anthology together

Migrating a B2 to LTI & REST

A Deep Dive

July 2022

#AT22

Forward-Looking Statement

Statements regarding our product development initiatives, including new products and future product upgrades, updates or enhancements represent our current intentions, but may be modified, delayed or abandoned without prior notice and there is no assurance that such offering, upgrades, updates or functionality will become available unless and until they have been made generally available to our customers.

Agenda

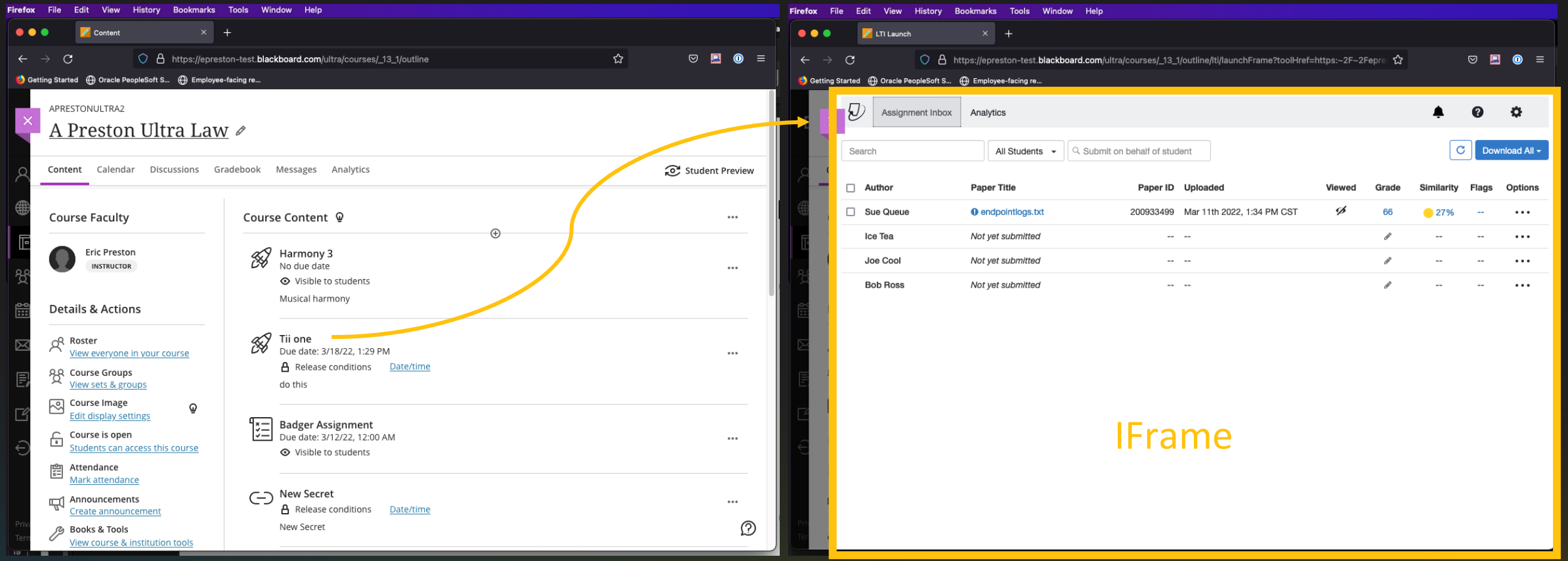
- Introductions
- Brief overview of B2, LTI, & REST
- Steps for building an application
- Breakdown of LTI 1.3 handling
- Using the REST API
- Brief introduction to the Ultra Extension Framework
- Deploying an application with Learn

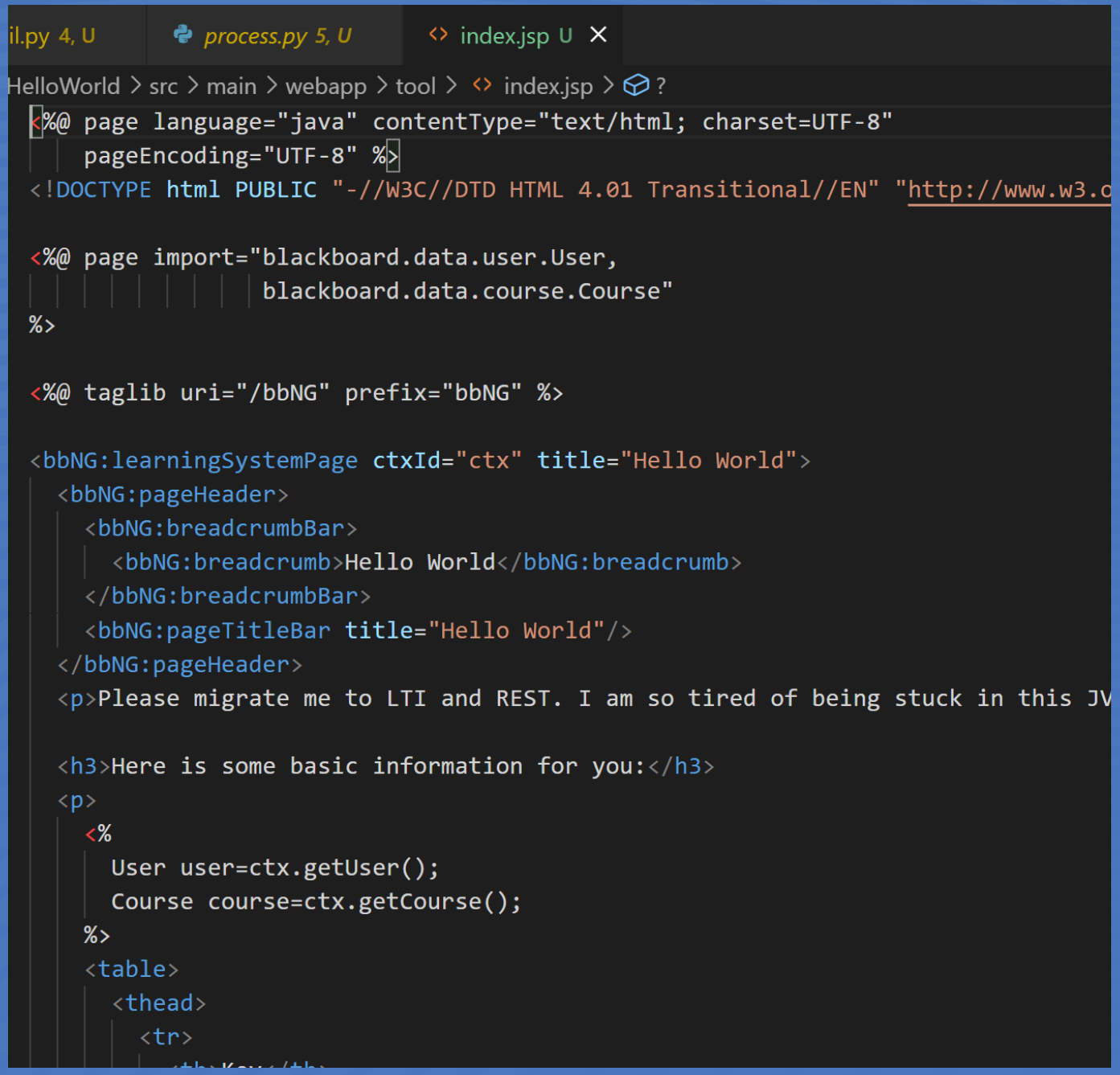
Introductions

Eric Preston

- Anthology + Blackboard + Xythos Staff Software Engineer for 16+ years
- Member of 1EdTech (FKA IMS Global) LTI and Caliper Working Groups
- Fun fact: I was at the northern-most point of the USA in June…Point Barrow, Alaska

The Problem or Opportunity – Connect Two Web Applications





Migrate Me, Please!!

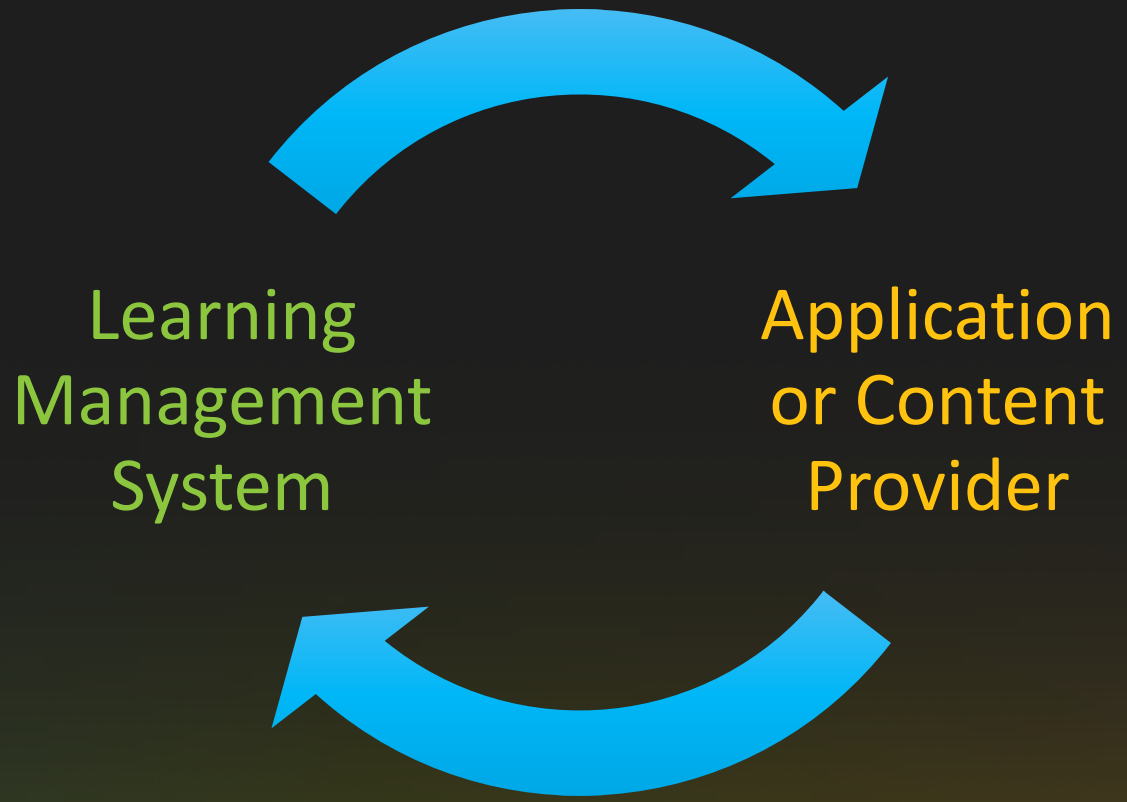
B2's are SO 2000s. Let's figure out how to get this in a modern web application infrastructure using modern integration techniques.

FREEDOM!!

The Solution – LTI & REST API

- A UI flow to connect LMS/VLE (Platform) to an Application (Tool)
- Provides Single-sign On (SSO) capabilities – a TRUST relationship between Platform and Tool
- Services to get data into and from a Platform
  - Content, e.g., Video, Assignment, Book
  - Roster
  - Grades
- Standards-based: works with any* LMS/VLE

