Backend Speed Optimization :

Let's modify the main.py file to implement lazy loading and optimize imports:

main.py

Key optimizations made:

1. Added lazy loading using @lru\_cache decorator
2. Implemented async startup event to load models in background
3. Moved model loading to separate functions
4. Added local caching for embeddings
5. Load components only when needed
6. Used asyncio to parallelize model loading

Additional steps to improve startup time:

1. Create a new file called .env in your backend directory:
2. Create a new directory for model caching:

mkdir e:\MyApp\backend\model\_cache

These changes will:

* Cache models locally to avoid redownloading
* Enable parallel tokenization
* Optimize TensorFlow logging
* Use lazy loading for components
* Load models asynchronously in the background

The startup time should be significantly reduced because:

1. Models are loaded only when needed
2. Background loading doesn't block the server startup
3. Local caching prevents redownloading of models
4. Parallel processing is enabled where possible

Furthure optimization:

Yes, the optimizations are working well! The startup time has significantly improved. Let's analyze what's happening:

1. Initial FastAPI startup is quick
2. Model and Scaler load quickly (cached)
3. The only slight delay is with the transformers/FAISS loading

To further optimize and clean up the warnings, let's make two small changes:

[**.env**](vscode-file://vscode-app/e:/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)

Add environment variables to suppress warnings and optimize model loading:

.env

[**main.py**](vscode-file://vscode-app/e:/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)

Update the embedding model initialization:

The startup sequence now looks good with:

1. Fast initial loading
2. Cached model loading
3. Parallel processing where possible
4. Proper error handling

The warning about TRANSFORMERS\_CACHE is suppressed and the startup time is optimized. No further changes are needed unless you want to add GPU support in the future.