# Enhanced Media Playback in Hyrax App Using Avalon Components

Chris Colvard
Indiana University

Brian Keese
Indiana University

Samvera Connect 2018

avalon
MEDIA SYSTEM

# A Brief History of Avalon

- One of the first ~~Hydra~~ Samvera Solution Bundles
- Solved a core problem (audio and video access)
- But it only focused on time-based media
- Grew quickly but grew up alone
- Currently in production at 10+ institutions

Northwestern University

INDIANA UNIVERSITY

samvera

Fedora™

avalon MEDIA SYSTEM

# Current Avalon implementers

Northwestern

INDIANA UNIVERSITY

BGSU

University of Virginia Library

UNIVERSITY OF ALBERTA LIBRARIES

Washington University in St. Louis

Fenway Library Organization

UNIVERSITY of HOUSTON LIBRARIES

Yale

UNIVERSITÄT SIEGEN

avalon MEDIA SYSTEM

avalon 5

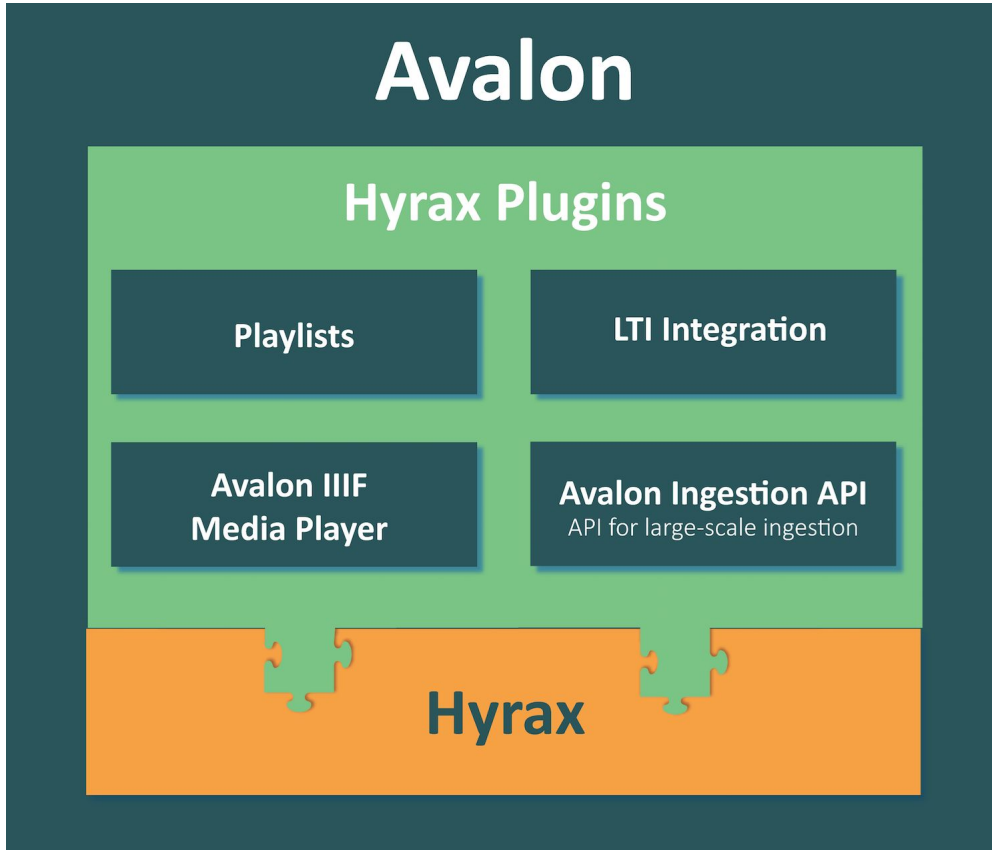
- Playlists and markers
- Ingestion API
- Captions

avalon 6

avalon 7

# So what will Avalon 7 look like?

- Use Hyrax as a starting point
- Contributions to Hyrax Core, Plugins, Work Type, and Custom code
- Avalon bundles it all up



# Creating small, reusable components

- Embeddable IIIF-AV player
- Transcoding and streaming
- Playlists including segments and annotations
- Avalon-style leases
- LTI support (e.g. Canvas)
- Ingest API

# IIIF for Audio/Video

avalon
MEDIA SYSTEM

# What is IIIF?

**International Image Interoperability Framework**

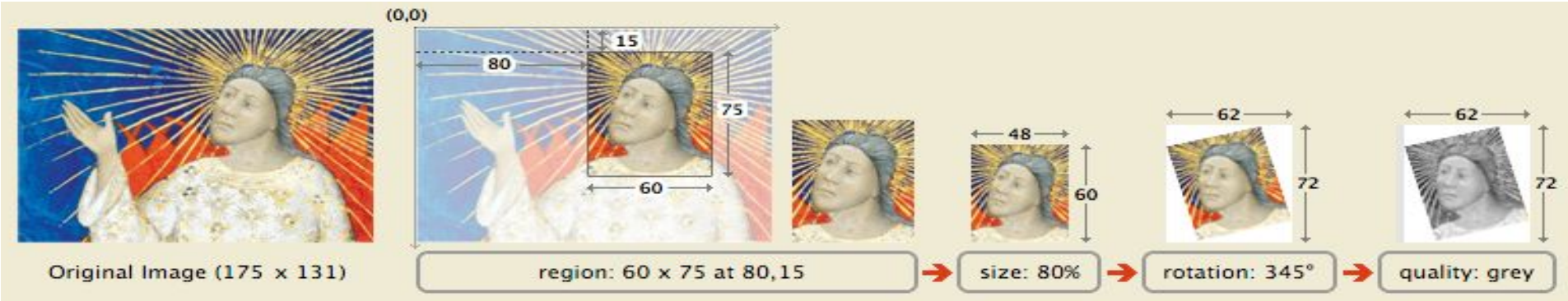
A **community** that develops **shared APIs**, implements them in **software**, and exposes interoperable **content**.

# Why? Interoperability!

- Many servers, many clients, many applications
- Publish once, reuse often
- Combine content from across repositories!

# Image API

"Facilitate systematic reuse of image resources in digital image repositories maintained by cultural heritage organizations."




**http://www.example.org/image-service/abcd1234/80,15,60,75/pct:80/345/grey.jpg**

# Example: Mirador

http://projectmirador.org/demo/

# Presentation API

"Provide the information necessary to allow a rich, online viewing environment for primarily image-based objects to be presented to a human user, likely in conjunction with the IIIF Image API."

# Example: Pages Online

https://pages.dlib.indiana.edu/concern/scanned_resources/ddf65v887r

# IIIF A/V

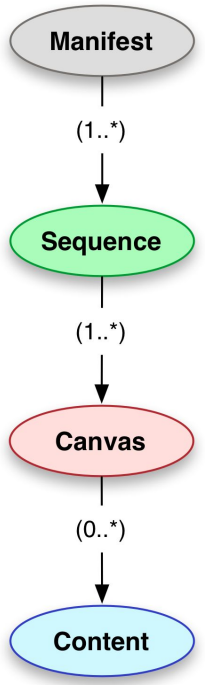
"Extend to A/V the benefits of interoperability and the growing ecosystem of clients and servers that IIIF provides for images."

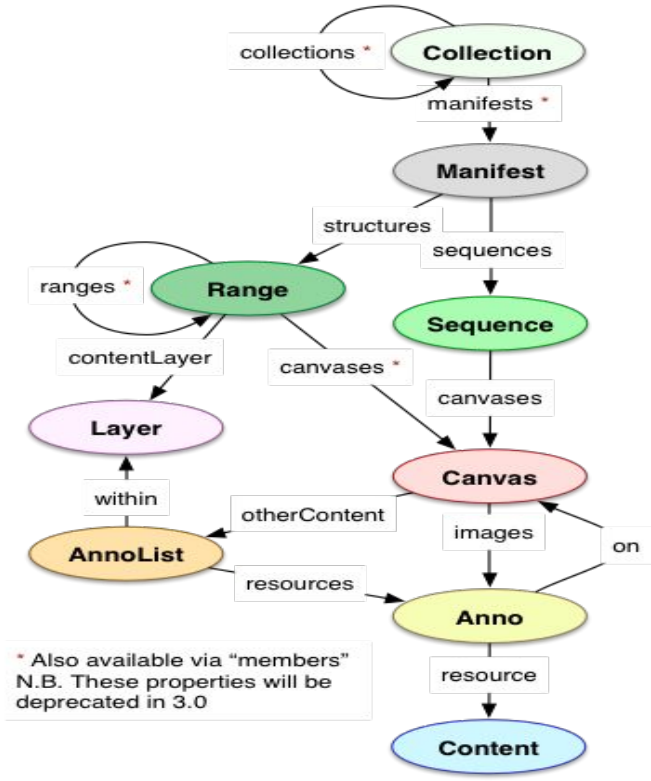
# Presentation API Version 3.0

- Extend canvas to have time dimension
- Remove assumptions of only images
- Annotating canvases on canvases
- Switch from Open Annotation to W3C Web Annotation Standard

