**Universidade Federal de Goiás**

**Instituto de Informática**

**Software Concorrente e Distribuído**

**Documento de Design de Software - Artistic Trivia**

**Alany Gabriely**

**André Lopes**

**Artur Lapot**

**Cauã Rebelo**

**Bruno Milioli**

**Filipe Paço**

**Tayna Crisllen**

**Goiânia**

**Julho de 2024**

1. **Overview do Projeto**
   1. **Contextualização**

Este projeto visa combinar elementos de aprendizado e diversão com a criatividade proporcionada por tecnologias avançadas de inteligência artificial. Utilizando um jogo de perguntas de múltipla escolha, onde os jogadores podem testar seus conhecimentos e competir entre si, juntamente com a capacidade de gerar imagens personalizadas a partir de descrições textuais, criaremos uma plataforma dinâmica e envolvente.

* 1. **Objetivos do Projeto**

Desenvolver uma plataforma interativa que combine um jogo de perguntas de múltipla escolha, e um gerador de imagens por inteligência artificial integrado ao Telegram. No jogo os usuários poderão testar seus conhecimentos e diversão ao jogar com um oponente onde os dois competem para ser o primeiro a alcançar 5 pontos. Utilizando o Stable Diffusion, a plataforma permitirá que os usuários criem imagens personalizadas a partir de descrições textuais, oferecendo uma experiência educativa, competitiva e criativa, que incentiva o aprendizado, a interação social e a expressão visual de ideias.

* 1. **Discussões relevantes**
  2. **Trabalhos relacionados**

Colocar os antigos, e outros tipos de trabalhos semelhantes, e arquitetura que eles usam

1. **Requisitos**
   1. **Requisitos de usuário**

RU 01 - Usuários podem iniciar um jogo de múltipla escolha e a plataforma escolher o oponente.

RU 02 - Usuários vão responder perguntas com quatro opções de resposta e apenas uma é a resposta correta.

RU 03 - Usuários devem receber feedback imediato se a resposta está correta ou não.

RU 04 - Usuários devem ser notificados quando um jogador alcançar 5 pontos e for declarado vencedor.

RU 06 - Usuários devem poder enviar descrições textuais através do Telegram para gerar imagens.

RU 07 - Usuários devem receber a imagem gerada pelo modelo de IA (Stable Diffusion) diretamente no Telegram.

* 1. **Requisitos funcionais**

RF 01 - O sistema deve selecionar perguntas aleatórias de um banco de dados e apresentá-las aos jogadores.

RF 02 - O sistema deve validar as respostas dos jogadores e atualizar suas pontuações.

RF 03 - O sistema deve receber e processar comandos de texto dos usuários através do Telegram.

RF 04 - O sistema deve enviar perguntas e receber respostas dos usuários através do Telegram.

RF 05 - O sistema deve enviar a imagem gerada pela IA de volta ao usuário no Telegram.

RF 06 - O sistema deve receber descrições textuais dos usuários e encaminhá-las para o modelo Stable Diffusion.

* 1. **Requisitos não funcionais**

RNF 01 - O sistema deve ser capaz de lidar com um grande número de usuários simultâneos sem sofrer queda de desempenho.

RNF 02 -O sistema deve garantir alta disponibilidade (pelo menos 99,9%) para que os usuários possam acessar a plataforma a qualquer momento.

RNF 03 - O sistema deve ter uma interface fácil de usar com informações claras e uma navegação intuitiva e clara, tanto para o jogo quanto para o gerador de imagens .

RNF 04 - O sistema deve ser capaz de se recuperar rapidamente de erros e manter a integridade dos dados.

RNF 05 - O sistema deve garantir que somente usuários autenticados possam acessar o jogo e o gerador de imagens.

RNF 06 - O sistema deve proteger os dados dos usuários contra acessos não autorizados.

RNF 07 - O sistema deve evitar fraudes, como a manipulação do jogo da pontuação do jogo.