TWO VERSIONS: 1.1 and 1.3/Advantage



Meet LTI 1.3/Advantage

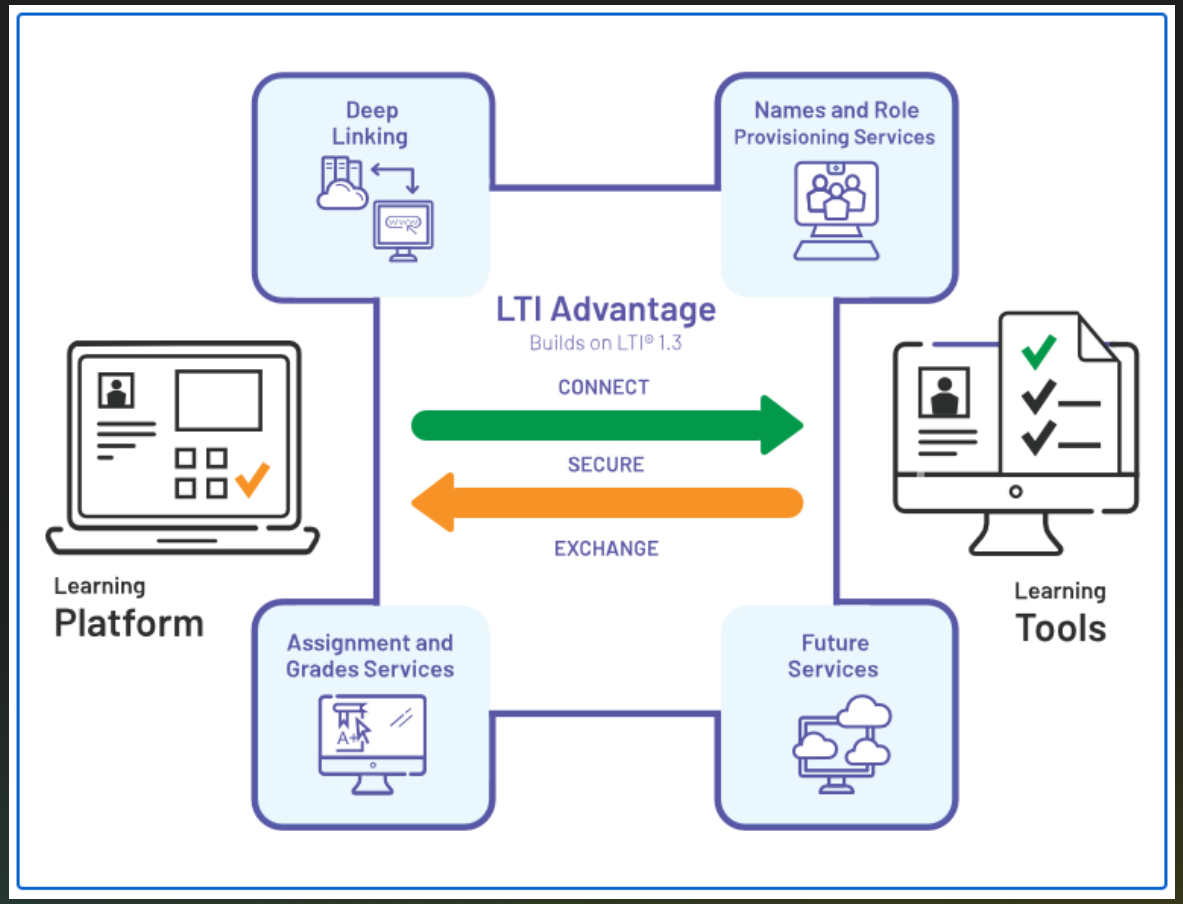
LTI 1.3 Core
The Launch

Deep Linking
Get Content

Names & Roles
Roster & Groups

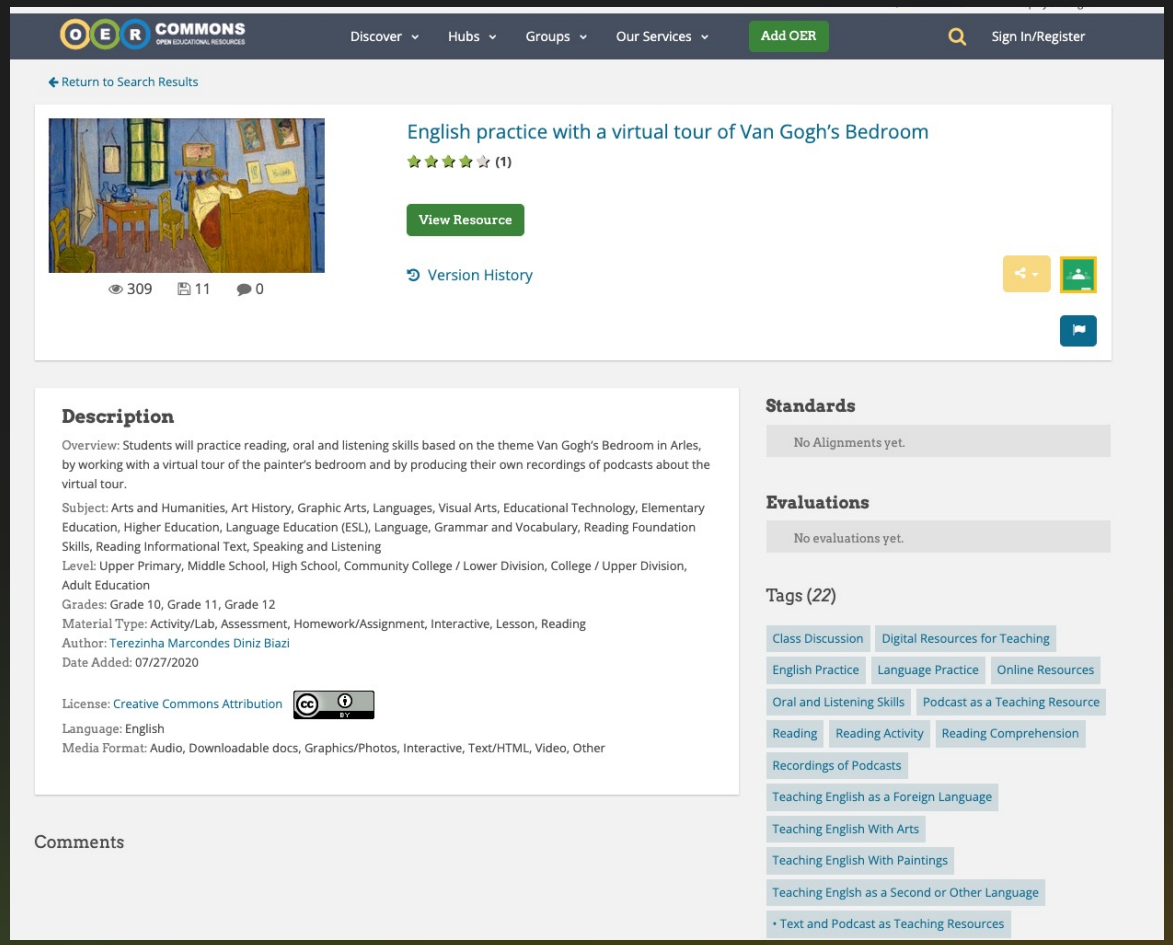
Assignment & Grades
Get & Create grades

Another Look at LTI Advantage



Building an Application

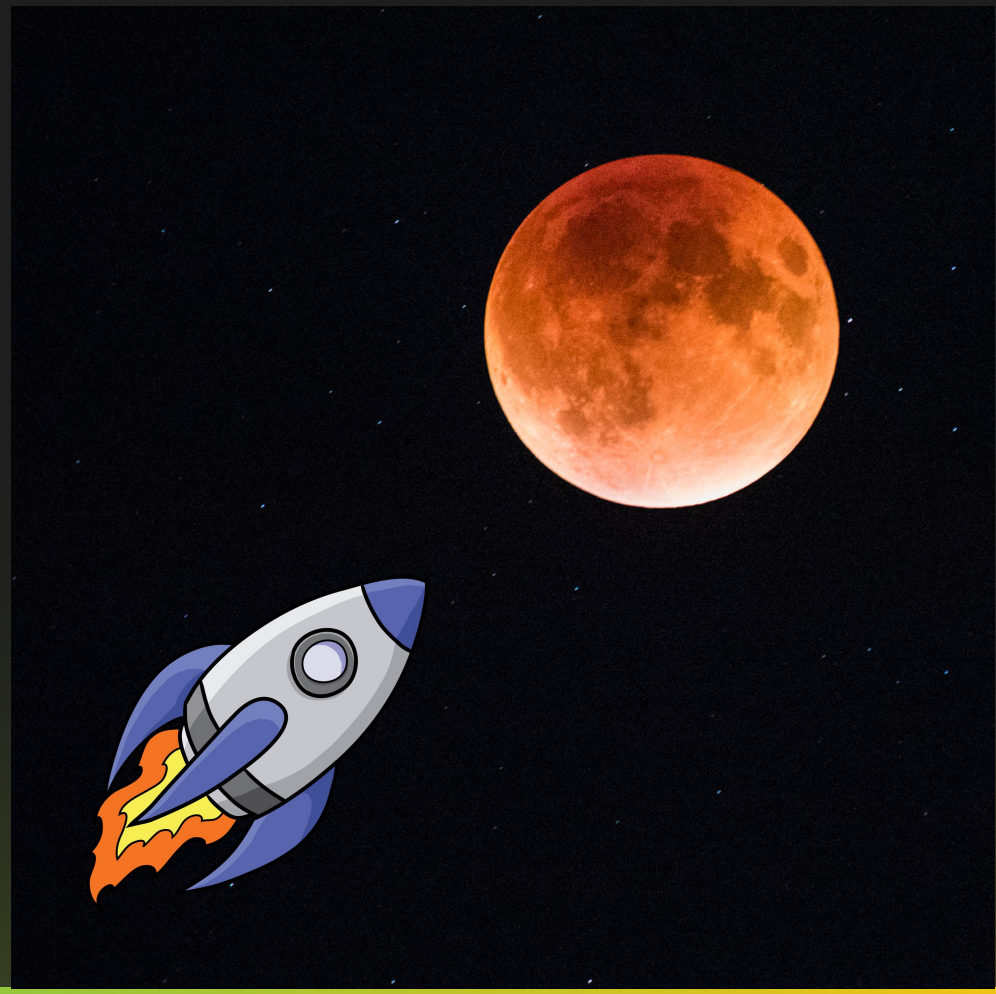
1. Build an awesome web application
   - Language and UI framework of choice
2. Support LTI Launch – accept FORM POST
3. Use REST to get and set Learn data
4. Configure Private and Public keys
5. Deploy it somewhere (AWS, Azure, …)
6. Get IMS Global certification
7. Register it with Blackboard ONCE
8. Deploy it to Learn (Admin)
9. Use it in a course or however you like (Instructors & Students)



LTI Launch – From one Web App to Another

- At its most basic it's an HTTP FORM POST with data – a JSON Web Token (JWT)
- OIDC login flow to protect against CSRF
- Parse JWT – header.body.signature
  - The payload is an OpenID id_token (JWT)
- Validate header
  - Get kid and public key from JWKS URL
  - Get algorithm and validate it is what you expect
- Validate signature
- Get REST & LTI OAuth2 access tokens*
- Get payload and do something with it

* Only if you want/need to



OAuth and OIDC

OAuth 2.0

OAuth 2.0 is an industry-standard protocol for authorization. OAuth 2.0 focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices.

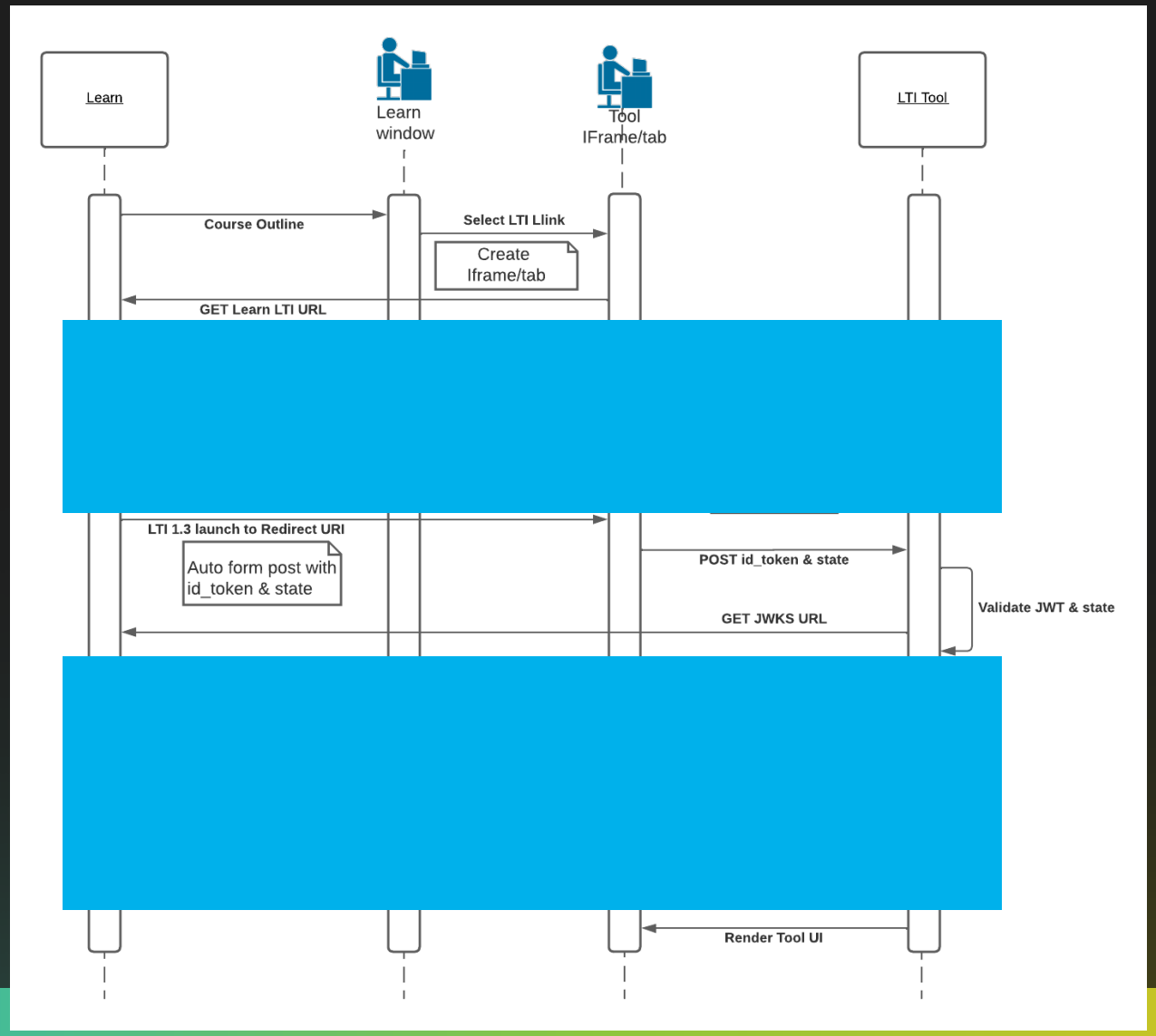
OpenID Connect (OIDC)

OpenID Connect 1.0 is an identity layer on top of the OAuth 2.0 protocol. It allows Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.

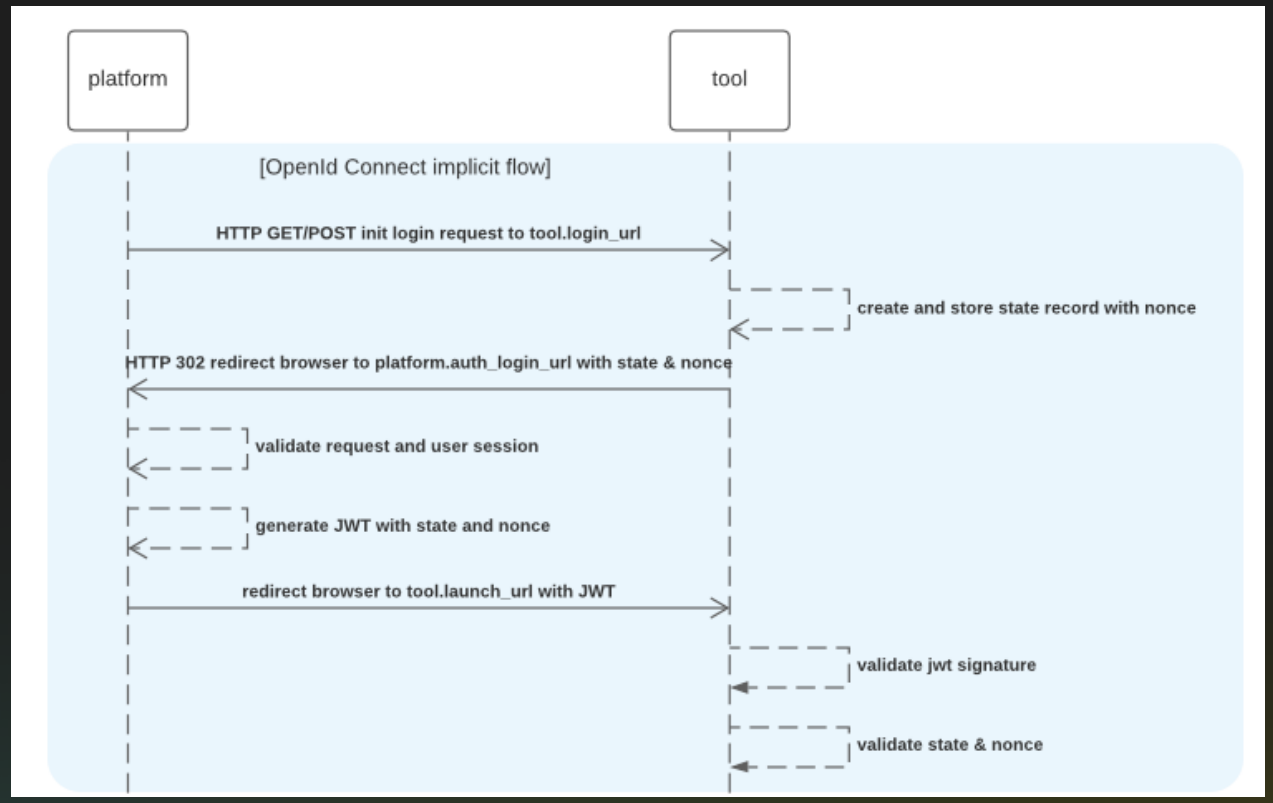
OAuth 2.0 Grant Types

- In OAuth 2.0, the term “grant type” refers to the way an application gets an access token. OAuth 2.0 defines several grant types, including the implicit flow, authorization code flow, and client credentials flow.
- Each grant type is optimized for a particular use case, whether that’s a web app, a native app, a device without the ability to launch a web browser, or server-to-server applications.

