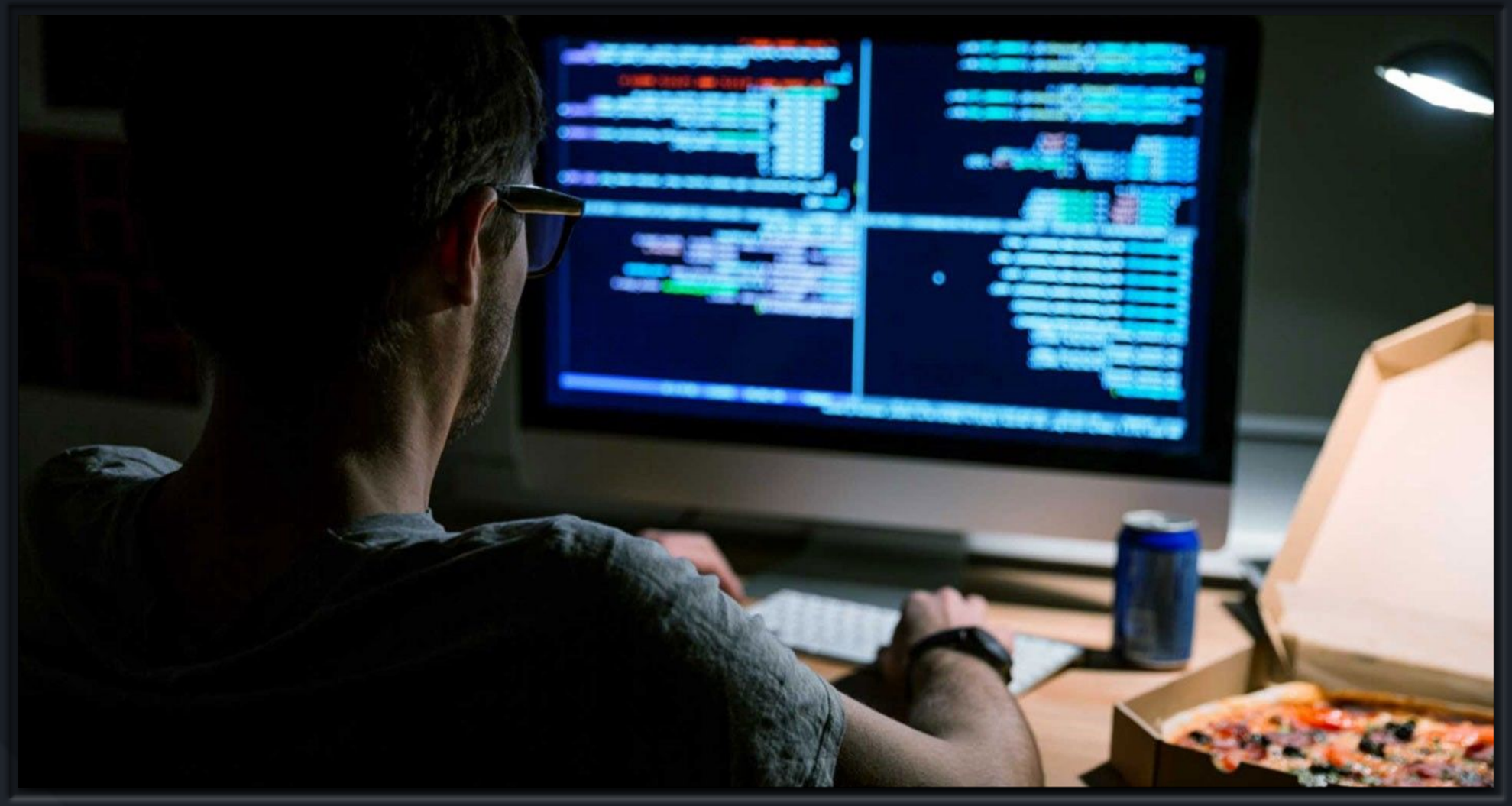
INTRO TO BACKEND DEVELOPMENT

DAY 13

04/26/2021

Instructor - Casey Wilson
TA - Kevin Dublin

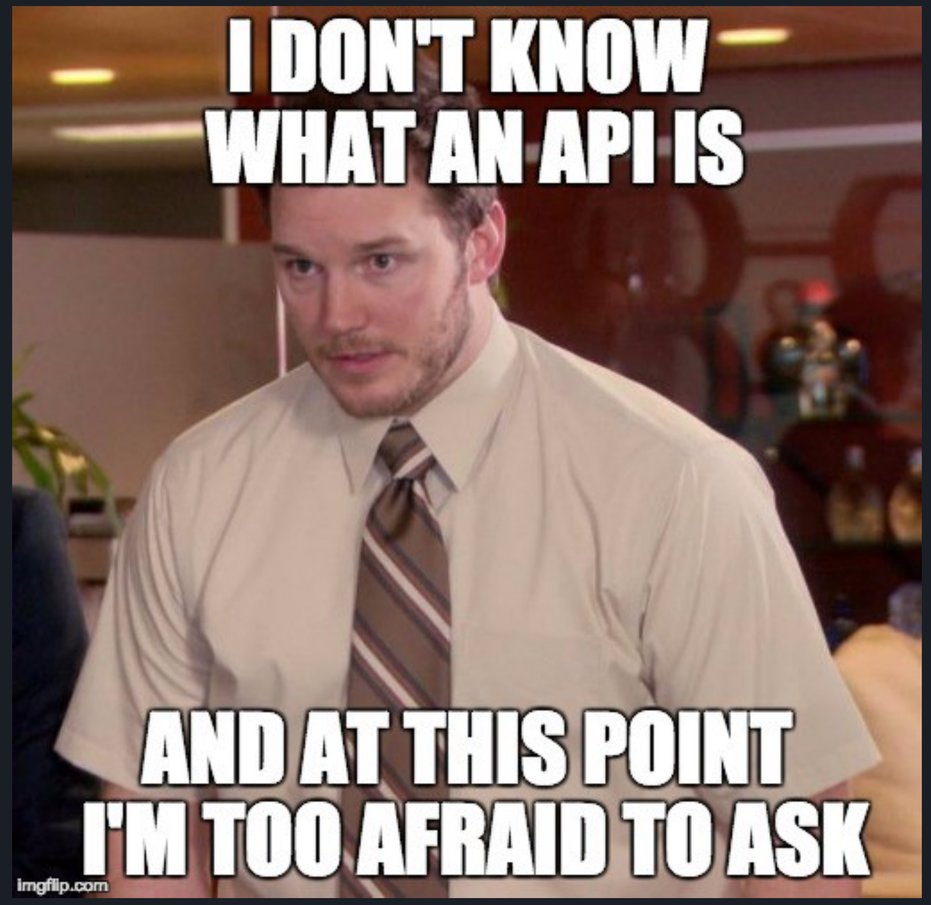
Take Home Challenge Review



Check In Time

Django Time - Round 4

Django - DRF Example



https://github.com/davidluque1/DRF-SIMPLE-CRUD
https://github.com/royalpranjal/Django-Rest-Framework/tree/master/test1

Django DRF - Serializer

```
#serializers.py

class AccountSerializer(serializers.ModelSerializer):
    class Meta:
        model = Account
        fields = ['id', 'account_name', 'users', 'created']
```

Django DRF - Generic View

```python
#views.py

class UserList(generics.ListCreateAPIView):
    queryset = User.objects.all()
    serializer_class = UserSerializer
    permission_classes = [IsAdminUser]
```

Django DRF - Viewset

```python
#views.py

class UserViewSet(viewsets.ModelViewSet):
    """
    A viewset for viewing and editing user instances.
    """
    serializer_class = UserSerializer
    queryset = User.objects.all()
```

Django DRF - Routing

```python
#urls.py

from django.urls import include, path
from rest_framework import routers
from tutorial.quickstart import views

router = routers.DefaultRouter()
router.register(r'users', views.UserViewSet)
router.register(r'groups', views.GroupViewSet)

# Wire up our API using automatic URL routing.
# Additionally, we include login URLs for the browsable API.
urlpatterns = [
    path('', include(router.urls)),
    path('api/', include('rest_framework.urls', namespace='rest_framework'))
]
```

Django DRF - Nested Serializers

```python
# models.py
class Album(models.Model):
    album_name = models.CharField(max_length=100)
    artist = models.CharField(max_length=100)

class Track(models.Model):
    album = models.ForeignKey(Album, related_name='tracks', on_delete=models.CASCADE)
    order = models.IntegerField()
    title = models.CharField(max_length=100)

# serializers.py
class AlbumSerializer(serializers.ModelSerializer):
    tracks = serializers.StringRelatedField(many=True)

    class Meta:
        model = Album
        fields = ['album_name', 'artist', 'tracks']
```

Django DRF - Authentication (Basic)

```python
# views.py

from rest_framework.authentication import SessionAuthentication, BasicAuthentication
from rest_framework.permissions import IsAuthenticated
from rest_framework.response import Response
from rest_framework.views import APIView

