



Migrating a B2 to LTI & REST

A Deep Dive

July 2022

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

The image shows two Firefox browser windows side-by-side. The left window displays the Blackboard Ultra course interface for 'APRESTONULTRA2'. It includes a sidebar with links like 'Content', 'Calendar', 'Discussions', etc., and a main area showing course content items such as 'Harmony 3', 'Tii one', 'Badger Assignment', and 'New Secret'. The right window shows a 'Gradebook' interface titled 'Assignment Inbox' with a table listing student submissions. A yellow arrow points from the 'Tii one' item in the first window to the 'Assignment Inbox' in the second window, illustrating the integration between the two applications.

| Author | Paper Title | Paper ID | Uploaded | Viewed | Grade | Similarity | Flags | Options |
|-----------|-------------------|-----------|----------------------------|--------|-------|------------|-------|---------|
| Sue Queue | endpointlogs.txt | 200933499 | Mar 11th 2022, 1:34 PM CST | 66 | 27% | -- | -- | ... |
| Ice Tea | Not yet submitted | -- | -- | -- | -- | -- | -- | ... |
| Joe Cool | Not yet submitted | -- | -- | -- | -- | -- | -- | ... |
| Bob Ross | Not yet submitted | -- | -- | -- | -- | -- | -- | ... |

IFrame

il.py 4, U process.py 5, U index.jsp U X

HelloWorld > src > main > webapp > tool > index.jsp > ?

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<%@ page import="blackboard.data.user.User,
    blackboard.data.course.Course"
%>

<%@ taglib uri="/bbNG" prefix="bbNG" %>

<bbNG:learningSystemPage ctxId="ctx" title="Hello World">
    <bbNG:pageHeader>
        <bbNG:breadcrumbBar>
            <bbNG:breadcrumb>Hello World</bbNG:breadcrumb>
        </bbNG:breadcrumbBar>
        <bbNG:pageTitleBar title="Hello World"/>
    </bbNG:pageHeader>
    <p>Please migrate me to LTI and REST. I am so tired of being stuck in this JV</p>

