicas de design para aumentar a qualidade do sistema.

Para o *frontend*, como já foi dito, foi utilizado o React JS com a *framework* Vite, além de algumas bibliotecas (*React Bootstrap, React – Icons e Recharts*) para adquirir ícones, gráficos e estilos de componentes prontos, trazendo um resultado inovador e responsivo:

Figura 4 - Landpage do sistema

FUNCIONALIDADES

O sistema Venturo é repleto de funcionalidades, contendo dez áreas de funcionalidades, sendo estas:

Contas a Pagar

- Cadastro de Fornecedores: Registra informações detalhadas sobre os fornecedores, centralizando os dados de parceiros comerciais;
- Lançamento de Despesas: Permite o registro de despesas, facilitando o controle financeiro;
- Pagamentos Programados: Agenda e gerencia os pagamentos futuros, garantindo que as obrigações financeiras sejam cumpridas pontualmente.

Contas a Receber

- Cadastro de Clientes: Registra informações detalhadas sobre os clientes, facilitando o acompanhamento de suas transações;
- Lançamento de Receitas: Permite o registro de receitas, otimizando o controle de fluxo financeiro;
- Recebimentos Programados: Agenda e gerencia os recebimentos futuros, proporcionando previsibilidade financeira;
- Fluxo de Caixa: Monitora o fluxo de caixa, permitindo uma análise detalhada das entradas e saídas de recursos financeiros.

Estoque

- Registro de Produtos: Permite registrar novos produtos no sistema, mantendo o controle do inventário;
- Atualização de Informações de Produtos: Facilita a atualização contínua das informações de produtos, como descrição, preço e quantidade;
- Entrada de Produtos: Registra a entrada de produtos no estoque, mantendo o controle atualizado;
- Saída de Produtos: Registra a saída de produtos do estoque, permitindo uma visão clara das vendas e movimentações;
- Histórico de Movimentações: Mantém um histórico detalhado das movimentações de estoque, ajudando no rastreamento e controle de produtos.

Vendas

- Cadastro de Nota Fiscal: Permite registrar notas fiscais no sistema;
- Emissão de Nota Fiscal: Facilita a emissão de notas fiscais conforme as vendas são realizadas;
- Histórico de Notas Fiscais: Mantém um registro detalhado de todas as notas fiscais emitidas, garantindo conformidade fiscal;
- Cadastro de Pedidos: Permite registrar novos pedidos de venda, organizando a gestão comercial;
- Consulta de Pedidos: Facilita a consulta de pedidos existentes, proporcionando maior agilidade no atendimento ao cliente;
- Cadastro de Clientes: Registra informações detalhadas sobre os clientes, como dados de contato e histórico de compras;
- Histórico de Compras: Mantém um histórico das compras realizadas por cada cliente, permitindo um melhor acompanhamento do relacionamento com o cliente;
- Vendas por Produto: Gera relatórios detalhados de vendas por produto, auxiliando na análise de desempenho dos itens vendidos;
- Vendas por Cliente: Gera relatórios detalhados de vendas por cliente, facilitando a análise do perfil de consumo.

Dashboard

- Dashboard: Oferece uma visão geral e interativa do desempenho empresarial, com gráficos e indicadores chave, proporcionando uma análise rápida e precisa dos principais indicadores financeiros e operacionais.

Cadastro de Usuários

- Definição de Perfis e Permissões: Permite definir perfis de usuários e configurar permissões de acesso, garantindo a segurança e controle de informações no sistema.

Auditoria

- Log de Ações de Usuários: Mantém um registro detalhado das ações realizadas por cada usuário, facilitando a rastreabilidade e o controle interno;
- Relatórios de Auditoria: Gera relatórios detalhados para auditoria, garantindo transparência nas operações e controle interno eficaz.

Suporte e Comunicação

- Sistema de e-mail entre Usuários: Integra um sistema de e-mail para comunicação interna entre os funcionários da empresa, melhorando a coordenação entre equipes;

Comunicação direta com o Suporte: Permite o envio de mensagens diretamente para a equipe de suporte Venturo, agilizando a resolução de problemas e dúvidas.

GLOSSÁRIO

Log arquivo ou estrutura de dados que armazena informações sobre eventos, erros ou outras atividades que ocorrem dentro de uma aplicação ou sistema.

SAAS Modelo de entrega de software em que a aplicação é hospedada na nuvem e acessada via internet, geralmente com cobrança baseada em uso ou assinatura. Exemplos incluem ferramentas de gestão, CRM, e aplicativos de produtividade.

SQL Builder Ferramenta ou biblioteca que facilita a construção de consultas SQL programaticamente, geralmente usada para evitar erros e melhorar a legibilidade do código. Muitas vezes inclui abstrações que tornam a consulta mais dinâmica e reutilizável.

Middleware Camada intermediária entre o sistema de gerenciamento de banco de dados (ou o servidor) e o cliente. Pode realizar tarefas como autenticação, manipulação de dados e log, antes de passar a solicitação para o backend ou frontend.

