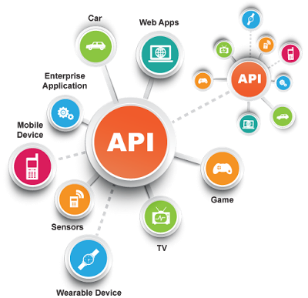
**SE495 Software and Systems Integration**

**Chapter 2: Enterprise Application Architecture (EAI) - Guided Project: Implementing a Publish-Subscribe Pattern for a Messaging System**

*Objective*: In this project, students will learn about the Publish-Subscribe pattern used in EAI and implement a simple messaging system using a message broker.

*Requirements*:

* Basic understanding of Python programming
* Familiarity with messaging systems and EAI patterns

*Tools and Technologies*:

* Python 3.x
* RabbitMQ message broker (or any other message broker of your choice)
* pika library in Python

**Step-by-step guide**:

1. **Install RabbitMQ**: Follow the installation guide for RabbitMQ on the official website (<https://www.rabbitmq.com/download.html>) to install and set up the message broker on your machine.
2. **Install the pika library**: In your Python environment, use the following command to install the pika library:

pip install pika

```

1. **Create a Python script for the publisher**: Write a Python script that uses the pika library to connect to the RabbitMQ server and publish messages to an exchange named "notifications". The messages should contain a topic and a payload (e.g., "tasks.new" and "A new task was created").
2. **Create a Python script for the subscriber**: Write another Python script that uses the pika library to connect to the RabbitMQ server and subscribe to the "notifications" exchange. The script should print the received messages to the console.
3. **Implement multiple subscribers**: Modify the subscriber script to allow multiple subscribers to listen to different topics. For example, one subscriber can listen to "tasks.new" messages, and another subscriber can listen to "tasks.completed" messages.
4. **Test the messaging system**: Run the publisher script and verify if the messages are being sent to the exchange. Then, run multiple instances of the subscriber script with different topics to ensure that they receive and print the relevant messages.
5. **Optional: Integrate the messaging system with the project from Chapter 1**: Modify the Python script from Chapter 1 to publish messages to the RabbitMQ exchange instead of directly sending messages to the Slack channel. Then, create a new subscriber that listens for messages and sends them to the Slack channel. This will demonstrate how the Publish-Subscribe pattern can be used to decouple components in an enterprise application architecture.