


```
class AttributeValidationMeta(type):
    def __new__(cls, name, bases, dct):
        # Iterate through attributes in the class dictionary
        for attribute_name, attribute_value in dct.items():
            # Check if the attribute is not a special method (e.g., __init__)
            if not attribute_name.startswith("__"):
                # Perform custom validation on the attribute value
                if not isinstance(attribute_value, (int, float, str)):
                    raise TypeError(f"Attribute '{attribute_name}' must be of type int, float, or str.")

        # Create the class using the default type.__new__
        return super().__new__(cls, name, bases, dct)

# Example usage of the metaclass
class MyClass(metaclass=AttributeValidationMeta):
    # Attributes with custom validation
    age = 25
    name = "John"

    # This would raise a TypeError during class creation
    # as 'salary' is not of type int, float, or str
    salary = [50000]

# Instantiate the class
my_instance = MyClass()
```

 Log: User created
Saving to database: {'user_id': 1, 'name': 'John'}