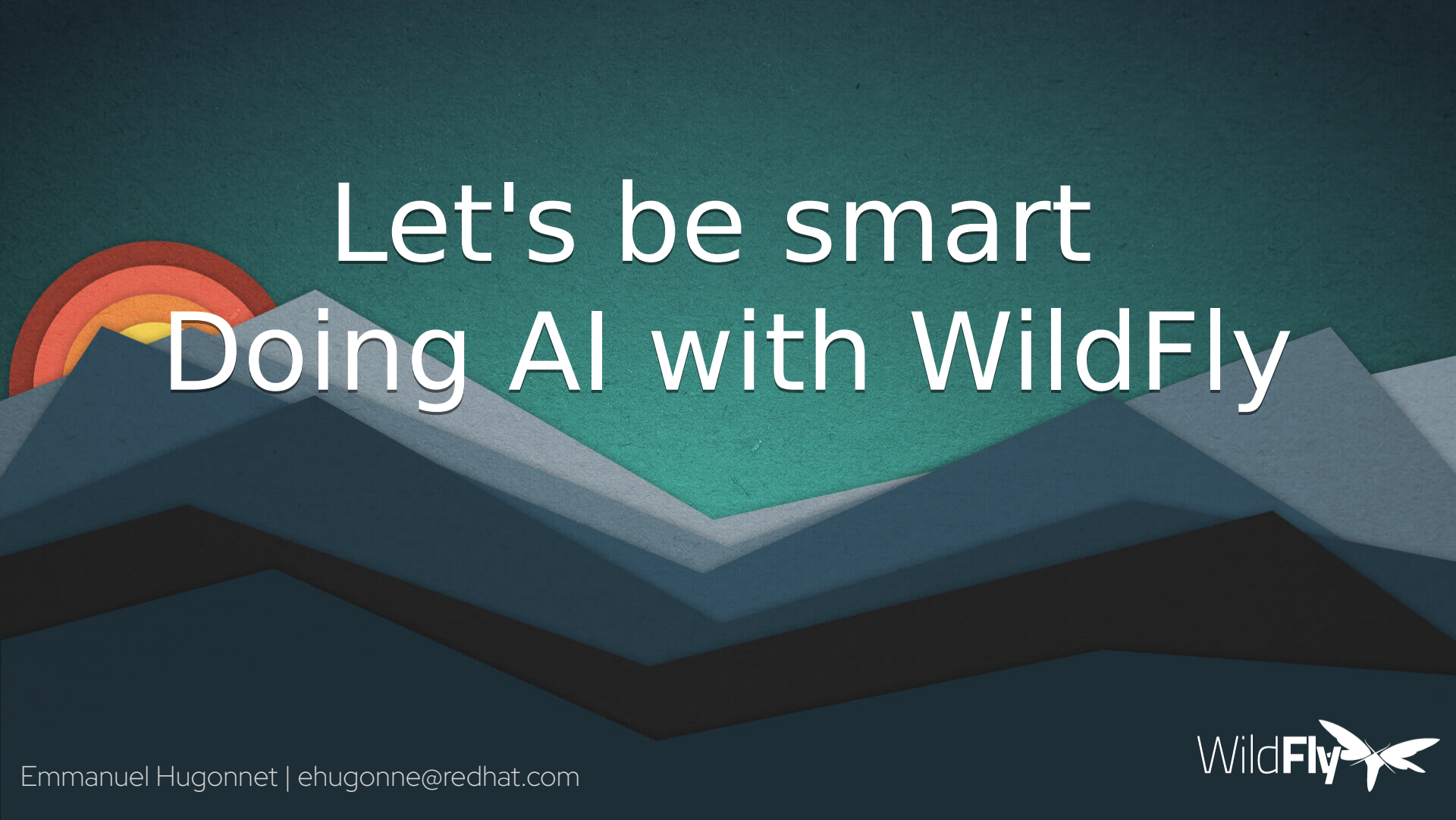


Emmanuel Hugonnet | ehugonne@redhat.com

WildFly

JakartaEE meets AI

Emmanuel Hugonnet | ehugonne@redhat.com

WildFly

Large Language Models (LLMs)

