UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Final Paper 2020

Title: EXHIBIT: An Online Tool for Digital Presentations

Author: Aa'isha Dout

Project Abbreviation: EXHIBIT

Supervisor(s): Dr Hussein Suleman

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	15
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	0
System Development and Implementation	*0*	*20*	15
Results, Findings and Conclusions	*10*	*20*	15
Aim Formulation and Background Work	*10*	*15*	15
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	
Total marks		**80**	

EXHIBIT: An Online Tool for Digital Presentations

Aa'isha Dout

dtxaai001@myuct.ac.za

University of Cape Town

ABSTRACT

Digital presentations are an ideal vehicle for showcasing digitised cultural heritage artefacts preserved in collections. This report discusses the development of EXHIBIT, an online tool created to integrate the upload of archival materials to such presentations. In addition, it facilitates direct manipulation of HTML and CSS to create templates for the exhibition, and includes a browse feature to view, download and edit templates and exhibits others have created. The tool was developed in iterations that focused on specific features, and was then subjected to various user and system tests. This report focuses on the template creator and browse components of the system, as well as the collaboration and commenting features.

KEYWORDS

digital libraries, digital exhibitions, virtual exhibitions, cultural heritage preservation, digital humanities

1 INTRODUCTION

Digital archives and collections are an integral part of cultural heritage preservation[36]. They preserve historical artefacts, often images, drawings and transcriptions, as multimedia objects with metadata. Metadata is additional information about the objects such as date produced and names of contributors, intended to aid with organising and contextualising items. An issue digital collections face is inaccessibility to those outside of the digitisation project's domain. Search features, while powerful and sufficient for those involved in the projects, often do not address cases of students, researchers and the general public wishing to explore collection contents – objects may be categorised by project-specific terms, for example[9, 15, 27].

Digital exhibitions, also called digital presentations, are an ideal medium to breach the ordinary user-expert divide, and have developed alongside digital collections and libraries since their inception[39]. Although definitions differ, the following components are cited as must-have features of a digital exhibition: a set of digital media objects (images, audio files, video etc.) and a logical combination for these objects that creates a narrative, such as a common subject, author, time period and so forth. User interaction (such as clicking/scrolling) is also common[20, 31]. Digital presentations allow both the creator and those viewing to explore and better understand digital collections' contents through object placement and examination in context.

1.1 Project context and aims

Many established tools are used to create digital presentations – in our research, we found PowerPoint, Prezi and Google Slides to be particularly popular, to name a few of the many tools available[38]. However, for digital presentations that contain archival material, most of these tools require the creator to either download and embed or copy and paste the multimedia objects into their presentations, potentially losing all the rich metadata present in their archival state and taking up space on the creator's computer.

EXHIBIT aims to address these issues by rolling the design aspects of digital presentation creation and integration of archival material into a single tool, with seamless archival upload. The Digital Bleek and Lloyd[2] and The Five Hundred Year Archive[1] are local, University of Cape Town affiliated examples of digital collections doing important preservation work, and subsets of their contents were used in the archival upload portion of this project.

Addressing difficulties in the creation of digital presentations in this context is important to further public appreciation of and interest in cultural heritage artefacts, a key objective of the field of cultural heritage preservation[35].

1.2 Solution outline

EXHIBIT is an online tool for the creation and showcasing of digital presentations containing cultural heritage artefacts. The tool is intended for professionals in GLAM sectors (galleries, libraries, archives and museums), referred to in this paper as the "expert" demographic/"(domain) expert users", who have knowledge of collection contents but not necessarily experience with markup and styling, as well as users who may have no domain knowledge (referred to as "ordinary" users). The system consists of three main components: the template creator, where users design the layout and style of their exhibition; the population stage, where users upload local and/or archival material into the templates they created; and the browse page, where users can view and edit existing templates and exhibits. In addition, the following features are available: collaboration, which allows users to edit templates and exhibits where the creator has defined them as a collaborator; commenting, which allows users to leave comments on templates and exhibits via the browse page; and downloading. Users may download exhibits and templates as HTML or PDF files.

This paper focuses on the components and features developed by the author, namely the template editor and browse components, and collaboration and commenting features.

1.3 Report structure

Section 2 provides an overview of the literature on digital exhibitions. Section 3 describes the design stage, including requirements gathering that informed prototype design, and Section 4 discusses implementation details. Software and usability testing is discussed in Section 5. Conclusions regarding the project are then made in

Section 6 and Section 7 concludes the paper with a discussion of future work.