Simplest LTI Launch Flow



OIDC Login Flow – To Protect Against XSRF Attacks



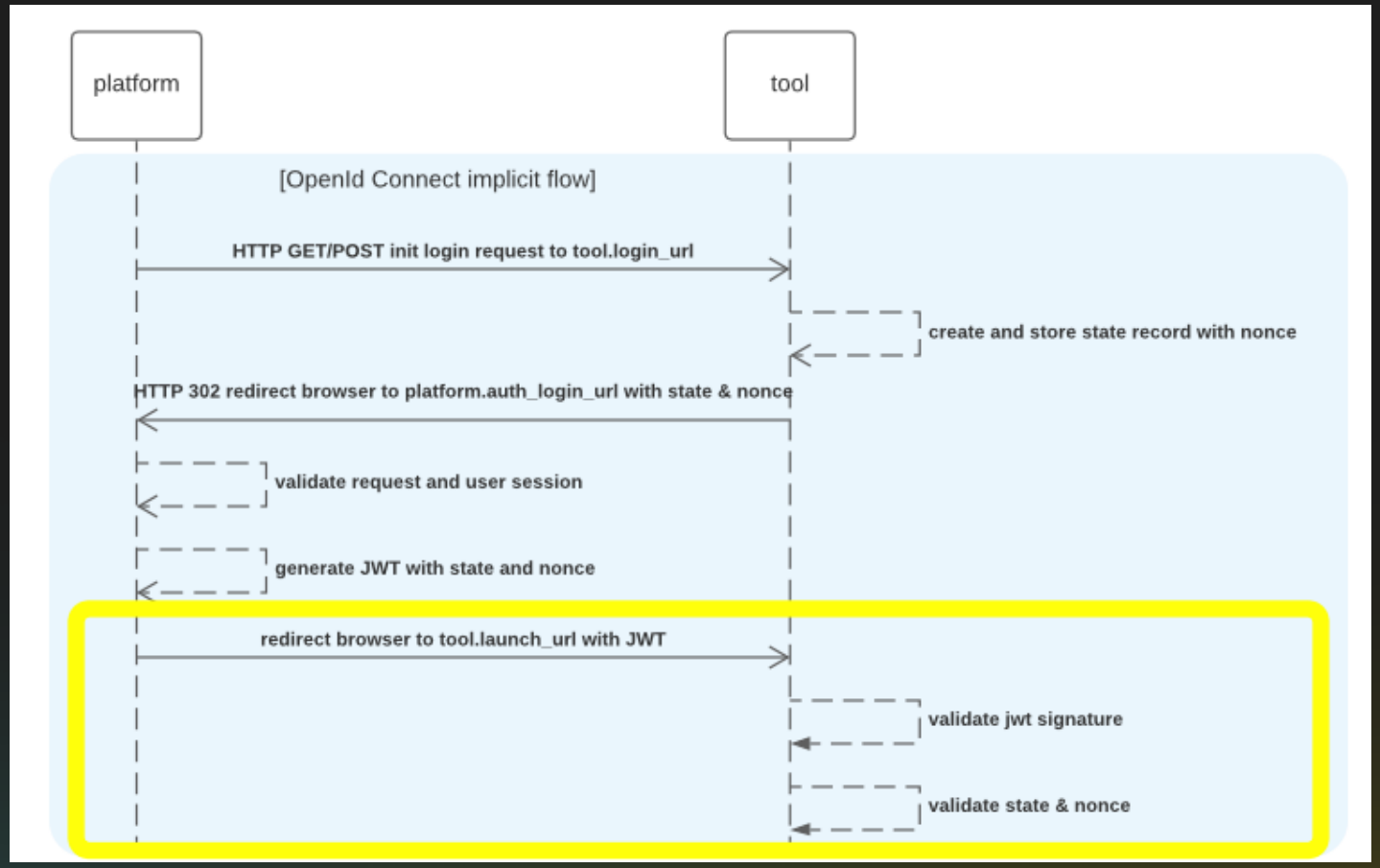
OIDC Login Request – Query Params

```python
def login(request):
    client_id = request.values.get("client_id")
    issuer = request.values.get("iss")
    login_hint = request.values.get("login_hint")
    lti_deployment_id = request.values.get("lti_deployment_id")
    lti_message_hint = request.values.get("lti_message_hint")
    target_link_uri = request.values.get("target_link_uri")

    if not client_id:
        abort(400, "InvalidParameterException - Missing client_id")
    if not issuer:
        abort(400, "InvalidParameterException - Missing issuer")
    if not login_hint:
        abort(400, "InvalidParameterException - Missing login_hint")
    if not lti_deployment_id:
        abort(400, "InvalidParameterException - Missing lti_deployment_id")
    if not target_link_uri:
        abort(400, "InvalidParameterException - Missing target_link_url")

```

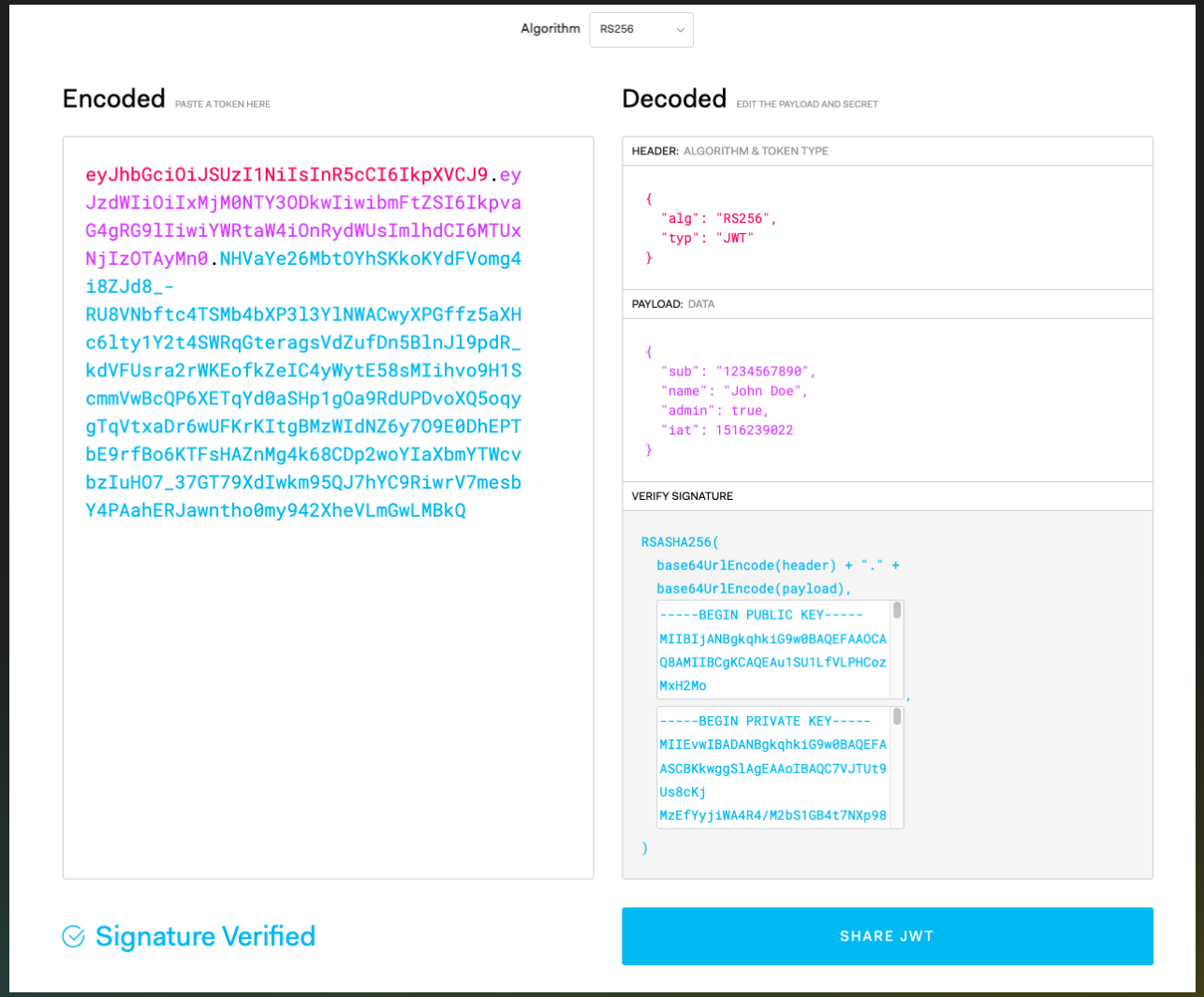
OIDC Login Request – Build Response

```
20      try:
21          lti_platform = LTIPlatform(LTIPlatformStorage()).load(client_id, issuer, lti_deployment_id)
22          state = LTIState(LTIStateStorage()).save()
23          auth_url = lti_platform.config.auth_login_url
24          query_params = dict(
25              scope="openid",
26              response_type="id_token",
27              client_id=client_id,
28              redirect_uri=target_link_uri,
29              state=state.record.id,
30              nonce=state.record.nonce,
31              login_hint=login_hint,
32          )
33
34          if lti_message_hint:
35              query_params.update(lti_message_hint=lti_message_hint)
36
37          query = "?" + urlencode(query_params)
38
39          redirect_url = urljoin(auth_url, query)
40          resp = redirect(redirect_url)
41          resp.set_cookie(
42              key="state",
43              value=state.record.id,
44              samesite="None",
45              secure=True,
46              httponly=True,
47          )
48
49          return resp
50      except Exception as e:
51          abort(500, e)
```

LTI Tool Launch



Inside a JWT – https://jwt.io



Sample JWT from Learn


```
{
  "aud": "63beec56-b8cf-4380-9f9c-953044ad7453",
  "sub": "d7443496cba24f1b96cc97e6473b9265",
  "https://purl.imsglobal.org/spec/lti/claim/deployment_id": "939c964e-07d6-4de5-86d3-4718a307ab8b",
  "https://purl.imsglobal.org/spec/lti/claim/version": "1.3.0",
  "iss": "https://blackboard.com",
  "locale": "en-US",
  "exp": 1656097351,
  "iat": 1656093751,
  "email": "[redacted]@gmail.com",
  "given_name": "Eric",
  "name": "Eric Preston",
  "family_name": "Preston",
  "https://purl.imsglobal.org/spec/lti/claim/roles": [
    "http://purl.imsglobal.org/vocab/lis/v2/membership#Instructor"
  ],
  "https://purl.imsglobal.org/spec/lti/claim/resource_link": {
    "id": "_24894_1",
    "title": "Py LTI Content"
  },
  "https://purl.imsglobal.org/spec/lti/claim/context": {
    "id": "b21a35dbbcbe49e29b96d67ad3e77e5a",
    "label": "ULTRA1",
    "title": "An Ultra Course",
    "type": [
      "http://purl.imsglobal.org/vocab/lis/v2/course#CourseOffering"
    ]
  },
  "https://purl.imsglobal.org/spec/lti/claim/tool_platform": {
    "name": "Blackboard, Inc.",
    "description": "Blackboard, Inc.",
    "guid": "17a1780b7811432980507a45584cbe2e",
```

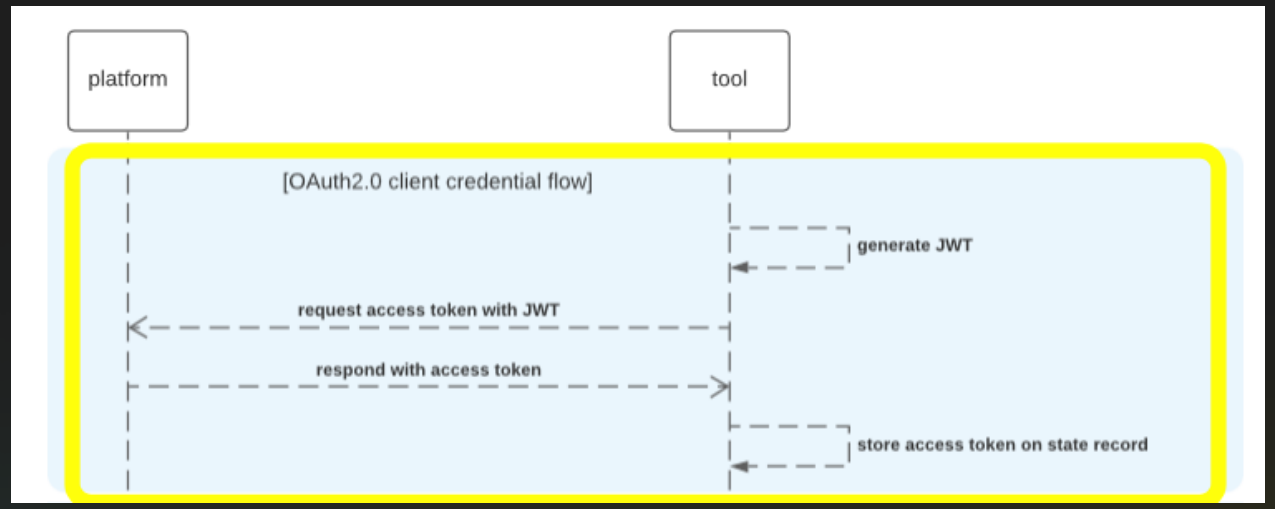
Handling Launch from Platform - Validate

```
1   def launch(request):
2       # Validate the request as per IMS standards: state and id_token
3       # ref: https://www.imsglobal.org/spec/security/v1p0/#step-3-authentication-response
4       # ref: https://www.imsglobal.org/spec/security/v1p0/#authentication-response-validation
5       request_cookie_state = request.cookies.get("state")
6       request_post_state = request.values.get("state")
7       id_token = request.values.get("id_token")
8
9       if not request_cookie_state:
10          abort(400, "InvalidParameterException - Missing state cookie")
11      if not request_post_state:
12          abort(400, "InvalidParameterException - Missing state")
13      if not id_token:
14          abort(400, "InvalidParameterException - Missing id_token")
15
16      if request_cookie_state != request_post_state:
17          abort(409, "InvalidParameterException - State Mismatch")
18
```

