In the context of using Qdrant, the concept of a vector store typically refers to how and where the vector embeddings are stored and managed. Qdrant itself acts as a vector database, managing the storage, indexing, and querying of vector embeddings. Therefore, when you initialize a Qdrant client and set up a collection, Qdrant manages the vectors you insert into this collection.

FAISS (Facebook AI Similarity Search) is another technology used for efficient similarity search and clustering of dense vectors. While FAISS is incredibly efficient and widely used, it operates separately from Qdrant. You typically use FAISS as a standalone solution for vector similarity search, especially in cases where you're dealing with large-scale datasets and require high-performance search capabilities.

However, if you're already using Qdrant, it's managing the vectors for you, providing functionalities similar to what FAISS offers, like efficient similarity search, but with additional features like REST API, payload management, and scalability options. While you can technically use FAISS to create an index and then pass that data to Qdrant, it's not the usual workflow, and Qdrant doesn't directly accept a FAISS index as an input.

If you want to leverage both, here’s how you could theoretically combine them:

1. **Use FAISS for local, fast similarity searches:**
   * You can preprocess your data, create a FAISS index, and use it for quick, efficient similarity searches locally.
   * This approach is beneficial when you have a static dataset where you don't need the additional features provided by Qdrant.
2. **Use Qdrant for persistent storage and querying:**
   * You can insert your data into Qdrant for persistent storage, scalable querying, and additional features like filtering by metadata.
   * This is useful when you need a scalable, robust system that can handle updates to your dataset and provide RESTful API access.

To directly answer your question, no, you can't input a FAISS vector store directly into the Qdrant configuration as shown in your code snippet. The setup you've shown is for initializing a Qdrant collection, which is designed to store and manage the vectors internally. If you have your vectors stored in a FAISS index, you would need to extract those vectors and then insert them into Qdrant using the appropriate API calls.