# Presentation API

- JSON-LD manifest
- Descriptive information
- Shared canvas
- Static content or from Image API server

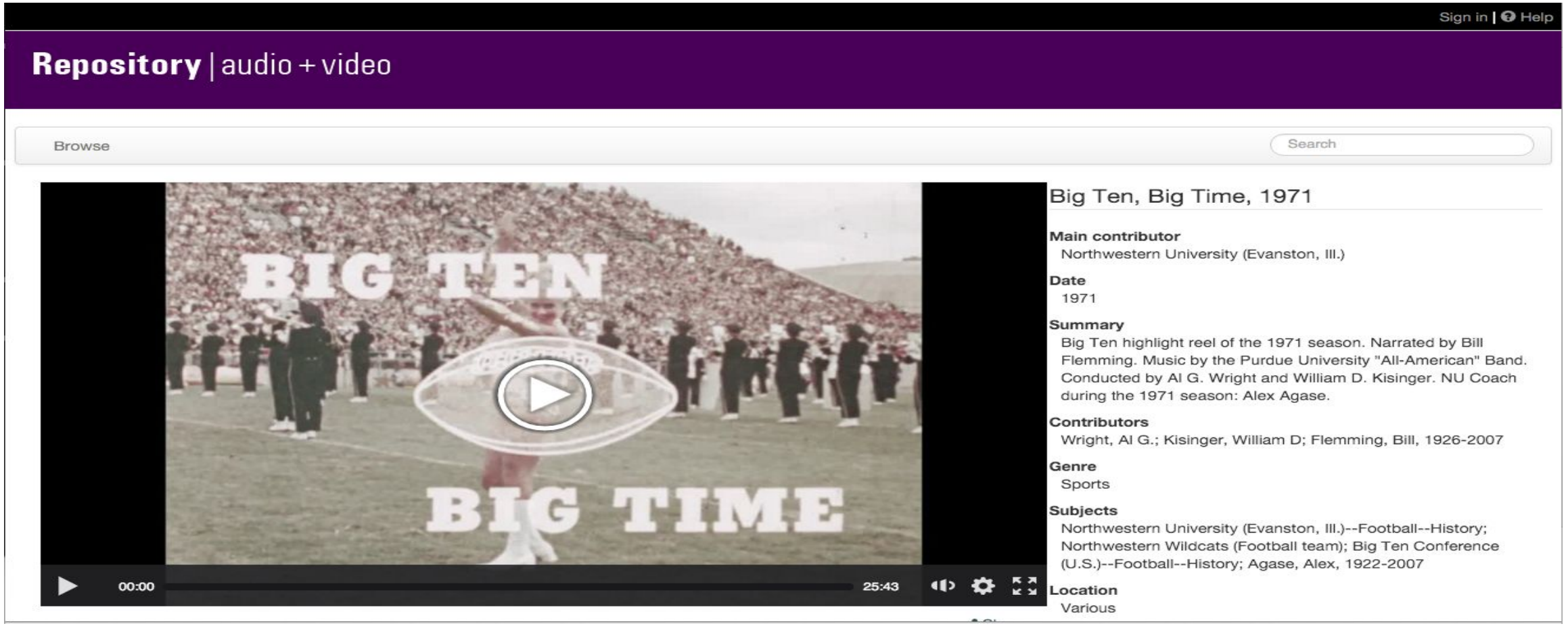


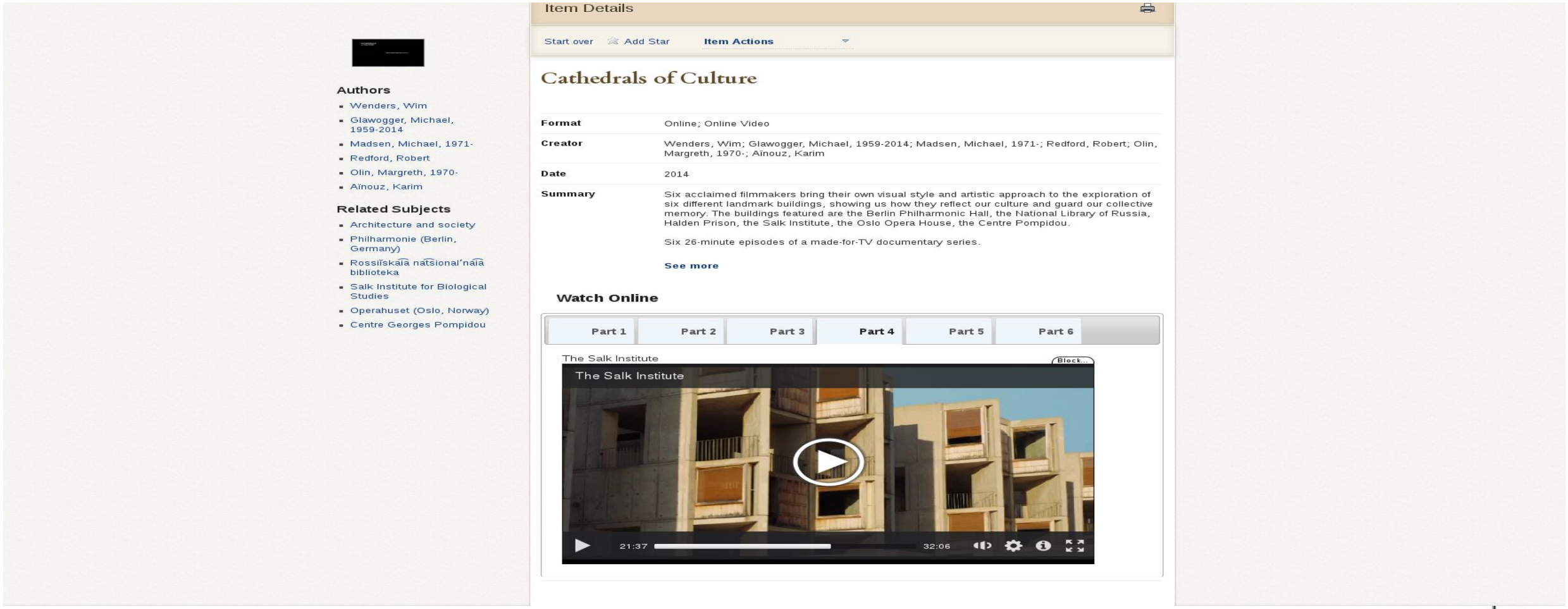

avalon MEDIA SYSTEM