Handling LTI Launch – Process JWT

```
18
19      try:
20          # Unpack the id_token (JWT) into an object, without validating
21          jwt_request = LTIJwtPayload(id_token)
22          # Load the config for this deployment using some of the properties of the id_token
23          platform = LTIPlatform(LTIPlatformStorage()).load(jwt_request.aud, jwt_request.iss, jwt_request.deployment_
24
25          # Will raise exceptions internally if the JWT doesn't validate
26          try:
27              jwt_request.verify(platform)
28          except Exception as e:
29              abort(401, e)
30
31          # Load the user's State record
32          state: LTIState = LTIState(LTIStateStorage()).load(request_cookie_state)
33          # Validate the state and nonce
34          if not state.validate(jwt_request.nonce):
35              abort(409, "InvalidParameterException - Unable to verify State")
36
37          # Add the id_token (JWT) to the user's State record
38          state.record.id_token = id_token
```

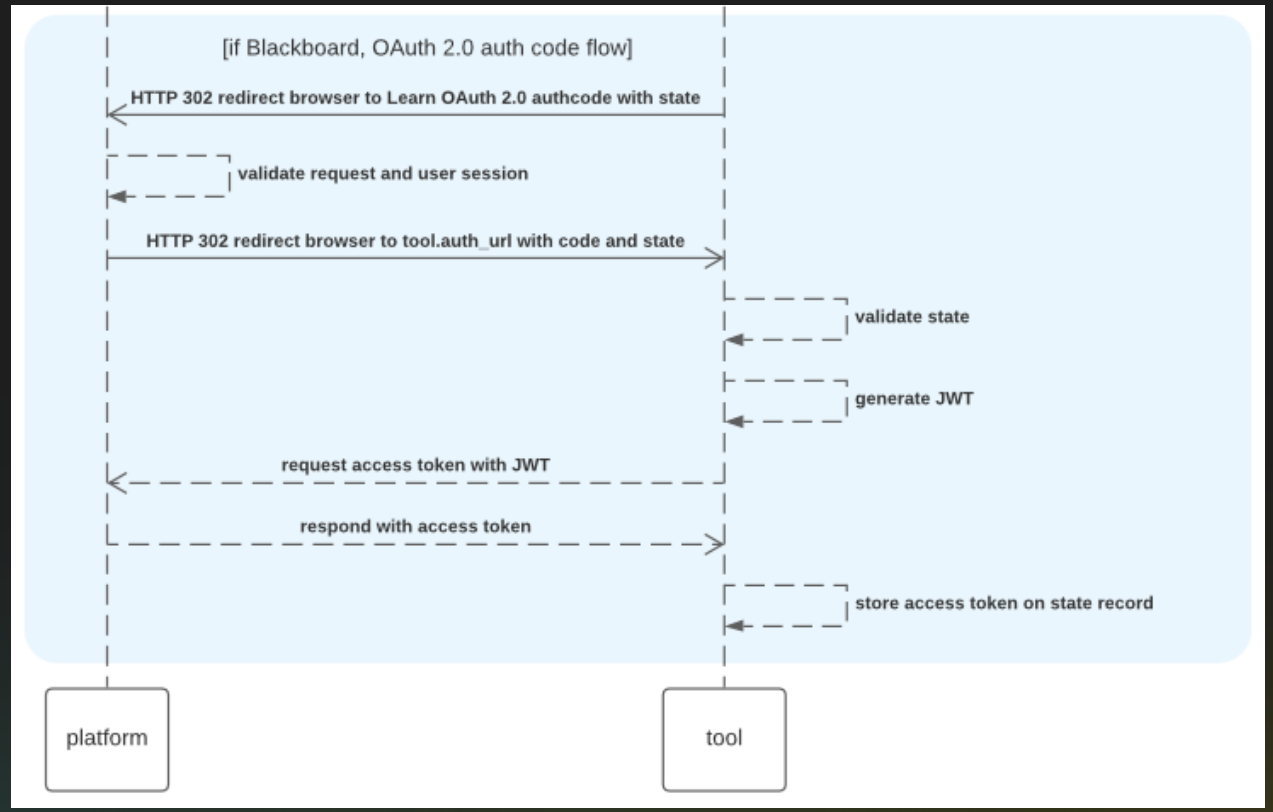
LTI OAuth2 Bearer Token Client Credentials Flow



Get LTI OAuth 2.0 Access Token

```python
jwt = LTIJwtPayload()
time_now = datetime.datetime.now(tz=datetime.timezone.utc)

payload = dict(
    aud=platform.config.auth_token_url,
    exp=timegm(
        (time_now + datetime.timedelta(seconds=int(300))).utctimetuple()
    ),
    jti=secrets.token_hex(16),
    iat=timegm(time_now.utctimetuple()),
    iss=platform.config.client_id,
    sub=platform.config.client_id,
)

jwtstring = jwt.encode(payload=payload, tool=tool)

auth_request = {
    "grant_type": "client_credentials",
    "client_assertion_type": "urn:ietf:params:oauth:client-assertion-type:jwt-bearer",
    "client_assertion": jwtstring,
    "scope": lti_scopes,
}

r = requests.post(platform.config.auth_token_url, data=auth_request)
if not r.ok:
    msg = f"Error retrieving access token from platform {platform.config.auth_token_url}. {r.reason}: {r.text}"
    logging.error(msg)
    raise Exception(msg)

# access token (bearer token) to be used to communicate with the Provider (LMS)
access_token = r.json()["access_token"]
return access_token
```

Learn REST OAuth2 Bearer Token Authorization Code Flow



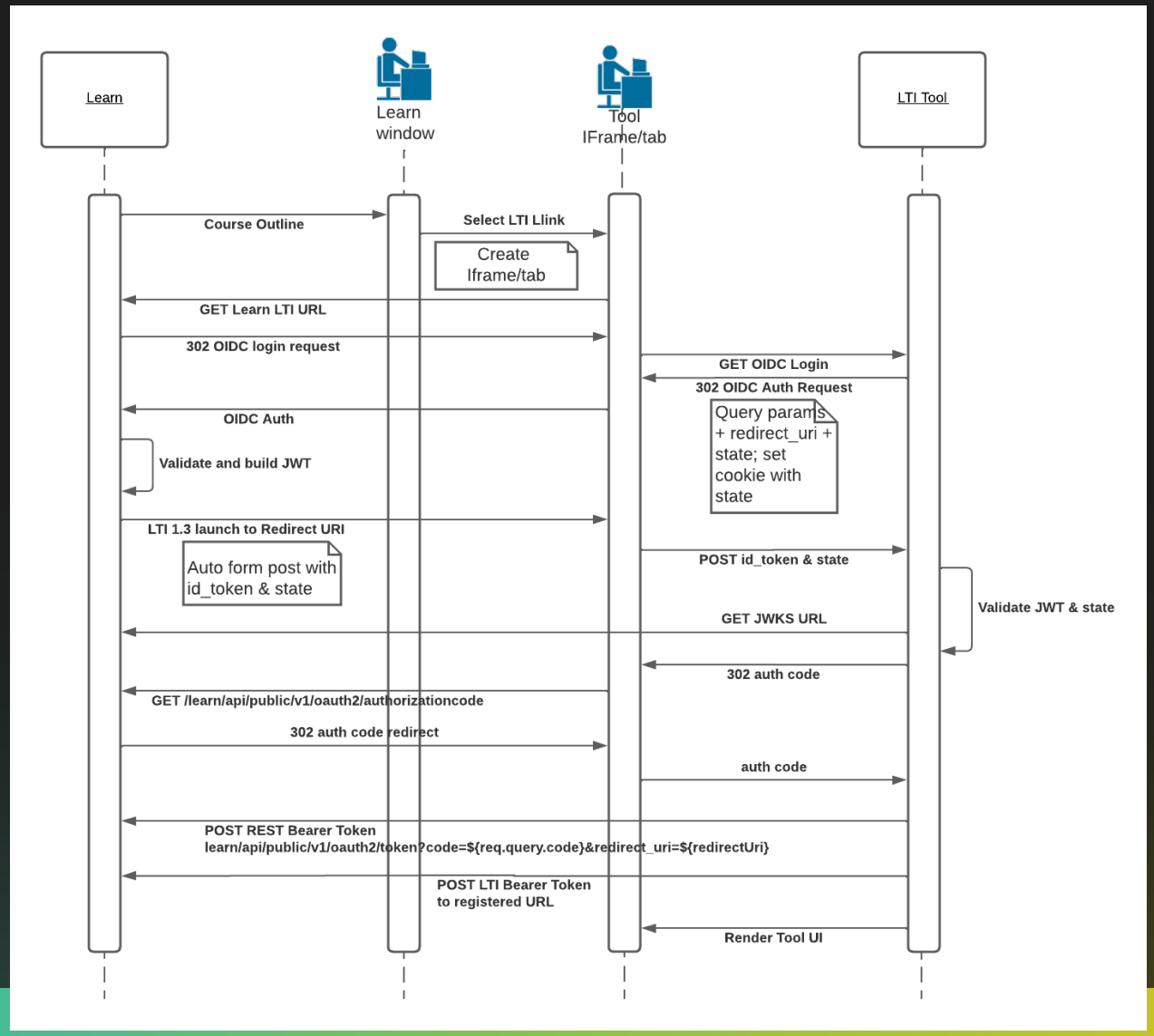
Construct Auth Code Request

```
52          # Blackboard Three-Legged OAuth (3LO) for Learn REST API
53          if jwt_request.platform_product_code == "BlackboardLearn":
54              ##################
55              # Learn REST access token for accessing the Learn REST API: Authorization Code grant
56              # https://docs.blackboard.com/rest-apis/learn/getting-started/3lo
57              # https://developer.blackboard.com/portal/displayApi/
58              # https://www.oauth.com/oauth2-servers/access-tokens/authorization-code-request/
59              ##################
60              params = {
61                  "redirect_uri": lti_tool.config.auth_code_url(),
62                  "response_type": "code",
63                  "client_id": lti_tool.config.learn_app_key,  # despite the naming this is the Learn Application Key
64                  "scope": "*",
65                  "state": request_post_state,
66              }
67              encoded_params = urlencode(params)
68
69              learn_url = jwt_request.platform_url.rstrip("/")
70              one_time_session_token = jwt_request.one_time_session_token
71              auth_code_url = f"{learn_url}/learn/api/public/v1/oauth2/authorizationcode?{encoded_params}&one_time_se
72              return redirect(auth_code_url)
73          else:
74              return render_ui(jwt_request, request_post_state, id_token)
```

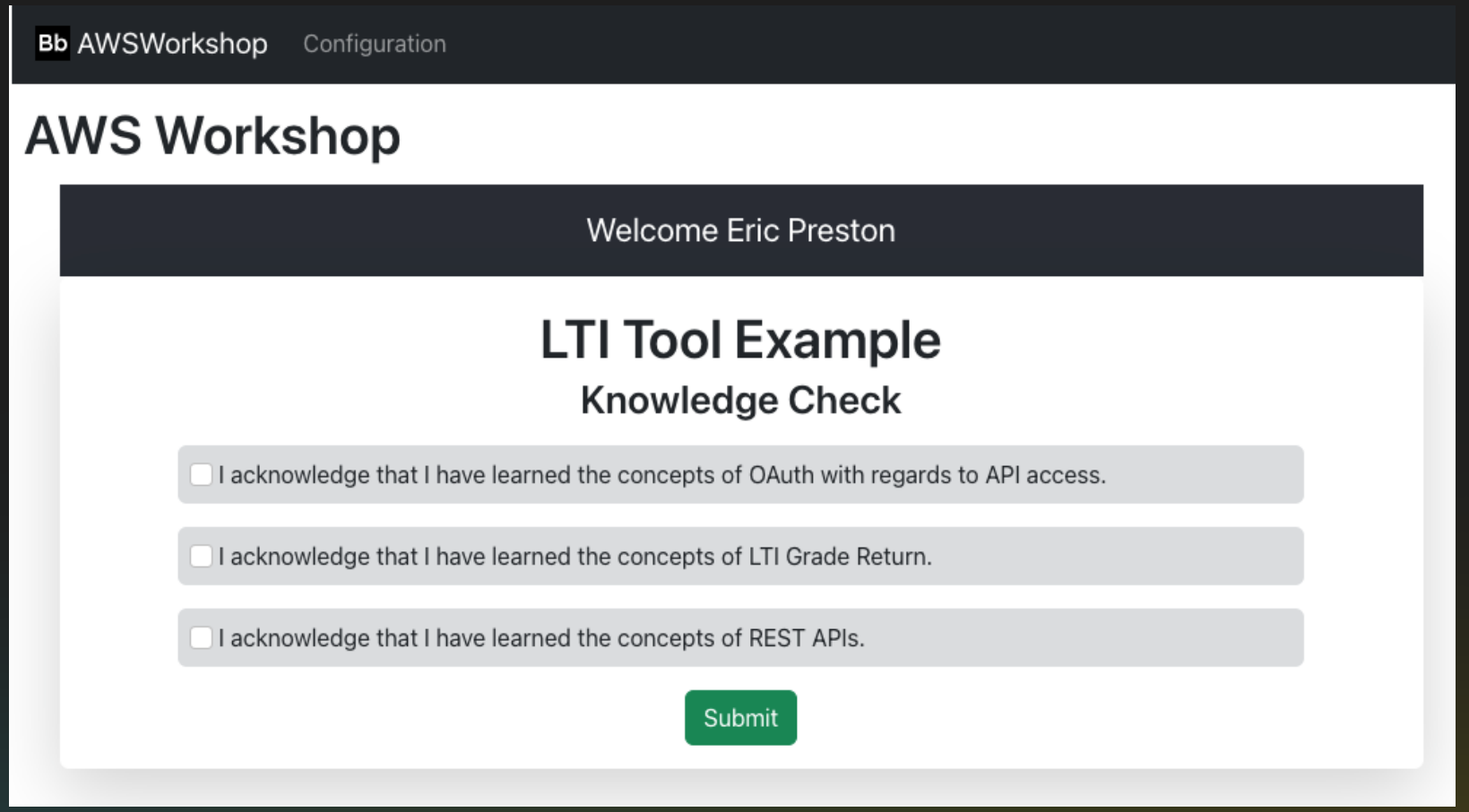
Handle Auth Code Response

```python
def authcode(request):
    auth_code = request.args.get("code", "")
    request_cookie_state = request.cookies.get("state")

    if not auth_code:
        abort(400, "InvalidParameterException - Missing auth code")
    if not request_cookie_state:
        abort(400, "InvalidParameterException - Missing state")

    state: LTIState = LTIState(LTIStateStorage()).load(request_cookie_state)
    if not state:
        abort(409, "InvalidParameterException - State not found")

    try:
        id_token = state.record.id_token
        jwt_request = LTIJwtPayload(id_token)

        lti_tool = LTITool(LTIToolStorage())
        auth_code_url = lti_tool.config.auth_code_url()

        # Get the Learn access token
        if jwt_request.platform_product_code == "BlackboardLearn":
            learn_url = jwt_request.platform_url.rstrip("/")
            learn_access_token = TokenClient().get_learn_access_token(learn_url, auth_code_url, auth_code)
            # Cache the REST access token
            state.record.set_platform_learn_rest_token(learn_access_token)
            state.save()

        return launch_controller.render_ui(jwt_request, request_cookie_state, id_token)
    except Exception as e:
        abort(500, e)
```

