# *BeLight Bible*

## *Manual do Programador*

TGPSI23R

Juan Loza – 2223223

BeLight Bible é uma aplicação de estudo bíblico, que permite a leitura estruturada da Bíblia, gestão de planos de leitura e interação personalizada com os versículos. O projeto integra funcionalidades de grifo, anotações e explicações por inteligência artificial, armazenando as interações em SQL Server.

Focado na usabilidade e no apoio à educação espiritual, BeLight Bible pretende ser uma ferramenta moderna para estudantes e grupos de estudo que procuram um contacto mais organizado e enriquecedor com as Escrituras.

# Índice



# 1 - Introdução

O projeto desenvolvido é a aplicação BeLight Bible. Esta aplicação foi criada para permitir a leitura estruturada da Bíblia completa com áudio, com funcionalidades de interação como grifos/destaques, anotações personalizadas, compartilhar versículos, obter explicações por inteligência artificial através de uma LLM, Planos de Leitura Bíblica Personalizados e uma opção para uma Bíblia Infantil com Histórias Ilustradas. Utilizando a linguagem C# com Windows Forms, a aplicação conecta-se a uma API para obter os textos bíblicos e utiliza uma base de dados SQL Server para guardar todas as interações dos utilizadores.

A escolha deste tema justifica-se pela oportunidade de desenvolver uma ferramenta educativa que alia a tecnologia ao estudo espiritual, incentivando a leitura regular e a reflexão, de forma moderna e acessível.

# 2 - Tecnologias e recursos

O desenvolvimento do projeto recorreu a um conjunto de tecnologias, linguagens de programação, bibliotecas e ferramentas específicas, selecionadas com base nos requisitos funcionais, na performance esperada e na facilidade de integração entre os componentes. Abaixo estão descritas todas as tecnologias e recursos utilizados.

## 2.1 - Tecnologias Utilizadas

- **Linguagem de Programação:**
    - **C# (.NET Framework)** – Linguagem principal utilizada para o desenvolvimento da aplicação desktop. Permitiu a integração com bases de dados, criação de interfaces gráficas e chamadas HTTP a APIs externas.
- **Interface Gráfica:**
    - **Windows Forms** – Framework da Microsoft utilizada para a construção da interface gráfica da aplicação.
    - **MaterialSkin** – Biblioteca open source baseada no Material Design da Google, utilizada para tornar a interface mais moderna e visualmente atrativa, oferecendo componentes estilizados como botões, caixas de texto e abas.
- **Base de Dados:**
    - **SQL Server (versão utilizada: SSMS 20)** – Utilizado como sistema de gestão de base de dados. Permitiu a estruturação e o armazenamento de dados como utilizadores, anotações, versículos destacados, histórico de leitura, entre outros.
- **Linguagem Suplementar:**
    - **Python com biblioteca GTTS (Google Text-to-Speech)** – Utilizado para gerar ficheiros de áudio com leitura natural dos capítulos bíblicos.
- **Consumo de APIs:**
    - Foram utilizadas chamadas HTTP através da biblioteca HttpClient para obter dados como livros, capítulos e versículos de uma API externa da Bíblia. O conteúdo recebido foi tratado com a biblioteca Newtonsoft.Json para desserialização eficiente de dados JSON, também utilizado para o processamento dinâmico das mensagens geradas em tempo real (modo streaming) pela LLM.

## 2.2 Bibliotecas e Recursos Adicionais

Algumas bibliotecas adicionais utilizadas ao longo do desenvolvimento incluem:

- System.Speech.Synthesis – Para a síntese de voz dentro do próprio C#, como alternativa ou complemento ao áudio gerado com Python.
- Newtonsoft.Json e Newtonsoft.Json.Linq – Para leitura e manipulação de dados em formato JSON.
- NAudio.Wave – Biblioteca para reprodução de áudio, permitindo que os ficheiros gerados ou fornecidos sejam reproduzidos dentro da aplicação.
- System.Data.SqlClient e System.Data.Entity – Para comunicação com a base de dados SQL Server.

- System.Windows.Forms, System.Drawing, System.Drawing2D – Para criação e personalização de elementos gráficos.
- System.Diagnostics, System.IO – Para manipulação de ficheiros e processos internos.

### 2.3 Recursos Recomendados

**Hardware:**

O projeto foi desenvolvido e testado num computador com as seguintes especificações mínimas recomendadas:

- **Processador:** Intel Core i5 ou superior
- **Memória RAM:** 8 GB
- **Armazenamento:** Pelo menos 1 GB de espaço livre
- **Sistema Operativo:** Windows 10 (64 bits) ou superior
- **Ligação à internet:** Necessária para chamadas às APIs externas (Bíblia e modelo de linguagem).

