Question 1 : 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_handler(sender, instance, created, **kwargs):
    if created:
        print(f'User {instance.username} has been created!')
```

Question 2 : 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 user_created_handler(sender, instance, created, **kwargs):
    print(f"Signal handler running in thread: {threading.current_thread().name}")
    if created:
        print(f"User {instance.username} has been created!")

# Simulating a user creation in the main thread
print(f"Main execution thread: {threading.current_thread().name}")
user = User.objects.create(username="testuser", password="securepassword")
```

Quesrion 3 : 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_handler(sender, instance, created, **kwargs):
    if created:
        print(f"User {instance.username} has been created!")
        instance.first_name = "UpdatedName"
        instance.save()  # This change is part of the same transaction

try:
    with transaction.atomic():
        user = User.objects.create(username="testuser",
password="securepassword")
        raise Exception("Simulating an error")  # This will trigger a rollback
except Exception as e:
    print(f"Transaction failed: {e}")

# Checking if the user exists
print(User.objects.filter(username="testuser").exists())  # Output: False
```