Full LTI Launch Flow



Render UI


AWSWorkshop
Configuration
AWS Workshop
Welcome Eric Preston
LTI Tool Example
Knowledge Check
I acknowledge that I have learned the concepts of OAuth with regards to API access.
I acknowledge that I have learned the concepts of LTI Grade Return.
I acknowledge that I have learned the concepts of REST APIs.
Submit

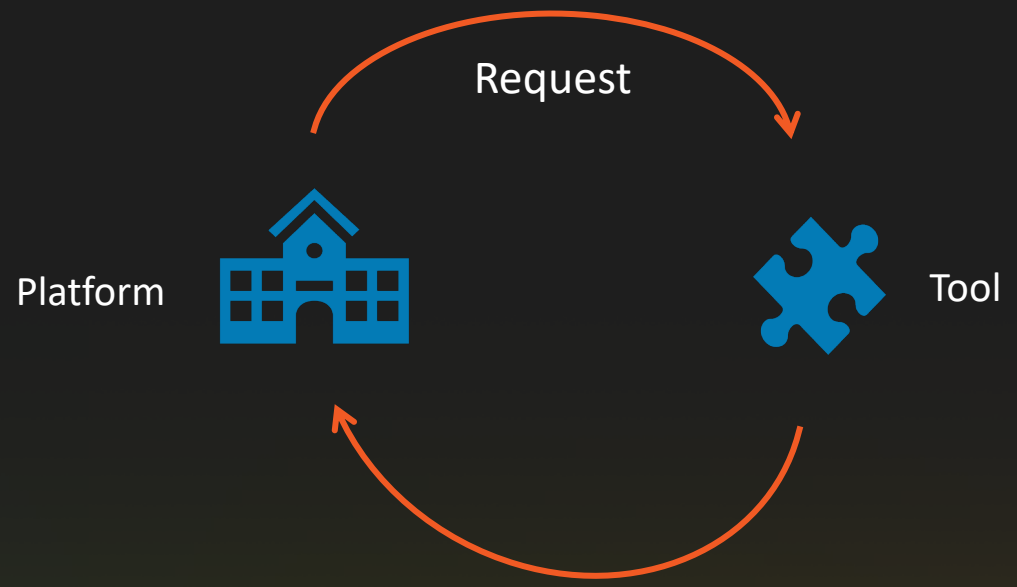
Post Grade to LMS

```
27      try:
28          id_token = state.record.id_token
29          jwt_request = LTIJwtPayload(id_token)
30
31          lti_token = state.record.get_platform_lti_token()
32          # Get Learn URL from the JWT
33          line_item_url = jwt_request.endpoint_lineitem.rstrip("/")
34
35          # Construct payload for Learning Tools Interoperability (LTI) Assignment and Grade Services (AGS) call
36          score_json = {
37              "userId": jwt_request.sub,
38              "scoreGiven": score,
39              "scoreMaximum": 100,
40              "comment": comment,
41              "timestamp": datetime.datetime.utcnow().replace(tzinfo=datetime.timezone.utc).isoformat(),
42              "activityProgress": "Completed",
43              "gradingProgress": "FullyGraded",
44          }
45
46          headers = {
47              "content-type": "application/vnd.ims.lis.v1.score+json",
48              "Authorization": f"Bearer {lti_token}",
49          }
50
51          # Make AGS call to update grade
52          response = requests.post(f"{line_item_url}/scores", json=score_json, headers=headers)
53
54          return render_template("submission_success.html", status=response.status_code, response=response.text)
55      except Exception as e:
56          abort(500, e)
```

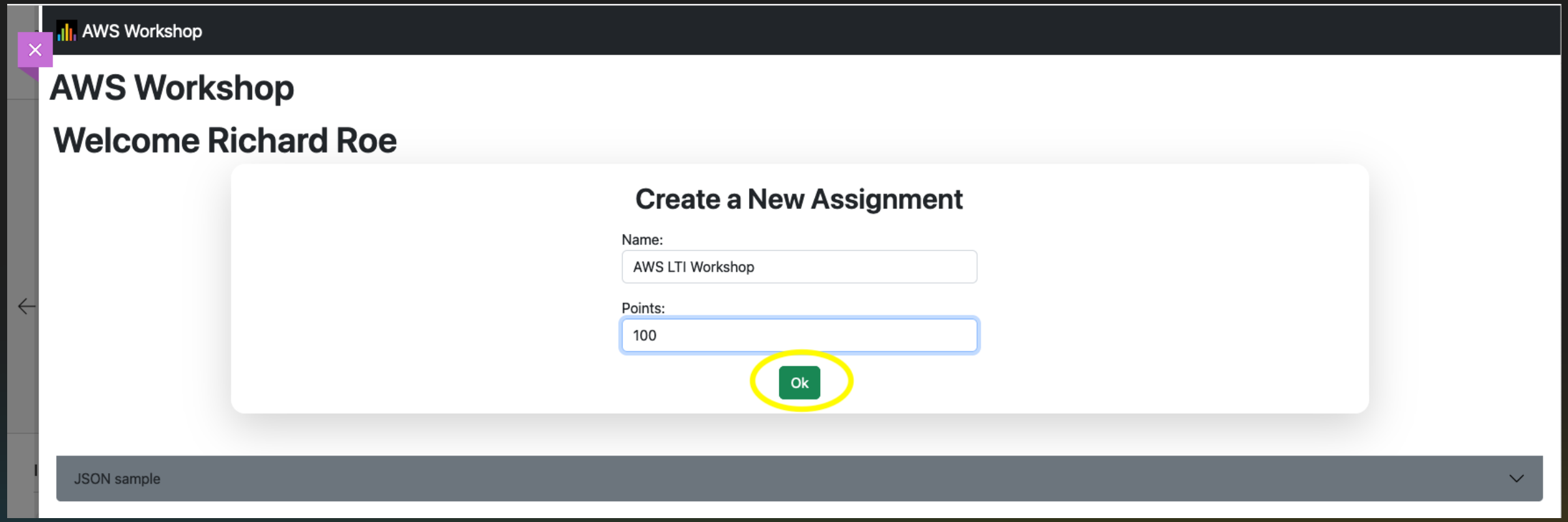
How Do We Get Content into LMS?

Deep Linking

- An LTI launch with a new message type
- The Tool provides a UI to create or select content
- The Tool sends one or more links back to the Platform that are LtiResourceLink messages



Provide a UI to Select Content



Create the Content Items

```python
def get_assignment_content(name, points):
    # Mock return value to simulate a assignment content item
    # Ideally we'd create an assignment in our database and create a content item with that unique identifier
    assignment_id = uuid.uuid4().hex

    tool = LTITool(LTIToolStorage())
    lti_launch_url = f"{tool.config.base_url()}/launch"

    content_item = dict(
        type="ltiResourceLink",
        title=name,
        text="Do this assignment",
        url=lti_launch_url,
        lineItem=dict(
            scoreMaximum=points, label=name, resourceId=assignment_id, tag="originality"
        ),
        custom=dict(
            assignment_id=assignment_id,
            userNameLTI="$User.username",
            userIdLTI="$User.id",
            contextHistory="$Context.id.history",
            resourceHistory="$ResourceLink.id.history",
        ),
    )

    return [content_item]
```

Create the Deep Link Response

```python
def get_message_claims(jwt_request: LTIJwtPayload, content_items) -> dict:
    claims = {
        "iss": jwt_request.aud,
        "aud": [jwt_request.iss],
        "exp": int(time.time()) + 600,
        "iat": int(time.time()),
        "nonce": "nonce-" + uuid.uuid4().hex,
        "https://purl.imsglobal.org/spec/lti/claim/deployment_id": jwt_request.deployment_id,
        "https://purl.imsglobal.org/spec/lti/claim/message_type": "LtiDeepLinkingResponse",
        "https://purl.imsglobal.org/spec/lti/claim/version": "1.3.0",
        "https://purl.imsglobal.org/spec/lti-dl/claim/content_items": content_items,
        "https://purl.imsglobal.org/spec/lti-dl/claim/data": jwt_request.deep_linking_settings_data,
    }
    return claims
```

Send Content to Platform as a FORM POST

```python
def create_assignment(request):
    name = request.form.get("name")
    points = request.form.get("points")
    id_token = request.form.get("id_token")
    request_cookie_state = request.form.get("state")

    if not name or not points or not id_token:
        abort(400, "InvalidParameterException - Missing required parameter")

    state: LTIState = LTIState(LTIStateStorage()).load(request_cookie_state)
    if not state:
        abort(409, "InvalidParameterException - State not found")

    try:
        jwt_request = LTIJwtPayload(id_token)
        lti_tool = LTITool(LTIToolStorage())

        content = get_assignment_content(name, points)

        return_url = jwt_request.deep_linking_settings_return_url
        deep_link_claims = get_message_claims(jwt_request, content)

        jwt = LTIJwtPayload()
        jwtstring = jwt.encode(payload=deep_link_claims, tool=lti_tool)

        pretty_body = json.dumps(deep_link_claims, sort_keys=True, indent=2, separators=(",", ": "))

        return render_template(
            "confirm_assignment.html",
            pretty_body=pretty_body,
            jwt=jwtstring,
            return_url=return_url,
        )
    except Exception as e:
        abort(500, e)
```

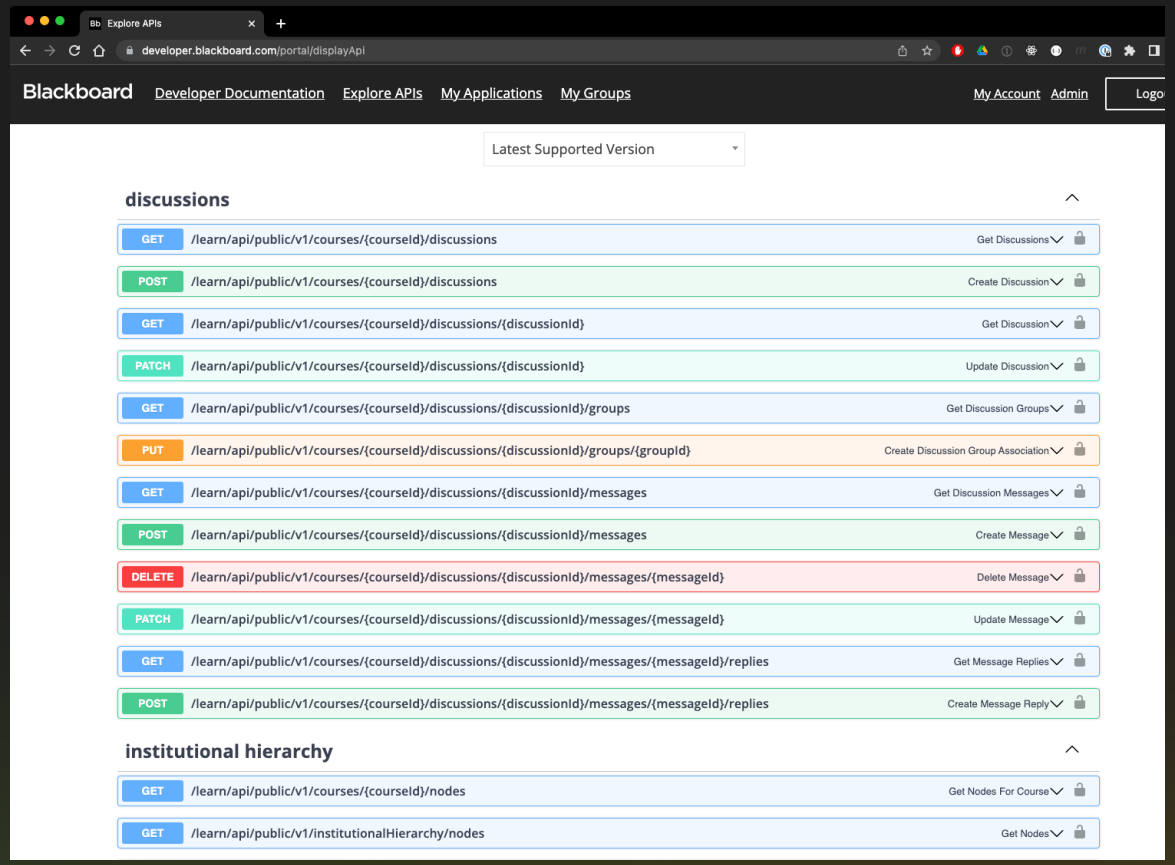
When LTI Is Not Enough

Everything we've done so far is 100% standards-based LTI

What happens if LTI isn't enough?