    <h3>Here is some basic information for you:</h3>
    <p>
        <%
            User user=ctx.getUser();
            Course course=ctx.getCourse();
        %>
        <table>
            <thead>
                <tr>
                    <th>Key</th>
                    <th>Value</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>User</td>
                    <td>${user}</td>
                </tr>
                <tr>
                    <td>Course</td>
                    <td>${course}</td>
                </tr>
            </tbody>
        </table>
    </p>
</bbNG:learningSystemPage>
```

Migrate Me, Please!!

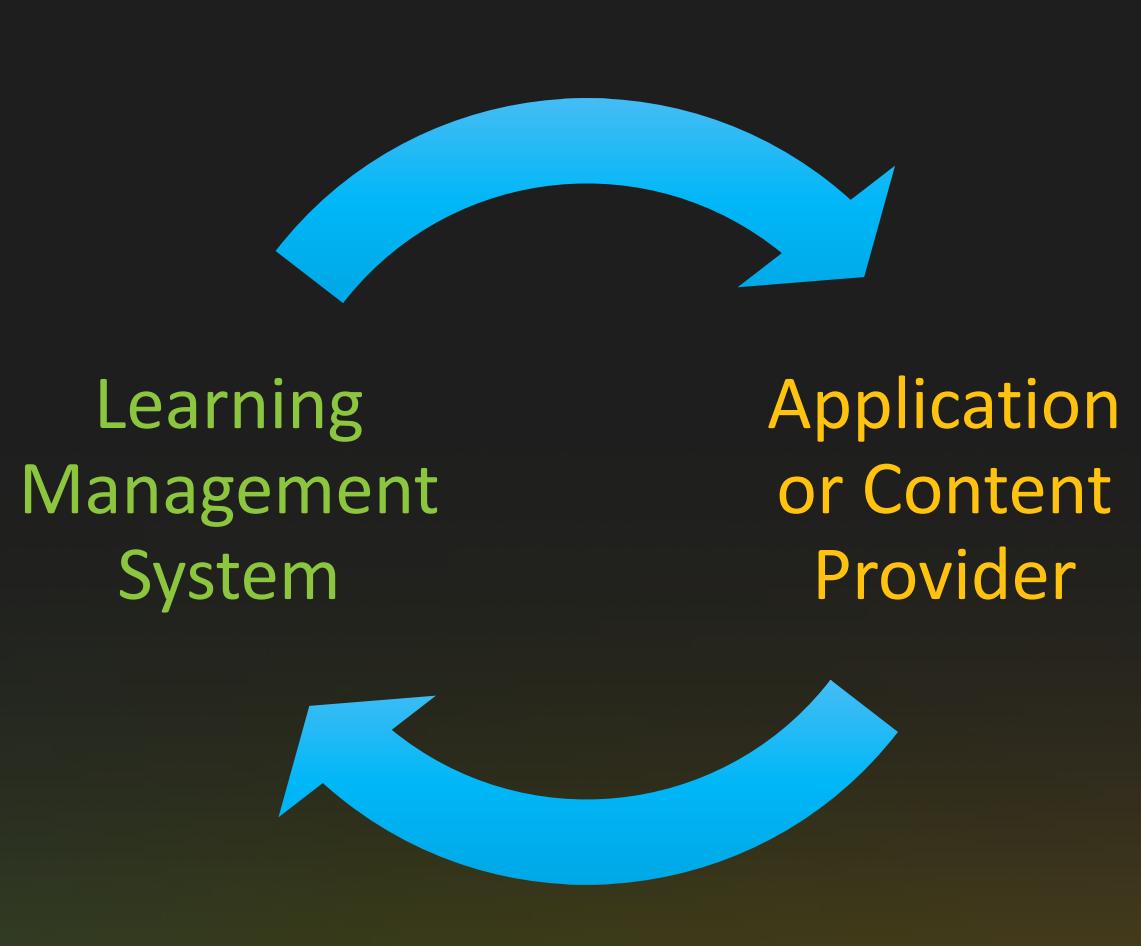
B2's are SO 2000s. Let's figure out how to get this in a modern web application infrastructure using modern integration techniques.

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

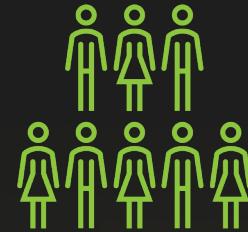


Meet LTI 1.3/Advantage

LTI 1.3 Core
The Launch



Deep Linking
Get Content

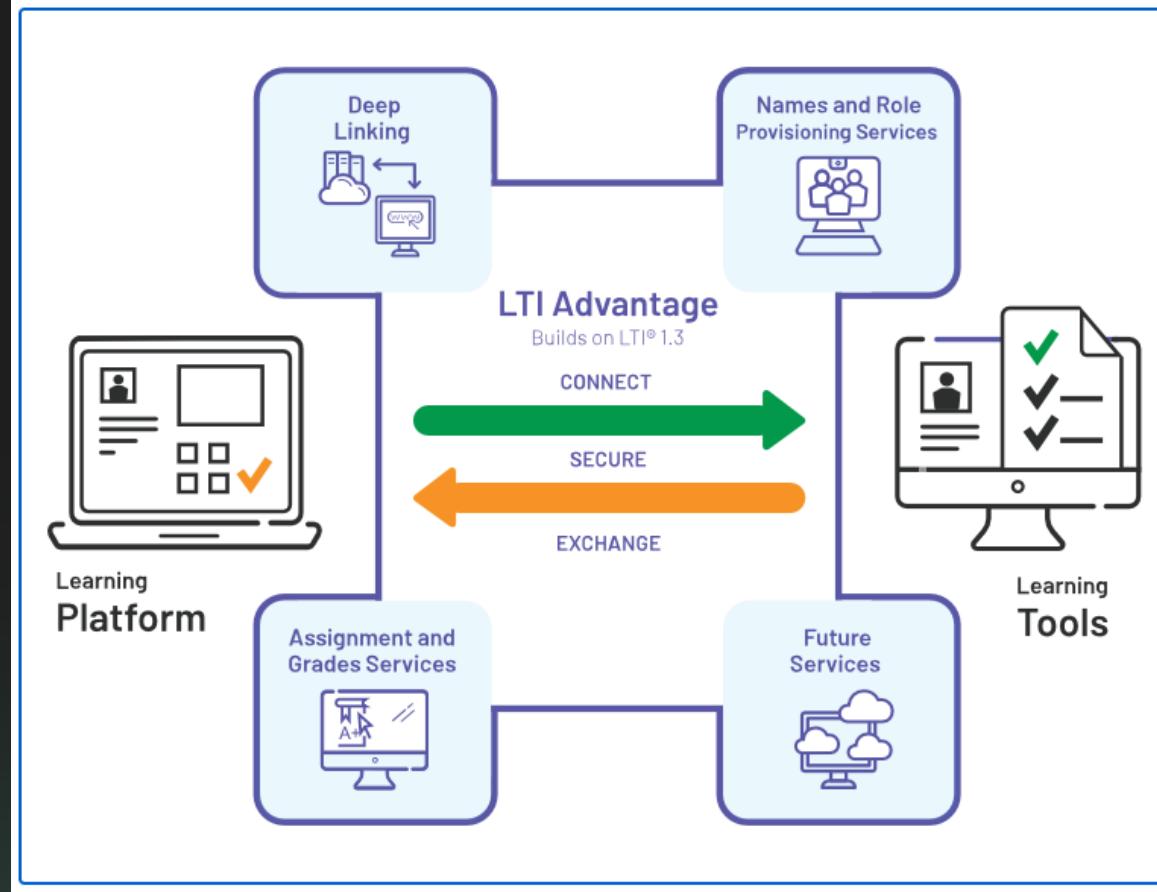


Names & Roles
Roster & Groups



Assignment & Grades
Get & Create grades

Another Look at LTI Advantage



Building an Application

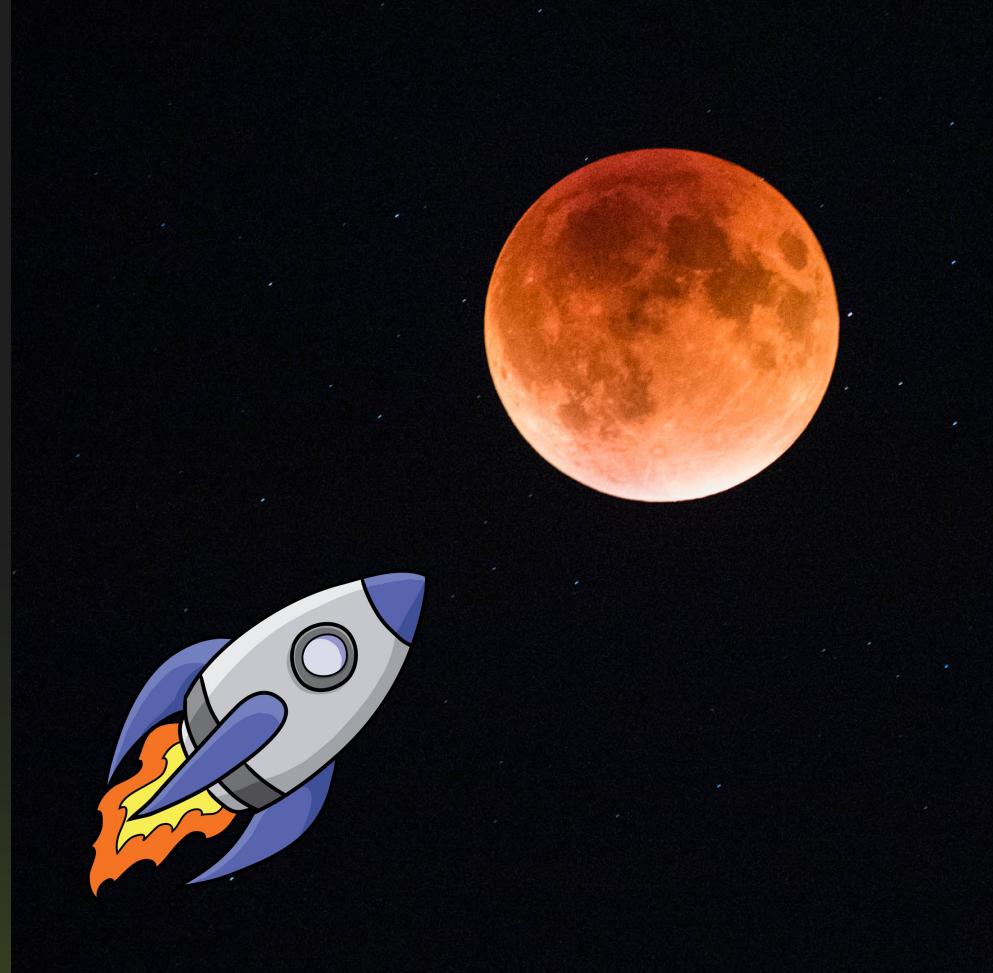
1. Build an awesome web application
 - Language and UI framework of choice
2. Support LTI Launch – accept FORM POST
3. Use REST to get and set Learn data
4. Configure Private and Public keys
5. Deploy it somewhere (AWS, Azure, ...)
6. Get IMS Global certification
7. Register it with Blackboard **ONCE**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

