UNIVERSIDADE FEDERAL DE MINAS GERAIS
SISTEMAS DE INFORMAÇÃO
MONOGRAFIA DE SISTEMAS DE INFORMAÇÃO II (MSI II)

AILTON VINAUD JÚNIOR

IMPLEMENTAÇÃO E VALIDAÇÃO DE CHATBOT PARA AGENDAMENTO DE ATENDIMENTO VOLTADO PARA MICROEMPREENDEDORES

BELO HORIZONTE
2024

AILTON VINAUD JÚNIOR

IMPLEMENTAÇÃO E VALIDAÇÃO DE CHATBOT PARA AGENDAMENTO DE ATENDIMENTO VOLTADO PARA MICROEMPREENDEDORES

Monografia do curso de Sistema de Informação

Orientador: Dr. Marco Túlio Valente

Tipo de Pesquisa: Projeto Tecnológico

BELO HORIZONTE

2024

LISTA DE FIGURA

LISTA DE GRÁFICOS

LISTA DE TABELA

Sumário

1. INTRODUÇÃO

No atual cenário empresarial, microempreendedores enfrentam um desafio significativo ao equilibrar suas atividades de atendimento ao cliente com o complexo gerenciamento de agendamentos e das operações comerciais. Muitas vezes, esses empreendedores trabalham sozinhos e frequentemente carecem de soluções eficazes para a gestão de compromissos e para a administração do seu empreendimento.

É notório que muitos microempreendedores iniciam suas jornadas profissionais movidos por uma profunda expertise em seus serviços ou produtos, porém, frequentemente enfrentam dificuldades na gestão do negócio em si. Eles podem ser altamente competentes na execução do serviço oferecido, mas têm pouca experiência na administração, atendimento ao cliente, gerenciamento financeiro, estratégias de marketing e logística. Neste cenário, os empreendedores acabam sobrecarregados com múltiplas áreas que são fundamentais para o sucesso do empreendimento, mas que não fazem parte das habilidades adquiridas durante suas formações.

É comum encontrar profissionais como manicures, professores, psicólogos, cabeleireiros, fisioterapeutas, personal trainers entre outros, que lidam diariamente com a dupla demanda de prestar um serviço de alta qualidade aos seus clientes e, ao mesmo tempo, de gerenciar todos os aspectos operacionais e administrativos do seu negócio. Nesse contexto, a sobrecarga de tarefas torna-se uma realidade que frequentemente resulta em desafios consideráveis.

Um grande desafio para estes profissionais é se tornar disponível mesmo com a agenda repleta de atendimentos. O profissional precisa focar em cada atendimento realizado, dando total atenção ao seu cliente que agendou o serviço a ser prestado. Todavia, mesmo durante o atendimento, o empreendedor não pode deixar de dar atenção aos demais clientes que querem realizar novos agendamentos, tirar dúvidas ou mesmo avisarem sobre remarcação de horários e imprevistos passíveis de cancelamentos da consulta.

O profissional então entende que seu negócio está em crescimento acelerado, mas não possui receita suficiente para viabilizar a contratação de uma pessoa responsável pelo atendimento ao público, bem como agendamentos e afins, tampouco

possui expertise e recursos financeiros para contratar softwares robustos voltados para sua área de atendimento seja ela barbearia, salão de beleza, saúde etc.

Por vezes, os softwares disponíveis no mercado possuem preço significativo para um microempreendedor custear, além de serem complexos de utilizar ou envolverem adaptação por parte do profissional e dos clientes que buscam atendimento (realização de download de aplicativos no smartphone e cursos para entender o funcionamento do aplicativo ou página web).

1.2. Objetivos

1.2.1. Objetivo Geral

Verificar a viabilidade de mercado e desenvolver um software de *chatbot* capaz de facilitar o agendamento de atendimento para microempreendedores, por meio de integração com o *WhatsApp for Business* e o *Google Calendar*.

1.2.2 Objetivo Específico

O objetivo é implementar o chatbot como MVP (*minimum viable product*) e validá-lo com um cliente em potencial no mercado.

2 REFERENCIAL TEÓRICO

2.1 Design Thinking e o Double Diamond

O *Design Thinking* é uma metodologia focada na geração de soluções inovadoras que atendem às necessidades reais dos usuários, caracterizando-se pelo seu ciclo iterativo e pela ênfase na empatia. O modelo do *Double Diamond*, desenvolvido pelo *Design Council*, descreve essa abordagem através de quatro fases principais: Descobrir, Definir, Desenvolver e Entregar.

As três primeiras etapas desse modelo foram abordadas no semestre anterior, focando na identificação e definição das necessidades dos microempreendedores, bem como na ideação e prototipação de uma solução viável. Agora, o projeto avança para a etapa entregar, onde o objetivo é criar um MPV e prepará-lo para o lançamento, representando a etapa final do modelo *Double Diamond*.

2.2 A solução idealizada

A solução idealizada no trabalho de MSI I é a criação de um *chatbot*, que utilize a arquitetura DDD (*Domain Driven Design*) ao integrar plataformas do *WhatsApp for Business* e *Google Calendar* por meio de um MVP que é validado por um dos microempreendedores selecionados na etapa de descobrir. O cliente que fez a validação foi selecionado por meio dos questionários e entrevistas realizados na etapa inicial do trabalho do semestre anterior.

2.3 Produto Mínimo Viável (MVP)

No contexto de startups e desenvolvimento ágil de produtos, o conceito de MVP emerge como uma estratégia crucial para validar ideias de negócios com o mínimo de recursos possível. Por meio dele, é possível testar hipóteses de mercado, adaptar-se rapidamente às necessidades dos usuários e iterar o produto de maneira eficiente.

Este princípio foi fundamental na fase inicial do desenvolvimento do nosso *chatbot*, assegurando que a solução atendesse às necessidades reais dos microempreendedores com a máxima economia de tempo e recursos.

2.4 Arquitetura DDD (Domain-Driven Design)

A complexidade inerente ao desenvolvimento de software exige uma abordagem estruturada que facilite a comunicação entre os envolvidos e a implementação de funcionalidades relevantes para o negócio. O Domain-Driven Design, proposto por Eric Evans, oferece um framework para lidar com essa complexidade, centrando o desenvolvimento no domínio do negócio e na linguagem ubíqua.

A adoção do DDD no desenvolvimento do *chatbot* permitiu uma modelagem mais precisa e alinhada às necessidades específicas dos microempreendedores, facilitando a criação de um sistema coeso e adaptado aos processos de trabalho.

2.5 API do WhatsApp for Business

A API do *WhatsApp for Business* é uma ferramenta que permite receber e enviar mensagens por meio da aplicação do *WhatsApp*.

Integrar o *chatbot* com esta API permitirá a automação de comunicações diretas com os clientes, uma funcionalidade crucial para o sucesso do projeto. A escolha por utilizar esta plataforma garantiu que a solução proposta estivesse adaptada ao meio mais utilizado de agendamento pelos clientes finais que consomem serviços agendados.

O questionário enviado no semestre anterior aponta que 100% dos empreendedores recebem solicitações de agendamentos de serviços por meio do *WhatsApp*.

2.6 API do Google Calendar

A gestão eficiente de agendas é fundamental para o sucesso de microempreendedores, tornando essencial a integração do nosso *chatbot* com serviços de calendário.

A API do *Google Calendar* é uma ferramenta que oferece acesso direto à aplicação do *Google Calendar* ou *Google Agenda*. Por meio dela o *chatbot* foi capaz de manipular eventos, agendamentos e notificações na agenda do microempreendedor, facilitando a sincronização de compromissos e a automatização de processos.

3 DESCRIÇÃO DO MVP

3.1 Apresentação do MVP

O produto criado é uma aplicação que integra as APIs do *WhatsApp* for *Business* com o *Google Calendar*. O funcionamento do produto se baseia em interagir com o cliente que deseja realizar agendamento de serviço por meio da plataforma do *WhatsApp*.