- Neural Networks
  - Transformer based
  - Recognize, Predict, and Generate text
  - Trained on a VERY large corpus of text
  - Deduce the statistical relationships between tokens
  - Can be fine-tuned

A LLM predicts the next token based on its training data and statistical deduction

Retrieval Augmented Generation

You can do *a lot*

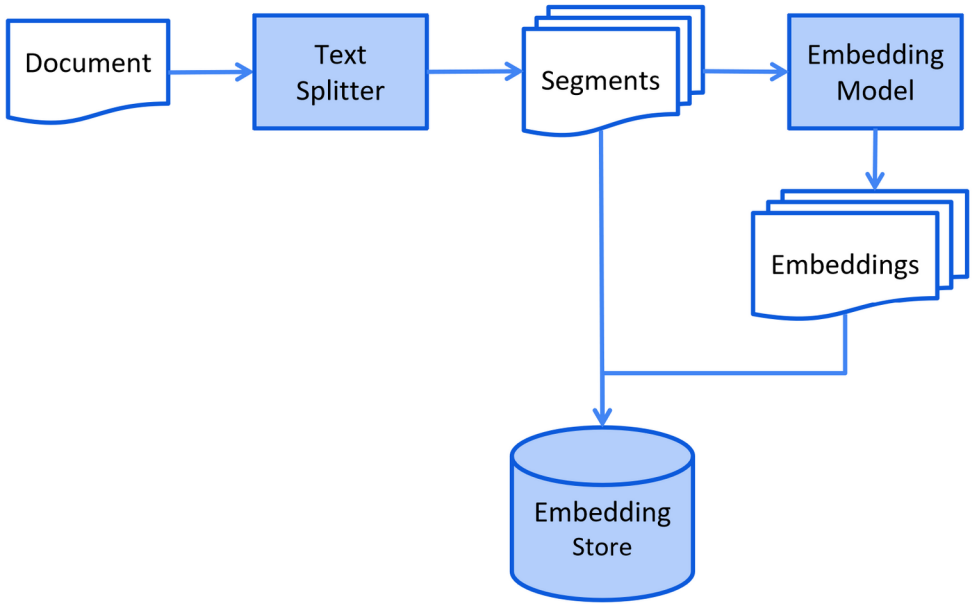
with a generic

model

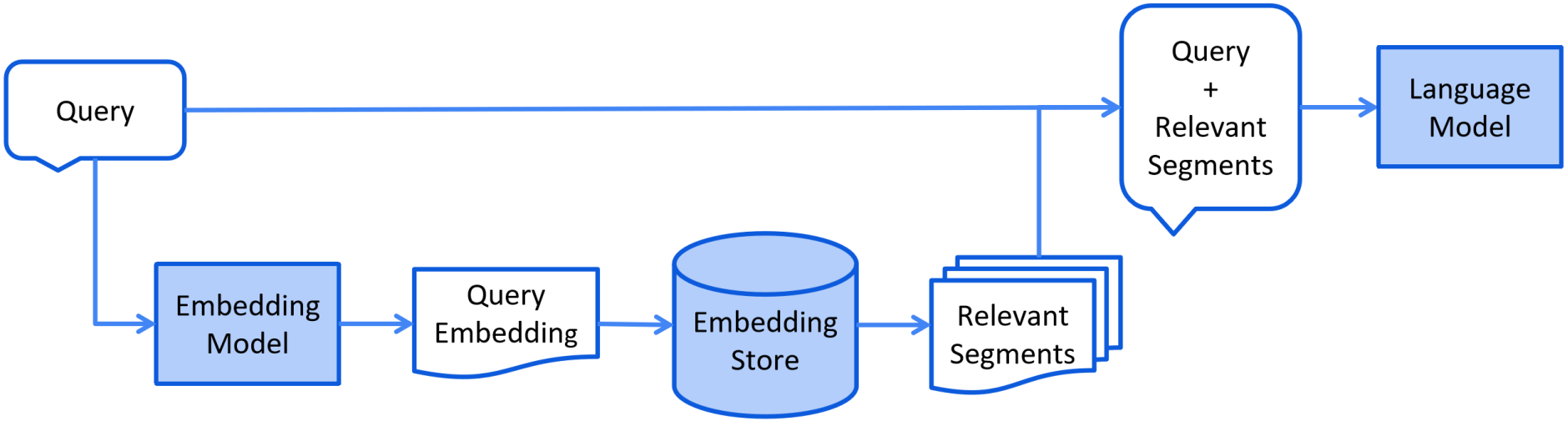
…and enough context

(docs, emails, other data)