The screenshot shows a resource page on the OER Commons platform. At the top, there is a navigation bar with links for 'Discover', 'Hubs', 'Groups', 'Our Services', 'Add OER', and 'Sign In/Register'. Below the navigation bar, the title of the resource is 'English practice with a virtual tour of Van Gogh's Bedroom'. It has a rating of 4 stars from 1 user. A 'View Resource' button is present. To the right of the title, there are icons for 'Version History' and other sharing options. The main content area includes a thumbnail image of a painting, statistics (309 views, 11 comments), and a 'Description' section. The 'Description' section provides an overview, subject, level, grades, material type, author, and date added. It also mentions the license (Creative Commons Attribution) and media format (Audio, Downloadable docs, Graphics/Photos, Interactive, Text/HTML, Video, Other). On the right side, there are sections for 'Standards' (No Alignments yet), 'Evaluations' (No evaluations yet), and 'Tags' (22), which include categories like Class Discussion, English Practice, Oral and Listening Skills, Reading, Recordings of Podcasts, Teaching English as a Foreign Language, Teaching English With Arts, Teaching English With Paintings, Teaching English as a Second or Other Language, and Text and Podcast as Teaching Resources.

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.

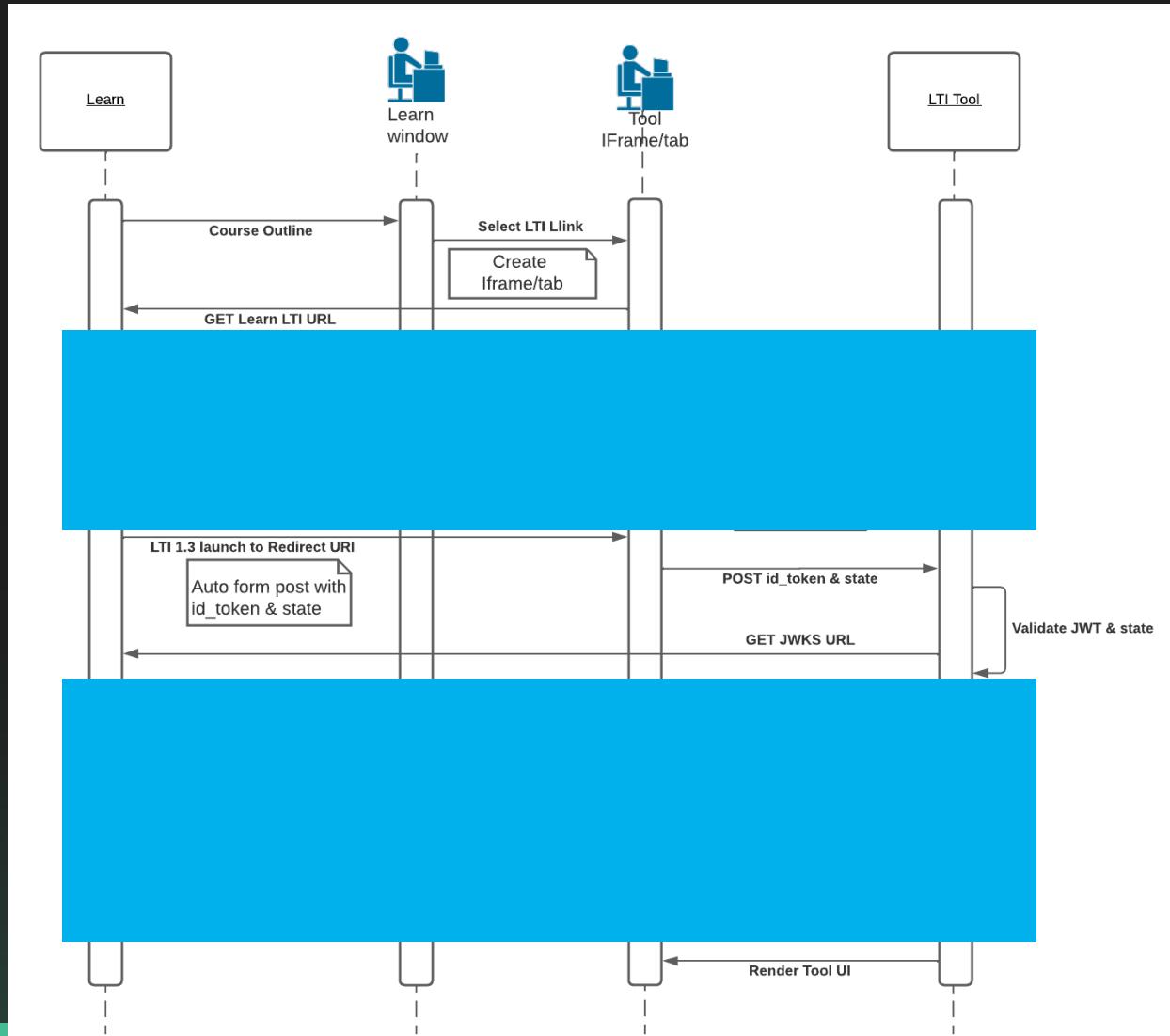
OpenID Connect (OIDC)

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.

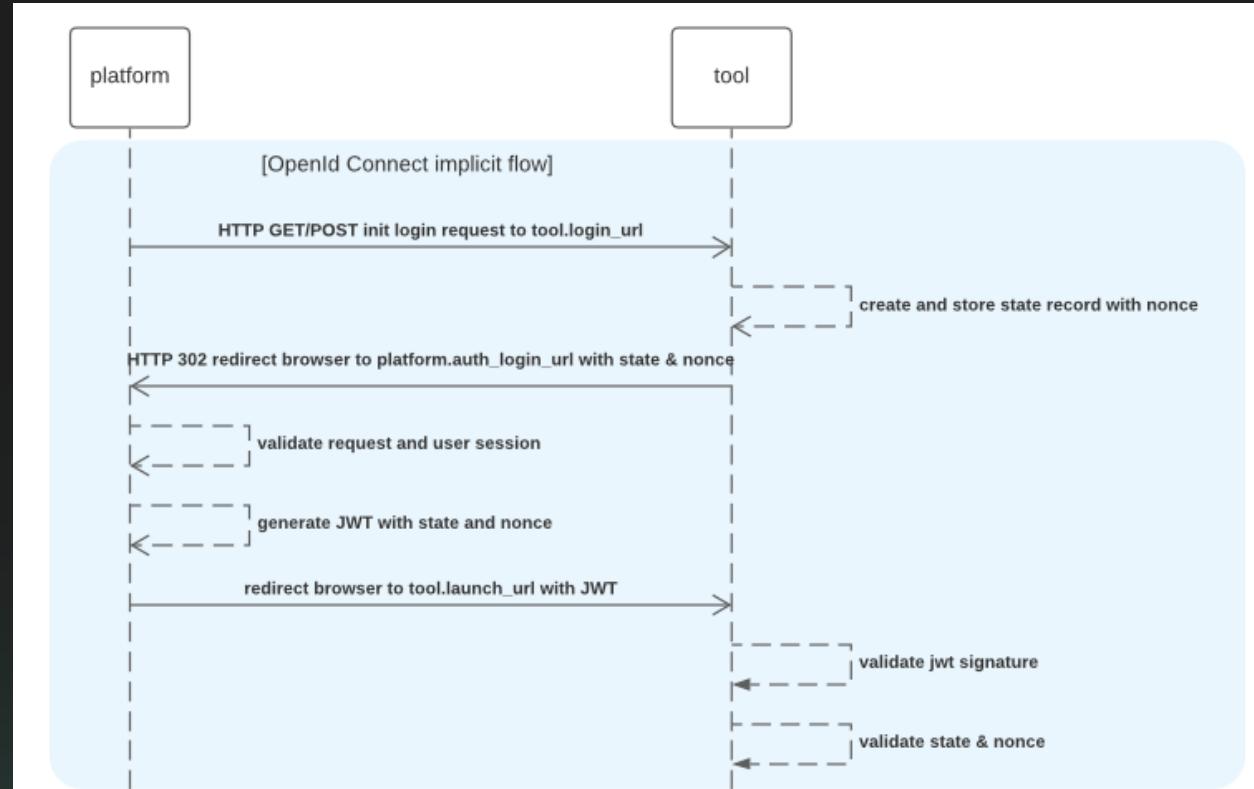
OAuth 2.0 Grant Types

- 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



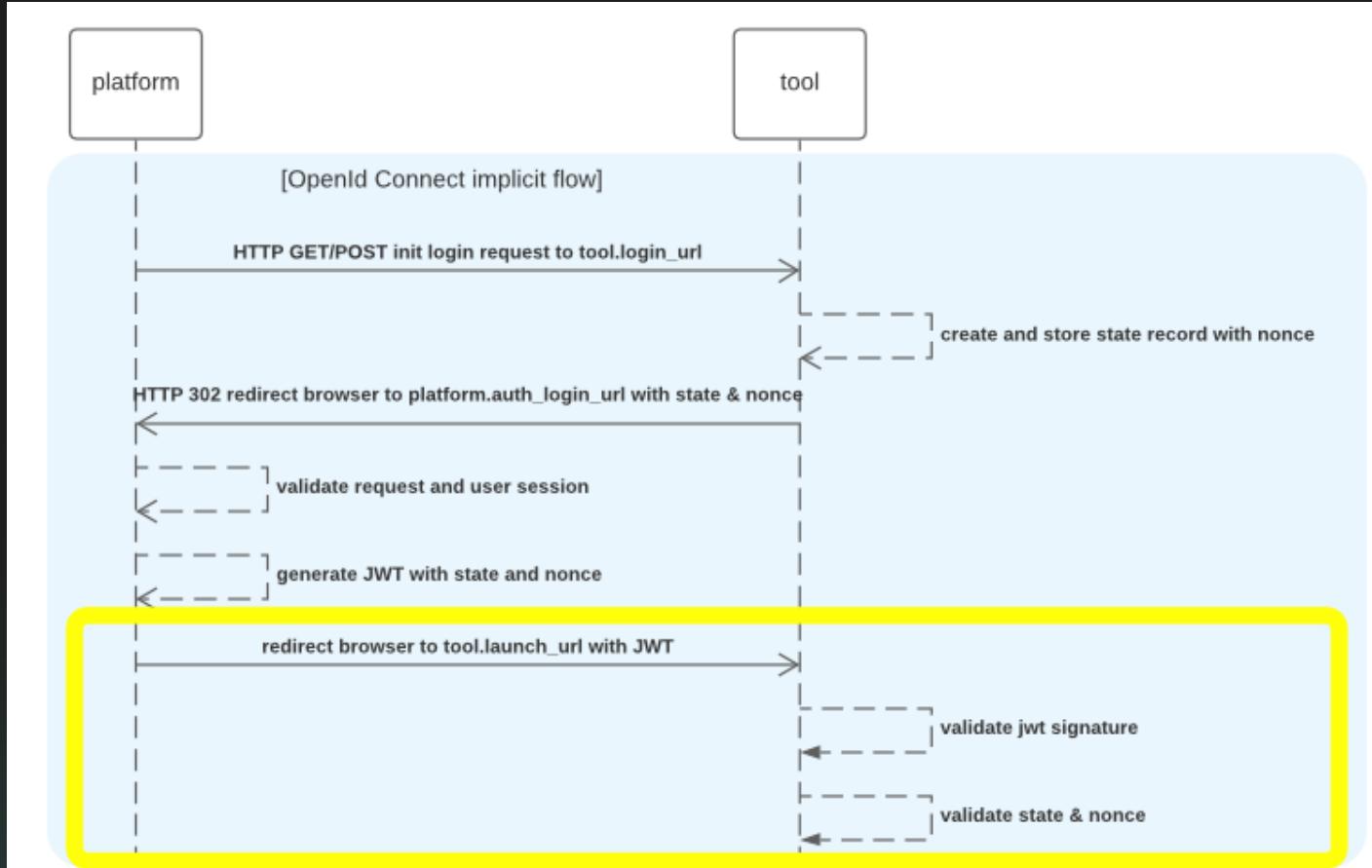
OIDC Login Request – Query Params