O funcionamento da aplicação é iniciado assim que ocorre o recebimento de uma mensagem originada de um cliente que deseja atendimento. O cliente envia uma mensagem para o número do *WhatsApp* do microempreendedor e, após o recebimento, o produto *chatbot* responde a mensagem com uma série de opções parametrizadas que conduzirão a um agendamento automático de serviços.

O objetivo do produto é garantir uma interação fluida entre *chatbot* e cliente de forma a resultar em um agendamento de serviço que será assinalado na conta *google calendar* dos empreendedores. Para este MPV, conforme será detalhado no próximo capítulo de validação, foram controladas as agendas de dois profissionais de uma barbearia da cidade de Araxá/MG.

3.2 Principais funcionalidades

O *chatbot* em sua versão MVP, possui três funcionalidades: **agendamento de serviço**, **cancelamento de serviço** e **falar com um profissional**.

A principal funcionalidade é a de **agendamento de serviço**, que possui as seguintes etapas:

1. Cliente envia uma mensagem para o número *whatsapp* do estabelecimento comercial;
2. *Chatbot* responde com uma mensagem de saudação e traz opções de serviços;
3. Cliente escolhe uma opção de serviço;
4. *Chatbot* solicita dia para agendamento;
5. Cliente digita dia em que deseja realizar o serviço;
6. *Chatbot* solicita período de atendimento;
7. Cliente envia opção referente ao período de atendimento;
8. *Chatbot* solicita escolha de qual profissional fará o atendimento;
9. Cliente escolhe profissional que irá atendê-lo;
10. *Chatbot* verifica na agenda do google daquele profissional quais horários estão disponíveis;
11. *Chatbot* envia lista de até seis horários disponíveis na agenda do profissional para a realização do serviço escolhido;
12. Cliente escolhe horário;

13. *Chatbot* agenda atendimento no calendário do profissional escolhido;
14. *Chatbot* envia mensagem de confirmação para cliente.

A segunda funcionalidade estava fora do escopo MVP, todavia, o empreendedor responsável pela barbearia que fez a validação do produto entendeu que seria requisito necessário para que a validação ocorresse. Dessa forma, foi implementado nesta versão a opção de **cancelamento de serviço** que contempla as seguintes etapas:

1. Cliente envia uma mensagem para o número *whatsapp* do estabelecimento comercial;
2. *Chatbot* responde com uma mensagem de saudação e traz opções de serviços;
3. Cliente escolhe uma opção de cancelar um agendamento;
4. *Chatbot* verifica quais agendamentos o cliente possui em aberto;
5. *Chatbot* envia a lista com todos os agendamentos em aberto para o cliente;
6. Cliente escolhe qual agendamento deseja cancelar;
7. *Chatbot* exclui o agendamento apontado na agenda do empreendedor que estava responsável pelo atendimento;
8. *Chatbot* envia uma mensagem confirmando que o agendamento foi cancelado.

A terceira funcionalidade é: **falar com um profissional**. Para esta funcionalidade é enviada uma mensagem para o cliente com o link do *whatsapp* pessoal do empreendedor.

É importante mencionar que o *whatsapp* for business vinculado ao produto *chatbot* é de uso exclusivo da aplicação. A empresa Meta não permite cadastro de um número *whatsapp* for business em mais de um dispositivo e o *chatbot* é contabilizado como dispositivo de cadastro. Dessa forma, caso um cliente queira falar com o empreendedor, o mesmo deve ser redirecionado para o *whatsapp* cadastrado no telefone do dono do estabelecimento comercial, este redirecionamento é feito por meio de um link parametrizado com o número do telefone do empreendedor.

As etapas deste fluxo são as seguintes:

1. Cliente envia uma mensagem para o número *whatsapp* do estabelecimento comercial;

2. *Chatbot* responde com uma mensagem de saudação e traz opções de serviços;
3. Cliente escolhe uma opção de falar com um profissional;
4. *Chatbot* envia uma mensagem com o link que direciona para *whatsapp* do empreendedor;
5. Cliente clica no link e inicia a conversa com o empreendedor diretamente.

3.3 Arquitetura

A solução foi desenhada utilizando o *Domain-Driven-Design*, conforme explanado na metodologia deste trabalho.

A solução foi dividida em três partes principais: um *serverless*, a api principal e um banco de dados *sqlserver*. Todas as aplicações foram desenvolvidas em C# .NET. A seguir, será apresentado o detalhamento de cada um deles.

Serverless: A aplicação *serverless* foi criada com o objetivo de receber a mensagem enviada pelo WhatsApp via *webhook*. A ideia aqui é desacoplar a autenticação do *webhook* na api WhatsApp for business da Meta e processar o objeto recebido na mensagem, estruturando o JSON recebido em um objeto conhecido pela API principal do *chatbot*.

API Principal: A API é o componente principal do *chatbot* e realiza a comunicação entre o *WhatsApp for Business* e o *Google Calendar*. O design da API foi criado para ser altamente disponível, escalável e de baixa manutenção, com uma solução orientada pelas premissas do Design Orientado ao Domínio (DDD) e separada em camadas, por meio da arquitetura em camadas descrita a seguir.

1. **Camada de Apresentação (API)**

Esta camada é responsável pela interação com os clientes, expondo *endpoints RESTful* através dos controladores (*controllers*). A camada tem como responsabilidade receber as requisições HTTP, validá-las e encaminhá-las para a camada de aplicação. É composta pelas *Controllers*.

2. **Camada de Aplicação**

É responsável pela lógica das operações através dos serviços e casos de uso (use cases). Tem como função principal implementar a lógica de negócio, coordenar as operações necessárias para atender às requisições recebidas pela camada de apresentação e orquestrar a interação entre a camada de domínio e a de infraestrutura. É composta pelos serviços que fazem comunicação com o WhatsApp e Google *Calendar*, bem como pelos casos de uso que tratam as mensagens e os fluxos de agendamento e cancelamentos.

3. **Camada de Domínio**

É responsável por modelar os objetos que abstraem o contexto de negócio. Tem como função definir as entidades e valores de objetos que representam a essência do negócio. É composta pelos modelos de objetos, extensões de classe, constantes, classes utilitárias para modelagem da aplicação e interfaces de repositórios (portas).

4. **Camada de Infraestrutura**

É responsável por gerenciar a comunicação com o banco de dados. Tem como função implementar os repositórios que interagem com a camada de domínio e fornecer a implementação concreta das interfaces definidas dos repositórios na camada de domínio (adaptadores). É composta pelos repositórios e pelo contexto de comunicação com o banco de dados.

3.4 Implementação

3.4.1 Recebimento de Mensagens

A API principal começa com as mensagens formatadas enviadas do lado da aplicação *serverless*. A integração acontece por meio de um *webhook*, onde as mensagens enviadas pelo *WhatsApp for Business* são consumidas, validadas e formatadas em JSON bem-estruturado.

3.4.2 Processamento e Envio

Assim que a mensagem é recebida pela API principal, ela é imediatamente respondida com um status 200, informando ao *serverless* e à API do *WhatsApp for Business* que a mensagem está sendo tratada devidamente.

Criar este processo de resposta imediata foi fundamental para evitar problemas de repetição de mensagens disparadas pela API da Meta. Quando o *WhatsApp for*

Business dispara o *webhook*, ele aguarda um retorno de que a mensagem foi processada. Caso haja um atraso no processamento, a Meta interpreta isso como um problema e reenviará a mensagem, resultando em mensagens duplicadas caso o processamento da API do *chatbot* leve mais do que alguns milissegundos.

Para contornar essa situação, foi necessário criar uma resposta imediata para a API da Meta assim que o *webhook* e acionado. Em seguida, de forma assíncrona o caso de uso *MessageHandlerUseCase* é chamado para processar a mensagem recebida. O *MessageHandlerUseCase* atua como um roteador de mensagens, tratando as respostas em texto adequadamente e acionando os serviços necessários para agendamento e cancelamento.