Use the public REST API

- Attendance
- Discussions
- Course Content
- Group Assignments
- Announcements

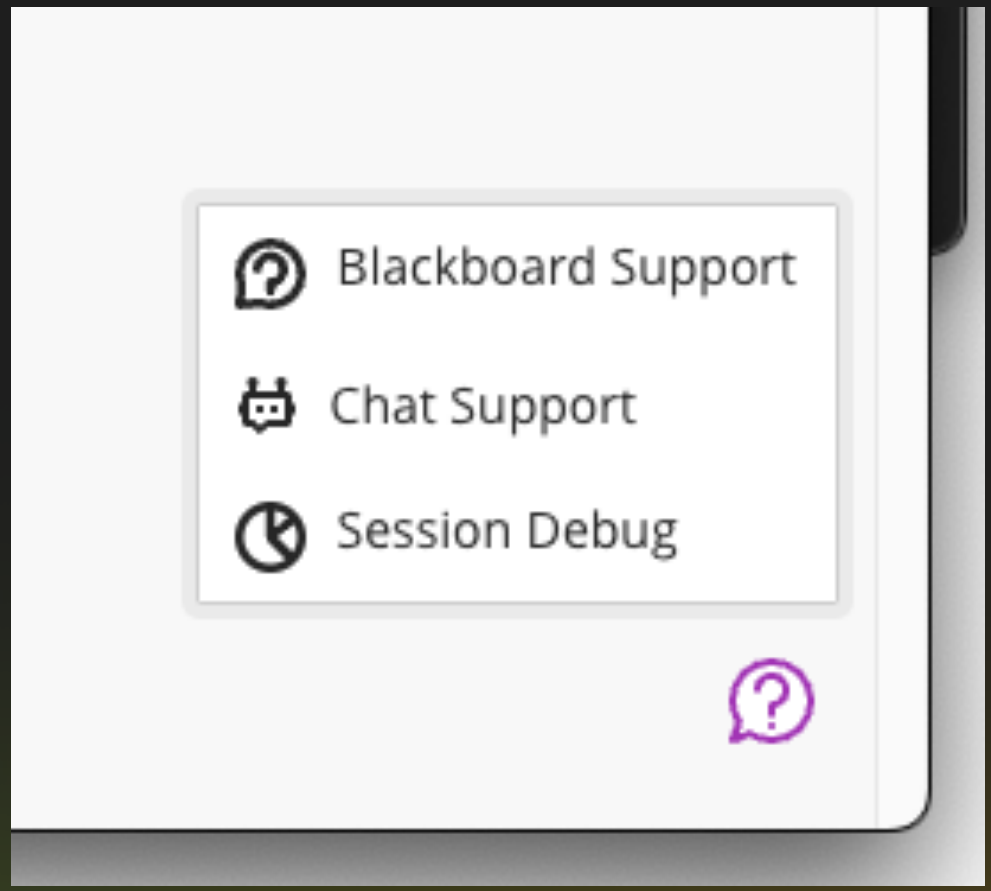


Call the Learn REST API with Access Token

```python
def get_course_info(self, jwt_request: LTIJwtPayload, request_cookie_state):
    state: LTIState = LTIState(LTIStateStorage()).load(request_cookie_state)
    learn_access_token = state.record.get_platform_learn_rest_token()
    learn_url = jwt_request.platform_url.rstrip("/")
    course_uuid = jwt_request.context_id
    headers = {"Authorization": f"Bearer {learn_access_token}"}
    course_info_url = f"{learn_url}/learn/api/public/v2/courses/uuid:{course_uuid}"
    response = requests.get(course_info_url, headers=headers)

    if response.status_code == 200:
        return response.json()
    else:
        self.__log().error(
            f"Error getting course info via Learn public API, status: {response.status_code}"
        )
        return {}
```

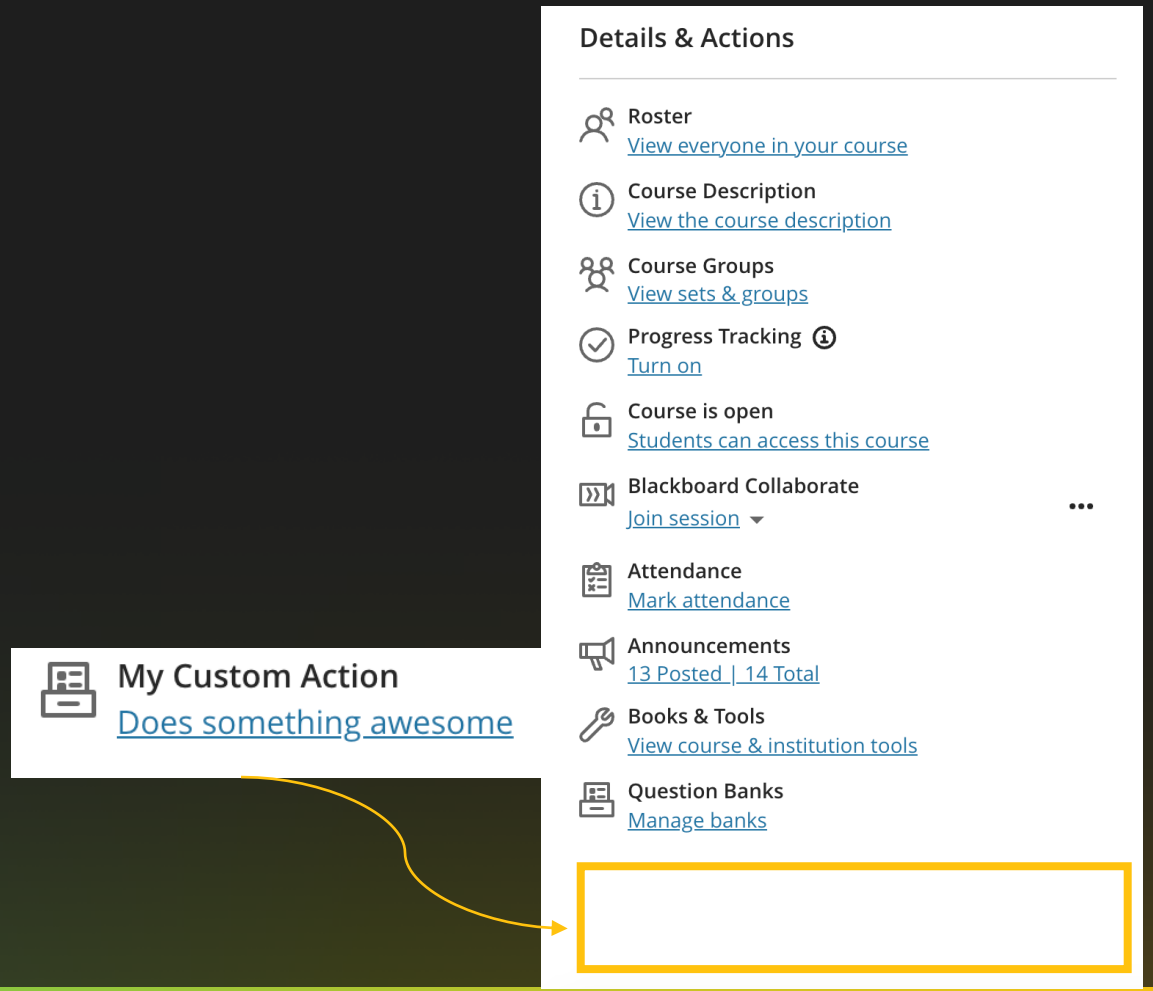
Ultra Extension Framework

Ultra Extension Framework (UEF) allows you to build an integration that communicates with Ultra so you know what is happening and can request actions based on that knowledge.

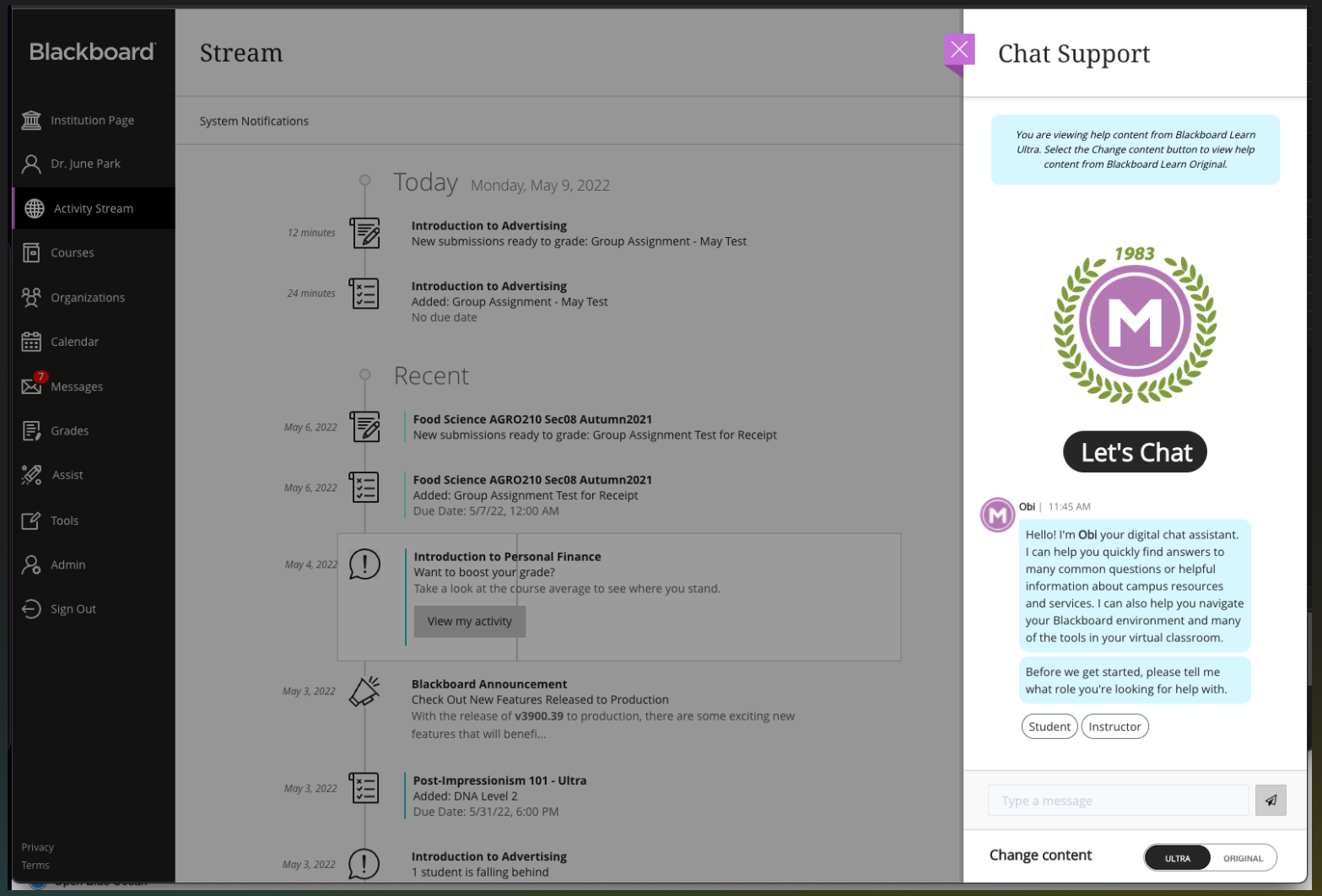


UEF Areas of Customization

- Modals
- Panels
- Base Nav
- Course Details
- Help
- LTI Launches
- Notifications



Chatbot – a UEF Example from Help



Register Your Application with Learn

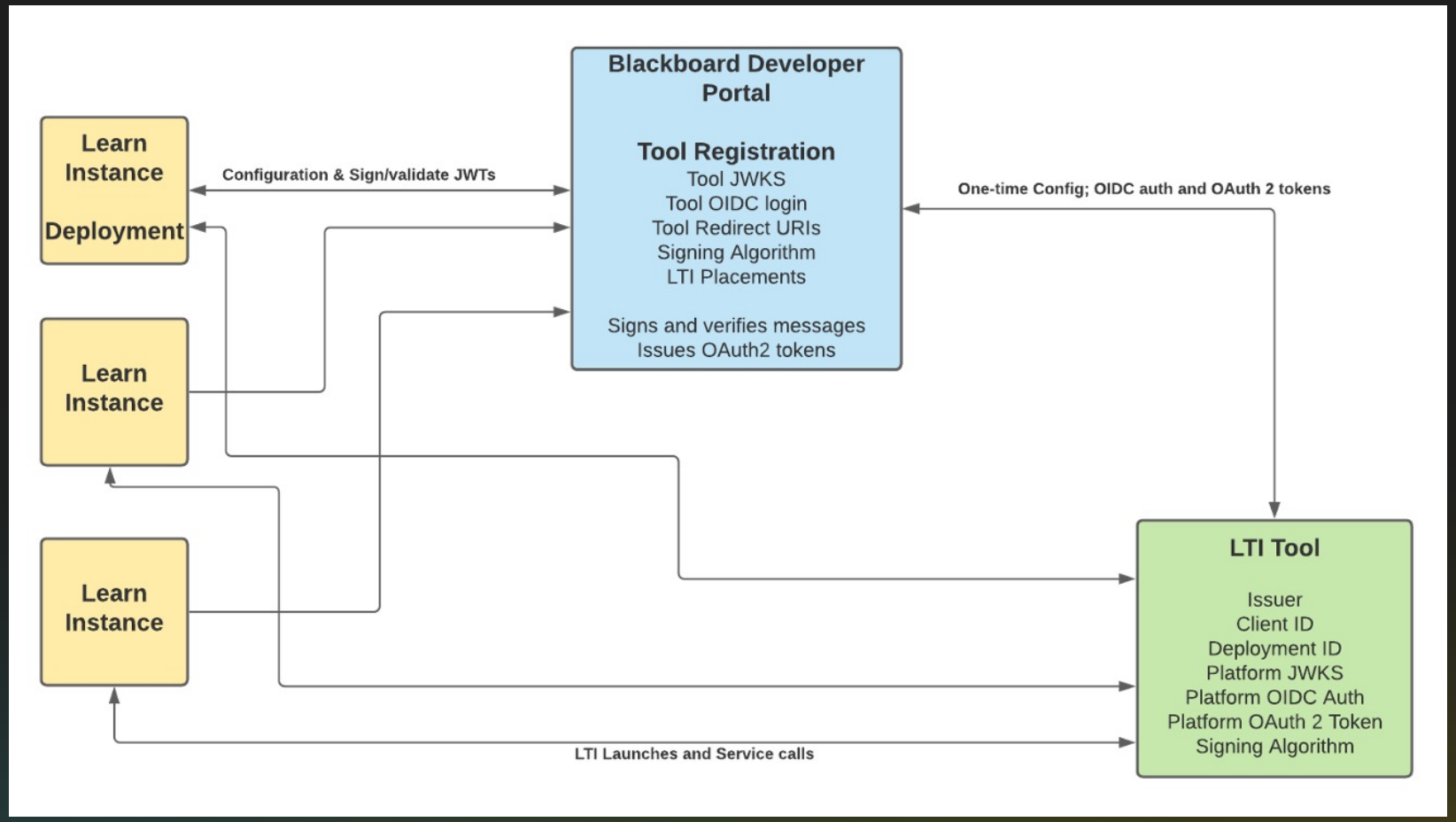
Platform Needs

- OIDC Login Initiation URI - CSRF
- Redirect URI(s) – LTI Launch
- JWKS URL - tool public keyset
- Signing Algorithm