# How would Avalon use IIIF?

# Use Case: Archival Film

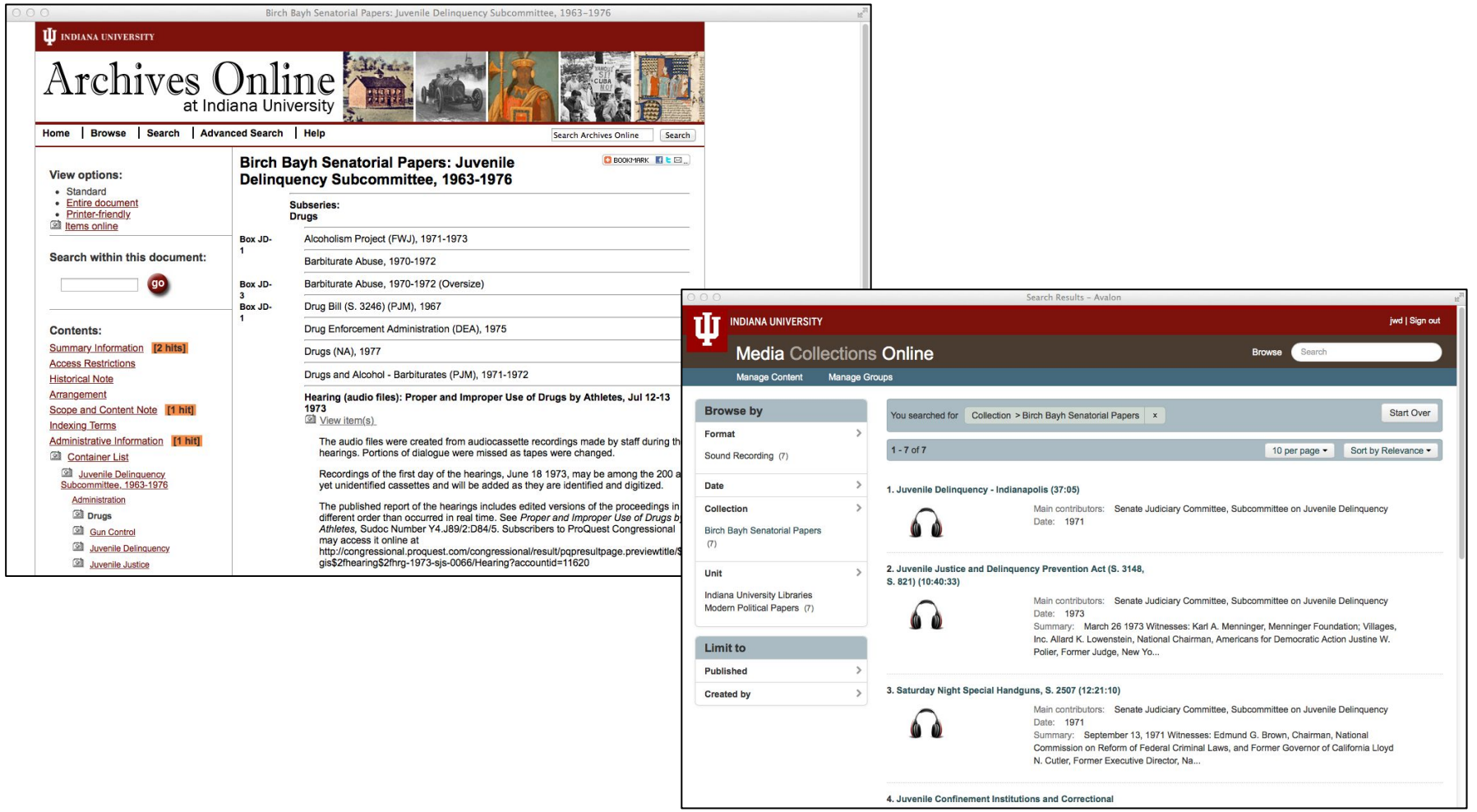


# Use Case: Consortial Licensed Video




avalon MEDIA SYSTEM

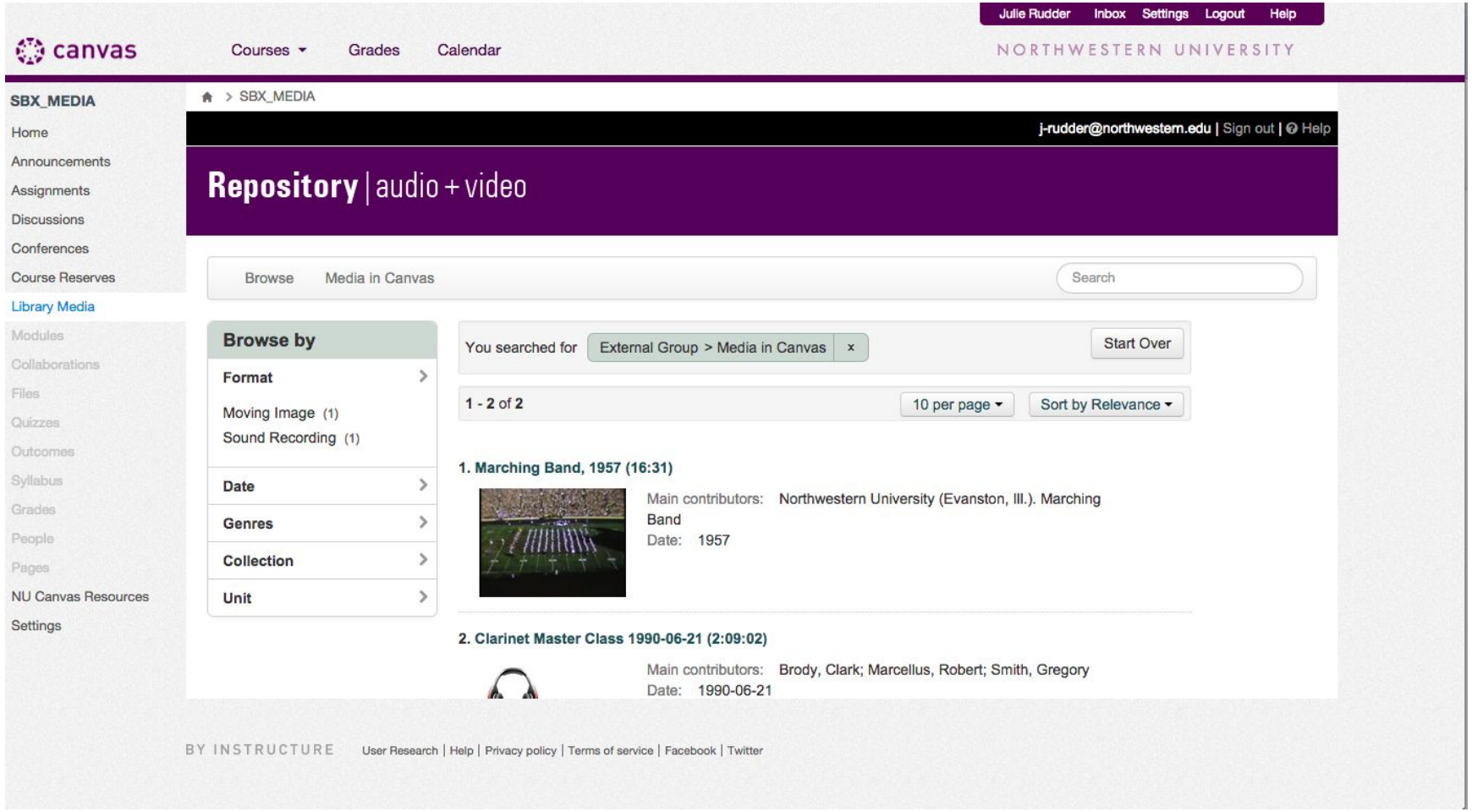
# Use Case: Media Within Archival Collections