O *MessageHandleUseCase* desempenha um papel fundamental ao listar os horários disponíveis para agendamento. Optou-se por sempre apresentar ao cliente, no máximo, seis horários disponíveis na agenda dos empreendedores. Assim, caso haja mais de seis horários disponíveis, o caso de uso seleciona os dois primeiros, os dois últimos e dois intermediários. Essa abordagem, combinada com a divisão do dia em períodos (manhã, tarde, noite ou dia todo), garante que a listagem de horários enviada ao cliente seja bem distribuída, evitando que ele receba uma lista extensa e confusa ao fazer sua escolha para agendamento.

Os acionamentos realizados pelo *MessageHandlerUseCase* podem incluir a chamada do caso de uso *ResponseActionHandlerUseCase*, que funciona como um roteador de ações concretas do chatbot, tais como realizar agendamentos ou cancelar um atendimento. O *ResponseActionHandlerUseCase* por sua vez pode chamar o *AppointmentCancellerUseCase*, que lida com todos os processos de cancelamento de atendimentos, ou o *AppointmentSchedulerUseCase,* responsável por todos os processos de agendamento de serviços.

Os serviços de *AppointmentService*, *ClientService*, *GoogleService*, *GoogleAuthService*, *LogMessageService*, *LogService, MessageService, WhatsAppConfigService*, e *WhatsAppService* podem ser acionados a qualquer momento por um dos casos de uso, conforme necessidade da ação requerida. Esse design garante que todas as interações com o usuário sejam processadas de forma eficiente e que a integridade das operações de agendamento e cancelamento seja mantida.

3.4.3 Serviços da aplicação

O *AppointmentService* é crucial para a gestão de agendamentos, lidando com operações como criação, atualização, redefinição e recuperação de compromissos. Ele interage com o repositório de agendamentos (*IAppointmentRepository*) para buscar agendamentos existentes, criar quando necessário, atualizar detalhes e redefinir dados. O serviço também valida e configura datas de compromisso, garantindo que as entradas dos clientes sejam válidas e ajustadas conforme necessário.

O *AppointmentService* também faz integração dos compromissos ao *Google Calendar*, assegurando que os agendamentos sejam sincronizados corretamente. Ele converte listas de horários livres para JSON, facilitando a comunicação e a sincronização dos dados.

O Serviço de cliente - *ClientService* é fundamental para gerenciar informações e operações relacionadas aos clientes. Ele interage com o repositório de clientes (*IClientRepository*) para recuperar, inserir e atualizar dados. O serviço permite buscar clientes pelo número de telefone e, caso não encontre, insere um novo registro no sistema, esta solução foi desenhada a fim de não exigir cadastro inicialmente na versão MVP. Além disso, o *ClientService* é responsável por atualizar as informações dos clientes e redefinir seus dados, garantindo que todas as interações e estados de operação sejam devidamente gerenciados.

O Serviço de Autenticação Google - *GoogleAuthService* gerencia a autenticação e obtenção de *tokens* de acesso para interações seguras com a API do Google. Utilizando as credenciais de uma conta de serviço, este serviço verifica se há um *token* de acesso válido em cache e, caso contrário, solicita um novo *token* de acesso.

Uma vez obtido o *token*, ele é armazenado em *cache* junto com o horário de obtenção para reutilização eficiente até que expire. Além disso, o *GoogleAuthService* facilita a realização de requisições à API do Google, anexando o token de acesso aos cabeçalhos de autorização das requisições HTTP, garantindo que todas as comunicações com os serviços do Google sejam realizadas de forma segura e autenticada

O Serviço Google - *GoogleService* gerencia a interação com o Google *Calendar*, utilizando *tokens* de acesso obtidos pelo *GoogleAuthService* para garantir

comunicações seguras e autenticadas. Ele permite criar, deletar e consultar eventos no *Google Calendar*

Utilizando credenciais *OAuth2*, ele cria uma instância do *CalendarService* para executar operações como inserção e exclusão de eventos e consultas de disponibilidade. A funcionalidade de consulta de disponibilidade (*GetFreeBusyInfoAsync*) permite verificar a ocupação de múltiplos calendários entre dois horários, retornando à disponibilidade de cada calendário consultado. Este serviço é essencial para sincronizar agendamentos do *chatbot* com o *Google Calendar* e proporcionar uma gestão eficiente e automatizada dos compromissos.

O Serviço de Registro de Mensagens - *LogMessageService* é responsável por registrar e armazenar logs de mensagens enviadas e recebidas pelo *chatbot*. Utilizando o repositório de logs de mensagens (*ILogMessageRepository*) e a unidade de trabalho (*IUnitOfWork*), o serviço assegura que cada mensagem é registrada com detalhes completos, incluindo o remetente, o corpo da mensagem e o estado (enviado ou recebido). Ao receber uma mensagem de texto enviada pelo *chatbot*, o serviço cria um log com as informações pertinentes e insere no repositório, garantindo que todas as comunicações sejam auditáveis. Similarmente, ao receber uma mensagem, o serviço registra os detalhes da mensagem recebida, assegurando a rastreabilidade e a integridade das operações de comunicação. Este serviço é crucial para entender o comportamento do cliente, analisar dificuldades de agendamentos e produzir uma base de dados que será utilizada na etapa de validação do produto.

O Serviço de Registro de Logs - *LogService* gerencia o registro de logs de diferentes tipos de eventos e mensagens no sistema. Ele interage com o repositório de logs (*ILogRepository*) e utiliza a unidade de trabalho (*IUnitOfWork*) para garantir que cada *log* é armazenado de forma consistente e confiável. O serviço permite a inserção de *logs* de erro (*InsertErrorLog*) e logs informativos (*InsertInformationLog*), associando cada *log* a um cliente específico. Ao registrar um evento, o serviço cria uma instância de *LogDbo* com os detalhes do cliente e a mensagem, e então insere o *log* no repositório, assegurando que todas as operações sejam salvas na base de dados.

Serviço de Mensagens - *MessageService* gerencia a recuperação e manipulação de mensagens baseadas nas interações dos clientes com o *chatbot*. Ele interage com

o repositório de mensagens (*IMessageRepository*) e o serviço de agendamentos (*IAppointmentService*) para fornecer respostas adequadas com base no fluxo de interação atual do cliente. O serviço determina o tipo de fluxo (agendamento, cancelamento ou saudação) em que o cliente está envolvido e, em seguida, recupera a mensagem correspondente da base de dados. Se o cliente não interagiu dentro de um limite de tempo definido ou está no início do fluxo, o serviço reinicializa o estado do cliente e reinicia o processo de interação, garantindo que a comunicação sempre esteja dentro de um contexto específico e impedindo interações longas que poderiam desencadear em tentativas de agendar atendimentos em horários que já não estão disponíveis mais.

O Serviço de Configuração do WhatsApp - *WhatsAppConfigService* é responsável por fornecer as configurações necessárias para a integração com o WhatsApp Business API. Ele utiliza o serviço de configuração (*IConfiguration*) para obter as credenciais e IDs necessários a partir de um arquivo de configuração, como o *appsettings*.json. O serviço retorna uma instância de *WhatsAppBusinessCloudApiConfig*, que contém o ID do número de telefone do *WhatsApp Business*, o ID da conta, o ID do negócio e o token de acesso. Isso garante que todas as interações com o *WhatsApp* Business API sejam autenticadas e configuradas corretamente, proporcionando uma base segura e configurável para as operações do *chatbot*.

O Serviço do *WhatsApp* - *WhatsAppService* gerencia a comunicação entre o chatbot e os clientes através do *WhatsApp* Business API. Ele utiliza o *WhatsAppConfigService* para obter as configurações necessárias para autenticação e configuração da API. O serviço fornece métodos para enviar mensagens genéricas e de texto, encapsulando a lógica de construção e envio das mensagens com o cliente do WhatsApp Business (*WhatsAppBusinessClient*). Além de enviar mensagens, o *WhatsAppService* também registra logs das mensagens enviadas através do *LogMessageService*, garantindo a rastreabilidade e a auditabilidade das comunicações. Em caso de erros durante o envio das mensagens, o serviço captura e propaga as exceções, permitindo um tratamento adequado dos erros.