RAG Ingestion



Retrieval Augmented Generation



JakartaEE meets AI

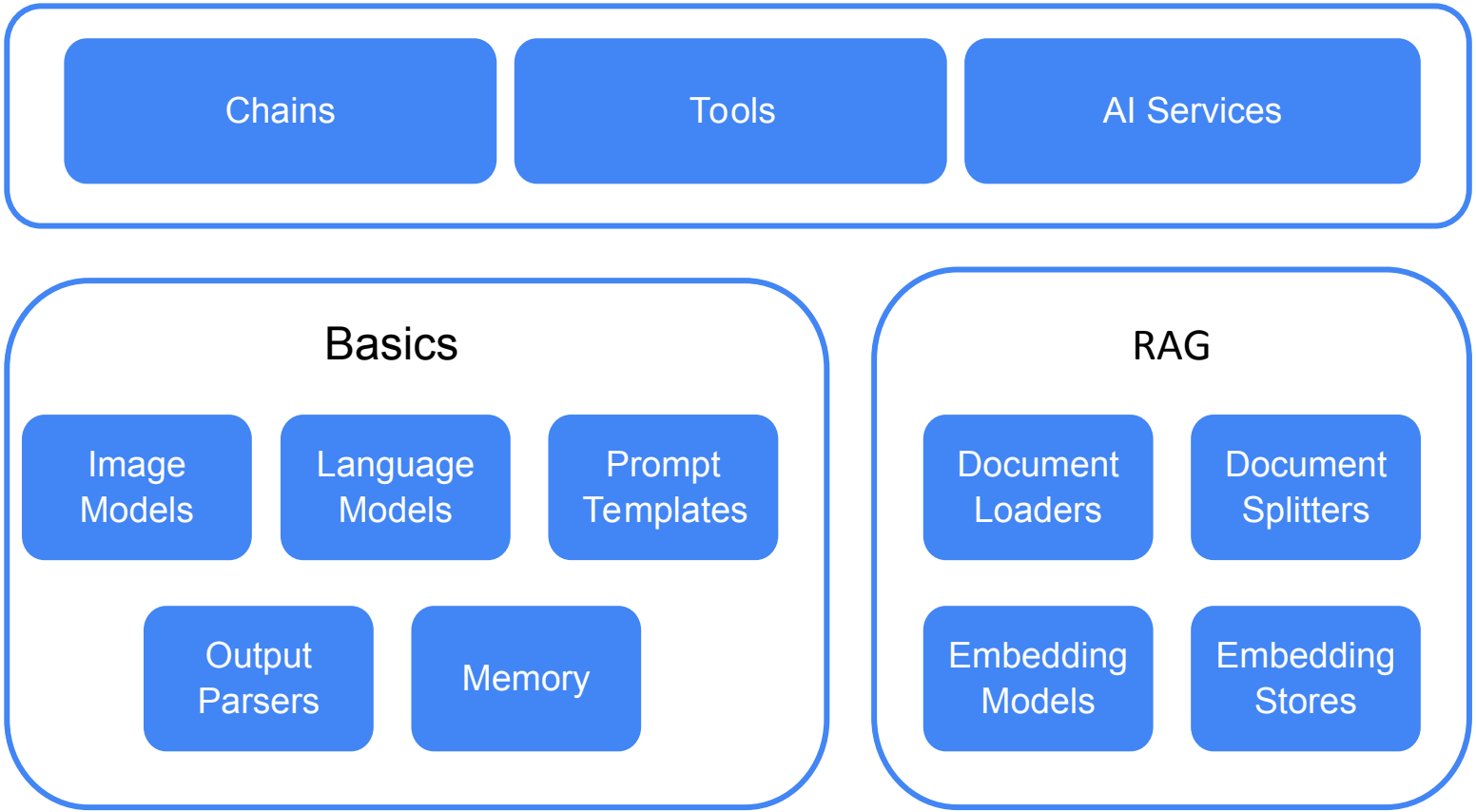
WildFly

Java

python

LangChain

Components of LangChain4j



JBoss CLI Code


```
[standalone@localhost:9990 /] /subsystem=ai/ollama-chat-model=ollama:add(model-
name="llama3.1:8b", base-url="http://127.0.0.1:11434", temperature="0.9")
{"outcome" => "success"}

[standalone@localhost:9990 /] /subsystem=ai/ollama-chat-model=ollama:read-resource
{
  "outcome" => "success",
  "result" => {
    "base-url" => "http://127.0.0.1:11434",
    "connect-timeout" => 0,
    "model-name" => "llama3.1:8b",
    "temperature" => 0.9
}
```