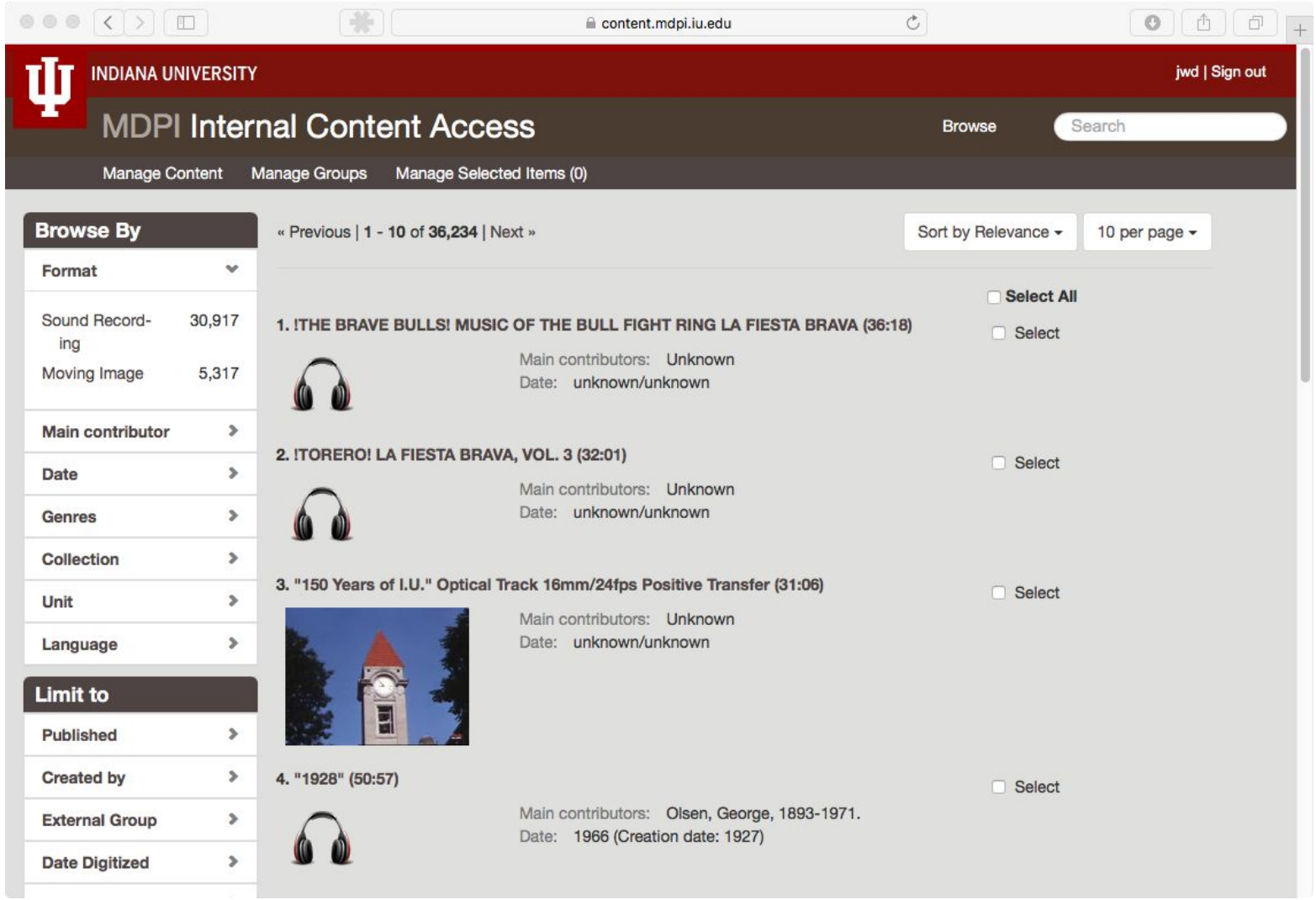

avalon MEDIA SYSTEM

# Use Case: Video Reserves in Learning Management System



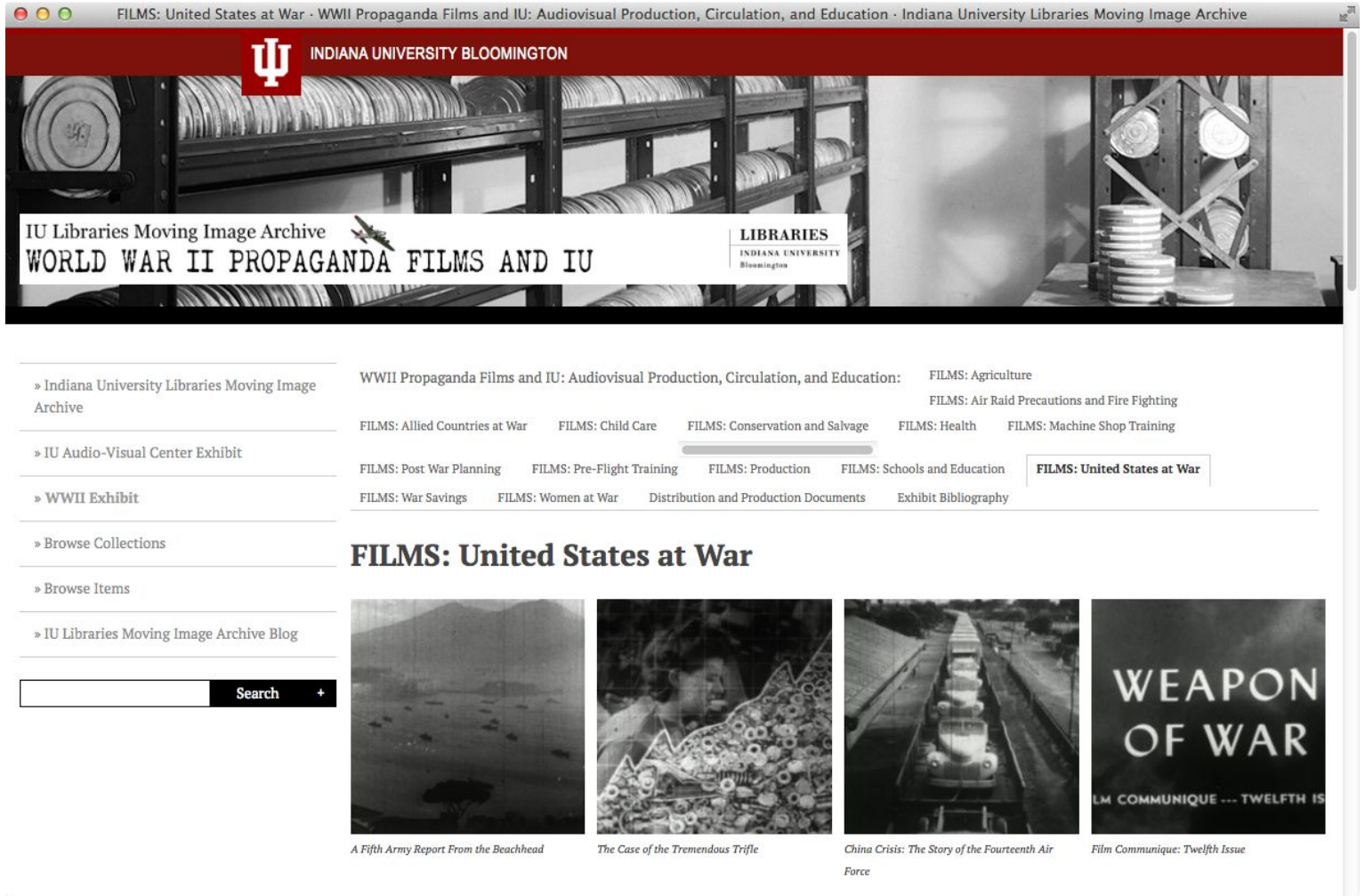

avalon MEDIA SYSTEM

# Use Case: Mass Digitization




avalon MEDIA SYSTEM