3.4.4 Repositórios

Os repositórios no sistema realizam a interação com o banco de dados, garantindo que todas as operações de persistência de dados sejam feitas de forma organizada e eficiente. Para garantir o princípio de *Single Responsability* do SOLID, cada repositório comunica apenas com uma tabela do banco de dados *SQLServer*. Os repositórios são: *AppointmentRepository, BusinessRepository, ClientRepository, LogMessageRepository, LogRepository, MessageRepository, ProfessionalRepository, ServiceRepository, StepOptionRepository.*

Os repositórios não implementam regra de negócio e funcionam como CRUDs para integração com o banco. A funcionalidade de cada tabela será descrita abaixo no detalhamento da estrutura do banco de dados.

3.4.5 Banco de Dados

O banco de dados *Sql Server* persiste as informações relevantes sobre agendamentos, usuários e interações. O DDD ajuda a moldar o banco de dados de tal forma que ele represente os conceitos de negócios diretamente, e a manutenção e escalabilidade do sistema sejam fáceis.

O **Anexo I** deste trabalho traz o diagrama arquitetural do banco, bem como os relacionamentos entre cada uma das nove tabelas do sistema. A seguir será detalhada a funcionalidade de cada tabela, seus campos.

1. *BusinessDbo* – Tabela que armazena os dados das empresas dos microempreendedores que fazem uso do sistema.

Estrutura:

- **id**_***business***: Identificador único da empresa (Primary Key).
- **num**_***phone***: Número de telefone da empresa.
- **nam**_***business***: Nome da empresa.

2. ***ClientDbo*** – Tabela que armazena os dados dos clientes que interagem com o sistema. Esta tabela é importante, não só na identificação do cliente, mas também para armazenar o estado e etapa em que o cliente se encontra. A tabela é uma das mais importantes para gerenciar o fluxo de atendimento do *chatbot*.

Estrutura:

- ***id_client***: Identificador único do cliente (*Primary Key*).
- ***id_business***: Identificador da empresa associada ao cliente.
- **num*_phone***: Número de telefone do cliente.
- ***nam_client***: Nome do cliente.
- ***id_next_step***: Próxima etapa no fluxo de interação do cliente.
- ***dt_last_contact***: Data e hora do último contato do cliente.
- ***has_scheduling_in_progress***: Indica se há um agendamento em progresso.
- ***has_canceling_in_progress***: Indica se há um cancelamento em progresso.
- ***id_last_whatsapp_message***: Identificador da última mensagem do *WhatsApp* recebida.

3. ***ProfessionalsDbo*** – Tabela que armazena os dados dos profissionais associados às empresas que utilizam o sistema. Esta tabela é essencial para gerenciar e identificar os profissionais disponíveis em cada empresa, facilitando a alocação e agendamento de serviços conforme a disponibilidade e as informações de contato dos profissionais.

   Estrutura:

   - ***id_professional:*** Identificador único do profissional (*Primary Key*).
   - ***id_business:*** Identificador da empresa à qual o profissional está associado (*Foreign Key*).
   - ***nam_professional:*** Nome do profissional.
   - ***nom_email:*** Endereço de e-mail do profissional.

4. ***ServicesDbo*** – Esta tabela é essencial para gerenciar e identificar os diversos serviços oferecidos por cada empresa, permitindo que os clientes escolham e agendem serviços conforme suas necessidades e preferências, além de facilitar a gestão do tempo e a alocação de recursos.

   Estrutura:

- ***id_service:*** Identificador único do serviço (*Primary Key*).
- ***id_business:*** Identificador da empresa à qual o serviço está associado (*Foreign Key*).
- ***nam_service:*** Nome do serviço.
- ***num_duration_minutes:*** Duração do serviço em minutos.

5. ***MessagesDbo –*** Tabela que armazena as mensagens utilizadas pelo sistema para interação com os clientes. Isso é importante para garantir que a comunicação com os clientes seja mais expressiva e envolvente, utilizando emojis para melhorar a clareza e a emoção transmitida nas mensagens.

   Estrutura:

   - ***id_message****:* Identificador único da mensagem (*Primary Key*).
   - ***id_business****:* Identificador da empresa associada à mensagem (*Foreign Key*).
   - ***id_step****:* Identificador do passo no fluxo de interação ao qual a mensagem pertence.
   - ***vlr_option****:* Valor da opção associada à mensagem.
   - ***vlr_message****:* Conteúdo da mensagem. O tipo do campo é *nvarchar*, o que permite salvar emojis nas mensagens que serão enviadas pelo WhatsApp.
   - ***id_flow_type****:* Identificador do tipo de fluxo ao qual a mensagem pertence, sendo 0 para fluxo de saudação, 1 para fluxo de agendamento, 2 para fluxo de cancelamento e 9 para fluxo de falar com atendente.

6. ***AppointmentDbo –*** Tabela que armazena os dados dos agendamentos realizados pelos clientes através do sistema. Permite gerenciar todos os aspectos dos agendamentos, desde a criação e confirmação até a eventual cancelamento, garantindo uma organização eficiente e detalhada dos compromissos entre clientes, profissionais e serviços oferecidos.

Estrutura:

- ***id_appointment:*** Identificador único do agendamento (*Primary Key*).
- ***id_client:*** Identificador do cliente que fez o agendamento (*Foreign Key*).
- ***id_business:*** Identificador da empresa onde o agendamento foi feito (*Foreign Key*).

- ***dt_appointment****:* Data e hora do agendamento.
- ***flag_morning:*** Indica se o agendamento é pela manhã (booleano).
- ***flag_evening:*** Indica se o agendamento é à tarde (booleano).
- ***flag_night:*** Indica se o agendamento é à noite (booleano).
- ***flag_any_time:*** Indica se o agendamento pode ser em qualquer horário (booleano).
- ***flag_scheduled*****:** Indica se o agendamento está confirmado (booleano).
- ***id_professional:*** Identificador do profissional escolhido para o agendamento, o cliente pode solicitar ser atendido por qualquer profissional disponível também.
- ***id_service:*** Identificador do serviço a ser realizado no agendamento.
- ***json_available_times****:* Horários disponíveis em formato JSON.
- ***id_scheduled_professional:*** Identificador do profissional agendado.
- ***id_scheduled_calendar:*** Identificador do calendário do microempreendedor agendado.
- ***id_google_event:*** Identificador do evento no *Google Calendar*.
- ***flag_canceled:*** Indica se o agendamento foi cancelado (booleano).

7. ***LogMessagesDbo*** – Tabela que armazena os registros de mensagens enviadas e recebidas pelo sistema. Essa tabela é crucial para manter um histórico detalhado das comunicações via *WhatsApp*, permitindo monitorar e auditar as interações do *chatbot* com os clientes. Cada log inclui informações essenciais como o número de telefone do cliente, o conteúdo da mensagem, o identificador da mensagem no *WhatsApp*, o tipo de transação e a data e hora do log. Isso garante a rastreabilidade e a transparência das operações de comunicação do sistema e tornando possível a análise e validação do *chatbot*.

Estrutura:

- ***id_log:*** Identificador único do log de mensagem (Primary Key).
- ***num_client_phone:*** Número de telefone do cliente.
- ***vlr_message*****:** Conteúdo da mensagem.
- ***id_whatsapp_message:*** Identificador da mensagem no WhatsApp.
- ***desc_transaction_type:*** Tipo de transação (enviada ou recebida).

- ***dt_log:*** Data e hora do log da mensagem.

8. ***LogsDbo –*** Tabela que armazena os registros de eventos e informações gerais do sistema. Essa tabela é crucial para monitorar e registrar eventos significativos e informações gerais do sistema, permitindo a auditoria e o diagnóstico de operações. Cada log inclui informações como o identificador do cliente, o tipo do log, a mensagem detalhando o evento ou a informação, e a data e hora do registro. Isso assegura que todas as operações e eventos importantes sejam documentados, facilitando a manutenção e a resolução de problemas no sistema.