Multi-Tenat Arquitetura de software em que uma única instância de uma aplicação serve múltiplos clientes (ou "tenants"), isolando dados e configurações específicas de cada um, mas compartilhando a mesma infraestrutura.

Multi-Schemas Estrutura de banco de dados onde múltiplos esquemas são utilizados para organizar os dados, permitindo um nível de separação entre as informações enquanto compartilham o mesmo banco de dados físico.

Sofware Refere-se a qualquer tipo de aplicação ou sistema desenvolvido para realizar tarefas específicas em um dispositivo, podendo ser tanto local quanto baseado em nuvem.

Single-Tenant Modelo de arquitetura onde uma instância de uma aplicação ou serviço é dedicada a um único cliente (tenant), o que garante maior controle sobre dados e recursos, mas pode ser menos escalável que o modelo multi-tenant.

API (Application Programming Interface) Conjunto de regras e definições que permite a comunicação entre diferentes sistemas de software. As APIs permitem que aplicações interajam e compartilhem dados ou funcionalidade

Framework Estrutura ou conjunto de ferramentas de desenvolvimento que fornece funcionalidades reutilizáveis e diretrizes para construção de software. Exemplos incluem frameworks para frontend (React, Angular) e backend (Django, Node.js).

Token Valor usado para autenticar e autorizar acessos a sistemas, serviços ou recursos. Frequentemente utilizado em APIs para manter o estado de sessão do usuário e garantir segurança.

Refresh Token Token utilizado para renovar um "access token" expirado sem a necessidade de reautenticar o usuário. É comum em sistemas de autenticação baseados em OAuth2.

Gestalt Teoria psicológica que pode ser aplicada no design de interfaces, sugerindo que o usuário percebe um conjunto de elementos como uma

totalidade, em vez de um amontoado de partes. Importante para a criação de interfaces intuitivas e eficazes.

Frontend Parte da aplicação com a qual os usuários interagem diretamente. Normalmente envolve HTML, CSS, JavaScript, e bibliotecas como ReactJS, Angular ou Vue.js.

Backend Parte do sistema que lida com a lógica de negócios, manipulação de dados, autenticação e comunicação com bancos de dados. O backend normalmente não é visível para o usuário, mas é crucial para o funcionamento da aplicação.

ReactJS Biblioteca JavaScript de código aberto para construir interfaces de usuário, particularmente eficaz em aplicativos de uma única página (SPA). Usada para criar componentes reutilizáveis.

React Icons Conjunto de ícones vetoriais reutilizáveis para uso com React. Permite que você adicione ícones facilmente a um projeto React, integrando bibliotecas populares de ícones, como FontAwesome e Material Icons.

Recharts Biblioteca React para a construção de gráficos e visualizações de dados. Utiliza D3.js por trás para a renderização de gráficos dinâmicos, como barras, linhas e gráficos de pizza.

Bootstrap Framework front-end popular para desenvolver sites responsivos e móveis. Inclui componentes prontos (botões, tabelas, formulários etc.) e facilita o design e desenvolvimento de interfaces consistentes.

Dashboard Interface visual que exibe informações e métricas em tempo real, geralmente em um formato gráfico ou tabelado. É comum em sistemas de monitoramento, administração e análise de dados.

KnexJS Biblioteca SQL para Node.js que funciona como um *query builder*, simplificando a criação de consultas SQL e a interação com bancos de dados. Suporta múltiplos bancos de dados, incluindo PostgreSQL, MySQL e SQLite.

ERP (Enterprise resource planning) Software de gestão empresarial integrado que ajuda na administração de recursos e processos de uma organização, como finanças, produção, vendas e inventário. Exemplo: SAP, Microsoft Dynamics.

REFERÊNCIAS BIBLIOGRÁFICAS

KNEX.JS. Knex.js: A SQL Query Builder for Javascript. Disponível em: https://knexjs.org/. Acesso em: 6 jul. 2024.

REACT.JS. React.js: The library for web and native user interfaces. Disponível em: https://react.dev/. Acesso em: 17 jul. 2024.

RECHARTS. Recharts: A composable charting library built on React components. Disponível em: https://recharts.org/. Acesso em 20 ago. 2024.

REACT ICONS. React icons: Include popular icons in your React projects easly with react-icons. Disponível em: https://react-icons.github.io/react-icons/. Acesso em 12 ago. 2024.

VIA CEP. Via Cep: Webservice gratuito de alto desempenho para consulta de Código de Endereçamento Postal (CEP) do Brasil. Disponível em: https://viacep.com.br/. Acesso em 17 de set. 2024.

DOCKER. Docker: The #1 containerization software for developers and teams. Disponível em: https://www.docker.com/products/docker-desktop/. Acesso em 24 jul. 2024.

JWT. Jwt: JSON Web Tokens are an open, industry standard RFC 7519 method for representing claims securely between two parties. Disponível em: https://jwt.io/. Acesso em 24 jul. 2024.