

TDS-Dito: Ideias e coisas a fazer na extensão

Legendas:



Pronto



Em andamento, sem prazo




Em estudo/a fazer (pode ser cancelado)

Inteligência Artificial (Carol Clockin Web API)






- API para uso da inteligência artificial.
- Definir tratamento para **What do you want me to explain?** ou **Explain What?**, inclusive de códigos já processados.
- Apresentado lista de opções ao final da resposta (**Show me...**). Como proceder?
- Eliminar frases duplicadas.
- Eliminar sugestões de código já existente.
- Adicionar tempo de processamento em todas as funções da API.
- Configuração do nível de detalhes das respostas.
Nessa etapa, apresentar sempre de forma sucinta, com opção para detalhamento.
- Controle de acesso (*login*).
- Mecanismo de avaliação de respostas pelo usuário, que poderá ser utilizada para refinamento do modelo.
- Configuração do idioma das respostas.
- Adicionar suporte para outros idiomas.
- Em alguns retornos de explicação de um bloco, retornou referências a linha (**First line, Second Line...**). Melhor seria textos mais corridos, sem ser linha a linha e de acordo com nível de detalhe desejado.

API: Inteligência Artificial (interface IaApiInterface)




- API para a inteligência artificial.
- Identificação de usuário (*login*).
Usar **vscode.AuthProvider** para autenticação.
- Mensagens associadas a processamento.
Encapsular o processamento e passar como *callback* para ChatApi.Dito e esse passa a tratar o retorno da mensagem (**messageId**).

-  No caso de erro 504 e com informação de tempo para nova tentativa, agendar nova tentativa. Mensagem com tempo: `The server encountered a temporary error and could not complete your request. Please try again in 30 seconds.`

API: Bate-papo (classe ChatApi)

-  API para bate-papo.
-  Implementar sistema de tradução L10N.
-  Detalhar a `ajuda(help)` dos comandos.
-  Melhorar/implementar tratamento de argumentos em comandos.
-  Implementar processo de abertura de chamado (comando `open_issue`).

Visual: Bate-papo

-  Executar o comando ao acionar `Enter`.
-  Associar visualmente mensagem de resposta com a mensagem de entrada.
Tratar `MouseOver` sobre a mensagem de resposta ou de entrada e destacar as duas. Dessa forma, se o usuário disparar vários processos, pode acompanhar qual mensagem está sendo respondida.
-  Implementar `goto` em ligações (*links*) de posicionamento em fontes.

Visual: Editor de texto

-  Colocar *codeLens* no código que esta sendo processado.