Sample configuration


```
    <subsystem xmlns="urn:jboss:domain:ai:1.0">
      <chat-language-models>
        <ollama-chat-model name="ollama" base-url="http://127.0.0.1:11434" log-requests="true" log-
responses="true" model-name="llama3.1:8b" temperature="0.9"/>
      </chat-language-models>
      <embedding-models>
        <in-memory-embedding-model name="all-minilm-l6-v2" module="dev.langchain4j.embeddings.all-minilm-
l6-v2" embedding-class="dev.langchain4j.model.embedding.onnx.allminilml6v2.AllMiniLmL6V2EmbeddingModel"/>
      </embedding-models>
      <embedding-stores>
        <in-memory-embedding-store name="in-memory" path="embeddings.json" relative-
to="jboss.server.config.dir"/>
      </embedding-stores>
      <content-retrievers>
        <embedding-store-content-retriever name="embedding-store-retriever" embedding-model="all-minilm-
l6-v2" embedding-store="in-memory" max-results="2" min-score="0.7"/>
      </content-retrievers>
    </subsystem>
```

RAG code


```
@Inject
@Named(value = "embedding-store-retriever")
ContentRetriever retriever;

RetrievalAugmentor augmentor;
private static final String PROMPT_TEMPLATE = "You are a WildFly expert who understands well
how to administrate the WildFly server and its components\n"
            + "Objective: answer the user question delimited by  ---\n"
            + "---\n"
            + "{{userMessage}}\n"
            + "---"
            + "\n Here is a few data to help you:\n"
            + "{{contents}}";

    @PostConstruct
    public void createBasicRag() {
        augmentor = DefaultRetrievalAugmentor.builder()
                .contentRetriever(retriever)
                .contentInjector(DefaultContentInjector.builder()
                        .promptTemplate(PromptTemplate.from(PROMPT_TEMPLATE)).build())
                .queryRouter(new DefaultQueryRouter(retriever)).build();
    }
```

RAG Code 2

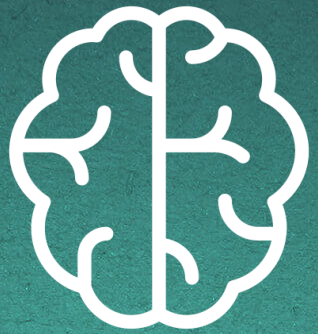

```
    @Inject
    @Named(value = "ollama")
    ChatLanguageModel chatModel;
    RetrievalAugmentor augmentor;

    @OnMessage
    public String sayHello(String question, Session session) throws IOException {
        ChatMemory chatMemory =
MessageWindowChatMemory.builder().id(session.getUserProperties().get("httpSessionId")).maxMessages
(3).build();
        ConversationalRetrievalChain chain = ConversationalRetrievalChain.builder()
                .chatLanguageModel(chatModel)
                .chatMemory(chatMemory)
                .retrievalAugmentor(augmentor)
                .build();
        String result = chain.execute(question).replace("\n", "<br/>");
        return result;
    }
```

AI Service


```
@RegisterAIService(chatLanguageModelName = "ollama", tools =
{org.wildfly.ai.websocket.Weather.class,org.wildfly.ai.websocket.Calculator.class}, scope =
SessionScoped.class)
public interface SimpleAIService {
    @SystemMessage("""
                   You are an AI named Bob answering general question.
                   Your response must be polite, use the same language as the question, and be
relevant to the question.""")
    String chat(@UserMessage String question);
}
```