## 3 – Implementação

### 3.1 Cronograma de desenvolvimento do projeto

| Período | Fase | Principais Tarefas |
|---|---|---|
| 12 - 13 maio | Início do Projeto | Criação do repositório, arquivos iniciais, estrutura básica do projeto. |
| 14 - 19 maio | Bíblia Infantil e Leitor Bíblico | Tela completa da aba Bíblia para crianças e adultos. |
| 20 - 28 maio | Funcionalidades do Chatbot | Adição de 'Explicar', copiar texto, melhorias visuais e semânticas. |
| 02 - 04 junho | Versículo do Dia e Anotações | Cards de versículo do dia, salvar e compartilhar, melhorias na tela de anotações. |
| 05 - 09 junho | Grifos e Visualização | Visualização de grifos, clique em versículos, menu de leitura, ajustes de layout. |

| | | |
|---|---|---|
| 10 - 11 junho | Planos de Leitura - Início | Criação da aba 'Planos de Leitura', cards no banco de dados, tela de admin com CRUD parcial. |
| 12 - 15 junho | Planos de Leitura - Expansão | Formulários de planos, associação com utilizadores, melhorias ao chatbot com LLaMA3-70B |
| 16 - 17 junho | Funcionalidades Admin | Validações, restrições de input, testes com chatbot, melhorias na tela de planos diários. |
| 19 - 20 junho | Definições e Áudio da Bíblia | Tela de definições, integração com áudio em Python, responsividade dos cards. |
| 22 - 23 junho | Refinamento e Finalização dos Planos | Edição de planos, botão de cancelar, separação de capítulos, ajustes de layout e fontes, README e LICENSE. |
| 23 – 25 junho | Ajustes de Bugs | Ajustes de Bugs de finais |

## 3.2 Aspetos técnicos do desenvolvimento do projeto

**Fluxo de Dados**

O fluxo de dados da aplicação pode ser descrito da seguinte forma:

- **Autenticação de Utilizador**
  O utilizador insere as suas credenciais, que são validadas contra os registos armazenados na tabela Users. Em caso de sucesso, o sistema inicia uma sessão para o utilizador autenticado.

- **Consulta e Interação com Versículos**
  Os versículos são obtidos por via de chamadas HTTP a uma API externa. O utilizador pode interagir com os versículos através das funcionalidades de sublinhado, anotação e marcação. As interações são persistidas nas tabelas VersiculoSublinhado, VersiculoAnotado e VersiculoSalvo.

- **Planeamento de Leitura**
  O utilizador pode aderir a um plano de leitura previamente definido na tabela PlanoLeitura. Cada plano contém uma sequência de dias e capítulos associados, definidos na tabela PlanoLeituraModeloDia. Ao iniciar um plano, são gerados os registos correspondentes nas tabelas PlanoLeituraUtilizador e PlanoLeituraDia, permitindo o acompanhamento do progresso individual.

- **Chatbot e Cache de Respostas**
  As perguntas formuladas pelo utilizador são primeiro verificadas na tabela RespostasCache. Caso não exista uma resposta em cache, a aplicação envia a questão para um serviço externo de inteligência artificial. A resposta obtida é armazenada em cache para futuras chamadas reptetidas.

- **Último Ponto de Leitura**
  A tabela UltimoPontoLeitura regista o último capítulo lido por cada utilizador, permitindo retomar facilmente a leitura.

**Modelo de Dados**

A base de dados, designada BeLightBibleDB, foi estruturada para permitir a gestão eficiente dos dados dos utilizadores e das interações com o conteúdo bíblico. Abaixo apresenta-se uma descrição sucinta das tabelas principais:

- **Users**: Armazena informações de registo e autenticação dos utilizadores.
- **VersiculoSublinhado / VersiculoAnotado / VersiculoSalvo**: Registam interações personalizadas com versículos específicos, permitindo ao utilizador destacar, comentar e guardar os mesmos.
- **PlanoLeitura / PlanoLeituraModeloDia**: Definem os planos de leitura disponíveis, com a distribuição de capítulos ao longo dos dias.
- **PlanoLeituraUtilizador / PlanoLeituraDia**: Representam a adesão de um utilizador a um plano e o respetivo progresso diário.
- **RespostasCache**: Armazena os pares pergunta-resposta gerados pelo sistema de inteligência artificial, incluindo os vetores semânticos em formato JSON.
- **UltimoPontoLeitura**: Guarda a última referência lida por cada utilizador.