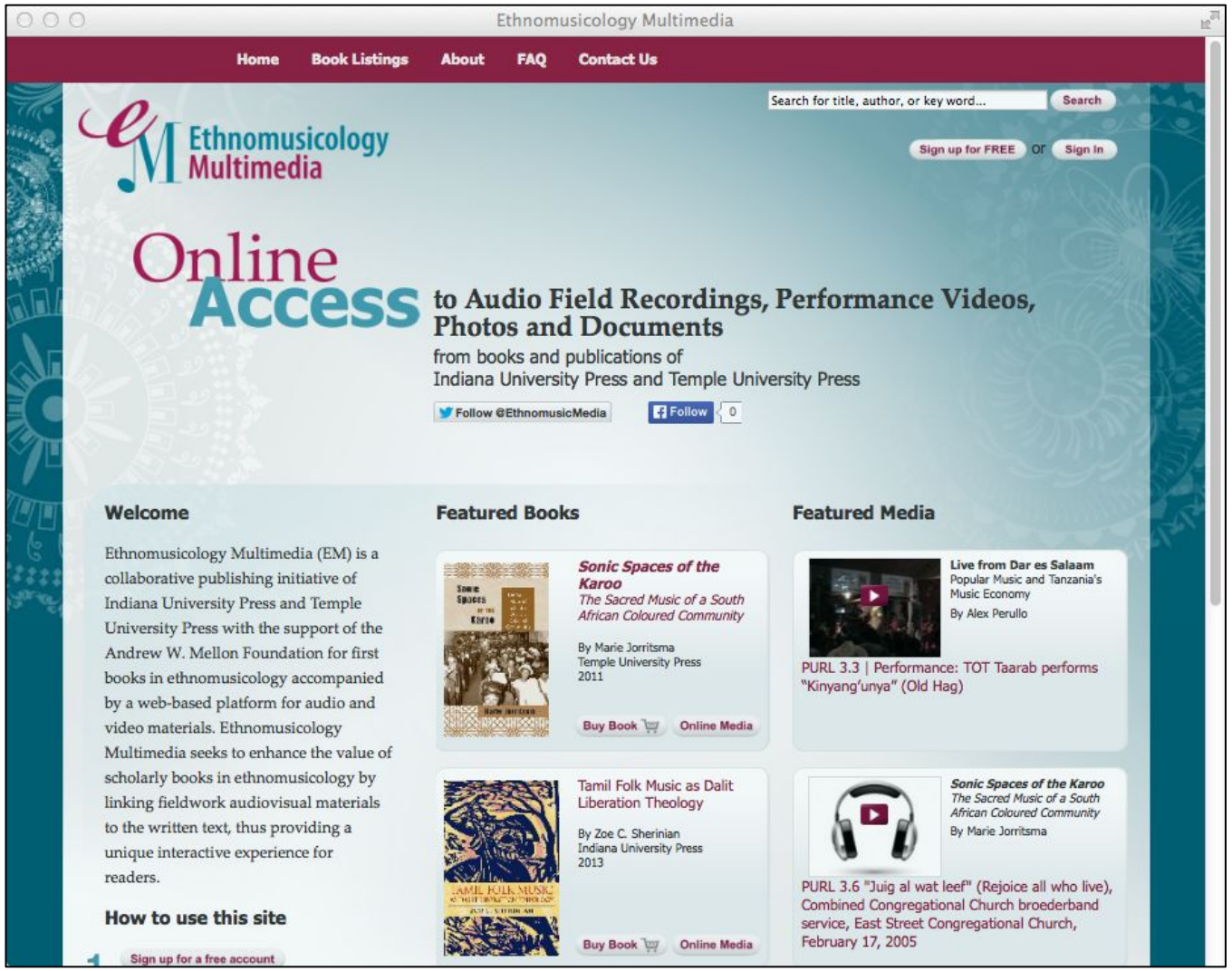
# Use Case: Online Exhibits



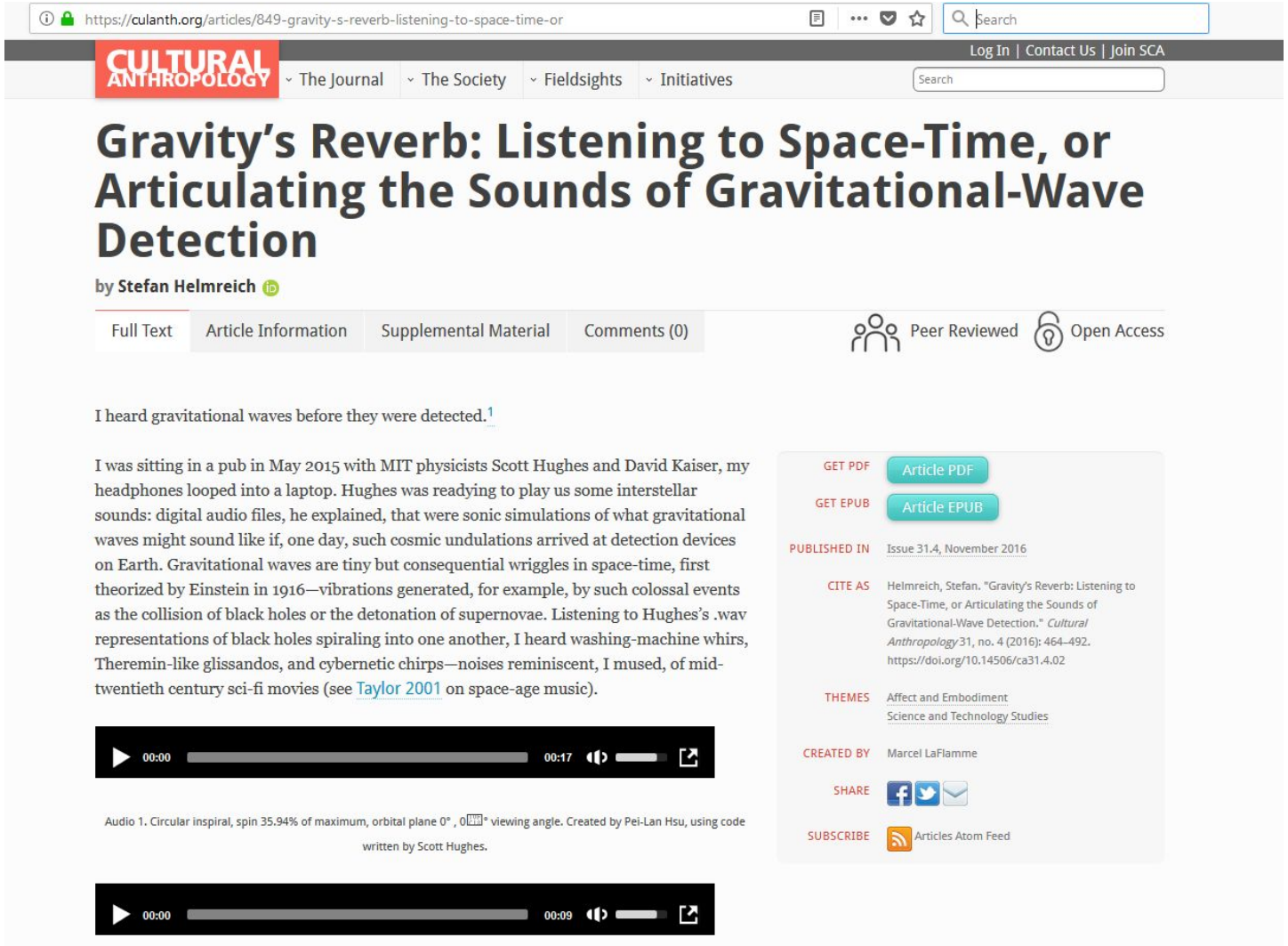



avalon MEDIA SYSTEM

# Use Case: Online Publishing

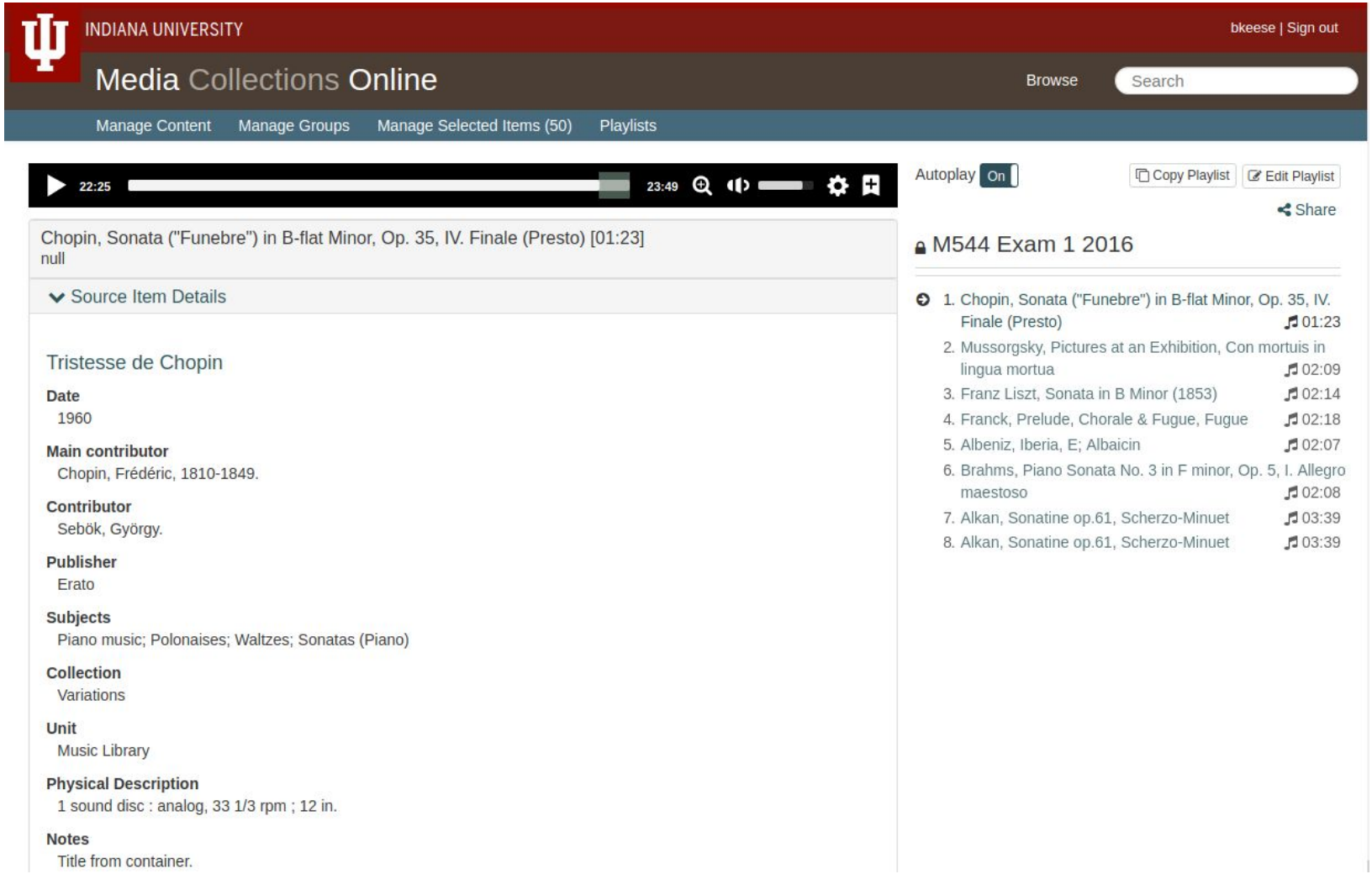


# Use Case: Online Publications

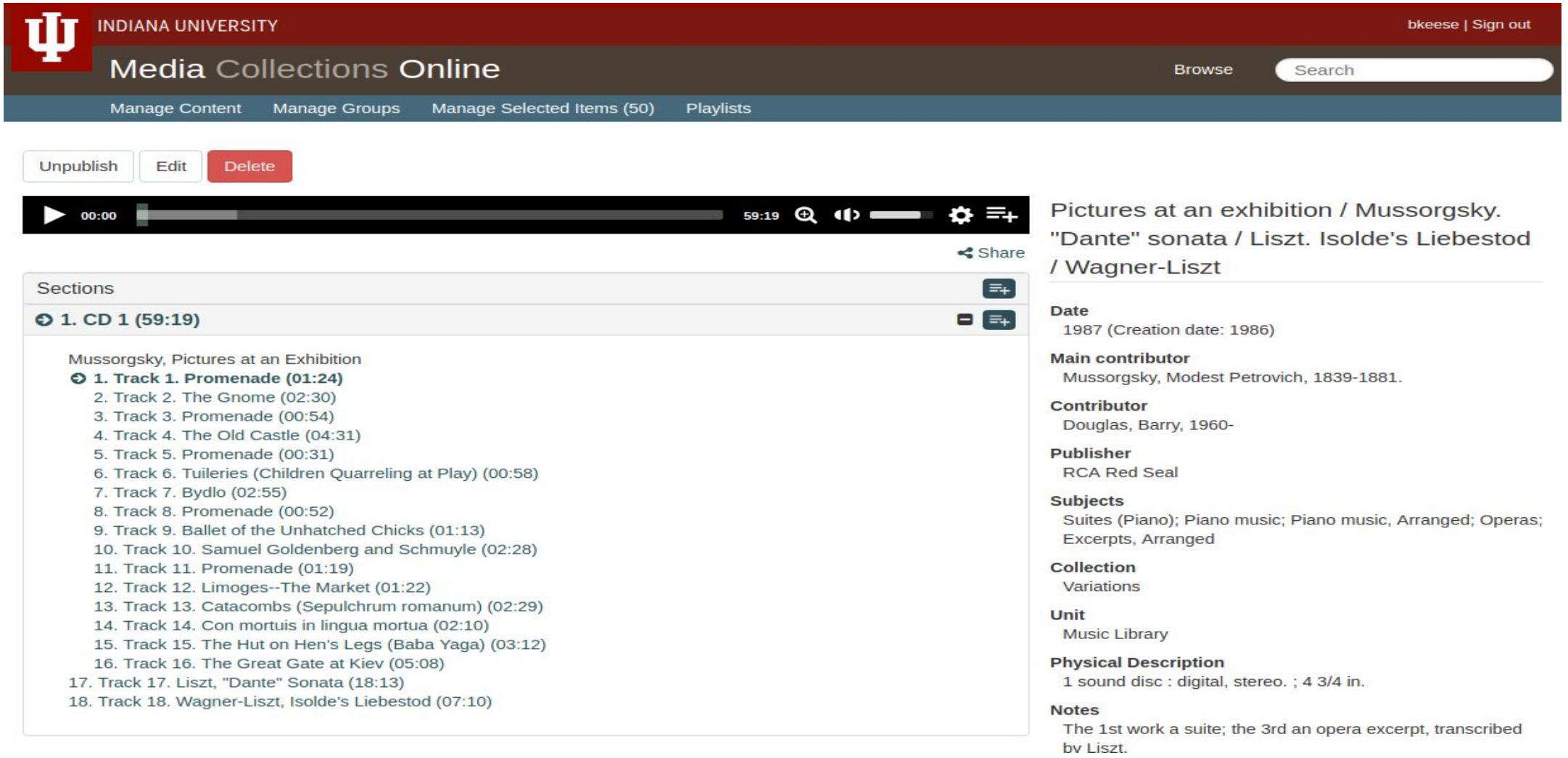