class ExampleView(APIView):
    authentication_classes = [SessionAuthentication, BasicAuthentication]
    permission_classes = [IsAuthenticated]

    def get(self, request, format=None):
        content = {
            'user': str(request.user),  # `django.contrib.auth.User` instance.
            'auth': str(request.auth),  # None
        }
        return Response(content)
```

Django DRF - Authentication (Djoser)

```
#  pip install djoser
# urls.py
urlpatterns = [
    (...),
    url(r'^auth/', include('djoser.urls')),
    url(r'^auth/', include('djoser.urls.authtoken')),
]

# settings.py
REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework.authentication.TokenAuthentication',
        (...)
    ),
}
```

Django DRF - Authentication (Knox)

```
# pip install django-rest-knox
# views.py
from rest_framework.permissions import IsAuthenticated
from rest_framework.response import Response
from rest_framework.views import APIView

from knox.auth import TokenAuthentication

class ExampleView(APIView):
    authentication_classes = (TokenAuthentication,)
    permission_classes = (IsAuthenticated,)

    def get(self, request, format=None):
        content = {
            'foo': 'bar'
        }
        return Response(content)
```

Django DRF - Authentication (OAuth)

```
# pip install djangorestframework-oauth
# views.py

@api_view(http_method_names=['POST'])
@permission_classes([AllowAny])
@psa()
def exchange_token(request, backend):
    serializer = SocialSerializer(data=request.data)

    if serializer.is_valid(raise_exception=True):
        user = request.backend.do_auth(serializer.validated_data['access_token'])

        if user:
            token, _ = Token.objects.get_or_create(user=user)
            return Response({'token': token.key})

        else:
            return Response(
                {'errors': {'token': 'Invalid token'}},
                status=status.HTTP_400_BAD_REQUEST,
            )
```

Django DRF - Filtering

```
# pip install django-filters
# views.py
class ProductList(generics.ListAPIView):
    queryset = Product.objects.all()
    serializer_class = ProductSerializer
    filter_backends = [DjangoFilterBackend]
    filterset_fields = ['category', 'in_stock']
```

Django DRF - Pagination

```
#settings.py


REST_FRAMEWORK = {
    'DEFAULT_PAGINATION_CLASS':
'rest_framework.pagination.PageNumberPagination',
    'PAGE_SIZE': 10
}
```

Questions?

Take Home Challenge