RNF 08 - O sistema deve operar perfeitamente dentro do ambiente do Telegram , aproveitando ao máximo todas as suas funcionalidades.

RNF 09 -O sistema deve ser compatível com uma variedade de dispositivos (computadores, tablets e celulares).

1. **Fundamentos de SD relacionados ao projeto**
   1. **Princípios de Sistemas Distribuídos**

cc

* 1. **Fundamentos de arquitetura de Sistemas Distribuídos e dos estilos arquiteturais**

Para garantir escalabilidade, flexibilidade e eficiência, o sistema precisa aplicar os conceitos arquitetônicos e de sistemas distribuídos por fundamentos.

O Frontend (Telegram Bot) será responsável por interagir com os usuários, enviar e receber mensagens deles, exibir perguntas de múltipla escolha e enviar imagens geradas pelo gerador de imagens .

O Backend (Servidor de Aplicação), é responsável por processar a lógica do jogo, manter o estado do jogo e progressão do jogador, e início da geração de imagens de IA .

O Gerador de Imagens IA é responsável por processar descrições textuais para gerar imagens.

O Banco de Dados é encarregado de guardar as informações, dados avaliados e histórico dos jogos.

### 

* Divisão de Componentes

Para garantir escalabilidade, flexibilidade e eficiência, o sistema precisa aplicar os conceitos arquitetônicos e de sistemas distribuídos por fundamentos.

O Frontend (Telegram Bot) será responsável por interagir com os usuários, enviar e receber mensagens deles, exibir perguntas de múltipla escolha e enviar imagens geradas pelo gerador de imagens .

O Backend (Servidor de Aplicação), é responsável por processar a lógica do jogo, manter o estado do jogo e progressão do jogador, e início da geração de imagens de IA .

O Gerador de Imagens IA é responsável por processar descrições textuais para gerar imagens.

O Banco de Dados é encarregado de guardar as informações, dados avaliados e histórico dos jogos.

* Estilos Arquiteturais

O sistema deve separar uma aplicação de microsserviços em pequenos serviços independentes, como classificação, geração de imagens e perguntas. Cada serviço pode ser desenvolvido, implementado, de forma independente, desenvolvido, implementado, dimensionado e dimensionado.

O sistema deve empregar os eventos para comunicação entre serviços conforme a Arquitetura Orientada a Eventos , principalmente para atualizar os estados do jogo e notificar novos pontos.

Serão implementadas APIs RESTful para a comunicação entre o servidor de aplicação e o Telegram Bot.

Funções serverless serão empregadas em componentes serverless com cargas de trabalho variáveis, como o gerador de imagens AI.

* Tecnologias e Ferramentas

API do Telegram Bot ,para integração com o Telegram .  
Use Django/Flask ou Node.js/Express, para construir o servidor do aplicativo.

Para funções sem servidor (particularmente geradores de imagens), usara AWS Lambda/Azure Functions.

MongoDB/PostgreSQL, para o armazenamento dos dados.

Para gerar imagens por descrições de texto, a API Stable Diffusion.

* Segurança e Escalabilidade

A autorização e a autenticação devem garantir que apenas usuários autorizados possam acessar e jogar.  
Deve-se proteger contra o sistema de abuso, especialmente na geração de imagens. Para garantir disponibilidade, o balanceamento de carga equilibrando deve distribuir cargas de trabalho entre diversas instâncias de servidores.

Deve-se usar caching para perguntas de múltipla escolha e imagens filtradas para reduzir a carga do servidor.

* 1. **Fundamentos de paradigmas de comunicação em Sistemas Distribuídos**
* Paradigmas de Comunicação

1. Comunicação por passagem de mensagens ;

2. Chamadas de Procedimento Remoto (RPCs);

3. Comunicação assíncrona;

4- Comunicação Síncrona;

5. Publicar/Assinar ou Pub/Sub;

* Aplicação na Plataforma

1. Comunicação por Mensagens

Para executar ações, os componentes trocam mensagens, sem acesso direto à memória.