```
1 def login(request):
2     client_id = request.values.get("client_id")
3     issuer = request.values.get("iss")
4     login_hint = request.values.get("login_hint")
5     lti_deployment_id = request.values.get("lti_deployment_id")
6     lti_message_hint = request.values.get("lti_message_hint")
7     target_link_uri = request.values.get("target_link_uri")
8
9     if not client_id:
10         abort(400, "InvalidParameterException - Missing client_id")
11     if not issuer:
12         abort(400, "InvalidParameterException - Missing issuer")
13     if not login_hint:
14         abort(400, "InvalidParameterException - Missing login_hint")
15     if not lti_deployment_id:
16         abort(400, "InvalidParameterException - Missing lti_deployment_id")
17     if not target_link_uri:
18         abort(400, "InvalidParameterException - Missing target_link_url")
19
```

OIDC Login Request – Build Response

```
20     try:
21         lti_platform = LTIPlatform(LTIPlatformStorage()).load(client_id, issuer, lti_deployment_id)
22         state = LTISState(LTISStateStorage()).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>

Algorithm RS256

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJdWIiOiIxMjM0NTY3ODkwIiwbmFtZSI6Ikpvag4gRG9lIiwiYWRtaW4iOnRydWUsImlhhdCI6MTUxNjIzOTAyMn0.NHVaYe26Mbt0YhSKkoKYdFVomg4i8ZJd8_-RU8VNbfte4TSMb4bXP313Y1NWACwyXPGffz5aXHc6lty1Y2t4SWRqGteragsVdZufDn5BlnJ19pdR_kdVFUsra2rWKEofkZeIC4yWytE58sMiihvoH1ScmmVwBcQP6XETqYd0aSHp1g0a9RdUPDvoXQ5oqygTqVtxaDr6wUFKrKItgBMzWIdNz6y709E0DHEPTbE9rfBo6KTFsHAZnMg4k68CDp2woYIaXbmYTWCvbzIuH07_37GT79XdIwkm95QJ7hYC9RiwrV7mesbY4PAahERJawntho0my942XheVLmGwLMBkQ
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{ "alg": "RS256", "typ": "JWT" }
```

PAYOUT: DATA

```
{ "sub": "1234567890", "name": "John Doe", "admin": true, "iat": 1516239022 }
```

VERIFY SIGNATURE

```
RSASHA256(  
base64UrlEncode(header) + "." +  
base64UrlEncode(payload),  
-----BEGIN PUBLIC KEY-----  
MIIBIjANBgkqhkiG9w0BAQEFAAOCA  
Q8AMIIIBCgKCAQEau1SU1LfVLPHCoz  
MxH2Mo  
-----BEGIN PRIVATE KEY-----  
MIIEvwIBADANBgkqhkiG9w0BAQEFA  
ASCBKKvggSlAgEEAoIBAQCC7JTUt9  
Us8cKj  
MzEfYyjiWA4R4/M2bS1GB4t7NXp98  
)
```

Signature Verified

SHARE JWT

Sample JWT from Learn

```
1  {
2    "aud": "63beec56-b8cf-4380-9f9c-953044ad7453",
3    "sub": "d7443496cba24f1b96cc97e6473b9265",
4    "https://purl.imsglobal.org/spec/lti/claim/deployment_id": "939c964e-07d6-4de5-86d3-4718a307ab8b",
5    "https://purl.imsglobal.org/spec/lti/claim/version": "1.3.0",
6    "iss": "https://blackboard.com",
7    "locale": "en-US",
8    "exp": 1656097351,
9    "iat": 1656093751,
10   "email": "████████@gmail.com",
11   "given_name": "Eric",
12   "name": "Eric Preston",
13   "family_name": "Preston",
14   "https://purl.imsglobal.org/spec/lti/claim/roles": [
15     "http://purl.imsglobal.org/vocab/lis/v2/membership#Instructor"
16   ],
17   "https://purl.imsglobal.org/spec/lti/claim/resource_link": {
18     "id": "_24894_1",
19     "title": "Py LTI Content"
20   },
21   "https://purl.imsglobal.org/spec/lti/claim/context": {
22     "id": "b21a35dbbcbe49e29b96d67ad3e77e5a",
23     "label": "ULTRA1",
24     "title": "An Ultra Course",
25     "type": [
26       "http://purl.imsglobal.org/vocab/lis/v2/course#CourseOffering"
27     ]
28   },
29   "https://purl.imsglobal.org/spec/lti/claim/tool_platform": {
30     "name": "Blackboard, Inc.",
31     "description": "Blackboard, Inc.",
32     "guid": "17a1780b7811432980507a45584cbe2e",
33     "url": "https://blackboard.com/api/v1/lti/tool"
34   }
35 }
```

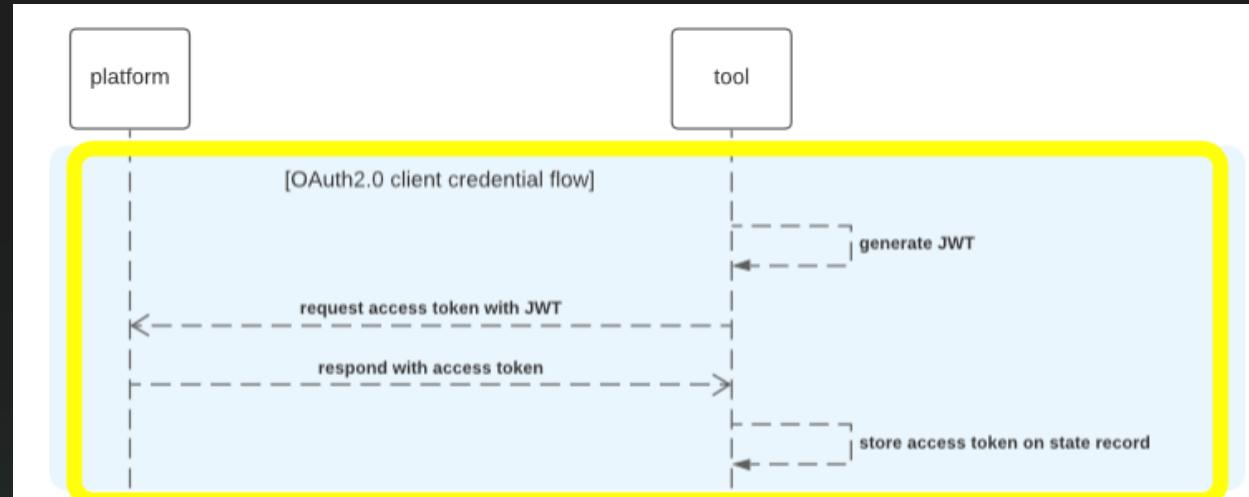
Handling Launch from Platform - Validate

```
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")
```

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 = LTIPPlatform(LTIPPlatformStorage()).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: LTISState = LTISState(LTISStateStorage()).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
```