***Requisitos Iniciais Implementados***

| Código | Descrição | Estado |
|---|---|---|
| REQ0001 | Conectar a uma API para obter livros, capítulos e versículos. | Feito |
| REQ0002 | Apresentar livros e capítulos em ComboBox e versículos num painel dinâmico. | Feito |
| REQ0003 | Permitir grifar/sublinhar versículos (armazenando livro, capítulo, versículo e cor). | Feito |
| REQ0004 | Permitir adicionar anotações pessoais a versículos. | Feito |
| REQ0005 | Implementar explicação de versículos com recurso a uma LLM local via API. | Feito |
| REQ0006 | Disponibilizar planos de leitura predefinidos e acompanhar progresso. | Feito |
| REQ0007 | Implementar uma tela de login e autenticação. | Feito |
| REQ0008 | Disponibilizar operações CRUD sobre anotações e grifos. | Feito |
| REQ0009 | Utilizar Windows Forms para a interface com o utilizador. | Feito |
| REQ0010 | Utilizar SQL Server como base de dados. | Feito |
| REQ0011 | Utilizar menus contextuais para interagir com versículos. | Feito |

| | | |
|---|---|---|
| REQ0012 | Registar histórico de leitura e anotações. | Feito (incluindo também registro de grifos e versículos salvos com data) |
| REQ0014 | Implementar painel administrativo com acesso restrito a administradores. | Feito |
| REQ0018 | Criar, editar ou apagar planos predefinidos. | Feito |
| REQ0019 | Associar capítulos a dias em cada plano de leitura. | Feito |

***Requisitos Não Implementados***

| Código | Descrição | Estado | Justificativa |
|---|---|---|---|
| REQ0013 | Alternância entre modo claro e escuro na interface (Dark/Light Mode). | Não Feito | Devido à limitação de tempo na fase final do desenvolvimento, priorizou-se a entrega de recursos essenciais à funcionalidade da aplicação. |
| REQ0015 | Listar todos os utilizadores registados (Painel Administrativo). | Não Feito | Por gestão de tempo e complexidade adicional no desenvolvimento de permissões e filtros. |
| REQ0016 | Ativar/desativar contas de utilizador. | Não Feito | A funcionalidade foi adiada por depender de um sistema mais robusto de gestão de estado de conta. |
| REQ0017 | Redefinir palavra-passe. | Não Feito | Exigiria uma integração extra com envio de email ou recuperação |

| | | | |
|---|---|---|---|
| | | | segura, o que não foi viável no tempo disponível. |
| REQ0020 | Ver número de utilizadores inscritos em cada plano de leitura. | Não Feito | A análise de dados por plano exigiria consultas específicas e testes adicionais. |
| REQ0021 | Visualizar estatísticas de uso: anotações e grifos por dia/mês. | Não Feito | Deixado para uma futura versão, dada a complexidade de gráficos e agregações temporais. |

***Novos Requisitos Adicionados Durante o Desenvolvimento***

| Código | Descrição | Estado |
|---|---|---|
| REQ001 | Visualizar 'versículo do dia', com opções de salvar, copiar, compartilhar (WhatsApp, email) ou gerar imagem do versículo. | Feito |
| REQ002 | Retomar automaticamente a última leitura (livro e capítulo salvos por último). | Feito |
| REQ003 | Utilizar áudio da Bíblia por capítulo, com voz natural via recurso de Text-to-Speech. | Feito |
| REQ004 | Leitor bíblico infantil com histórias ilustradas. | Feito |
| REQ005 | Interface de definições com opção de logout e manutenção da sessão. | Feito |
| REQ006 | Interface de gestão de anotações, grifos e versículos salvos, com opções de editar, apagar ou | Feito |

| | | |
|---|---|---|
| | navegar até o versículo correspondente. | |

### Funcionalidades Implementadas (Principais)

A maioria dos **requisitos funcionais e técnicos definidos no início do projeto** foram implementados com sucesso:

- Autenticação de utilizador
- Leitura de versículos com interação (grifar, anotar, salvar)
- Explicação de versículos com IA
- Planos de leitura personalizados e progresso
- Versículo do dia com partilha
- Recurso de Text-to-Speech para leitura em voz alta
- Interface para crianças com histórias bíblicas
- Tela de definições com logout

### Funcionalidades Não Implementadas

Apesar do esforço, algumas funcionalidades ficaram por implementar, como:

- Alternância entre modo claro e escuro
- Redefinição de palavra-passe
- Estatísticas e gráficos de uso
- Gestão detalhada de utilizadores no painel administrativo

### *Considerações Finais (relacionadas as funcionalidades)*

Apesar de algumas funcionalidades administrativas e analíticas não terem sido implementadas devido a restrições de tempo, o projeto cumpriu todos os requisitos principais definidos inicialmente, bem como adicionou novas funcionalidades de valor significativo para a experiência do utilizador. Estas melhorias refletem a evolução natural do projeto ao longo do seu desenvolvimento.

### Classes Principais

Abaixo descrevem-se as classes mais relevantes para o funcionamento da aplicação:

**HistoriasKidsTab**

Responsável por renderizar e controlar a aba de histórias bíblicas infantis. Permite o carregamento dinâmico de histórias com imagens e textos curtos adaptados para crianças.

- **Principais Responsabilidades:**
    - Exibir conteúdo ilustrado.
    - Adaptar o layout para uma leitura mais acessível.
    - Carregar histórias de arquivos locais.

**Livro**

Classe de apoio que representa um livro da Bíblia. Usada na construção da interface de seleção de livros e capítulos.

**Utilizador**

Modelo que representa os dados do utilizador autenticado.