Entre o servidor de aplicação e o bot do Telegram : O bot avisa o servidor de aplicação da resposta quando um usuário responde a uma pergunta.

O gerador de imagens AI e o servidor de aplicação: O gerador de imagens recebe a URL da imagem gerada como resposta do servidor de aplicação, que envia uma descrição textual ao gerador.

2. RPC (Remote Procedure Call)

A função de um programa pode ser realizada em outro espaço de endereço, normalmente em outro servidor, como o uso do RPC (Remote Procedure Call).  
O gerador de imagens AI e o servidor de aplicação: A função remota pode ser chamada pelo servidor de aplicativo para gerar uma imagem devido a uma descrição textual.

3. Comunicação Assíncrona

O sistema continua a operar enquanto a resposta é enviada, permitindo que o sistema continue funcionando enquanto espera.

Bot do Telegram e o Servidor de Aplicação :as respostas do usuário podem ser processadas de forma assíncrona, com atualizações enviadas ao bot quando estiverem disponíveis.  
Gerador de imagens AI e servidor de aplicação: o servidor de aplicação envia a solicitação de geração de imagem e continua com outras tarefas, aceitando a imagem gerada quando pronta.

4. Comunicação Síncrona

O remetente espera pela resposta do destinatário antes de prosseguir com o processamento.  
Entre o servidor de aplicação e o bot do Telegram: para atividades críticas que exigem feedback rápido , como validação instantânea dos resultados dos usuários .

5. Pub/Sub (Publish/Subscribe)

Uma comunicação de estratégia, em que os publicadores enviam mensagens para um canal sem saber quem são os assinantes. Os receptores registram o interesse nos ​canais.

Notificações de Pontuação e Atualizações de Jogo: quando um usuário alcança um novo ponto ou termina um jogo, o servidor de aplicação publica um evento de atualização de jogo, e o bot do Telegram, que está inscrito nesses eventos, envia notificações aos usuários.

* 1. **Robustez em sistemas distribuídos, nomeação, coordenação, consenso, consistência, replicação e tolerância a falhas**
* Nomeação

Identificadores Únicos: atribuir identificadores únicos a cada usuário, sessão de jogos e imagem gerada.

Serviço de Nomeação: para mapear nomes lógicos para endereços físicos, como DNS para serviços internos.

* Coordenação

Zookeeper/etcd: para gerenciar a configuração distribuída, bloqueios de exclusão mútua e sincronização entre componentes durante a coordenação de serviços.

Gerenciamento de Sessão: coordenar entre os vários jogadores usando um coordenador central que garante que as respostas sejam processadas na ordem correta.

* Consenso

Algoritmos de Consenso**:** Raft ou Paxos para garantir que todos os participantes do sistema participem em estados críticos, tais como a pontuação do jogo ou o estado atual da partida.

Coordenação de Pontuação: garantir que a pontuação dos jogadores seja consistente e acordada por todos os nós envolvidos no processamento do jogo.

* Consistência

Consistência Eventual: operações não críticas onde a latência é mais importante, como atualizações de perfil de usuário.

Consistência Forte: para dados críticos como pontuação e estado do jogo, é importante que tenhamos a mesma visão do estado entre todos.

* Replicação

Banco de Dados Replicado**:** para armazenar questões, respostas e pontuações

Replicação de Imagens: distribuir imagens registradas para garantir alta disponibilidade e recuperação rápida em caso de falhas.

* Tolerância a Falhas

Detecção de falhas: implementar mecanismos para identificar falhas de componentes usando ferramentas, como Heartbeat ou outros sistemas de monitoramento .

Failover Automático: para que quando um de nós falhar, o tráfego seja redirecionado para nós com segurança.