Provisioning


```
<plugin>
    <groupId>org.wildfly.plugins</groupId>
    <artifactId>wildfly-maven-plugin</artifactId>
    <version>${version.wildfly.maven.plugin}</version>
    <configuration>
        <feature-packs>
            <feature-pack>
                   <location>org.wildfly:wildfly-galleon-pack:${version.wildfly.bom}</location>
            </feature-pack>
            <feature-pack>
                <location>org.wildfly:wildfly-ai-feature-pack:${version.wildfly.ai.feature.pack}</location>
            </feature-pack>
        </feature-packs>
        <layers>
            <layer>cloud-server</layer>
            <layer>ollama-chat-model</layer>
            <layer>default-embedding-content-retriever</layer>
            <!-- Providing the following layers
            <layer>in-memory-embedding-model-all-minilm-l6-v2</layer>
            <layer>in-memory-embedding-store</layer>
            -->
            <!-- Existing layers thart can be used:
            <layer>ollama-embedding-model</layer>
            <layer>openai-chat-model</layer>
            <layer>mistral-ai-chat-model</layer>
            <layer>neo4j-embedding-store</layer>
            <layer>weaviate-embedding-store</layer>
            <layer>web-search-engines</layer>
            -->
        </layers>
...
</plugin>
```



Demo

Emmanuel Hugonnet | ehugonne@redhat.com

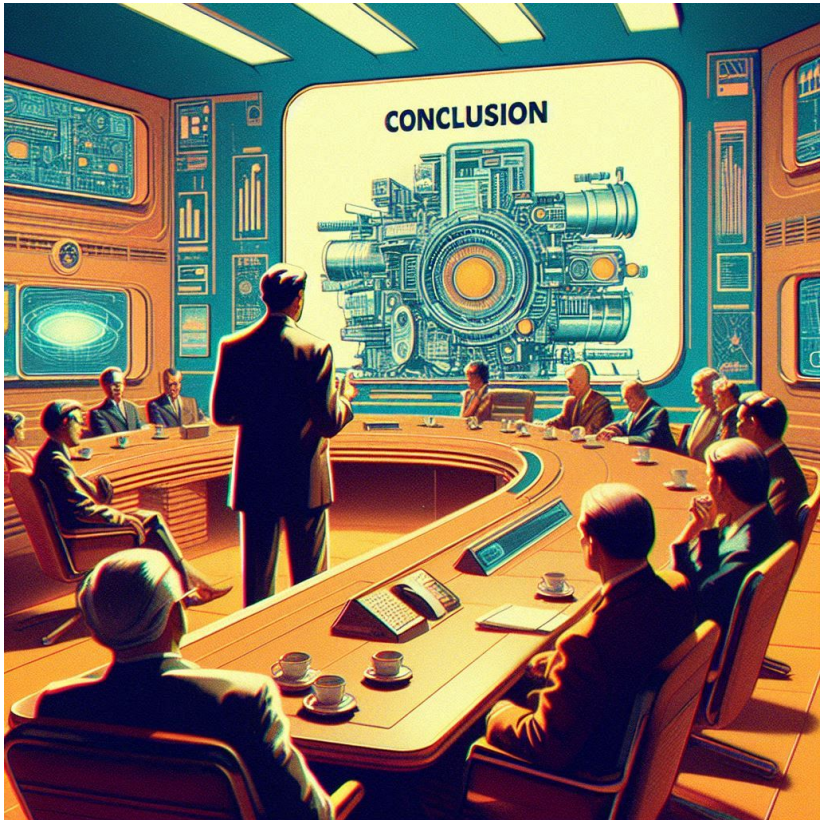
WildFly

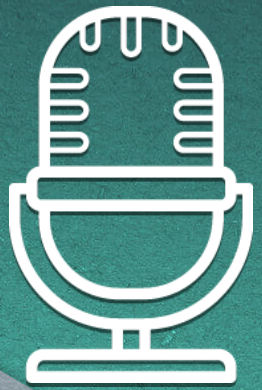
What’s next

- Multi modal suuport
- Chat Streaming support
- OpenTelemetry integration
- Replace OkHTTP by JDK or JAXRS clients

Conclusion

- Use WildFly Management to create required resources
- Layers to trim down the configuration to our needs
- CDI injection of those resources
- AI Service support





Questions

Emmanuel Hugonnet | ehugonne@redhat.com

WildFly