- **Campos principais:** UserId, Username, Email, Password, etc.
- Utilizada em sessões.

**Versiculo**

Classe central da lógica de leitura. Gerencia tudo relacionado à interação do utilizador com os versículos da Bíblia.

- **Funções principais:**
    - Sublinhado e anotação de versículos.
    - Armazenamento de versículos salvos.
    - Copiar texto, gerar imagem, partilhar.
    - Associação com áudio e explicação via LLM.

**Estilo**

Classe de suporte visual. Define estilos reutilizáveis e trata eventos relacionados à aparência e comportamento da interface gráfica.

- **Responsável por:**
    - Aplicar temas visuais (cores, fontes).
    - Eventos de hover, clique e tooltips.
    - Feedback visual para ações do utilizador.

**ApiBibleService**

Classe de comunicação com a API da Bíblia.

- **Responsabilidades:**
    - Buscar livros, capítulos e versículos.
    - Enviar e receber dados em JSON.
    - Tratar erros de rede e formatação de resposta.

## IA e Cache de Respostas

Para otimizar a experiência de explicação de versículos e minimizar chamadas repetidas ao modelo LLM, foram criados os seguintes métodos:

**EnviarParaOllama(string pergunta)**

- Envia a questão para o modelo LLaMA 3.3 70B hospedado no Groq Cloud.
- Recebe uma resposta textual detalhada.
- Utiliza HTTP Client para consumir a API do modelo.

**ObterEmbeddingOllamaAsync(string texto)**

- Chama o modelo **nomic-embed-text** (ollama: modelo local) para obter um vetor de embedding semântico do texto.
- Utilizado para indexar e comparar perguntas semanticamente semelhantes.

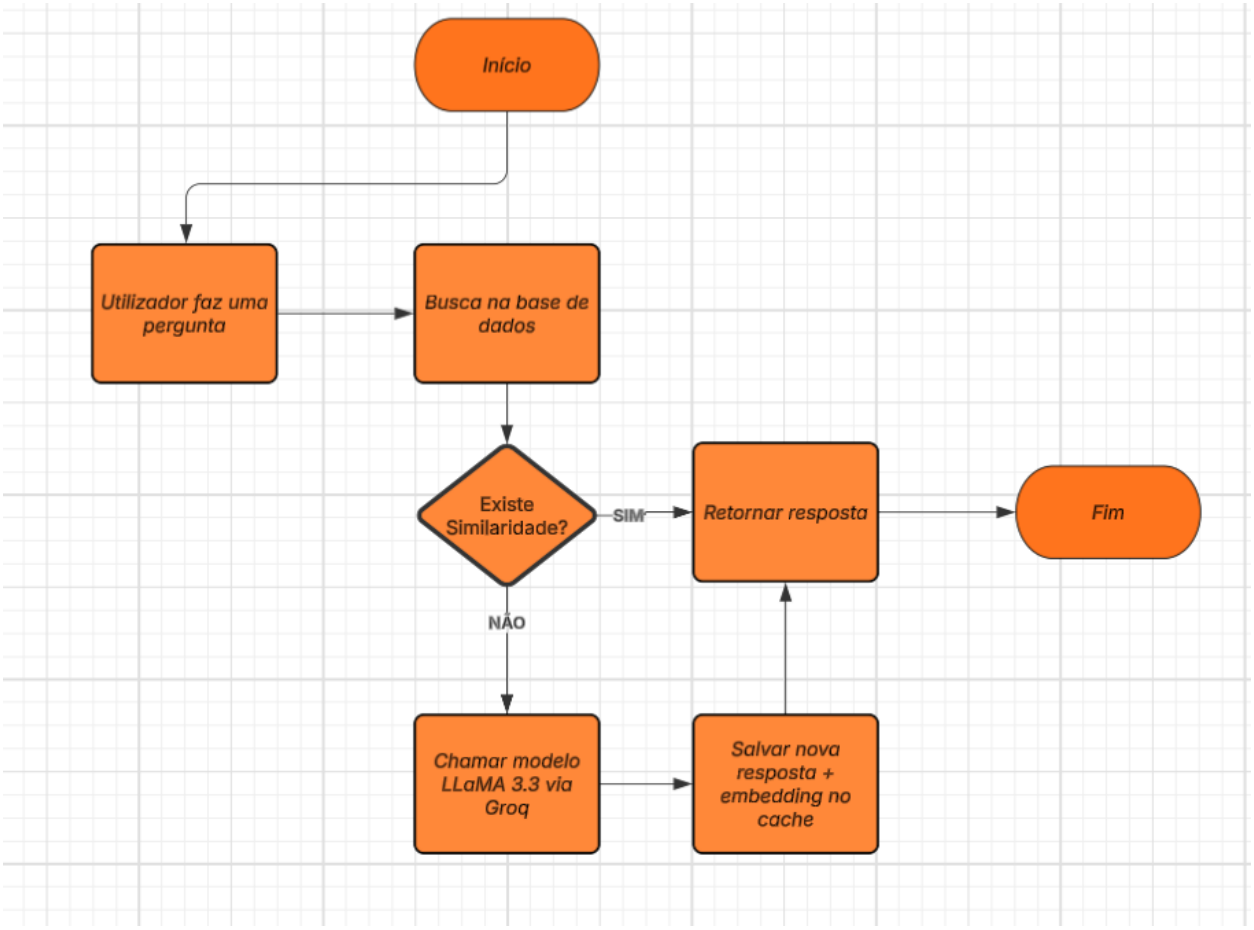
**BuscarRespostaCacheAsync(string pergunta)**