Retry Mechanism: implementar mecanismos de tentativa e erro para operações críticas que podem falhar devido a problemas temporários

1. **Resultados**
   1. **Design arquitetural**

Telegram Bot (Frontend)

Servidor de Aplicação (Backend)

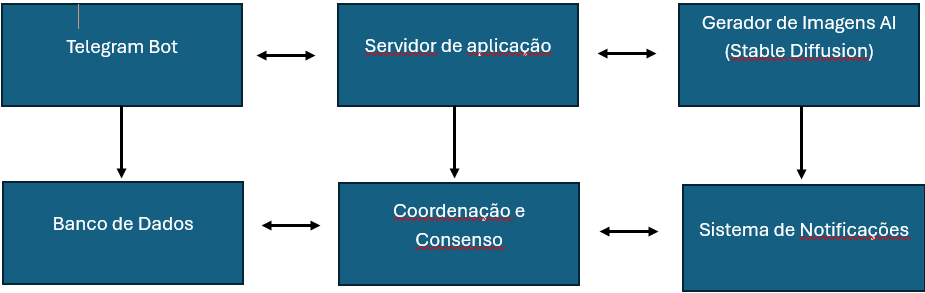
Gerador de Imagens AI (Stable Diffusion)

Banco de Dados

Serviços de Coordenação e Consenso

Sistema de Notificações

Diagrama de Arquitetura

****

* 1. **Design dos dados**

1. **Limitações, trabalhos futuros e perspectiva do projeto**
   1. **Limitações**

As limitações e desafios que ao desenvolver um sistema que combina um jogo de perguntas de múltipla escolha e um gerador de imagens por inteligência artificial integrado ao Telegram são:

* Limitações Técnicas

Limites de taxa na API do Telegram podem afetar a capacidade de resposta do bot quando muitos usuários estão ativos ao mesmo tempo.

O uso de modelos de indivíduo artificial, como o Difusão Estável, exige um alto grau de habilidades computacionais. O desempenho pode ser prejudicado se a plataforma não possuir acesso a recursos computacionais suficientes.

* Escalabilidade

Manter o estado do jogo para vários usuários simultâneos pode ser desafiador e exigir uma arquitetura de distribuição de dados eficaz.  
É fundamental dividir a carga de trabalho entre os servidores de maneira eficiente , mas isso pode ser difícil, principalmente se houver picos de usos inesperados.

* Consistência e Sincronização

Garantir que o estado jogo e a pontuação sejam mantidos de forma consistente ao longo de várias sessões e dispositivos pode ser desafiador, principalmente em um ambiente distribuído .Um desafio técnico importante é fornecer atualizações em tempo real para ambos os jogadores sem atrasos perceptíveis.

* Segurança e Privacidade

Proteger os dados pessoais e as atividades dos usuários contra acessos não autorizados e garantir que o gerador de imagens por IA não seja empregado para produzir conteúdo impróprio ou ofensivo.

* Experiência do Usuário

O sistema pode ser complicado de usar, principalmente para usuários menos experientes, devido à integração de diversas funcionalidades, como geração de imagens e jogo de perguntas. A qualidade das imagens geradas pelo Stable Diffusion pode variar.

* Custos Operacionais

Pode ser caro executar modelos IA , principalmente se o sistema precisar ser dimensionado para oferecer suporte a muitos usuários.

Manter o sistema operando e fornecer suporte ao usuário pode consumir muito tempo e exigir uma equipe dedicada o que pode gerar um alto custo.

* Dependência de Terceiros

O sistema é altamente dependente da API Telegrama, e quaisquer alterações ou limitações na API podem afetar a funcionalidade do sistema.

Continuar acompanhando as versões de melhorias dos modelos de IA pode ser desafiador e exigir investimento.

* Legalidade e Conformidade

Garantir que o sistema esteja em conformidade com os regulamentos de proteção de dados (como o GDPR) e outras leis pertinentes é crucial. E também empregar modelos de IA licenciado corretamente e garantir que não haja direitos autorais visíveis.

* Confiabilidade e Disponibilidade

Garantir a disponibilidade do sistema durante a manutenção e atualizações pode ser um desafio.

* 1. **Trabalhos futuros**