Estrutura:

- ***id_log***: Identificador único do log (*Primary Key*).
- ***id_client****:* Identificador do cliente associado ao log (*Foreign Key*).
- ***type_log****:* Tipo do log (e.g., *Information*, *Error*).
- ***vlr_log_message****:* Mensagem do log, detalhando o evento ou a informação.
- ***dt_log***: Data e hora em que o log foi registrado.

9. ***StepOptionsDbo*** – Tabela que armazena as opções de passos nos fluxos de interação do sistema. Essa tabela é crucial para definir as opções disponíveis em cada passo dos fluxos de interação do sistema, permitindo uma navegação estruturada e funcional através das diferentes etapas, como agendamento, remarcação, cancelamento e suporte ao cliente.

Estrutura:

- ***id_step:*** Identificador único do passo no fluxo (*Primary Key*).
- ***id_business:*** Identificador da empresa associada ao passo (*Foreign Key*).
- ***vlr_option:*** Valor da opção associada ao passo.
- ***has_appointment:*** Indica se o passo envolve agendamento (booleano).
- ***has_rescheduling:*** Indica se o passo envolve remarcação de agendamento (booleano).
- ***has_appointment_cancellation:*** Indica se o passo envolve cancelamento de agendamento (booleano).

- **h*as_customer_support:*** Indica se o passo envolve suporte ao cliente (booleano).

3.4.6 Segurança e Autenticação

As mensagens recebidas são autenticadas com a ajuda de *tokens* de acesso que são fornecidos pelo *WhatsApp for Business*. Da mesma forma, todas as ações feitas no *Google Calendar* são autenticadas por *OAuth2*, para que apenas os usuários permitidos tenham a capacidade de acessar e editar agendas.

3.4.7 Fluxo de Dados

- Mensagem Recebida: O usuário envia uma mensagem para o *WhatsApp* da pequena empresa;
- *Webhook*: O *webhook* do aplicativo serverless envia a mensagem. Estruturação: A mensagem está autenticada e formatada no estilo canônico de JSON;
- Roteamento: O *request* formatado vai para a API principal, que o redireciona para o serviço relevante;
- Interação com APIs: A API principal interage com o *Google Calendar* e com o *WhatsApp for Business* para processar a solicitação do usuário;
- Resposta ao Cliente: A API principal responde de volta ao cliente via *WhatsApp* com a confirmação da reserva, o cancelamento ou o link de contato.

4 VALIDAÇÃO DO MVP

Durante a Monografia de Sistemas de Informação I, foi elaborado um questionário a fim de compreender melhor a rotina do público-alvo: microempreendedores que trabalham com serviços de agendamento e possuem agenda saturada com atendimentos.

Entre os respondentes deste questionário, foi selecionada a empresa Vitor Barbearia para realizar a validação do MVP. Além de se enquadrar perfeitamente no público-alvo definido, o proprietário, Vitor Morais, se dispôs a realizar a validação, pois se encontrava em um cenário de expansão em que um software de agendamento

poderia se tornar um aliado de grande valor. A seguir serão apresentadas as características e o contexto geral da empresa citada.

4.1 Descrição da empresa

A Vítor Barbearia é uma empresa nova, que nasceu há dois anos. Fundada pelo empreendedor Vitor Morais, um jovem de 24 anos que decidiu empreender ao abrir o próprio negócio. A empresa conta com mais um barbeiro, o Thomas, e está localizada no centro da cidade de Araxá-MG. Apesar de ser um empreendimento novo, a barbearia está em expansão, com mudança programada para um espaço maior e com adicional de um terceiro profissional no quadro de barbeiros.

Em se tratando de agendamentos, o proprietário, antes de iniciar a validação do *chatbot*, recebia mensagens e ligações de clientes e controlava todos os agendamentos em duas agendas de papel: uma para os atendimentos próprios e outra para os atendimentos do Thomas. Apesar de os profissionais realizarem serviços apenas por meio de agendamentos, é comum que clientes transeuntes no comércio local entrem no estabelecimento e agendem horários de última hora, caso algum barbeiro esteja disponível no momento.

Os serviços realizados pela barbearia incluem: corte de cabelo, barba, selagem e sobrancelha. Os serviços têm duração de 30 minutos cada, à exceção de sobrancelha que normalmente leva menos de dez minutos. O público da empresa é majoritariamente masculino, embora existam clientes do sexo feminino que realizam corte ou mesmo levam os filhos para cortarem o cabelo.

4.2 Preparativos para validação

Foram realizadas três reuniões com os microempreendedores a fim de explicar o funcionamento do *chatbot*, bem como do Google *Calendar*, principal aplicativo que seria a interface do barbeiro com os agendamentos realizados. Durante a reunião foi realizada a instalação do aplicativo Google Agenda (*Calendar*) no celular de cada barbeiro e foi ministrada uma miniaula explicativa do funcionamento do app.

Foi criada uma conta do google *calendar* para o barbeiro Thomas e, em seguida, um calendário google foi criado. Foi realizado o compartilhamento do

calendário com o proprietário da barbearia e com o autor do presente trabalho, para controle e acompanhamento dos agendamentos realizados pelo *chatbot*. O mesmo procedimento foi feito com o Vitor, de modo que o proprietário da barbearia e o autor deste trabalho teriam visibilidade e controle de todas as agendas.

Foi disponibilizado para a empresa um novo número vinculado ao *WhatsApp for Business* do *chatbot*. Os empreendedores foram orientados a programar uma mensagem automática, instruindo os clientes que desejassem realizar agendamentos a clicar em um link que os direcionaria ao *WhatsApp* do *chatbot*. Esta abordagem foi utilizada, pois caso o número oficial da barbearia fosse cadastrado diretamente no *chatbot*, os empreendedores perderiam acesso ao histórico de conversas dos clientes. Como a validação do produto ainda era experimental, optou-se por utilizar um novo número temporariamente, para melhor compreender o comportamento dos clientes e minimizar problemas de comunicação da barbearia com os clientes caso o *chatbot* fosse descontinuado.

As reuniões e interações foram necessárias para garantir uma boa adaptação por parte dos profissionais ao abandonarem a agenda de papel e começarem a controlar os atendimentos em um aplicativo no *smartphone*. Por meio do *app* google agenda, os barbeiros conseguiam agendar atendimentos manualmente para clientes que abordavam os profissionais dentro da barbearia, bem como excluir algum agendamento ou bloquear a agenda em caso de compromissos pessoais, feriados e folgas. Como o *chatbot* tem integração direta com o google agenda, cada compromisso inserido ou excluído pelo barbeiro no *app* do google refletia instantaneamente na disponibilidade de horários ofertados para os clientes via *WhatsApp*.

4.3 Dados coletados

O *chatbot* esteve em funcionamento de 28/05/2024 a 18/06/2024. Durante as três semanas de validação, o chat foi acessado por 121 clientes distintos. No total, foram realizadas 191 interações, categorizadas da seguinte forma:

- 108 agendamentos;
- 9 cancelamentos;

- 5 foram redirecionados para atendente da barbearia;
- 3 registraram erro no *chatbot*;
- 30 não encontraram horário disponível;
- 10 encontraram horários disponíveis que não eram compatíveis com a disponibilidade do cliente;
- 25 abandonaram o *chatbot* sem motivo aparente;
- 1 tentou agendar dois cortes ao mesmo tempo e não conseguiu;

A seguir, será detalhada cada uma das categorizações apresentadas.