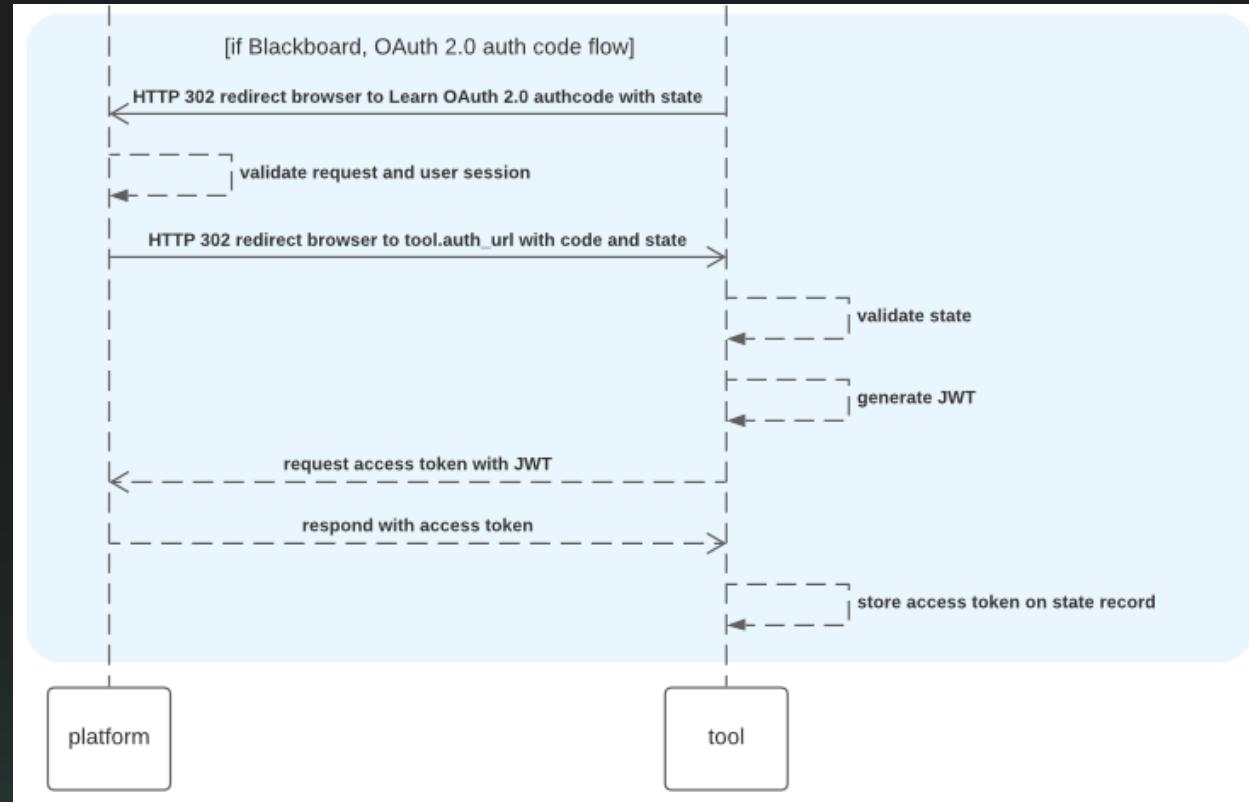
LTI OAuth2 Bearer Token Client Credentials Flow



Get LTI OAuth 2.0 Access Token

```
1 jwt = LTIJwtPayload()
2 time_now = datetime.datetime.now(tz=datetime.timezone.utc)
3
4 payload = dict(
5     aud=platform.config.auth_token_url,
6     exp=timegm(
7         (time_now + datetime.timedelta(seconds=int(300))).utctimetuple()
8     ),
9     jti=secrets.token_hex(16),
10    iat=timegm(time_now.utctimetuple()),
11    iss=platform.config.client_id,
12    sub=platform.config.client_id,
13 )
14
15 jwtstring = jwt.encode(payload=payload, tool=tool)
16
17 auth_request = {
18     "grant_type": "client_credentials",
19     "client_assertion_type": "urn:ietf:params:oauth:client-assertion-type:jwt-bearer",
20     "client_assertion": jwtstring,
21     "scope": lti_scopes,
22 }
23
24 r = requests.post(platform.config.auth_token_url, data=auth_request)
25 if not r.ok:
26     msg = f"Error retrieving access token from platform {platform.config.auth_token_url}. {r.reason}: {r.text}"
27     logging.error(msg)
28     raise Exception(msg)
29
30 # access token (bearer token) to be used to communicate with the Provider (LMS)
31 access_token = r.json()["access_token"]
32 return access_token
```

Learn REST OAuth2 Bearer Token Authorization Code Flow



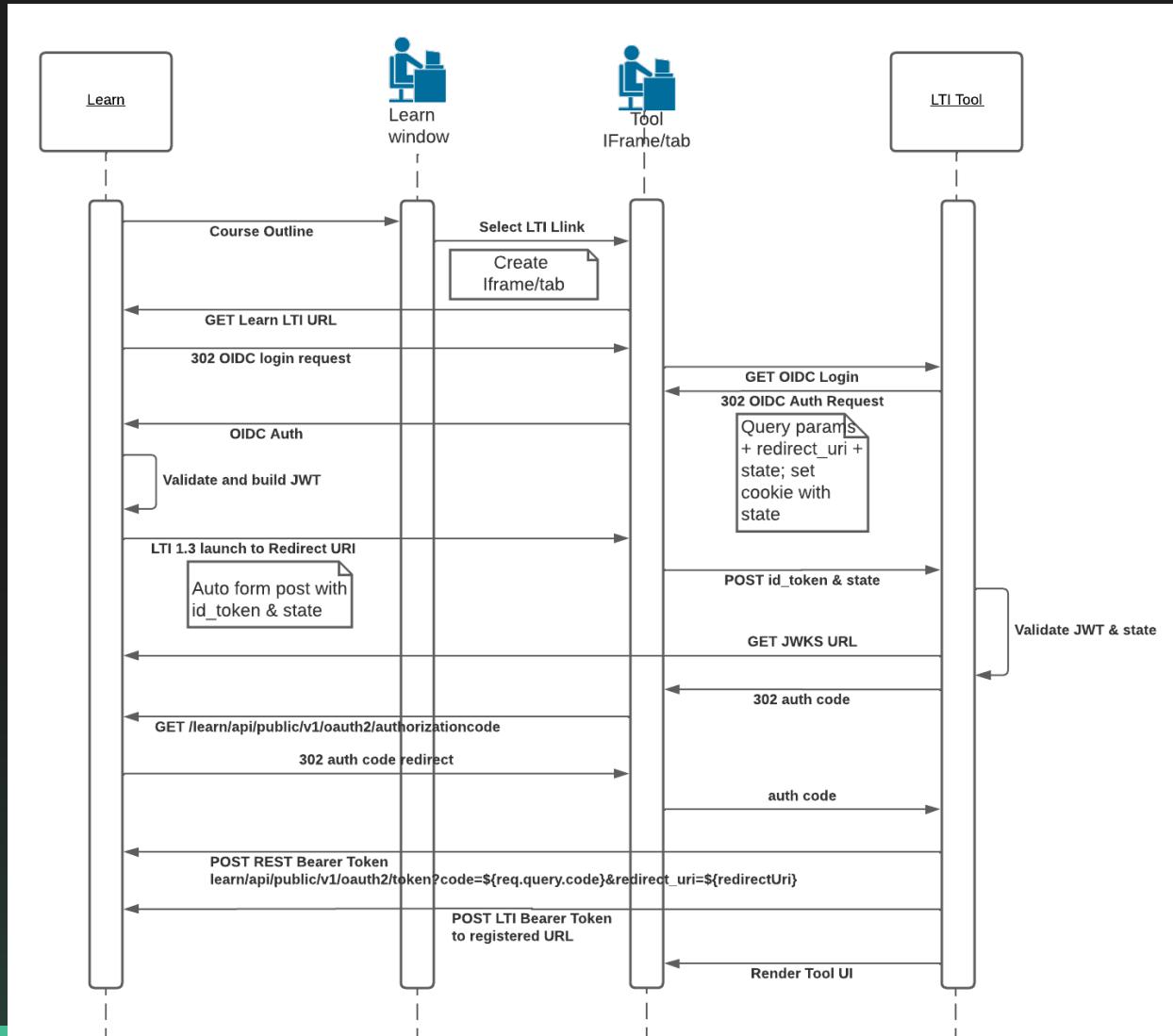
Construct Auth Code Request

```
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