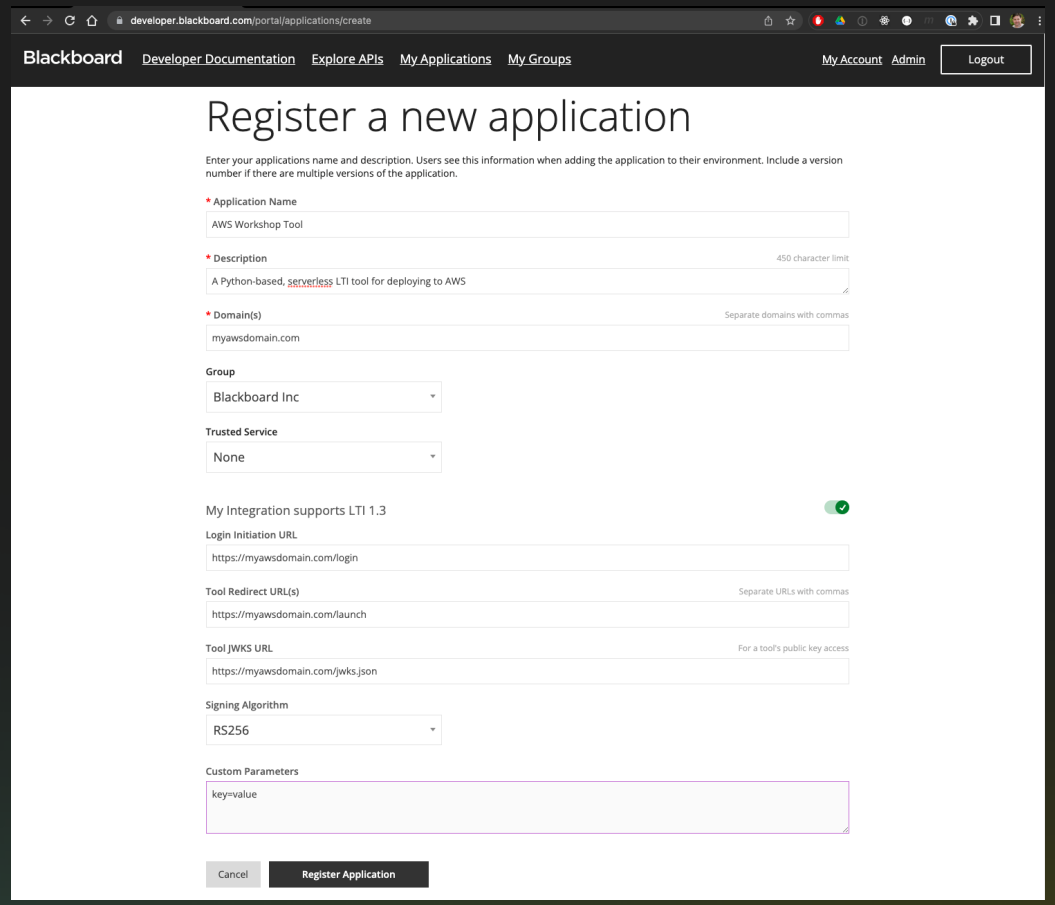
Tool Needs

- Issuer – of keys
- Client ID – OIDC identity
- OIDC Auth URL – CSRF
- OAuth 2 Token URL – service calls
- JWKS URL – platform public keyset
- Signing Algorithm
- Deployment ID
- REST key & secret