- Antes de consultar o LLM, busca se a pergunta já foi feita.
- Compara o vetor da nova pergunta com os vetores existentes usando CosineSimilarity.
- Se similaridade for alta (ex: > 0.85), retorna a resposta armazenada.

**CosineSimilarity(List<double> vector1, List<double> vector2)**

- Calcula a **semelhança semântica** entre dois vetores.
- Usado para evitar chamadas repetidas à API.
- Base para o sistema de cache inteligente.

## Fluxograma (da interação com a IA)




*Fluxograma do funcionamento das interações com a LLM*

## Escalabilidade e Sustentabilidade do Uso de IA (Groq Cloud + Nomic)

O sistema atual utiliza o modelo **LLaMA 3.3 70B (Groq Cloud)** para gerar explicações de versículos e o modelo **nomic-embed-text** para realizar buscas semânticas rápidas e relevantes em cache.

**Custo por Tokens – Groq Cloud**

- **Modelo:** Meta LLaMA 3.3 70B versatile
- **Hospedagem:** Groq Cloud
- **Velocidade de geração:** até 500 tokens/ms (extremamente rápido)
- **Preço (estimado em junho de 2025):**
    - Entrada (input): **$0.0005 por mil tokens**

- Saída (output): **$0.0015 por mil tokens**

Exemplo de custo médio por pergunta:

- Pergunta com 50 tokens e resposta com 200 tokens:
  - (50 × $0.0005) + (200 × $0.0015) = $0.0025

**Custo dos Embeddings e nomic-embed-text**

- **Preço médio:** ~$0.10 por 1.000 chamadas, dependendo do host (pode ser gratuito para uso moderado)
- Cada versículo ou pergunta é convertido num vetor de ~768 dimensões.
- Utilizado **apenas uma vez por pergunta nova**. Reutilizado a partir do cache posteriormente.

**Escalabilidade: 1.000, 5.000 utilizadores**

**1.000 usuários ativos/mês**

- Estimativa: 2 perguntas por dia por utilizador - 60.000 perguntas/mês
- Custo estimado:
  - LLM: ~$150/mês
  - Embeddings: ~$6
  - Total: **~$156/mês**

**5.000 usuários ativos/mês**

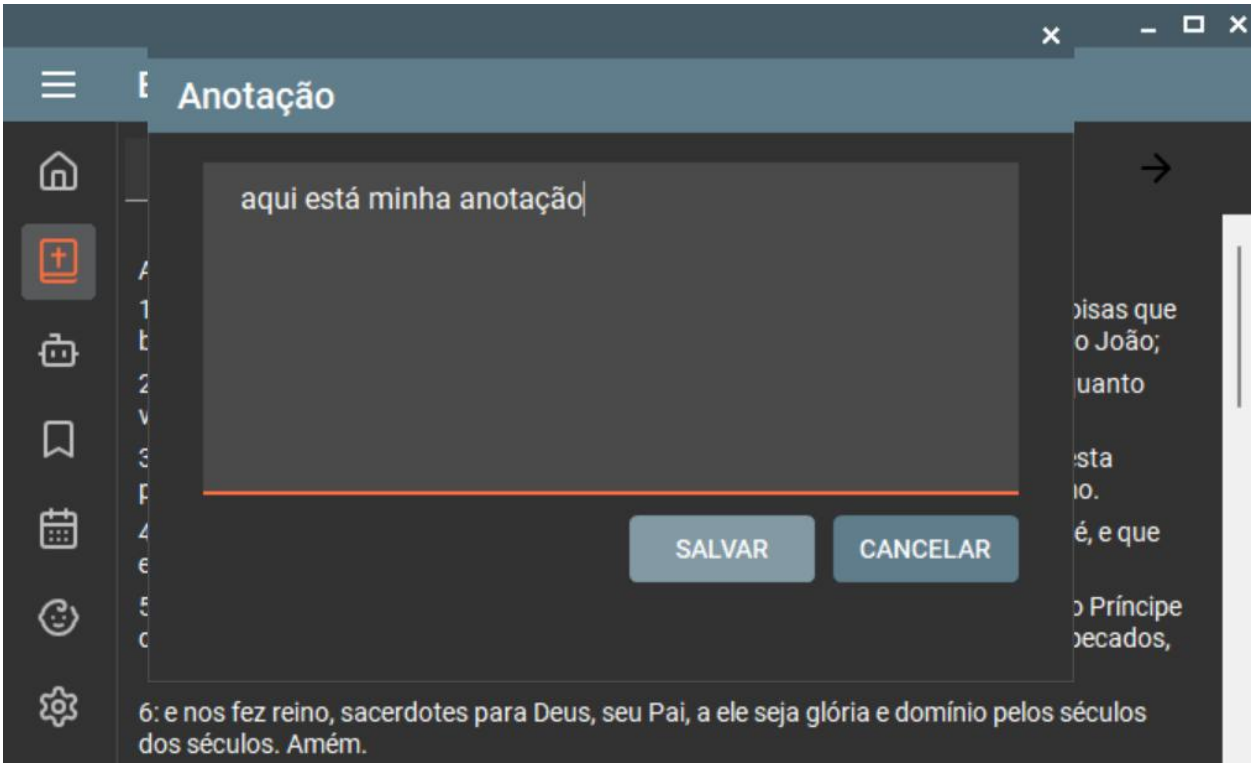
- Estimativa: 2 perguntas por dia por utilizador - 300.000 perguntas/mês
- Custo estimado:
  - LLM: ~$750
  - Embeddings: ~$30
  - Total: **~$780/mês**
- **Sustentável com apoio institucional ou doações.**