```
1  def authcode(request):
2      auth_code = request.args.get("code", "")
3      request_cookie_state = request.cookies.get("state")
4
5      if not auth_code:
6          abort(400, "InvalidParameterException - Missing auth code")
7      if not request_cookie_state:
8          abort(400, "InvalidParameterException - Missing state")
9
10     state: LTIStrate = LTIStrate(LTIStrateStorage()).load(request_cookie_state)
11     if not state:
12         abort(409, "InvalidParameterException - State not found")
13
14     try:
15         id_token = state.record.id_token
16         jwt_request = LTIJwtPayload(id_token)
17
18         lti_tool = LTITool(LTIToolStorage())
19         auth_code_url = lti_tool.config.auth_code_url()
20
21         # Get the Learn access token
22         if jwt_request.platform_product_code == "BlackboardLearn":
23             learn_url = jwt_request.platform_url.rstrip("/")
24             learn_access_token = TokenClient().get_learn_access_token(learn_url, auth_code_url, auth_code)
25             # Cache the REST access token
26             state.record.set_platform_learn_rest_token(learn_access_token)
27             state.save()
28
29         return launch_controller.render_ui(jwt_request, request_cookie_state, id_token)
30     except Exception as e:
31         abort(500, e)
```

Full LTI Launch Flow



Render UI

The screenshot shows a user interface for a knowledge check. At the top, there is a header bar with the Bb AWSWorkshop logo and Configuration link. Below the header, the title "AWS Workshop" is displayed. A dark banner below the title contains the text "Welcome Eric Preston". The main content area has a light gray background and features the title "LTI Tool Example" followed by "Knowledge Check". Below the title, there are three statements, each preceded by an unchecked checkbox:

- 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.

At the bottom center of the main content area is a green "Submit" button.

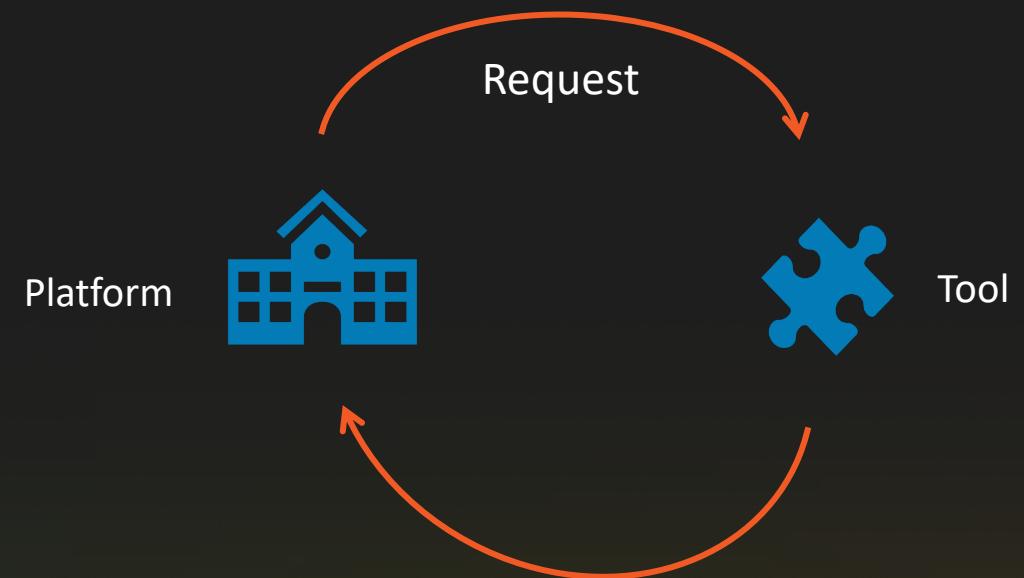
Post Grade to LMS

```
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

The screenshot shows a web-based application titled "AWS Workshop". The main header says "AWS Workshop" and "Welcome Richard Roe". A sub-header "Create a New Assignment" is displayed above a form. The form has two fields: "Name:" containing "AWS LTI Workshop" and "Points:" containing "100". A green "Ok" button is at the bottom, with a yellow circle highlighting it. A "JSON sample" button is at the bottom left, and a "v" icon is at the bottom right.

AWS Workshop

Welcome Richard Roe

Create a New Assignment

Name: AWS LTI Workshop

Points: 100

Ok

JSON sample

Create the Content Items

```
38 def get_assignment_content(name, points):
39     # Mock return value to simulate a assignment content item
40     # Ideally we'd create an assignment in our database and create a content item with that unique identifier
41     assignment_id = uuid.uuid4().hex
42
43     tool = LTITool(LTIToolStorage())
44     lti_launch_url = f"{tool.config.base_url}/launch"
45
46     content_item = dict(
47         type="ltiResourceLink",
48         title=name,
49         text="Do this assignment",
50         url=lti_launch_url,
51         lineItem=dict(
52             scoreMaximum=points, label=name, resourceId=assignment_id, tag="originality"
53         ),
54         custom=dict(
55             assignment_id=assignment_id,
56             userNameLTI="$User.username",
57             userIdLTI="$User.id",
58             contextHistory="$Context.id.history",
59             resourceHistory="$ResourceLink.id.history",
60         ),
61     )
62
63     return [content_item]
```

Create the Deep Link Response

```
65 def get_message_claims(jwt_request: LTIJwtPayload, content_items) -> dict:
66     claims = {
67         "iss": jwt_request.aud,
68         "aud": [jwt_request.iss],
69         "exp": int(time.time()) + 600,
70         "iat": int(time.time()),
71         "nonce": "nonce-" + str(uuid.uuid4().hex),
72         "https://purl.imsglobal.org/spec/lti/claim/deployment_id": jwt_request.deployment_id,
73         "https://purl.imsglobal.org/spec/lti/claim/message_type": "LtiDeepLinkingResponse",
74         "https://purl.imsglobal.org/spec/lti/claim/version": "1.3.0",
75         "https://purl.imsglobal.org/spec/lti-dl/claim/content_items": content_items,
76         "https://purl.imsglobal.org/spec/lti-dl/claim/data": jwt_request.deep_linking_settings_data,
77     }
78     return claims
```

Send Content to Platform as a FORM POST

```
1 def create_assignment(request):
2     name = request.form.get("name")
3     points = request.form.get("points")
4     id_token = request.form.get("id_token")
5     request_cookie_state = request.form.get("state")
6
7     if not name or not points or not id_token:
8         abort(400, "InvalidParameterException - Missing required parameter")
9
10    state: LTISState = LTISState(LTISStateStorage()).load(request_cookie_state)
11    if not state:
12        abort(409, "InvalidParameterException - State not found")
13
14    try:
15        jwt_request = LTIJwtPayload(id_token)
16        lti_tool = LTITool(LTIToolStorage())
17
18        content = get_assignment_content(name, points)
19
20        return_url = jwt_request.deep_linking_settings_return_url
21        deep_link_claims = get_message_claims(jwt_request, content)
22
23        jwt = LTIJwtPayload()
24        jwtstring = jwt.encode(payload=deep_link_claims, tool=lti_tool)
25
26        pretty_body = json.dumps(deep_link_claims, sort_keys=True, indent=2, separators=(", ", ": "))
27
28        return render_template(
29            "confirm_assignment.html",
30            pretty_body=pretty_body,
31            jwt=jwtstring,
32            return_url=return_url,
33        )
34    except Exception as e:
35        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

The screenshot shows a web browser displaying the Blackboard Developer Documentation API Explorer at developer.blackboard.com/portal/displayApi. The page has a dark header with the Blackboard logo, 'Developer Documentation', 'Explore APIs', 'My Applications', 'My Groups', 'My Account', 'Admin', and 'Logo'. A dropdown menu 'Latest Supported Version' is open. The main content area is titled 'discussions' and lists various API endpoints for managing discussions:

- GET /learn/api/public/v1/courses/{courseId}/discussions (Get Discussions)
- POST /learn/api/public/v1/courses/{courseId}/discussions (Create Discussion)
- GET /learn/api/public/v1/courses/{courseId}/discussions/{discussionId} (Get Discussion)
- PATCH /learn/api/public/v1/courses/{courseId}/discussions/{discussionId} (Update Discussion)
- GET /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/groups (Get Discussion Groups)
- PUT /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/groups/{groupId} (Create Discussion Group Association)
- GET /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/messages (Get Discussion Messages)
- POST /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/messages (Create Message)
- DELETE /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/messages/{messageId} (Delete Message)
- PATCH /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/messages/{messageId} (Update Message)
- GET /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/messages/{messageId}/replies (Get Message Replies)
- POST /learn/api/public/v1/courses/{courseId}/discussions/{discussionId}/messages/{messageId}/replies (Create Message Reply)

Below this section is another titled 'institutional hierarchy' with two endpoints:

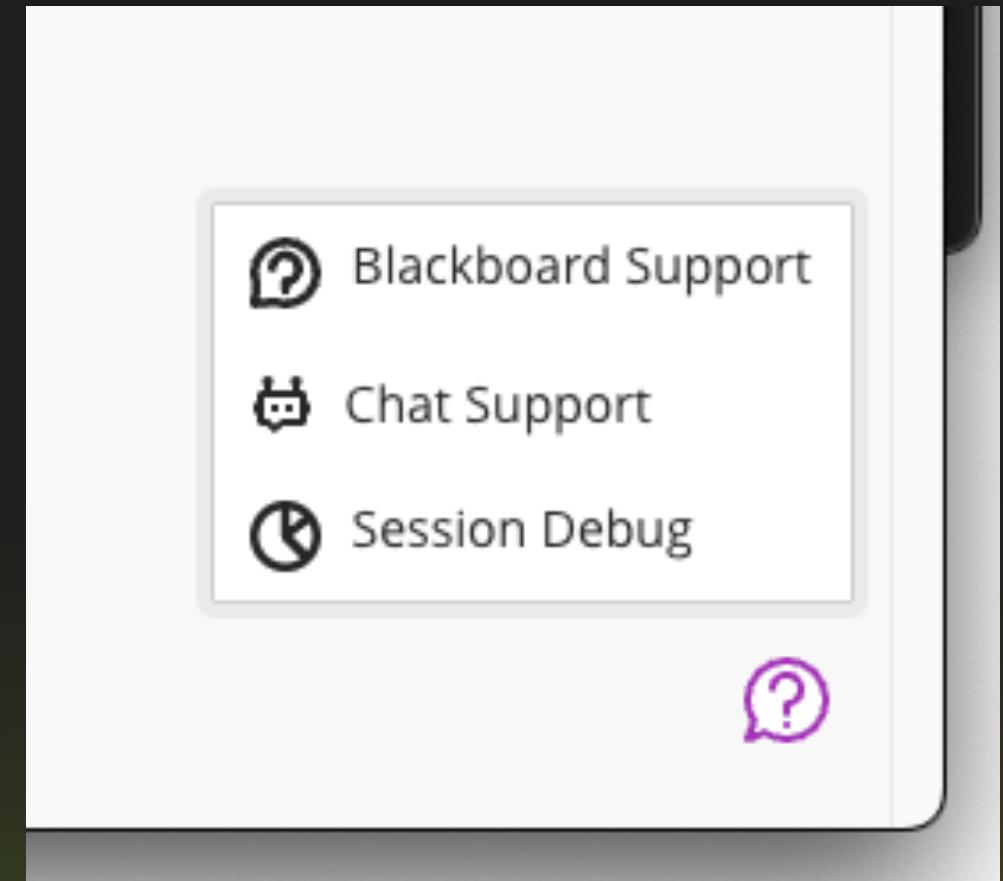
- GET /learn/api/public/v1/courses/{courseId}/nodes (Get Nodes For Course)
- GET /learn/api/public/v1/institutionalHierarchy/nodes (Get Nodes)

Call the Learn REST API with Access Token