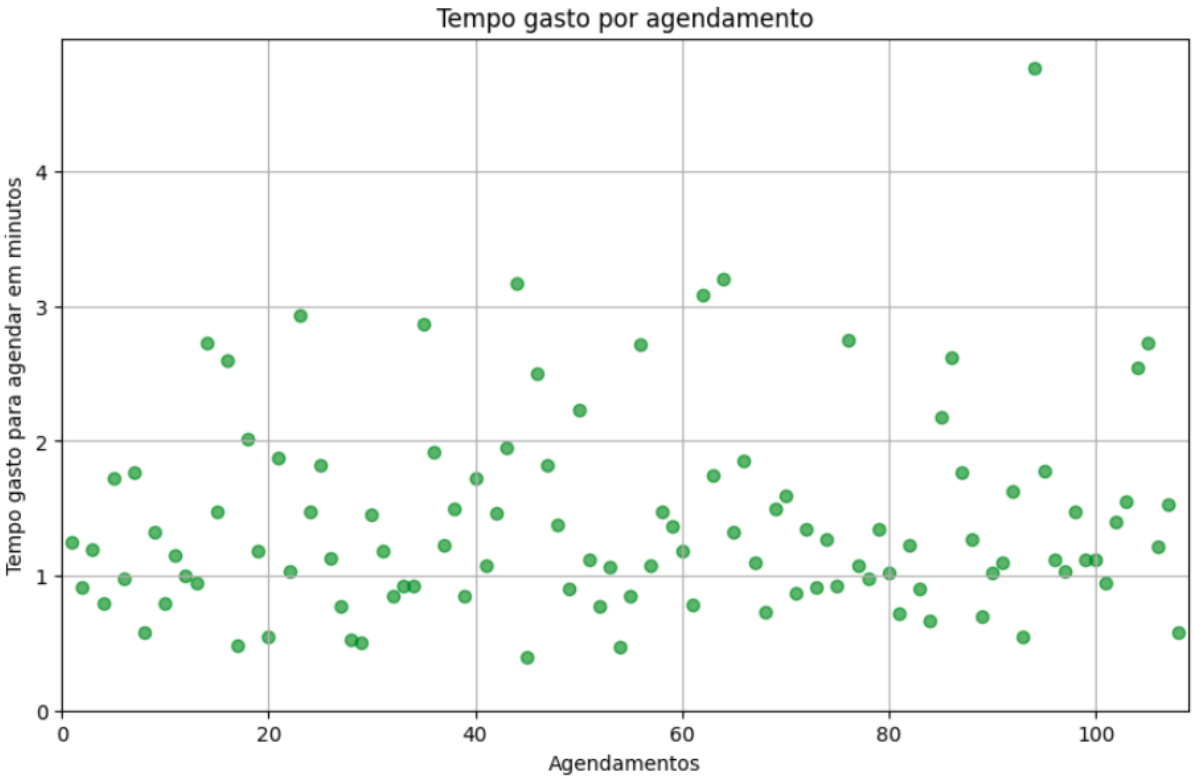
4.3.1 Agendamentos

Os 108 agendamentos realizados tiveram seu tempo de interação registrados nos logs da tabela *log_messages*. Esses dados foram organizados em uma planilha para análise, e um gráfico de dispersão foi construído para facilitar a interpretação dos dados.

Para computar o tempo de cada interação até o agendamento ser concluído, subtraiu-se o horário exato do agendamento realizado pelo horário da primeira mensagem enviada pelo cliente no *chatbot*. O tempo médio gasto em segundos para o agendamento foi de 1,41 minuto, ou seja, aproximadamente 1 minuto e 25 segundos.

O gráfico 1 de dispersão mostra o tempo gasto em minutos para cada agendamento realizado via *chatbot*.

Gráfico 1 – Tempo dos agendamentos



Fonte: Autor (2024)

O ponto *outlier* que ficou acima de quatro segundos refere-se a um cliente que demorou para responder ao *chatbot*, mas conseguiu concluir o agendamento de forma fluída e com poucas interações.

4.3.2 Cancelamentos

O cancelamento entrou no MPV como solicitação do cliente que requereu a feature como condição necessária ao início do processo de validação do produto. Ao todo foram realizados apenas seis cancelamentos que tiveram tempo médio de 0,42 minuto, ou seja, 25 segundos para serem realizados.

O gráfico 2 apresenta a dispersão dos cancelamentos em função do tempo gasto em minutos para a realização.

Gráfico 2 - Tempo do cancelamento

Fonte: Autor (2024)

O fluxo de cancelamento é simples (vide item 4.2) e por isso tem tempo de interação bem menor que o fluxo de agendamento.

4.3.3 Redirecionamento para atendente da barbearia

O número de redirecionamentos foi baixo porque os clientes eram redirecionados diretamente do chat do profissional. Conforme mencionado no item 7.2, os clientes entraram em contato no número oficial da barbearia e eram redirecionados para o *chatbot*. Dessa forma, caso não conseguissem o contato desejado com o *chatbot*, bastava entrar em contato com número oficial da barbearia novamente.

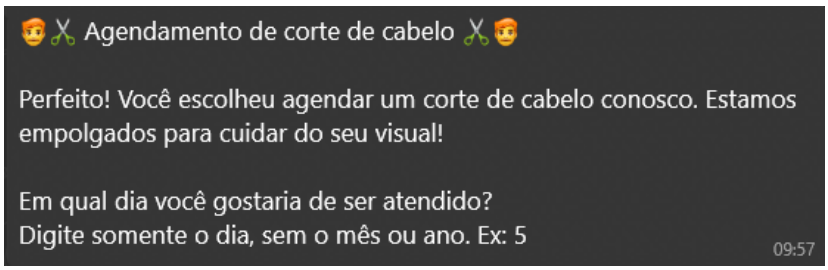
4.3.4 Registro de erro no chatbot

O *chatbot* também foi programado para enviar mensagens genéricas de erro. Caso alguma exceção genérica fosse lançada pela aplicação o chat informava: "*Ops! Ocorreu um erro inesperado! Basta enviar uma nova mensagem aqui para iniciarmos seu atendimento novamente!*" Dessa forma o atendimento era reiniciado, e o cliente poderia tentar realizar o agendamento novamente.

Essa abordagem foi utilizada para lidar com cenários não previstos e permitir uma análise posterior para entender possíveis erros e tratá-los no futuro. Os três erros ocorridos foram devidos a três clientes que tentaram agendar atendimento para um horário que estava no passado. Nos casos analisados, os clientes entraram em contato no período da noite tentando realizar agendamento para o mesmo dia, no período da manhã, o que gerou o erro.

Esse cenário de erro, mostrou que havia oportunidade de melhoria na forma de comunicar o dia em que o cliente deseja realizar atendimento. A figura 1 mostra como o chat solicitava o dia de agendamento.

Figura 1 - Padrão de agendamento



Fonte: Autor (2024)

Mesmo a mensagem estando clara, alguns clientes podem se confundir e não saber que dia cinco seria uma quarta-feira, por exemplo. Uma oportunidade de melhoria identificada seria enviar uma mensagem de confirmação informando qual dia da semana corresponde ao dia X escolhido e perguntar ao cliente se esse dia realmente é o desejado para o atendimento.

O dono da barbearia informou que um cliente pensou que o dia informado se referia ao dia da semana, e não ao dia do mês. O cliente achou que o dia dois seria segunda-feira, o dia três seria terça-feira, e assim por diante. Esse mal-entendido gerou um erro de agendamento, sendo um cenário muito difícil de prever antes da implementação.

4.3.5 Não encontraram horário disponível

Observou-se que as 30 interações que não encontraram horário disponível ocorreram devido a tentativas de agendamento em horários muito próximos do horário desejado para atendimento e em fins de semana, que são datas de alta demanda.

Alguns agendamentos também foram tentados para dias em que a barbearia não funciona, como, por exemplo, segunda-feira. A tabela 1 apresenta a relação dos dias da semana que estavam indisponíveis quando os clientes tentaram realizar seus agendamentos.