**Estratégias para Reduzir Custos e Escalar Melhor**

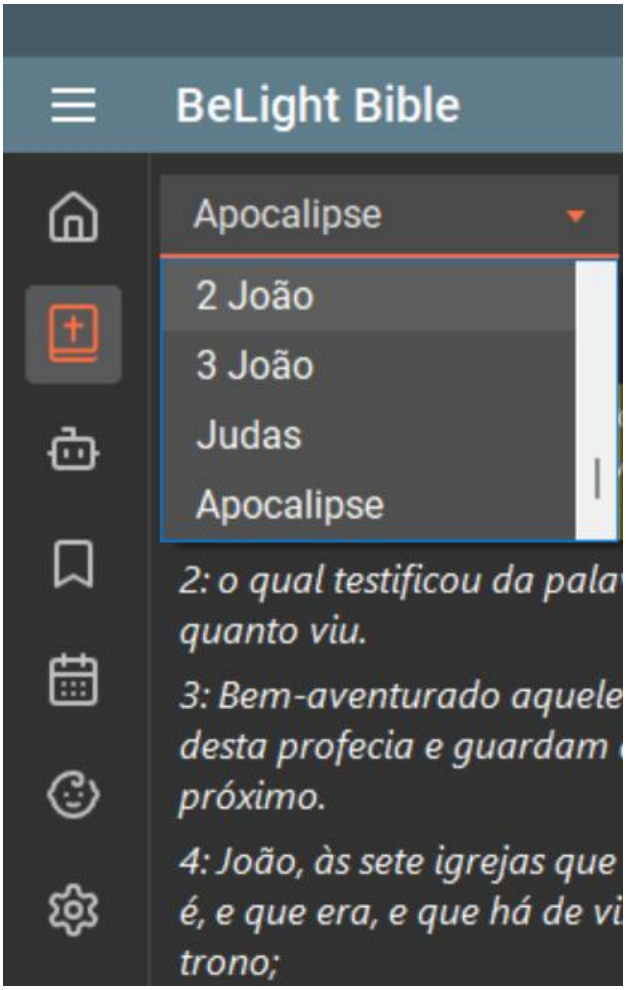
- **Uso de Cache Inteligente:**
  Reduz em até **70–80%** o número de chamadas à IA, graças ao sistema de embeddings e similaridade semântica (CosineSimilarity).
- **Pré-processamento de respostas frequentes:**
  Perguntas comuns (ex: "O que significa João 3:16?") podem ser respondidas com antecedência e armazenadas.

