Django Trainee at Accuknox

Question 1: Are Django signals executed synchronously or asynchronously by default?

Django Signals are executed synchronously.

Example : Pre Save : Modifies the User model before it is saved to the database.

```
from django.db.models.signals import pre_save
from django.dispatch import receiver
from django.contrib.auth.models import User

@receiver(pre_save, sender=User)
def user_save_bf(sender, instance, **kwargs):
    print("Before saving the user")
    print(instance.username)
```

Example : Post Save : After the model instance is saved to the database.

```
from django.db.models.signals import post_save
from django.dispatch import receiver
from django.contrib.auth.models import User

@receiver(post_save, sender=User)
def user_save_af(sender, instance, **kwargs):
    print("After saving the user")
    print(instance.username)
```

Create a new user

```
user=User.objects.create(username='Teacher1')
```

Question 2: Do Django signals run in the same thread as the caller?

The signal and the original action (Saving a user) happen together in sequence. It is Possible.

Example : Signal & Saving a user in same time.

```
import threading
from django.db.models.signals import post_save
from django.dispatch import receiver
from django.contrib.auth.models import User

@receiver(post_save, sender=User)
def signal_pass(sender, instance, **kwargs):
    print(f"Signal is running : {threading.current_thread().name}")


print(f"Caller : {threading.current_thread().name}")
user=User.objects.create(username="Teacher1")
```

Question 3: Do Django signals run in the same database transaction as the caller?

Django signals do not automatically run in the same database transaction . We Can Control the Signal is exected in relation.

User is save to database but signal is pass after the transaction is Completed.

Example:

```
from django.db import transaction
from django.db.models.signals import post_save
from django.dispatch import receiver
from django.contrib.auth.models import User

@receiver(post_save, sender=User)
def user_created(sender, instance, **kwargs):
    print("Signal : User saved.")

def user_create():
    user=User(username="Teacher1")
    user.save()

    transaction.on_commit(print("Transaction Completed"))
```

Topic: Custom Classes in Python

Description: You are tasked with creating a Rectangle class with the following requirements:

An instance of the Rectangle class requires length:int and width:int to be initialized.We can iterate over an instance of the Rectangle class

When an instance of the Rectangle class is iterated over, we first get its length in the format: {'length': <VALUE_OF_LENGTH>} followed by the width {width: <VALUE_OF_WIDTH>} make this as a simple human format. simple code and

Example:

```
class Rectangle:
    def __init__(self, length: int, width: int):
        self.length = length
        self.width = width

    def __iter__(self):
        yield {'length': self.length}
        yield {'width': self.width}


rect = Rectangle(100, 5)
for dimension in rect:
    print(dimension)
```