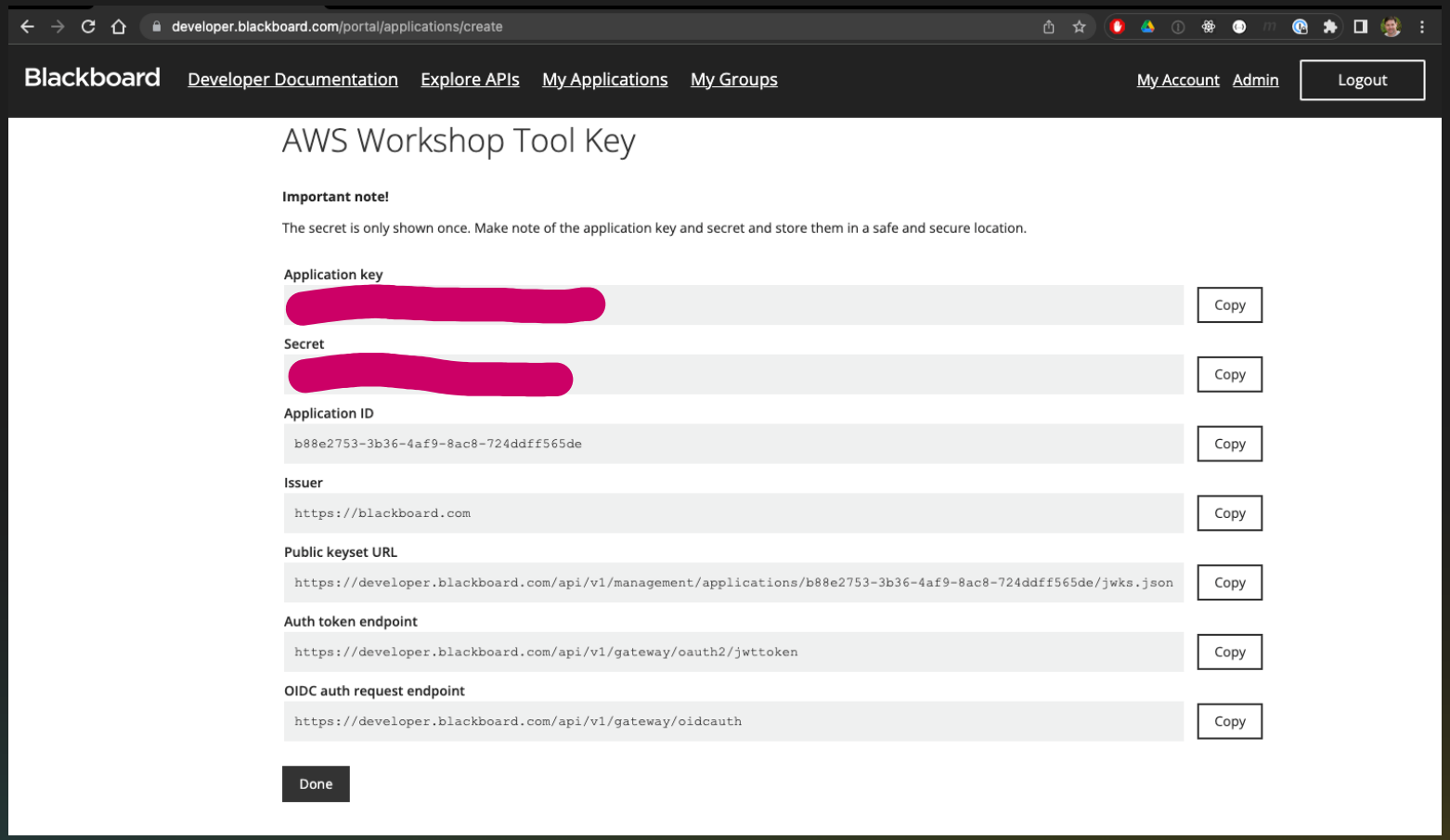
Learn LTI Architecture



Register Application with Developer Portal

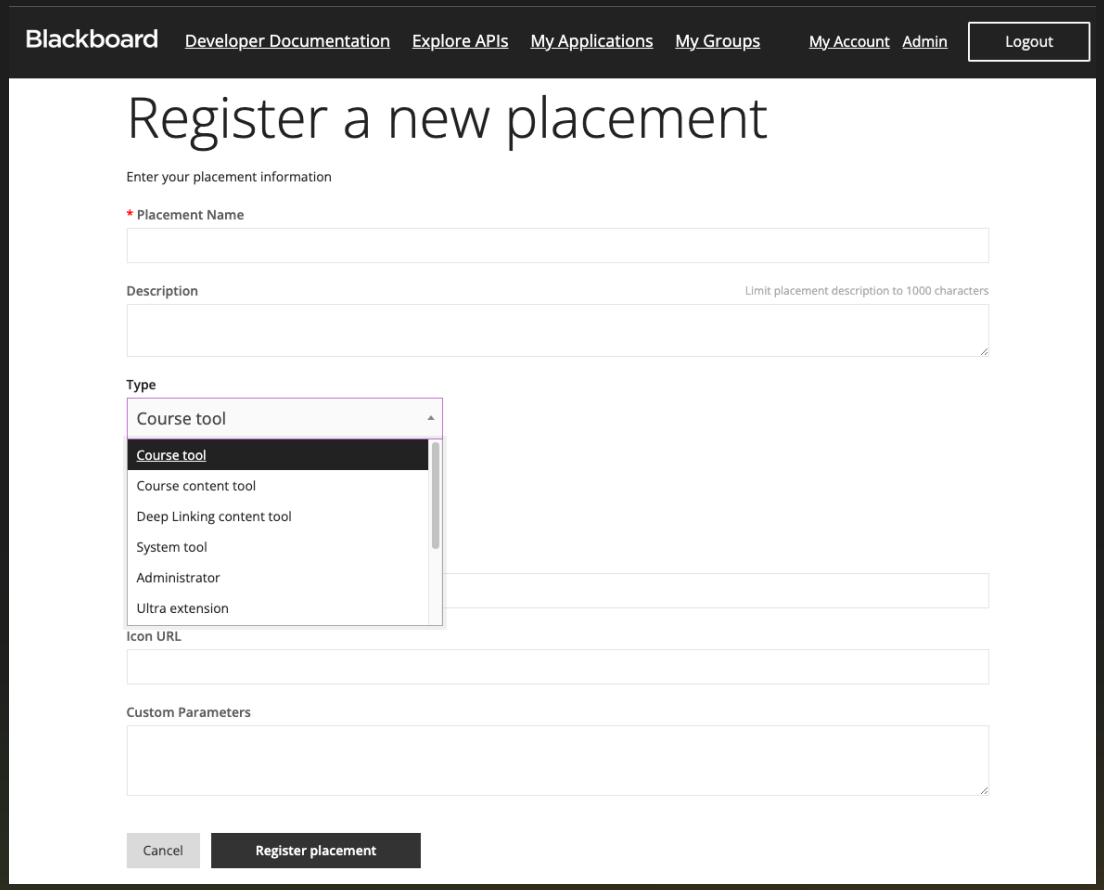


Copy Configuration Values

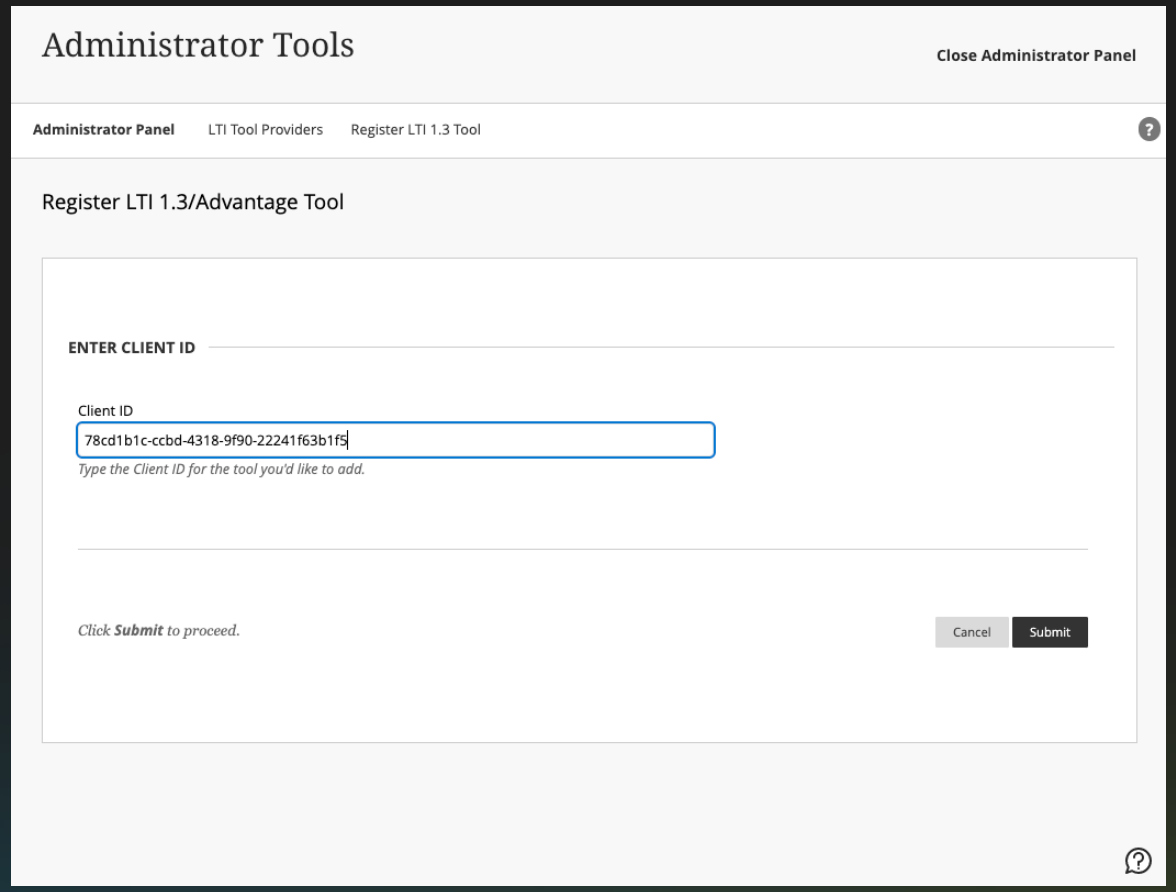


Define Placements

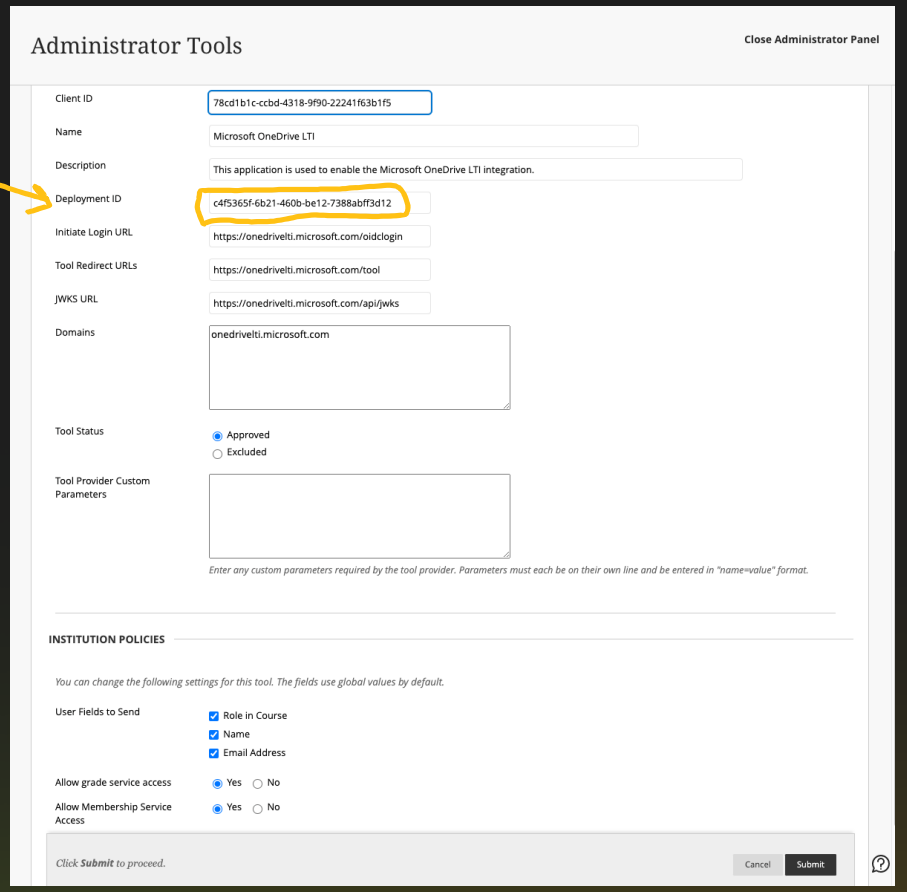
- A Placement defines where your tool appears in the Learn UI
- Seven types of placements:
  - Deep Linking – allow students or not
  - Course Content – gradable or not
  - Course – allow students or not
  - System
  - Admin
  - UEF – Extend Ultra, like JS Hacks for Ultra



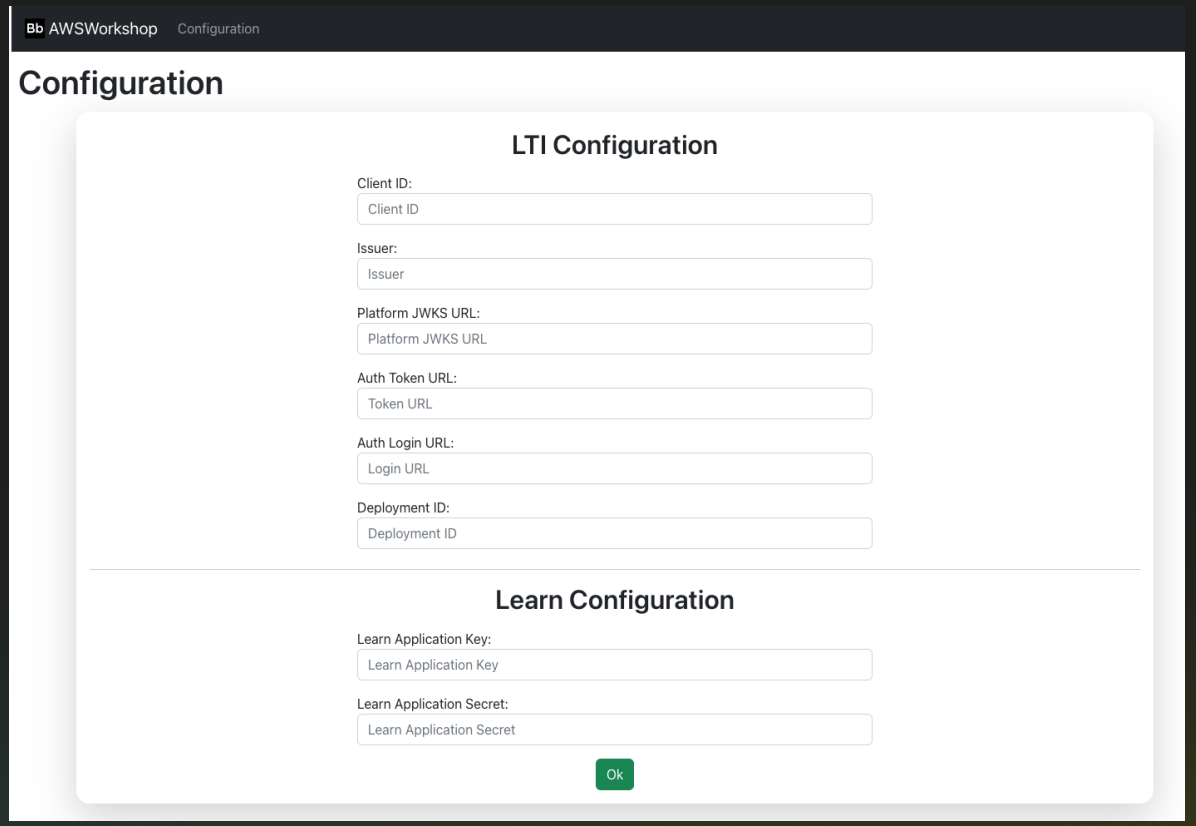
Deploy to Learn



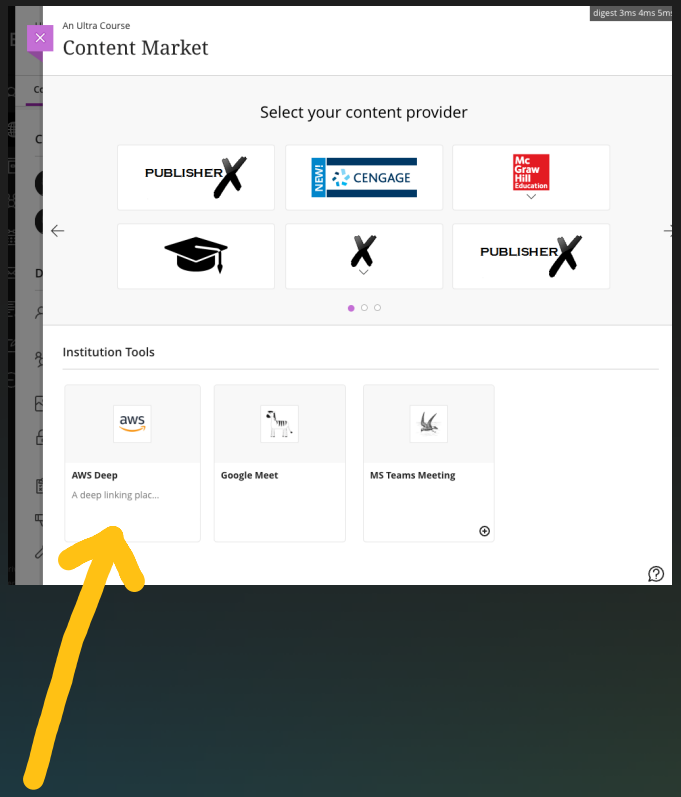
Copy Deployment ID

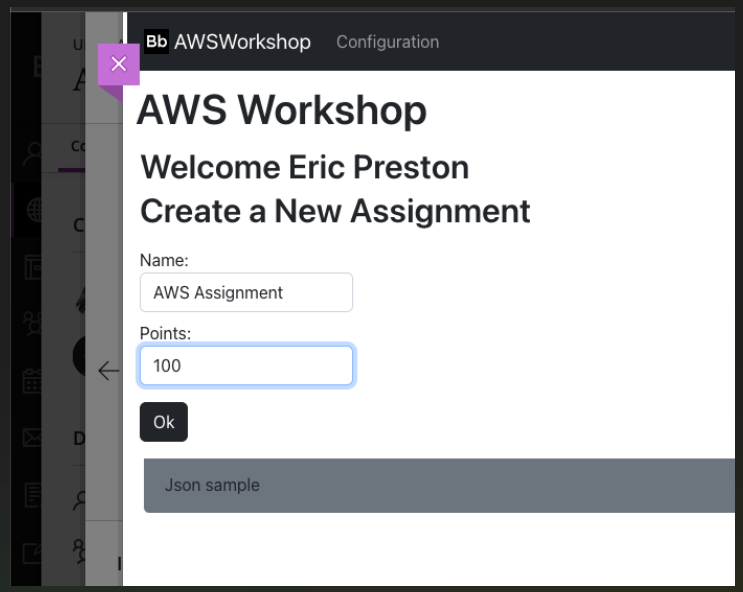


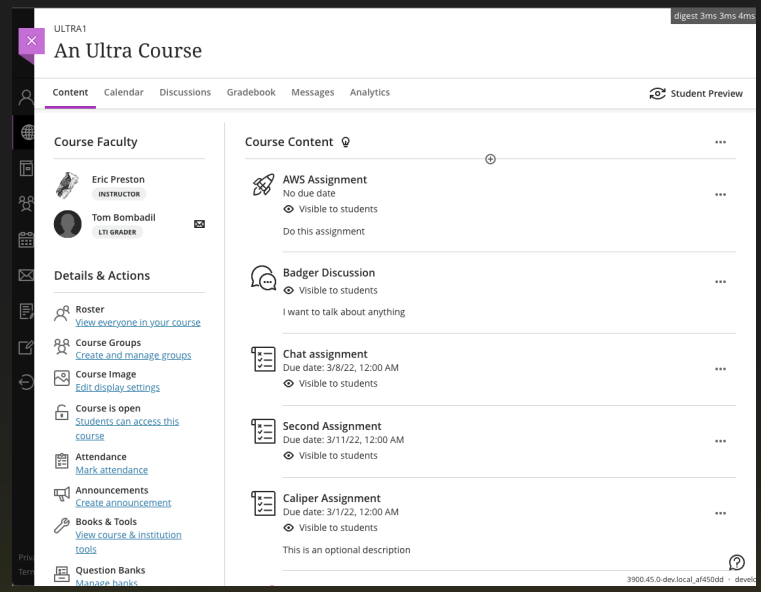
Update Tool Configuration



Use in Learn


An Ultra Course
Content Market
Select your content provider
PUBLISHER
NEW!
CENGAGE
Mc Graw Hill Education
PUBLISHER
Institution Tools
aws
AWS Deep
A deep linking plac...
Google Meet
MS Teams Meeting


Bb AWSWorkshop
Configuration
AWS Workshop
Welcome Eric Preston
Create a New Assignment
Name:
AWS Assignment
Points:
100
Ok
Json sample


ULTRA1
An Ultra Course
Content
Calendar
Discussions
Gradebook
Messages
Analytics
Student Preview
Course Faculty
Eric Preston
INSTRUCTOR
Tom Bombadil
LTI GRADER
Details & Actions
Roster
View everyone in your course
Course Groups
Create and manage groups
Course Image
Edit display settings
Course is open
Students can access this course
Attendance
Mark attendance
Announcements
Create announcement
Books & Tools
View course & institution tools
Question Banks
Course Content
AWS Assignment
No due date
Visible to students
Do this assignment
Badger Discussion
Visible to students
I want to talk about anything
Chat assignment
Due date: 3/8/22, 12:00 AM
Visible to students
Second Assignment
Due date: 3/11/22, 12:00 AM
Visible to students
Caliper Assignment
Due date: 3/1/22, 12:00 AM
Visible to students
This is an optional description

Specifications to read

- OAuth 2:
  - https://oauth.net/2/ (Client Credentials and Authorization Code flows)
- OpenID Connect:
  - https://openid.net/specs/openid-connect-core-1_0.html#ThirdPartyInitiatedLogin
- JWT:
  - https://tools.ietf.org/html/rfc7519
  - https://jwt.io
- LTI Advantage:
  - https://www.imsglobal.org/ims-security-framework
  - https://www.imsglobal.org/activity/learning-tools-interoperability

LTI & REST Resources

- https://docs.blackboard.com/ - Developer docs for LTI, REST, UEF, etc.
- https://developer.blackboard.com – The Developer Portal with Swagger docs on REST API
- Many sample projects at https://github.com/blackboard
  - https://github.com/blackboard/BBDN-lti-1p3-tool-example - Python example used here
  - https://github.com/blackboard/BBDN-LTI-Tool-Provider-Node - Node LTI/REST example
  - https://github.com/blackboard/BBDN-HelloWorld-B2 - Old-school B2 example
  - https://github.com/blackboard/BBDN-UEF-Python - UEF example
- https://github.com/IMSGlobal/ltibootcamp - Many open source resources

Questions? developers@blackboard.com

anthology

together