```
from swarms import MixtureOfAgents, Agent

from jamba_swarm.jamba_llm import Jamba


model = Jamba(

    max_tokens=4000,

)


jamba_prompt = """


from jamba_swarm.jamba_llm import Jamba


model = Jamba(

    max_tokens=4000,

)


# Run jamba

out = model.run(

    "Your task goes here",

)


"""


# System Prompts

app_designer_prompt = (

        "You are AppDesigner, responsible for designing the overall structure and layout of the
application. "
```

```
    "Your tasks include defining the user interface (UI) components, navigation flow, and ensuring that the design "
    "is user-friendly, visually appealing, and functional. You must consider the best practices for UI/UX design, "
    "accessibility standards, and ensure that the design is scalable for future enhancements. Provide a detailed "
    "blueprint of the application's architecture, including wireframes, mockups, and any design specifications that "
    "are necessary for the development team to implement the design accurately."
)

feature_engineer_prompt = (
    "You are FeatureEngineer, responsible for defining and implementing the features of the application. "
    "Your tasks include identifying the core functionalities that the application should offer, creating detailed "
    "feature specifications, and ensuring that each feature aligns with the overall goals of the project. You must "
    "consider the technical feasibility, user needs, and integration with existing systems. Provide a comprehensive "
    "list of features with detailed descriptions, user stories, and any necessary technical requirements. Additionally, "
    "outline the steps required for implementing each feature and any potential challenges or considerations that need "
    "to be addressed during development."
)
```

```
code_generator_prompt = (
    "You are CodeGenerator, responsible for generating the Python code for the application based on the design and features. "
    "Your tasks include translating the design specifications and feature requirements into clean, efficient, and maintainable "
    "Python code. You must follow best practices for software development, including code organization, documentation, and testing. "
    "Ensure that the code is modular, reusable, and adheres to the project's coding standards. Provide the complete source code for "
    "the application, along with any necessary configuration files, dependencies, and instructions for setting up and running the application in python code. Only generate the code only"
    f"The code should be well-structured, commented, and easy to understand. The code must also only use Jamba model for everything {jamba_prompt}"
)


quality_assurance_prompt = (
    "You are QualityAssurance, responsible for testing and ensuring the quality of the generated code. "
    "Your tasks include performing thorough testing of the application, identifying and reporting bugs, and verifying that all features "
    "function as intended. You must create and execute test cases, perform code reviews, and ensure that the application meets the defined "
    "quality standards. Provide detailed test reports, including the results of functional, performance, and security testing. Additionally, "
    "recommend any improvements or fixes needed to enhance the overall quality and reliability of
```

```
the application."
)



# nitialize AppDesigner
app_designer = Agent(
    agent_name="AppDesigner",
    system_prompt=app_designer_prompt,
    llm=model,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    context_length=150000,
    state_save_file_type="json",
    saved_state_path="app_designer.json",
)

# Initialize FeatureEngineer
feature_engineer = Agent(
    agent_name="FeatureEngineer",
    system_prompt=feature_engineer_prompt,
    llm=model,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    context_length=150000,
    state_save_file_type="json",
    saved_state_path="feature_engineer.json",
)

# Initialize CodeGenerator
code_generator = Agent(
    agent_name="CodeGenerator",
    system_prompt=code_generator_prompt,
    llm=model,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    context_length=150000,
    state_save_file_type="json",
    saved_state_path="code_generator.json",
)

# Initialize QualityAssurance
quality_assurance = Agent(
    agent_name="QualityAssurance",
    system_prompt=quality_assurance_prompt,
    llm=model,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    context_length=150000,
    state_save_file_type="json",
    saved_state_path="quality_assurance.json",
)


def run_jamba_swarm(task: str = None):
    # Initialize the MixtureOfAgents with verbose output and auto-save enabled
    moe_swarm = MixtureOfAgents(
        agents=[
            app_designer,
            feature_engineer,
            code_generator,
            quality_assurance,
        ],
        final_agent=quality_assurance,
        verbose=True,
        layers=3,
    )

    # Run the swarm
    return moe_swarm.run(task)
```

```
out = run_jamba_swarm(
    "Create an open source API server that can host Jamba with context for agents "
)
```