Tabela 1 - Relação de indisponibilidade semanal

Dia da semana	Tentativas de agendamento após agenda ser preenchida
Sexta-feira	10
Sábado	10
Segunda-feira	6
Quinta-feira	2
Quarta-feira	1
Terça-feira	1

Fonte: Autor (2024)

4.3.6 Horários disponíveis incompatíveis com a disponibilidade dos clientes

A categorização das interações foi essencial para verificar se os clientes não agendaram porque realmente não havia horários compatíveis com a disponibilidade deles ou se o *chatbot* não estava listando os horários de forma adequada (vide implementação no item 4.2.2).

Observou-se que os clientes que não encontraram horários disponíveis haviam recebido menos de cinco opções na listagem fornecida. Isso confirma que, de fato, não havia horários compatíveis com a disponibilidade dos clientes, e possibilitou o descarte da hipótese de que o *chatbot* tenha falhado em apresentar a agenda disponível para agendamento.

4.3.7 Abandono do chat sem motivo aparente

Por se tratar de um período inicial de validação, entende-se que os clientes da barbearia também passaram por uma adaptação para agendar serviços pelo *chatbot*. O tempo de três semanas pode não ter sido suficiente para alguns clientes, visto que o corte de cabelo normalmente é feito uma vez por mês.

Os empreendedores relataram que alguns clientes com dificuldade em lidar com recursos tecnológicos ficaram receosos de interagir com o *chatbot*, o que pode ter causado o abandono da interação sem motivo aparente. Segundo os empreendedores, idosos foram um grupo que enfrentou maior dificuldade de interação. Eles esperavam que o chat entendesse texto em linguagem natural, sem perceber que havia uma automatização no agendamento.

Durante o período de validação, foi identificado esse número de evasão durante o serviço de agendamento, e um relatório manual estava sendo gerado e enviado para o dono da barbearia, com o objetivo de prover uma listagem de clientes para que o barbeiro entrasse em contato e garantisse que o agendamento fosse concluído. Para facilitar o monitoramento foi criado um *dashboard* na ferramenta *Grafana*, que executava *queries* previamente configuradas no banco de dados e apresentava de forma gráfica as informações de:

- quantidade de agendamentos realizados no dia;
- quantidade de pessoas que chamaram no *chatbot*, mas não fizeram um agendamento (evasão);
- número de erros da aplicação;
- listagem com dados da evasão, tais como nome, *link* para contato do cliente e horário de contato;
- histórico da interação do último cliente a chamar o *chatbot*;

Esses dados eram disponibilizados para o empreendedor de forma a mapear de perto o comportamento dos clientes evitando assim possível perda de clientes durante o período de adaptação. O *dashboard* gerado pode ser visto no **Anexo II** deste trabalho.

4.3.8 Tentativa sem sucesso ao agendar dois cortes par ao mesmo horário

Uma cliente tentou agendar dois cortes, um para o marido e outro para o filho, conversando com o *chatbot* em linguagem natural em vez de usar as opções numéricas fornecidas pelo bot. Na versão MVP, não foi previsto o agendamento de mais de um serviço ao mesmo tempo. O cliente precisava chamar o *chatbot*, agendar um horário, esperar a confirmação e, em seguida, chamar o *chatbot* novamente para

agendar outro horário. A possibilidade de agendar mais de um atendimento seria abordada em uma segunda versão do sistema.

4.4 *Feedbacks* coletados

O *feedback* dos barbeiros foi muito positivo. Vitor, dono da barbearia, notou uma melhora significativa na interação com os serviços de agendamento. Os clientes também trouxeram *feedbacks* positivos, destacando como o agendamento ficou prático e fácil por meio do *chatbot*, o que trouxe um ar mais profissional à barbearia.

Vitor notou que, ao permitir que os clientes escolhessem seus próprios horários por meio do *chatbot*, a agenda da barbearia começou a ficar mais cheia no início da semana. Ele acredita que, ao fazer os agendamentos, tendia a marcar mais serviços para o fim de semana, presumindo que essas datas eram preferidas. Com a autonomia, os clientes começaram a agendar em dias como terça e quarta-feira, resultando em menos congestionamento nos fins de semana.

Outro *feedback* positivo foi a facilidade de controlar os agendamentos por parte dos profissionais. Antes, cada barbeiro tinha sua agenda de papel individual e o dono do estabelecimento controlava ambas. Com a digitalização dos agendamentos, os atendimentos ficaram mais acessíveis, facilitando o dia a dia.

Como *feedback* negativo, foi citado o fato de ter que clicar em um *link* separado para sair do *WhatsApp* oficial da barbearia e acessar o *chatbot*. Isso foi necessário para o processo de validação, mas em tempos de golpes cibernéticos, pode gerar desconforto e insegurança para os clientes que desejam agendar.

Por fim, devido ao fato de o *chatbot* prover apenas a interface de agendamento e cancelamento, o microempreendedor sentiu falta de um *software* capaz de controlar as finanças e o fluxo de caixa da empresa. Como a barbearia está em expansão, o empreendedor optou por migrar para um *software* mais robusto e completo, que consiga controlar as comissões dos barbeiros e gerar relatórios de fluxo de caixa. Embora o novo *software* não tenha a opção de *chatbot* e o agendamento seja feito por meio de uma aplicação web separada, o empreendedor acredita que ele suprirá melhor as necessidades da nova fase da Vitor Barbearia.

5 CONSIDERAÇÕES FINAIS

A implementação do MVP foi um sucesso em diversos aspectos. O produto foi desenvolvido utilizando boas práticas de programação, com interfaces bem projetadas e alta disponibilidade garantida por uma infraestrutura robusta na *Azure*, além de monitoramento em tempo real pelo *Grafana*. Embora seja apenas um MVP, o produto foi regido por uma arquitetura escalável, aplicando os princípios SOLID e DDD. O baixo índice de erros da aplicação mostra a qualidade do software entregue, o que reflete diretamente na experiência do usuário.

Quanto a validação pelo mercado foi possível observar que a ideia de um *chatbot* para agendamento foi muito bem aceita pelos clientes, devido à praticidade na interação. As respostas do *chatbot*, bem como o acionamento da api *Google Calendar* levam milissegundos para ocorrer. A média de um minuto e vinte e cinco segundos para se agendar um atendimento mostra a qualidade do diálogo e do entendimento entre chatbot e cliente. Os feedbacks positivos também fomentam a eficácia da ideia e o valor que um agendador automático pode trazer para o negócio.

Apesar do êxito inicial, a migração do microempreendedor para uma solução mais específica para barbearias mostra a inviabilidade do MVP como foi idealizado. Construir um produto que atenda diversas áreas de agendamento, como barbearias, clínicas de estética, manicures, e psicólogos, é interessante para a fábrica de software, pois tem o potencial de alcançar um maior número de empresas com um único produto. No entanto, para o microempreendedor, essa abordagem é menos atraente, já que existem muitos *players* no mercado que oferecem soluções completas e específicas para cada nicho, a preços acessíveis e com funcionalidades adaptadas às particularidades de cada área.

Embora o estudo tenha impactado positivamente 121 clientes, a interação foi com apenas um microempreendedor. O cenário da Vitor Barbearia não é representativo de todos os empreendedores do público-alvo. Mesmo assim, ficou claro que o MVP gerou valor para o mercado, mas a longo prazo, pode não se sustentar na preferência dos microempreendedores devido aos fatores mencionados.

Uma abordagem ideal seria construir um *chatbot* vinculado a uma solução mais robusta e nichada para cada área de prestação de serviços. Dessa forma, o MVP estaria integrado a um software completo e específico para cada mercado, com o diferencial de possuir um *chatbot* que oferece todos os benefícios descritos nesta monografia.

6 REFERÊNCIAS BIBLIOGRÁFICAS

Evans, Eric. **Domain-Driven Design: Tackling Complexity in the Heart of Software.** Addison-Wesley, 2004.

Google Developers. Google Calendar API Documentation.