2 BACKGROUND ON DIGITAL EXHIBITIONS

We began the project by researching existing digital presentation tools. The scope of this literature review was limited to tools with some connection to archival material, such as dedicated exhibition features of digital library tools and content management systems, or those not explicitly designed for such digital presentations but nevertheless commonly used for this purpose. We avoided well-known slide-oriented tools such as PowerPoint and Google Slides.

2.1 Integrated/dedicated exhibition services

Since the showcasing of contents is a important aspect of cultural heritage preservation, many toolkits for creating digital libraries contain software to create exhibitions out of the box.

2.1.1 Omeka. Omeka is a content management system (CMS) for digital collections, marketing itself to the cultural heritage field in particular[21]. Omeka comes in two versions: Omeka.org is the locally hosted, open source package and Omeka.net is a paid, account-based service that provides Web hosting and administrative support. In addition to user-friendly upload of artefacts and metadata logging, both versions provide tools to create digital exhibitions, referred to within Omeka as exhibition sites. These sites are based on default templates and themes, to minimise setup time, and therefore require some knowledge of HTML and CSS to modify if a user requires significant changes.

2.1.2 MOVIO (MOstre VIrtuali Online). Another CMS-based exhibition tool, MOVIO offers a path-oriented approach to digital exhibitions. The toolkit was developed as part of the AthenaPlus project to showcase the Europeana cultural heritage collection[20]. MOVIO's CMS enables the archive to import and catalogue resources in a manner adhering to the Dublin Core metadata model[29], while the Ontology Builder component allows creation of thematic paths. The tool defines digital objects as entities linked to other objects via relationships, allowing users to place these entities in a workspace and create logical paths connecting them[30]. The path may then be represented as is or located in relation to a timeline or a map. The Storyteller tool incorporates a "narration line", which may contain additional media objects, actual voice narration and user interaction elements. Each narration item has a permalink to be quoted or shared by an exhibition viewer. The MOVIO APP and MOVIO HUB provide access to the catalogue of exhibitions.

2.2 Re-purposed tools

Like popular slide-oriented tools, software developed for alternative purposes has long been used to create digital exhibitions [39]. While they occasionally lack metadata standards for multimedia objects, features like easy sharing, commenting and collaboration and familiar interfaces for these contribute to their popularity[8]. Website and blog-building tools are notable examples.

2.2.1 Wordpress. WordPress is a content management system (CMS) known primarily for its blog-publishing capabilities[23]. Like Omeka, WordPress comes in two versions: Wordpress.com – the hosted blogging service – and Wordpress.org – blogging software that can be downloaded and locally hosted.

While WordPress does not adhere to any GLAM metadata standards out of the box[23], the Scriblio plug-in mitigates this by allowing data to be structured according to the Dublin Core metadata standard for better searching and browsing, and allows for basic exhibition creation using hyperlinks[8].

2.3 Innovative tools and the future of digital exhibitions

2.3.1 Virtual exhibitions. While also occasionally used as a term for digital exhibitions as defined earlier, the Digital Exhibitions Working Group uses the term "virtual exhibitions" to refer strictly to digital exhibitions that seek to mimic their real life counterparts by incorporating 3D, virtual reality (VR) and augmented reality (AR) technology[20]. ARCO (Architecture for Digitization, Management and Presentation of Virtual Exhibitions)[43] and ViMEDEAS (Virtual Museum Exhibition Designer using an Enhanced ARCO Standard)[7] are two such examples. They are designed primarily for museums and galleries looking to digitise their physical collections. Their toolkits therefore include image-processing software for creating 3D models of artefacts, content management tools and visualisation tools for staging VR and AR exhibitions.

The move towards improving immersive quality of digital exhibitions[17] has resulted in even the more standard digital presentation software beginning to incorporate 3D, VR and AR; Omeka, for example, now offers the Omeka Everywhere suite for integration of touchscreen tables with in-museum digital presentations[22].

2.4 Influence on system design

Areas of interest emerged from investigating the above tools. The seamless integration of multimedia uploads provided by dedicated and integrated tools was one we aimed to emulate for the population stage of the system. For template editing, the direct manipulation aspects of tools such as MOVIO proved ideal to address cases of users not familiar with markup. Style customisation was also a recurring feature, which was identified as important to implement in the template editing phase, again with the option of easy-to-use manipulators (such as range selectors) that reflect changes visually. Since tools like MOVIO were feature-heavy for our purposes, we mainly drew inspiration from their intuitive user-interaction style. As stated in both MOVIO's[28]and Omeka's[37] documentation, the user interface and (UI) and user experience (UX) considerations were designed to engage users and prompt them to explore categories of artefacts they might not initially be interested in, effectively showcasing cultural heritage artefacts, in line with the aims of cultural heritage preservation research.