## *Interfaces*

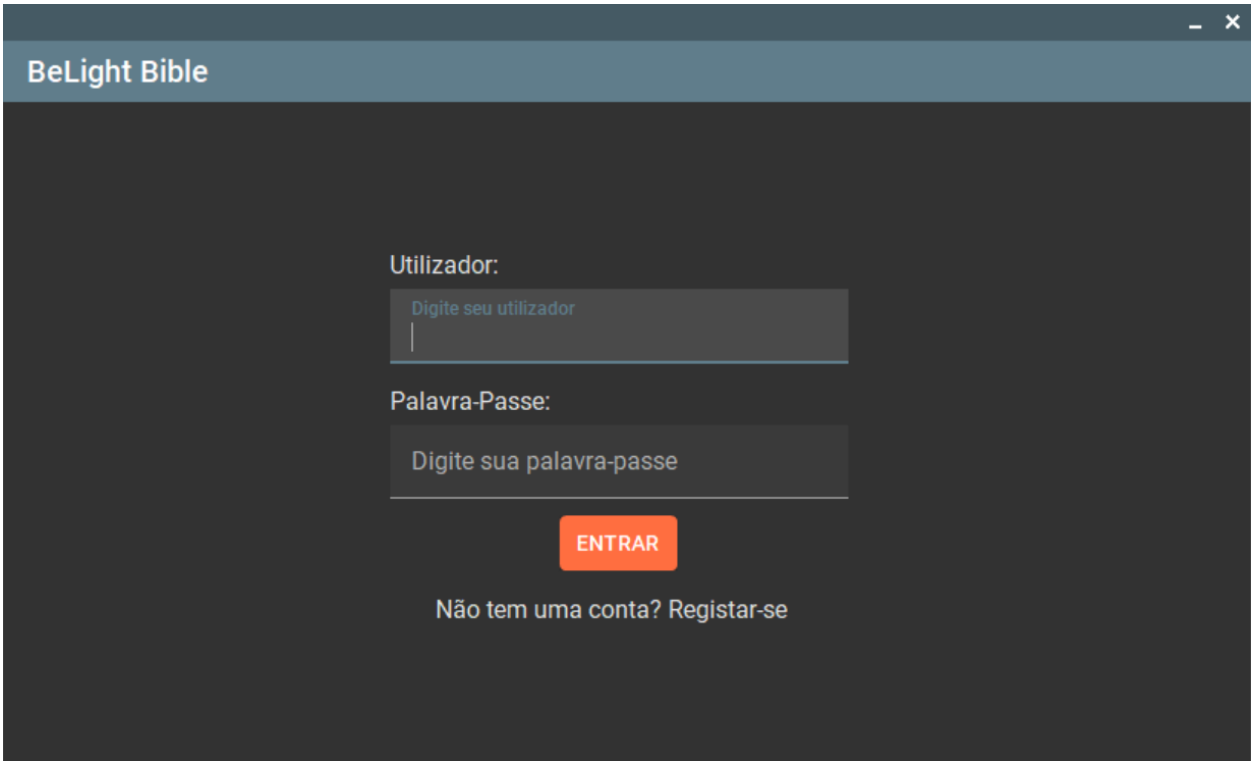

***Interface para visualizar os versículos.***



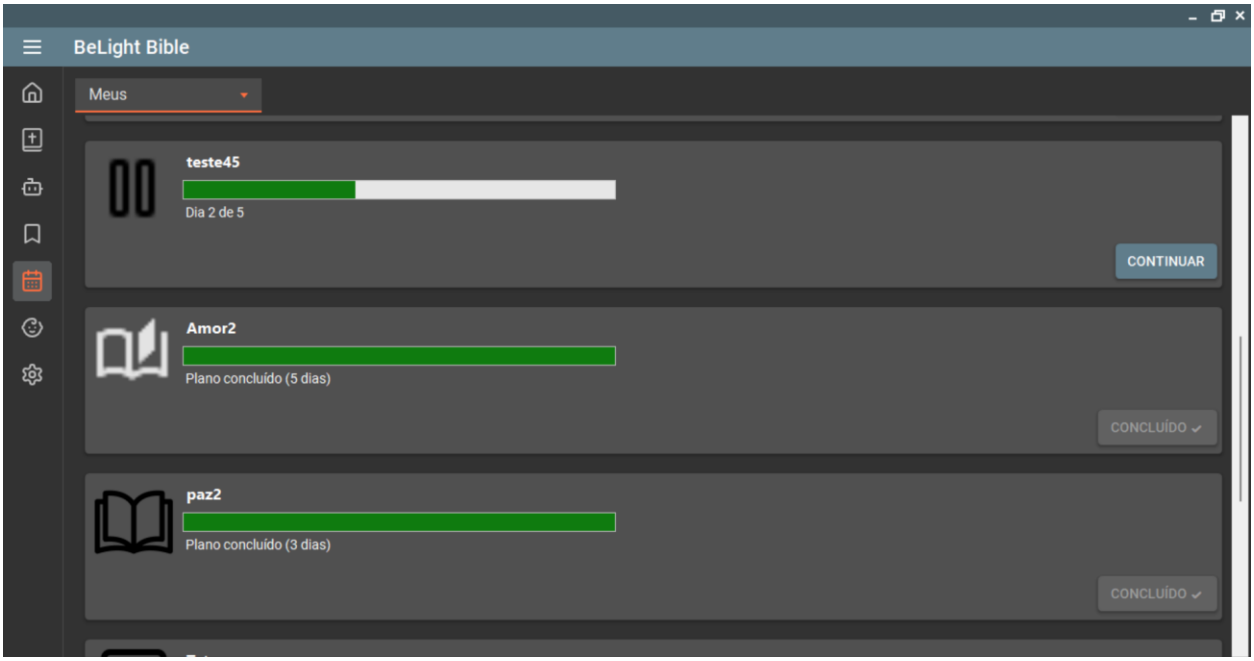

***Popover (componente visual), janela flutuante contextual para adicionar uma nota pessoal ao versículo.***




***”Sidebar” (componente visual) para lista dos livros  da bíblia*.**




***Interface de login.***




***Interface para o utilizador acompanhar o seu progresso de estudo***.


***Interface para selecionar e iniciar um plano de estudo.***

### *Diagrama E-R da Base de Dados*

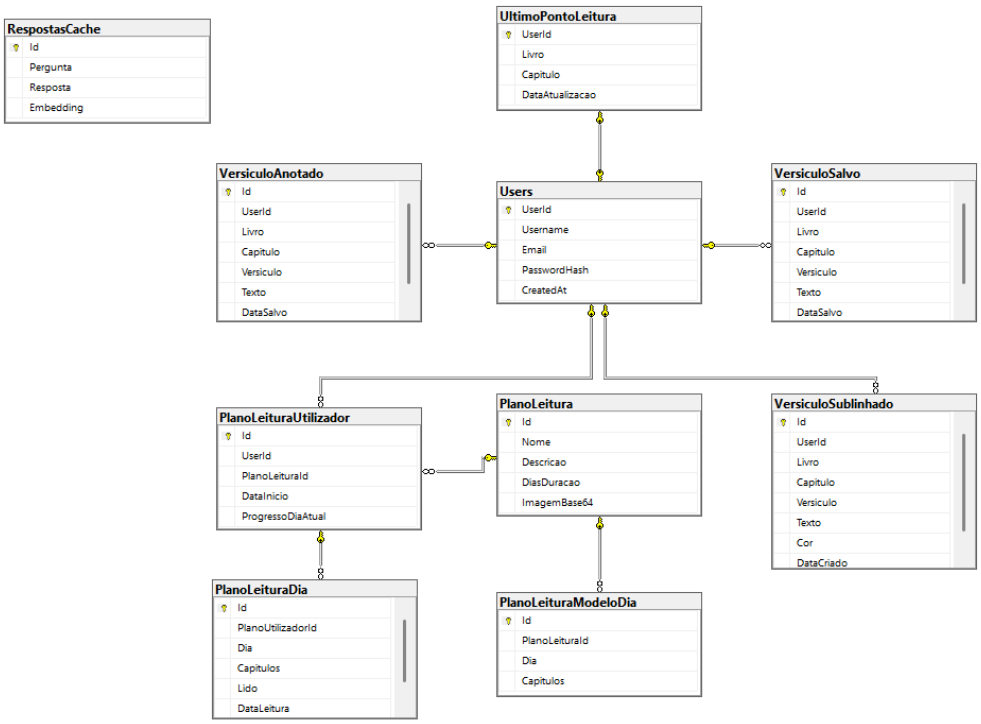



*Diagrama Relacional da Base de Dados: Relações entre as Tabelas Users, VersiculoSublinhado, VersiculoSalvo, VersiculoAnotado, UltimoPontoLeitura, PlanosLeitura, PlanoLeituraUtilizador, PlanoLeituraModeloDia, PlanoLeituraDia e RespostasCache.*

## 4 - Referências Bibliográficas


1. **ChatGPT (OpenAI)**
   OpenAI. "ChatGPT." Disponível em: https://chat.openai.com.
   Descrição: Utilizado para gerar explicações e sugestões relacionadas ao desenvolvimento do projeto, incluindo a conceção de funcionalidades, diagrama e fluxos do BeLight Bible.
2. **Canva**
   Canva. "Ferramenta de Design Canva." Disponível em: https://www.canva.com.