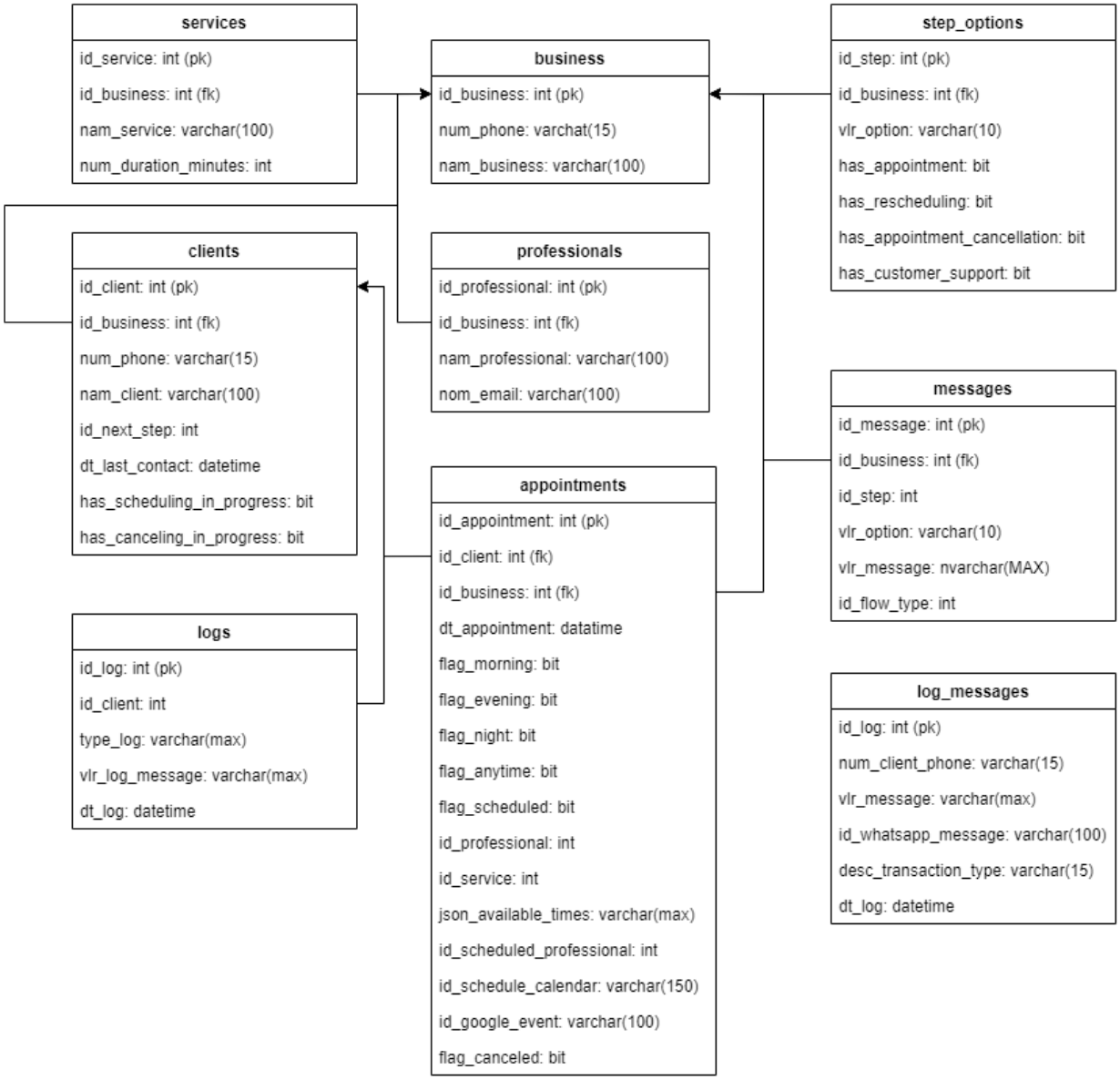
Ries, Eric. **The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses**. Crown Business, 2011.

VALENTE, Marco Tulio. **Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade**. Editora Independente, 2020.

WhatsApp Inc. WhatsApp Business API Documentation.

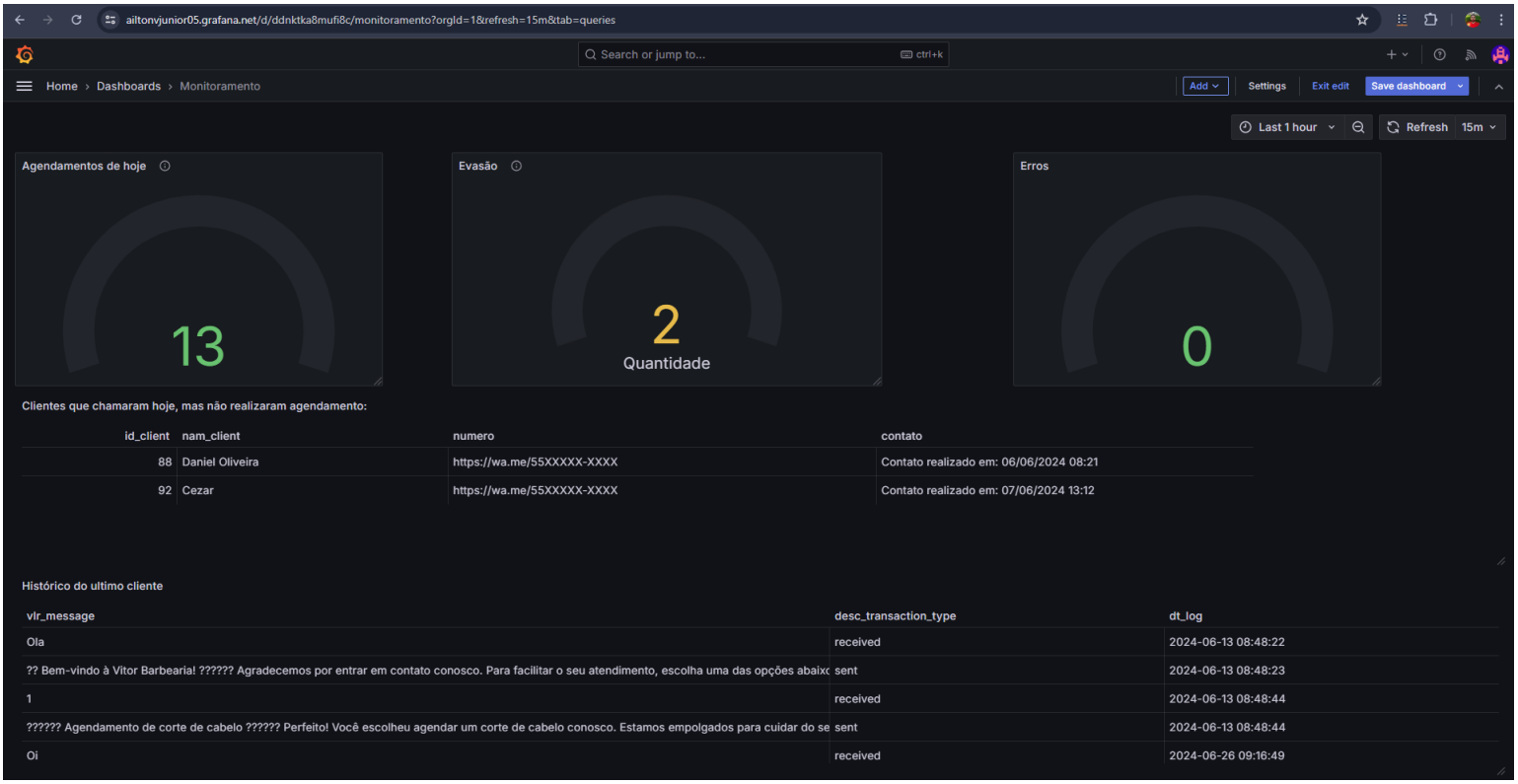
7 ANEXOS

Anexo I: Diagrama arquitetural do banco de dados



Fonte: Autor (2024)

Anexo II: *Dashboard* Grafana para acompanhamento diário de agendamentos e evasão



Fonte: Autor (2024)