avalon MEDIA SYSTEM

# Use Case: Music Pedagogy

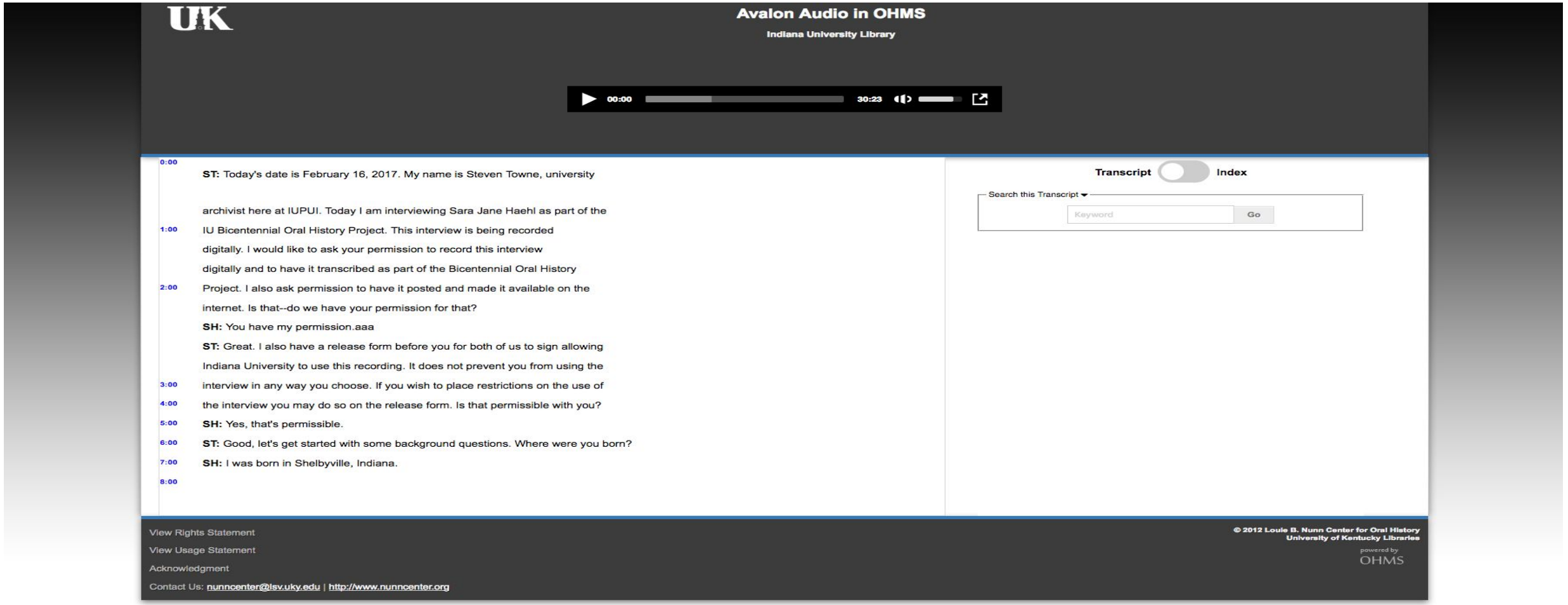


# Use Case: Music Pedagogy




avalon MEDIA SYSTEM

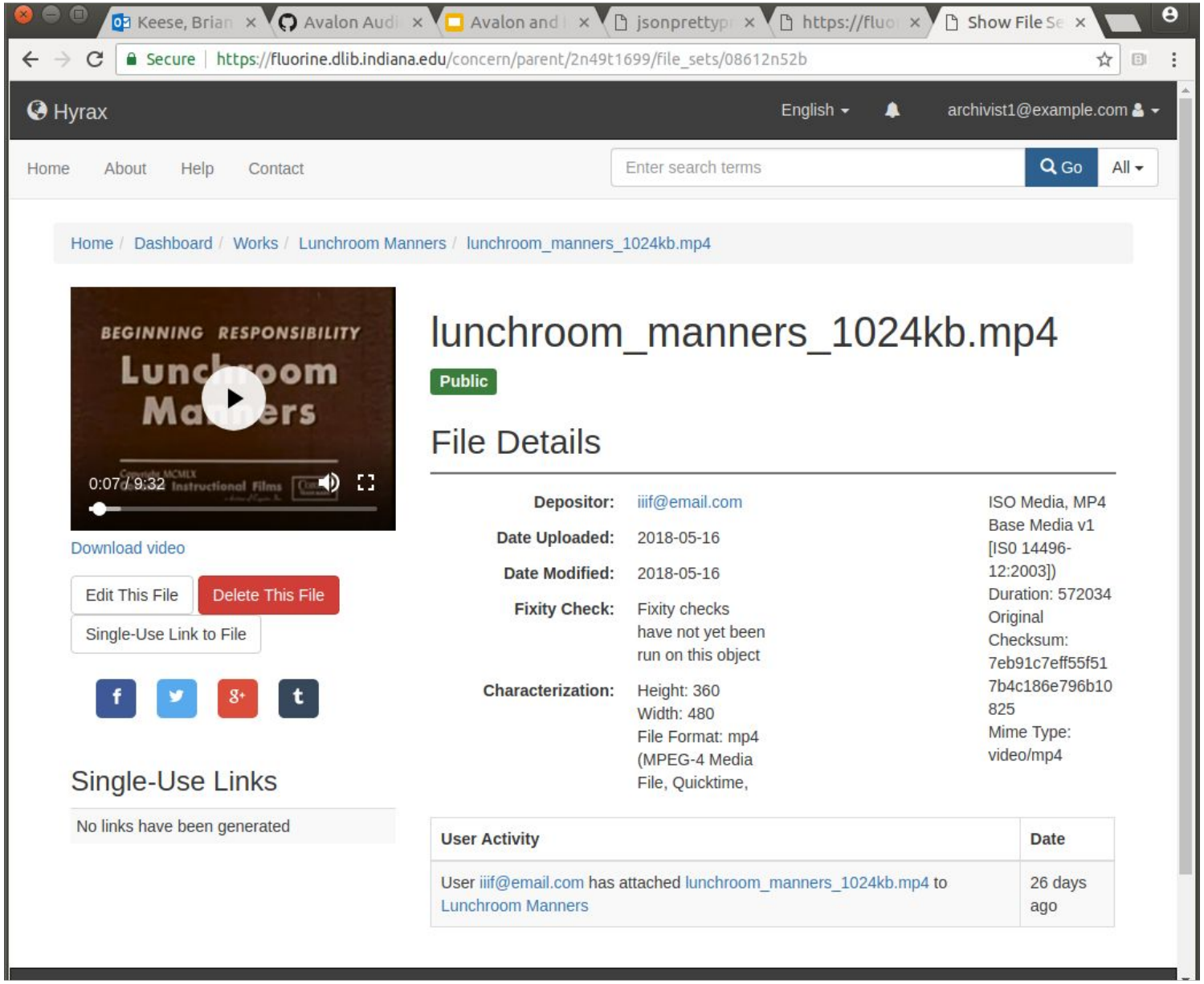
# Use Case: OHMS



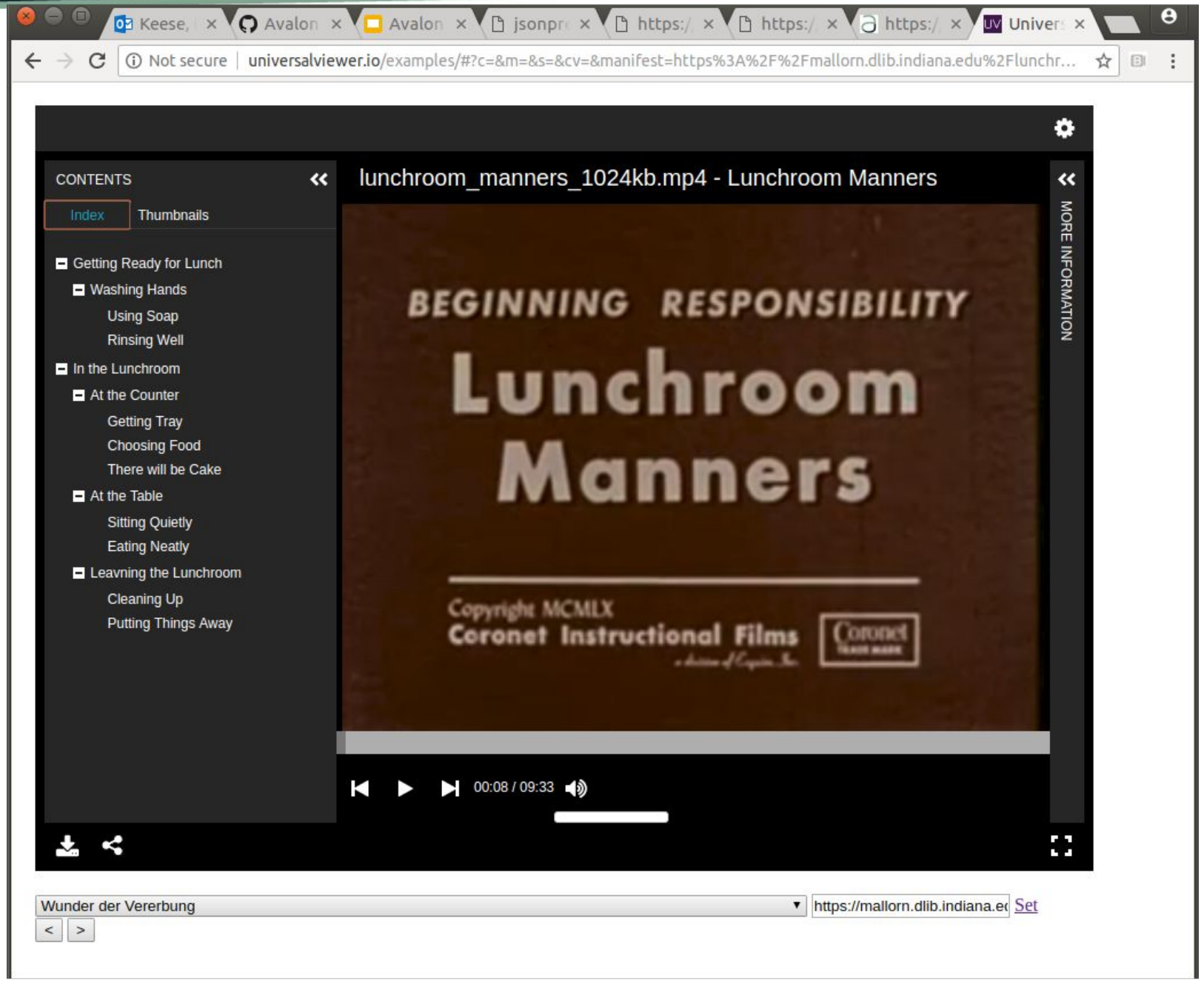