```
80  def get_course_info(self, jwt_request: LTIJwtPayload, request_cookie_state):  
81      state: LTISState = LTISState(LTISStateStorage()).load(request_cookie_state)  
82      learn_access_token = state.record.get_platform_learn_rest_token()  
83      learn_url = jwt_request.platform_url.rstrip("/")  
84      course_uuid = jwt_request.context_id  
85      headers = {"Authorization": f"Bearer {learn_access_token}"}  
86      course_info_url = f"{learn_url}/learn/api/public/v2/courses/uuid:{course_uuid}"  
87      response = requests.get(course_info_url, headers=headers)  
88  
89      if response.status_code == 200:  
90          return response.json()  
91      else:  
92          self.__log().error(  
93              f"Error getting course info via Learn public API, status: {response.status_code}"  
94          )  
95      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

The screenshot shows a course navigation bar on the right side of a dark-themed interface. At the top, there's a "Details & Actions" section with various course links. Below it is a "My Custom Action" section with a blue icon and the text "Does something awesome". A yellow arrow points from this section down to a large, empty yellow-bordered box at the bottom right. The background is dark, and the overall theme is modern and customizable.

Details & Actions

- Roster [View everyone in your course](#)
- Course Description [View the course description](#)
- Course Groups [View sets & groups](#)
- Progress Tracking [Turn on](#)
- Course is open [Students can access this course](#)
- Blackboard Collaborate [Join session](#) ...
- Attendance [Mark attendance](#)
- Announcements [13 Posted | 14 Total](#)
- Books & Tools [View course & institution tools](#)
- Question Banks [Manage banks](#)

Chatbot – a UEF Example from Help

Blackboard

Stream

System Notifications

Today Monday, May 9, 2022

- 12 minutes **Introduction to Advertising**
New submissions ready to grade: Group Assignment - May Test
- 24 minutes **Introduction to Advertising**
Added: Group Assignment - May Test
No due date

Recent

May 6, 2022

- Food Science AGRO210 Sec08 Autumn2021**
New submissions ready to grade: Group Assignment Test for Receipt
- May 6, 2022 **Food Science AGRO210 Sec08 Autumn2021**
Added: Group Assignment Test for Receipt
Due Date: 5/7/22, 12:00 AM

May 4, 2022

- Introduction to Personal Finance**
Want to boost your grade?
Take a look at the course average to see where you stand.
View my activity

May 3, 2022

- Blackboard Announcement**
Check Out New Features Released to Production
With the release of v3900.39 to production, there are some exciting new features that will benefit...
- May 3, 2022 **Post-Impressionism 101 - Ultra**
Added: DNA Level 2
Due Date: 5/31/22, 6:00 PM
- May 3, 2022 **Introduction to Advertising**
1 student is falling behind

Privacy Terms

X Chat Support

You are viewing help content from Blackboard Learn Ultra. Select the Change content button to view help content from Blackboard Learn Original.



Let's Chat

Obi | 11:45 AM
Hello! I'm Obi your digital chat assistant. I can help you quickly find answers to many common questions or helpful information about campus resources and services. I can also help you navigate your Blackboard environment and many of the tools in your virtual classroom.

Before we get started, please tell me what role you're looking for help with.

Student **Instructor...**

Type a message

Change content **ULTRA** **ORIGINAL**

Register Your Application with Learn

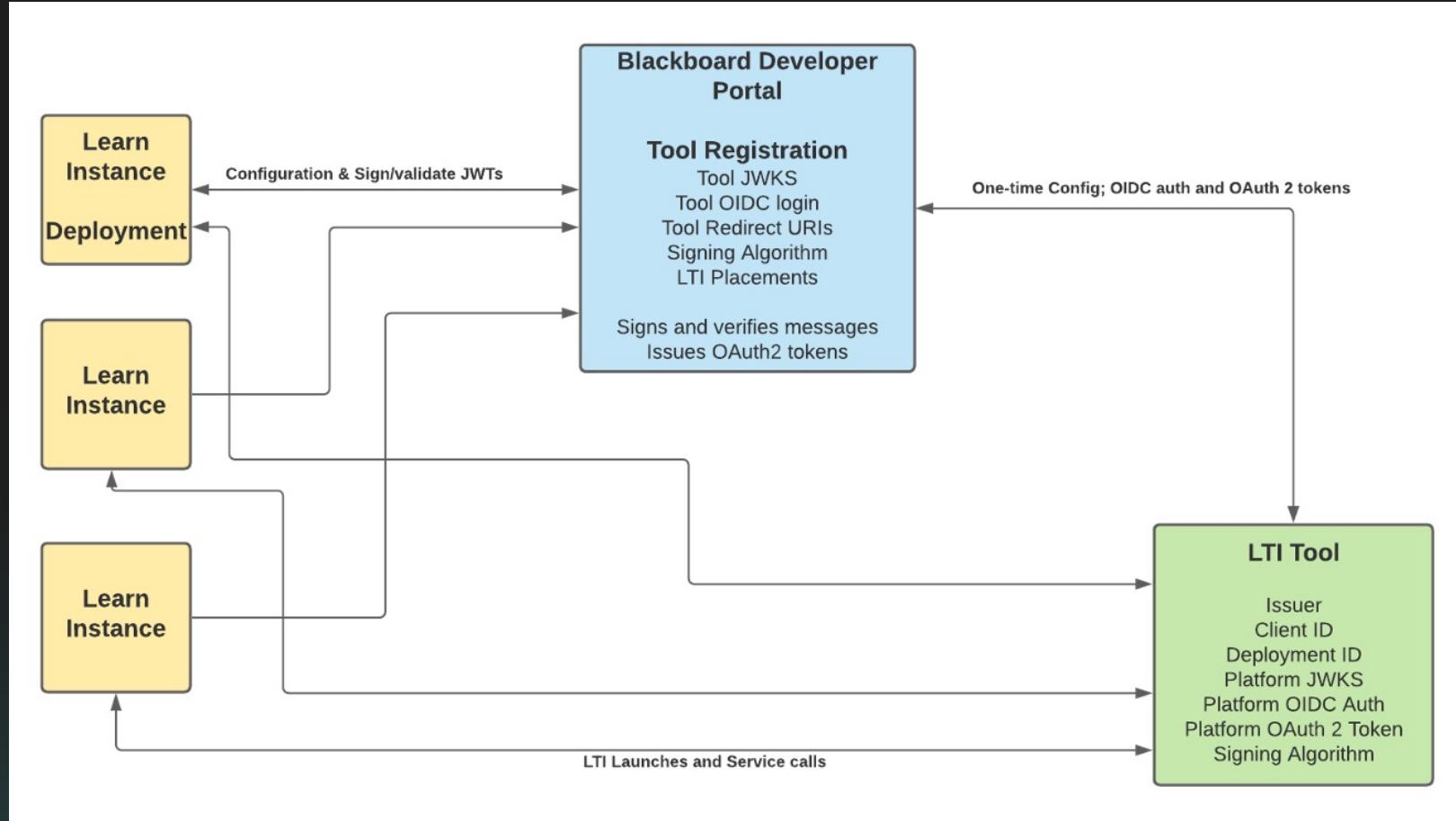
Platform Needs

- 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

The screenshot shows a web browser window with the URL `developer.blackboard.com/portal/applications/create`. The page title is "Register a new application". The form fields are as follows:

