Getting Started with PyTM

To begin using PyTM for threat modeling, follow these steps:

Step 1: Installation

First, ensure you have Python installed on your system. You can install PyTM using pip:

```
pip install pytm
```

Step 2: Code your threat model

1. Define system component: start by defining the components of your system. This includes identifying assets, processes, and data flows. Here's a simple example:

```
from pytm import TM, Actor, Process, Dataflow

tm = TM("My System Threat Model")
user = Actor("User")
process = Process("Web Application")
dataflow = Dataflow(user, process, "User Data")
```

2. Identify your threats: next, identify potential threats to your system. PyTM provides a variety of built-in threats, but you can also define custom threats based on your specific context.

```
from pytm import Threat

threat = Threat("Unauthorized Access", "An attacker gains access to user data.")
```

3. Analyze and mitigate: once you have identified threats, analyze their potential impact and likelihood. Based on this analysis, you can propose mitigations to reduce the risk.

```
from pytm import Mitigation

mitigation = Mitigation("Implement Authentication", "Use strong authentication
mechanisms.")
```

Step 3: Leverage LLMs for Threat Modeling

> **Example Prompt***
>
> ***Scenario***:*
>
> *You are working on developing a new {**web application**} that includes various functionalities for different types of users and integrates with external services. The application enables users to perform certain actions {*(e.g., view personal information, submit requests, manage data)***}, while administrators or managers can perform more elevated tasks (e.g., approving*

actions, generating reports). The system also integrates with external services, such as payment gateways, government portals, or third-party APIs.

Task:

Your task is to develop a threat model for this web application using a threat modeling tool such as **PyTM***. Consider all aspects of the application, including both user-facing and administrative functionalities, as well as integration points with external services. The threat model should cover potential security risks, threats, and vulnerabilities, as well as provide mitigation strategies.*

Deliverables: *Threat Model defined by PyTM Code*