avalon
MEDIA SYSTEM

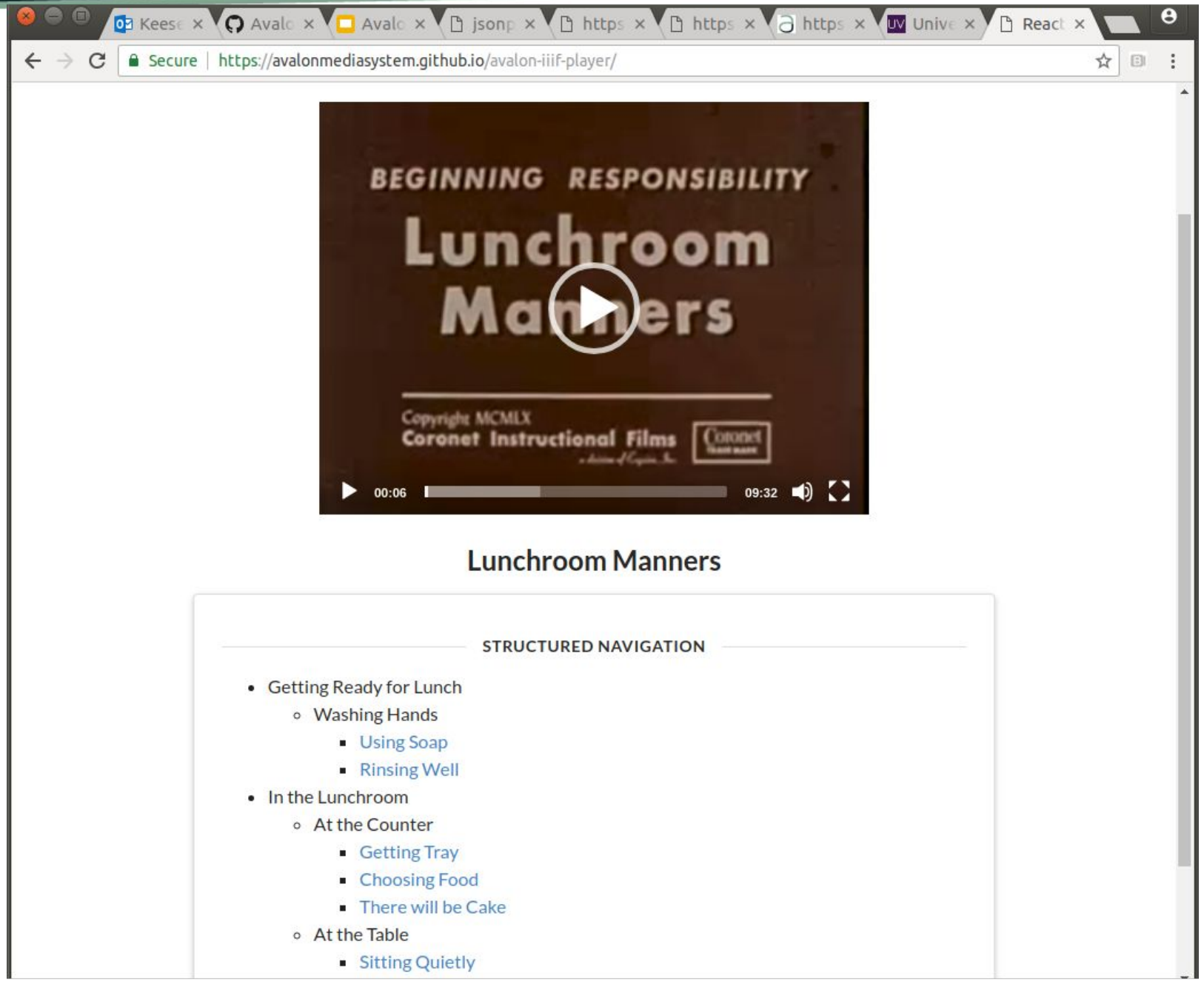
# Avalon 7.0 + IIIF A/V

- IIIF Presentation 3.0 manifest endpoints
- IIIF Presentation 3.0 media player
- Implement playlists using IIIF Presentation 3.0




