Arquivo: controller.py

```
from PyQt6.QtCore import *

class GerarAtasApiController(QObject):
    def __init__(self, icons, view, model):
        super().__init__()
        self.icons = icons
        self.view = view
        self.model = model.setup_model("controle_atas_api")
        self.setup_connections()

    def setup_connections(self):
        # Conecta os sinais da view aos métodos do controlador
        self.view.instructionSignal.connect(self.instrucoes)
        self.view.trSignal.connect(self.termo_referencia)
        self.view.homologSignal.connect(self.termo_homologacao)
        self.view.sicafSignal.connect(self.sicaf_widget)
        self.view.apiSignal.connect(self.api_consulta)
        self.view.atasSignal.connect(self.gerar_atas)

        # Configura os botões usando GerarAtasModel
        self.view.tr_widget.abrirTabelaNova.connect(self.abrir_tabela_nova)
        self.view.tr_widget.configurarSqlModelSignal.connect(self.configurar_sql_model)

        # Conecta o sinal de carregar tabela ao método do modelo
        self.view.tr_widget.carregarTabela.connect(self.caregar_tabela_com_dados)

        # Conecta o sinal `tabelaCarregada` para configurar o modelo SQL após carregar a tabela
        self.model.tabelaCarregada.connect(self.configurar_sql_model)

    def configurar_sql_model(self):
        # Aplica `self.model` ao `table_view` diretamente
        self.view.tr_widget.table_view.setModel(self.model)
        self.view.configurar_visualizacao_tabela_tr(self.view.tr_widget.table_view)

    def abrir_tabela_nova(self):
        # Lógica para abrir uma nova tabela
        self.model.abrir_tabela_nova()

    def caregar_tabela_com_dados(self):
        # Lógica para abrir uma nova tabela
        self.model.carregar_tabela()

    def instrucoes(self):
        # Atualiza para o widget de instruções
        self.view.content_area.setCurrentWidget(self.view.instrucoes_widget)

    def termo_referencia(self):
        # Define o widget atual para Termo de Referência
        self.view.content_area.setCurrentWidget(self.view.tr_widget)
        # Configura o QTableView com o modelo e ajusta a visualização
```

```
    self.view.tr_widget.table_view.setModel(self.model)
    self.view.configurar_visualizacao_tabela_tr(self.view.tr_widget.table_view)

def termo_homologacao(self):
    # Exibe o widget de Processamento para Termo de Homologação
    self.view.content_area.setCurrentWidget(self.view.homolog_widget)

def sicaf_widget(self):
    # Exibe o widget de Processamento para Termo de Homologação
    self.view.content_area.setCurrentWidget(self.view.sicaf_widget)

def api_consulta(self):
    # Atualiza para o widget de consulta API
    self.view.content_area.setCurrentWidget(self.view.api_widget)

def gerar_atas(self):
    # Atualiza para o widget de geração de atas
    self.view.content_area.setCurrentWidget(self.view.atas_widget)

def indicadores(self):
    # Atualiza para o widget de indicadores
    self.view.content_area.setCurrentWidget(self.view.indicadores_widget)
```