- Application Name:** AWS Workshop Tool
- Description:** A Python-based, ~~serverless~~ LTI tool for deploying to AWS
- Domain(s):** myawsdomain.com
- Group:** Blackboard Inc
- Trusted Service:** None
- My Integration supports LTI 1.3:** (checkbox checked)
- Login Initiation URL:** `https://myawsdomain.com/login`
- Tool Redirect URLs:** `https://myawsdomain.com/launch`
- Tool JWKS URL:** `https://myawsdomain.com/jwks.json`
- Signing Algorithm:** RS256
- Custom Parameters:** key-value

At the bottom of the form are two buttons: "Cancel" and "Register Application".

Copy Configuration Values

The screenshot shows a web browser window for the Blackboard developer portal at developer.blackboard.com/portal/applications/create. The page title is "AWS Workshop Tool Key". The navigation bar includes links for "Blackboard", "Developer Documentation", "Explore APIs", "My Applications", "My Groups", "My Account", "Admin", and "Logout". A note at the top says: "Important note! The secret is only shown once. Make note of the application key and secret and store them in a safe and secure location." Below this, several configuration values are listed with "Copy" buttons:

- Application key: [REDACTED]
- Secret: [REDACTED]
- Application ID: b88e2753-3b36-4af9-8ac8-724ddff565de
- Issuer: <https://blackboard.com>
- Public keyset URL: <https://developer.blackboard.com/api/v1/management/applications/b88e2753-3b36-4af9-8ac8-724ddff565de/jwks.json>
- Auth token endpoint: <https://developer.blackboard.com/api/v1/gateway/oauth2/jwttoken>
- OIDC auth request endpoint: <https://developer.blackboard.com/api/v1/gateway/oidcauth>

A "Done" button is located at the bottom left.

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

Blackboard [Developer Documentation](#) [Explore APIs](#) [My Applications](#) [My Groups](#) [My Account](#) [Admin](#) [Logout](#)

Register a new placement

Enter your placement information

* Placement Name

Description Limit placement description to 1000 characters

Type

Course tool

Course content tool

Deep Linking content tool

System tool

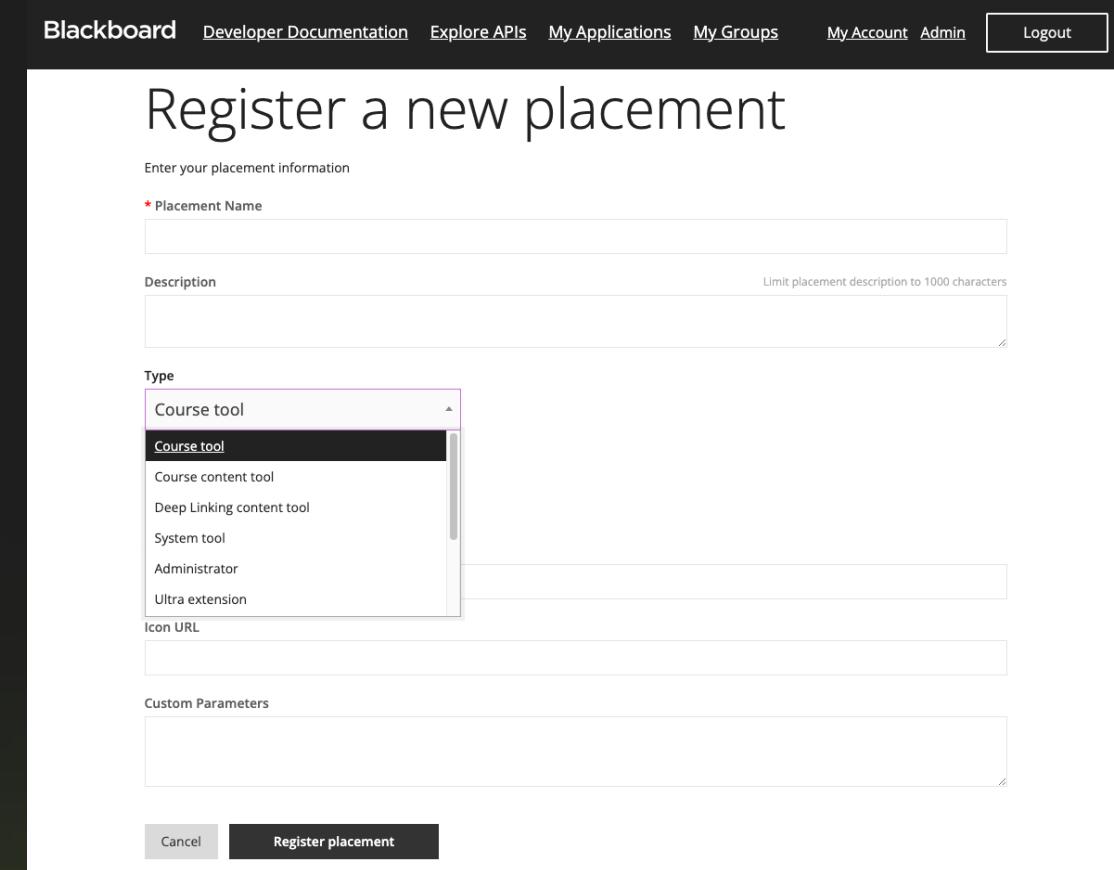
Administrator

Ultra extension

Icon URL

Custom Parameters

[Cancel](#) [Register placement](#)



Deploy to Learn

Copy Deployment ID

Administrator Tools

Close Administrator Panel

Administrator Panel LTI Tool Providers Register LTI 1.3 Tool

Register LTI 1.3/Advantage Tool

ENTER CLIENT ID

Client ID
78cd1b1c-ccbd-4318-9f90-22241f63b1f5

Type the Client ID for the tool you'd like to add.

Click **Submit** to proceed.

Cancel Submit

Administrator Tools

Close Administrator Panel

Client ID
78cd1b1c-ccbd-4318-9f90-22241f63b1f5

Name
Microsoft OneDrive LTI

Description
This application is used to enable the Microsoft OneDrive LTI integration.

Deployment ID
c4f5365f-6b21-460b-be12-7388abff3d12

Initiate Login URL
<https://onedrivelti.microsoft.com/oidlogin>

Tool Redirect URLs
<https://onedrivelti.microsoft.com/tool>

JWKS URL
<https://onedrivelti.microsoft.com/api/jwks>

Domains
onedrivelti.microsoft.com

Tool Status
 Approved
 Excluded

Tool Provider Custom Parameters
Enter any custom parameters required by the tool provider. Parameters must each be on their own line and be entered in "name=value" format.

INSTITUTION POLICIES

You can change the following settings for this tool. The fields use global values by default.

User Fields to Send
 Role in Course
 Name
 Email Address

Allow grade service access
 Yes
 No

Allow Membership Service Access
 Yes
 No

Click **Submit** to proceed.

Cancel Submit

Update Tool Configuration

Bb AWSWorkshop Configuration

Configuration

LTI Configuration

Client ID:

Issuer:

Platform JWKS URL:

Auth Token URL:

Auth Login URL:

Deployment ID:

Learn Configuration

Learn Application Key:

Learn Application Secret:

Ok

Use in Learn

An Ultra Course
Content Market

Select your content provider

PUBLISHER X CENGAGE McGraw Hill Education

Graduation cap icon

PUBLISHER X

Institution Tools

AWS Deep 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

Course Faculty

Eric Preston INSTRUCTOR Tom Bombadil LTI GRADER

Course Content

- AWS Assignment Due date: 3/8/22, 12:00 AM Visible to students Do this assignment
- Badger Discussion 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

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 Manage banks

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/ltitrainingcamp> - Many open source resources

Questions? developers@blackboard.com



anthology together

