Exercise: You have a Python notebook containing a pre-trained machine learning model that can predict the price of a cake based on its radius (in centimeters), number of layers, and topping. Your task is to create a Flask API that exposes the following endpoints:

1. Cake Price Prediction Endpoint:

Create an API endpoint **/predict** that accepts three parameters: radius (float), layers (integer), and topping (string).

Load the pre-trained model from the provided notebook.

Implement the endpoint to take the input parameters, pass them to the loaded model, and return the predicted cake price.

Include input validation to ensure that the parameters are of the correct data types and within acceptable ranges.

Topping Recommendations Endpoint:

1. Create an API endpoint **/toppings** that accepts two parameters: radius (float) and layers (integer).

Implement logic to generate a list of five recommended toppings based on the provided radius and layers.

For each recommended topping, include an associated price.

Return the list of recommended toppings and their prices as a JSON response.

Include input validation to ensure that the parameters are of the correct data types and within acceptable ranges.

Model Information Endpoint:

1. Create an API endpoint **/model-info** that does not accept any parameters.

Implement the endpoint to return a JSON response containing information about the machine learning model used for cake price prediction.

The JSON response should include the following information:

Python version

Model version (V1)

Any other relevant information about the model

1. Swagger Documentation:

Create Swagger documentation for all the API endpoints you have created, including descriptions of the endpoints, input parameters, and expected responses.

Make sure to include proper error handling and response codes in your Flask API implementation.