avalon MEDIA SYSTEM

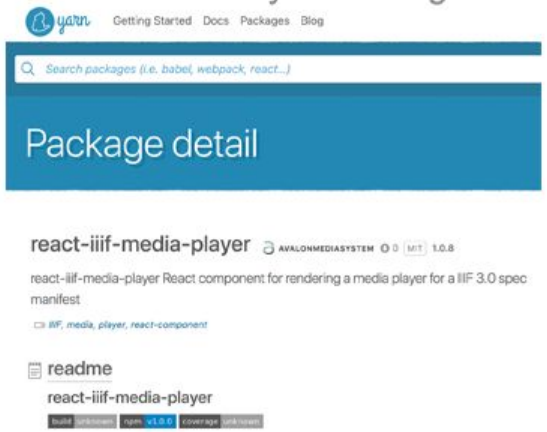

Not secure
universalviewer.io/examples/#?c=&m=&s=&cv=&manifest=https%3A%2F%2Fmallorn.dlib.indiana.edu%2Flunchr...
CONTENTS
Index
Thumbnails
Getting Ready for Lunch
Washing Hands
Using Soap
Rinsing Well
In the Lunchroom
At the Counter
Getting Tray
Choosing Food
There will be Cake
At the Table
Sitting Quietly
Eating Neatly
Leavning the Lunchroom
Cleaning Up
Putting Things Away
lunchroom_manners_1024kb.mp4 - Lunchroom Manners
BEGINNING RESPONSIBILITY
Lunchroom Manners
Copyright MCMLX
Coronet Instructional Films
MORE INFORMATION
00:08 / 09:33
Wunder der Vererbung
https://mallorn.dlib.indiana.e
Set
<
>



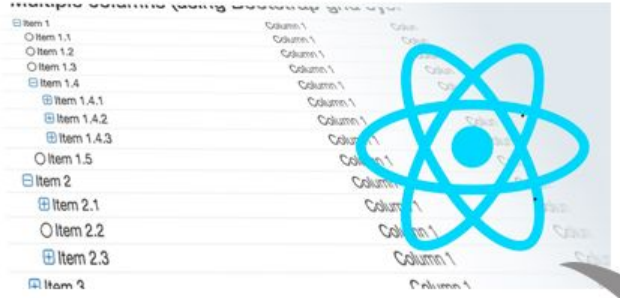

avalon
MEDIA SYSTEM

# Avalon IIIF Media Player

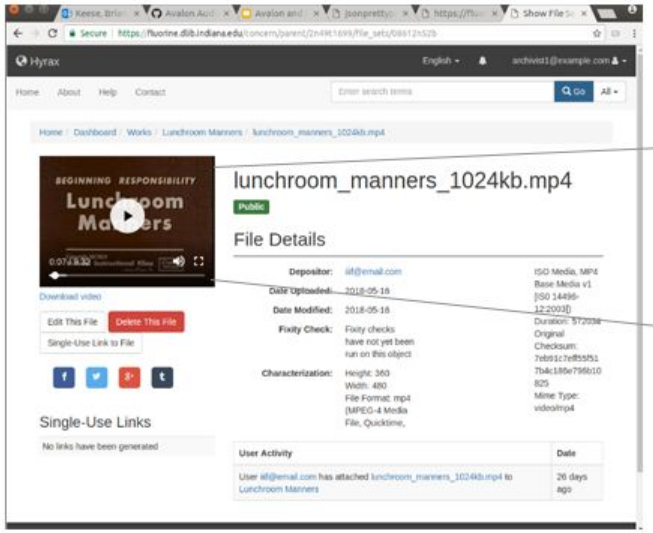
External Player Package



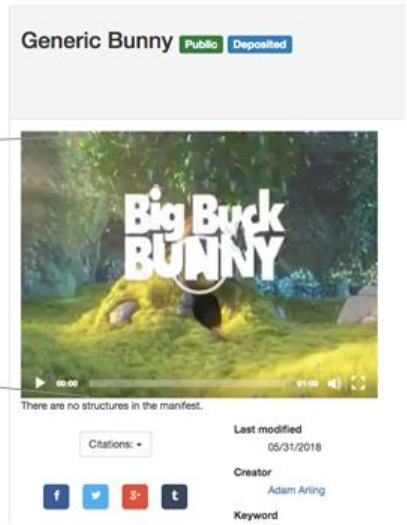

React JS Player Component



Avalon Bundle




MediaelementJS Player




avalon MEDIA SYSTEM

# Transcoding

# Opencast Matterhorn

Lecture capture software

Existing OSS with international community

Flexible workflow pipeline with Admin UI

**But...**

Differing use cases

Totally different stack (Java OSGI)

Heavyweight

Hard to install

Barrier to adoption

# Needs

Support multiple encoding backends

Lightweight

Creates multiple derivatives

Flexible for different institution’s needs

Encourage instead of hinder adoption

# Introducing ActiveEncode

(Modeled on ActiveJob)

# ActiveEncode Basic Usage

```
ActiveEncode::Base.engine_adapter = :ffmpeg

ActiveEncode::Base.create("file:///usr/local/bundle/gems/active_encode-0.4.1/spec/fixtures/fireworks.mp4",{
  outputs: [{ ffmpeg_opt:"-s 640x480", label:"mp4", extension:"mp4" }] })

=> #<ActiveEncode::Base:0x0000000007aee3e8 @input=#<ActiveEncode::Input:0x0000000007aecef8
  @url="file:///usr/local/bundle/gems/active_encode-0.4.1/spec/fixtures/fireworks.mp4", @width=960.0,
  @height=540.0, @frame_rate=29.671, @duration=6.024, @file_size=1629578,
  @audio_codec="mp4a-40-2", @video_codec="avc1", @audio_bitrate=69737, @video_bitrate=2092780,
  @created_at=2018-10-09 12:58:45 +0000, @updated_at=2018-10-09 12:58:45 +0000, @id=298>,
  @options={:outputs=>[{:ffmpeg_opt=>"-s 640x480", :label=>"mp4", :extension=>"mp4"}]},
  @id="8378768a-7b88-494a-a37e-a4f87aa9c54e", @created_at=2018-10-09 12:58:45 +0000,
  @updated_at=2018-10-09 12:58:45 +0000, @current_operations=[], @output=[], @state=:running,
  @percent_complete=1, @errors=[]>
```

# ActiveEncode::Base

**Class Methods**

create
find
default_options
engine_adapter
engine_adapter=

**Instance Methods**

new
create!
cancel!
reload
created?
cancelled?
completed?
running?
failed?

# ActiveEncode::Base

**Fields**

id
input
output
options
state
current_operations
percent_complete
errors

**Callbacks**

before/after/around create
before/after/around cancel
after_find
after_reload

# Custom encode job

```
class CustomEncode < ActiveEncode::Base
  after_create do
    logger.info "Created encode with id #{self.reload.id}"
  end

  def self.default_options(input)
    {preset: 'avalon-skip-transcoding'}
  end
end
```

# Encoding Backends

**Current**

Matterhorn (will be deprecated soon)
FFmpeg
Amazon Elastic Transcoder
Zencoder (prototype)

**Future**

???

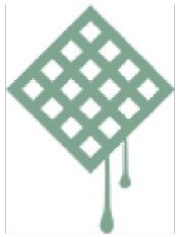
# Workshop Time

https://github.com/avalonmediasystem/connect2018-workshop/blob/master/README.md

avalonmediasystem.org

avalon-discuss-l@indiana.edu

github.com/avalonmediasystem



waffle.io/avalonmediasystem/

samvera.slack.com/messages/avalon

# So that's it

thanks for hanging with us

cjcolvar@indiana.edu

bkeese@indiana.edu

avalon MEDIA SYSTEM