Program Analysis report for flask calculator

Program Logic
The calculator will use 1 or 2 numbers and a type of operation. Some operations works as python operators and others are python functions from the **math** module except PI which is a constant.

The math modules functions use a single number where are operators use 2 numbers as operands

Operation	Type	Python equivalent
Addition	Operator	+
Subtraction	Operator	-
Multiplication	Operator	*
Division	Operator	/
Modulus	Operator	%
Exponential	Operator	**
log	function	math.log()
sin	function	math.sin()
cos	function	math.cos()
tan	function	math.tan()
pi	constant	math.pi

User interface
The user interface of the program is a simple form with 3 textboxes, 1 dropdown, 2 checkboxes and 1 button and one anchor <a> tag:

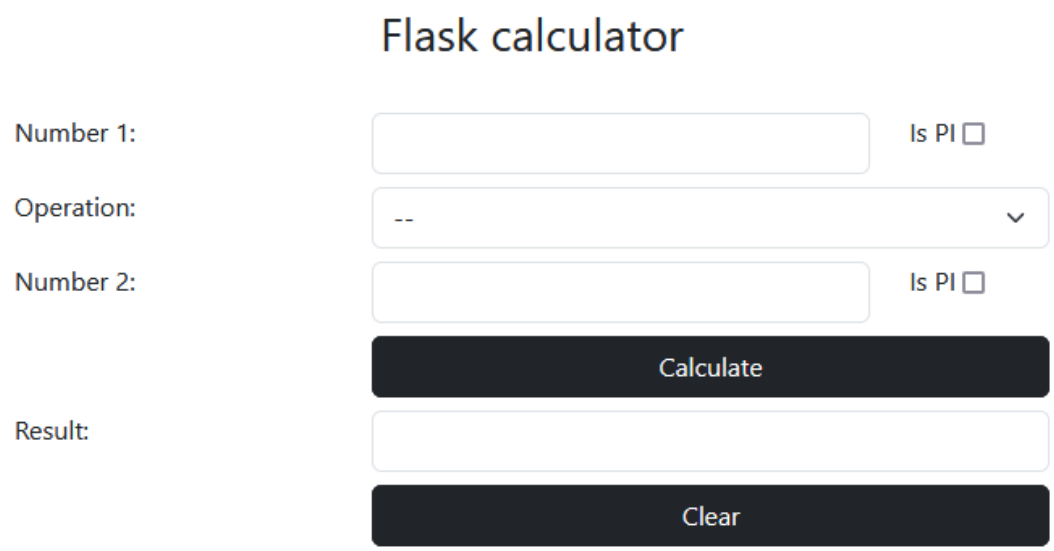
Input type	Label	Function
Textbox	Number 1	First number for operation – required all around
Textbox	Number 2	Second number for operation – required only for operators
Checkbox	Is PI	Set the value of textbox number 1 = math.pi
Checkbox	Is PI	Set the value of textbox number 2 = math.pi
Select	Operations	List of operations
Submit	Calculate	Validate the input fields and calculate the result
Textbox	Result	Displays the calculated result
Anchor tag	Clear	Redirects to inital root to clear the fields

Additional libraries
Bootstap css version 5.3.2 was used to style the html form and error notificactions.

jQuery version 1.9.1 was used to set / unset of value of PI to each of the number input fields

User interface

A screenshot of the user interface is shown below:



Server side validation

The following rules were applied:

- Both input fields must be numeric or float type
- If an operator is selected as operation, both input fields are required
- If a math function is selected as operation only the first number is required

Error notifications were generated using the **flash** function of the flask module

Flask routing, form requests and templating

Routing
The program uses 2 routes namely:

- "/" for landing page and also to clear form after an operation was made
- "/calculate" to display errors notifications and the calculated result

Form request
The html form uses POST as method. The posted values were sent as template vars to fill the input fields after calculation

Templating
The decorator @context_processor was used to send the common data accros the different routes.