   Descrição: Ferramenta de design gráfico utilizada para a criação de protótipos e elementos gráficos do projeto (**logótipo**).
3. **Ollama** Ollama. Disponível em: https://ollama.com. Descrição: Tecnologia utilizada para a criação e gestão de embeddings no sistema, permitindo o processamento semântico de textos para funcionalidades avançadas de pesquisa e recomendação no BeLight Bible.
4. **Groq Cloud** Groq. *Plataforma Groq Cloud*. Disponível em: https://groq.com/cloud.
   Descrição: Plataforma utilizada para acelerar o processamento do modelo de LLM no projeto, melhorando a performance das operações.

5. **Windows Forms** Microsoft. *Windows Forms*. Disponível em: https://docs.microsoft.com/en-us/dotnet/desktop/winforms/. Descrição: Framework para desenvolvimento da interface gráfica da aplicação.
6. **MaterialSkin** MaterialSkin Project. *MaterialSkin Library*. Disponível em: https://github.com/IgnaceMaes/MaterialSkin. Descrição: Biblioteca open source baseada no Material Design para estilização da interface gráfica.
7. **YouTube** YouTube. *Canal oficial e vídeos tutoriais*. Disponível em: https://www.youtube.com. Descrição: Utilizado para consulta de vídeos tutoriais e explicativos sobre desenvolvimento em C#, Windows Forms, consumo de APIs e outras tecnologias usadas no projeto BeLight Bible.