Canvas LTI Application Documentation

Md Ashraful Alam

C836836995

- I initiated my new project with the following command:

```
django-admin startproject ProjectName
```

- I moved into the project directory using:

```
cd ProjectName
```

- Within the project directory, I created a new app by executing:

```
python ./manage.py startapp AppName
```

- To configured it according to the models, I ran:

```
python ./manage.py migrate
```

- To ensure everything was set up properly, I started the development server using:

```
python manage.py runserver
```

- Initially, I installed django-sslserver to enable HTTPS:

```
sudo pip install django-sslserver
```

- Afterward, I launched the secure server with:

```
python ./manage.py runsslserver localhost:portNumber
```

- In the Django settings, I included my app and sslserver in the INSTALLED_APPS:

```
INSTALLED_APPS = [
  "myapp.apps.MyappConfig",
  "sslserver"
]
```

- **Routing and Configuration**
- **In myApp/urls.py, defined the LTI launch route:**

```
from django.urls import path
from . import views

urlpatterns = [
    path('lti/', views.index, name='lti_launch'),
]
```

- **In mysite/urls.py, included the app’s URL patterns:**

```
from django.contrib import admin

from django.urls import include, path


urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include('myApp.urls')),

]
```

```
/", admin.site.urls),

]
```

- **I set up the LTI configuration in local_settings.py:**

```
PYLTI_CONFIG =  {'consumers': {
 'consumerKey':
{'secret': 'secretKey'
 }
 }
}
```

- **Set Up the LTI Tool in Canvas**

Step : Configured the LTI Tool in Canvas

In Canvas, go to **Settings** > **Apps** > **View App Configurations**.

Added a new LTI app:

- **Launch URL**: http://127.0.0.1:8000/lti/
- **Consumer Key**: <consumer_key>
- **Shared Secret**: <shared_secret>
- **Custom Fields**

course_id=$Canvas.course.id

user_id=$User.id

Step : Created a View in Django to Handle LTI Launch

- **In myApp/views.py, created a view called index to handle the LTI POST request:**

Canvas API Configuration:

- Set the **Canvas API base URL** and provided an **API token** for authentication.

POST Request Handling (index view):

- This function processes POST requests from Canvas.
- It retrieves course_id and user_id from the request.
- If either is missing, it returns a 400 error.
- It fetches the list of users in the course and, for each user, retrieves their assignment submissions using the Canvas API.
- The data (course ID, users, and their assignments) is returned as a JSON response.

Helper Functions:

- **get_canvas_course_users**: Fetches the list of users in a given course using the Canvas API.
- **get_user_assignments**: Retrieves assignment submission data for a specific user in the course.

4. Running the Application

Start the Django development server to listen for requests:

```
python manage.py runserver
```

My video on YouTube :

https://www.youtube.com/watch?v=kb1ljBU3Ks4