3 DESIGN

After background research, we commenced research with users from our target demographics to inform system design. After finalising architecture and general system design, we executed prototype design in an iterative process that produced prototypes of increasing fidelity to the final implementation. These prototypes

were tested or analysed with users/evaluators where possible to improve them and refine features.

3.1 Requirements gathering

This project was conducted entirely under the lockdown restrictions necessitated by the COVID-19 pandemic. As a result, all our user interaction was conducted online, via messaging tools, emails, Google surveys and video-conferencing platforms.

The requirements gathering Google survey was circulated amongst employees of the Iziko museum, people involved in the archival projects mentioned in Section 1.1 and UCT students. We received 5 responses. Demographics questions were asked to establish the population the participant fell into and revealed that 3 users from the expert category and 2 students answered our survey. Following that, to determine the usefulness of the tool we intended to create, participants were asked for their estimate of the number and frequency of presentations – strictly containing cultural heritage/archival material – they made. They were also questioned about tools (software or otherwise) that they used to create presentations. All answered that they used solely software tools and not stationery to create these presentations. We also asked about digital presentation tools they were familiar with, PowerPoint and Google Slides proving popular amongst the expert demographic in particular.

To assist with the design of the browse feature, we inquired about which websites/methods participants used to view others' digital presentations. Most answered that they only viewed presentations when colleagues presented them, but a few respondents expressed familiarity with presentation marketplaces such as Slideshare[3].

Our closing questions were on specific features they use in existing presentation tools, and those which they do not have access to but would like to see in a digital presentation tool. Features commonly used included styling of elements, especially colour and size, deleting, copy-paste, undo/redo and group ("lasso") select.

The responses informed the refinement of a minimum feature set, and the paper prototypes we had already begun work on were adjusted accordingly.

3.2 Non-functional requirements

Non-functional requirements were devised based on responses to the survey and according to the aims of the project. They included the following:

3.2.1 Security. Although user account and authentication system was not a focus of project, sensible constraints on editing of templates – e.g. only the creator and collaborators are allowed to edit – should be enforced.

3.2.2 Usability. The system's interfaces should be intuitive, easy to learn and allow users to identify and correct errors.

3.3 System architecture

EXHIBIT is a dynamic Web application, and has a typical Web application architecture comprised of the following layers (also called components): browser layer, application logic layer, data access layer and database layer. A high-level view of the system architecture is shown in Figure 1, with only the author's components represented in each layer.

3.3.1 Browser layer. The browser layer handles layout and display of HTML documents. Since all modern browsers, notably Chrome, Firefox, Safari and Edge, conform to the HTML5 standard[41], these are the browsers we decided to support.

3.3.2 Application logic layer. For the system as a whole, this layer concerned the logic of the template creation, population and browse components, as well as the commenting, collaboration and download features.

3.3.3 Data access layer. This layer exists to enable interaction with data stored on the server at a high level.

3.3.4 Database layer. This section was divided into three areas of interest. Most importantly, templates and exhibit objects are stored along with their metadata. The database of archival material and metadata also formed part of this layer. Finally, basic user information formed a user database to facilitate the collaboration feature.

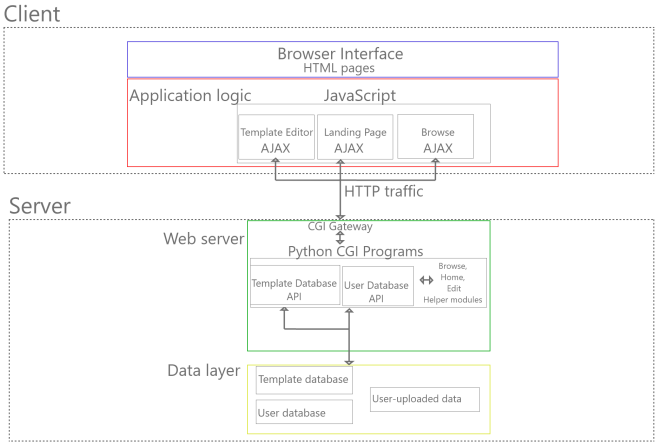

Figure 1: High level view of the system architecture

3.4 Design artefacts

Unified Modeling Language (UML) diagrams were devised based on the requirements gathered, in order to clarify and document system structure and behaviour between team members. Artefacts relevant to the author's work are included in Figure 2: the activity diagram models the basic behaviour of the template editing component and the use case diagram highlights the most important interactions with the browse page.

3.5 Prototyping

3.5.1 Paper prototyping. Initial prototyping was done using stationery (see Figure 3). Low fidelity prototypes were used to gain more accurate feedback on intended functionality and interface, rather than superficial aesthetics[33]. User interaction was simulated using movable parts, for example a viewing cutout for the

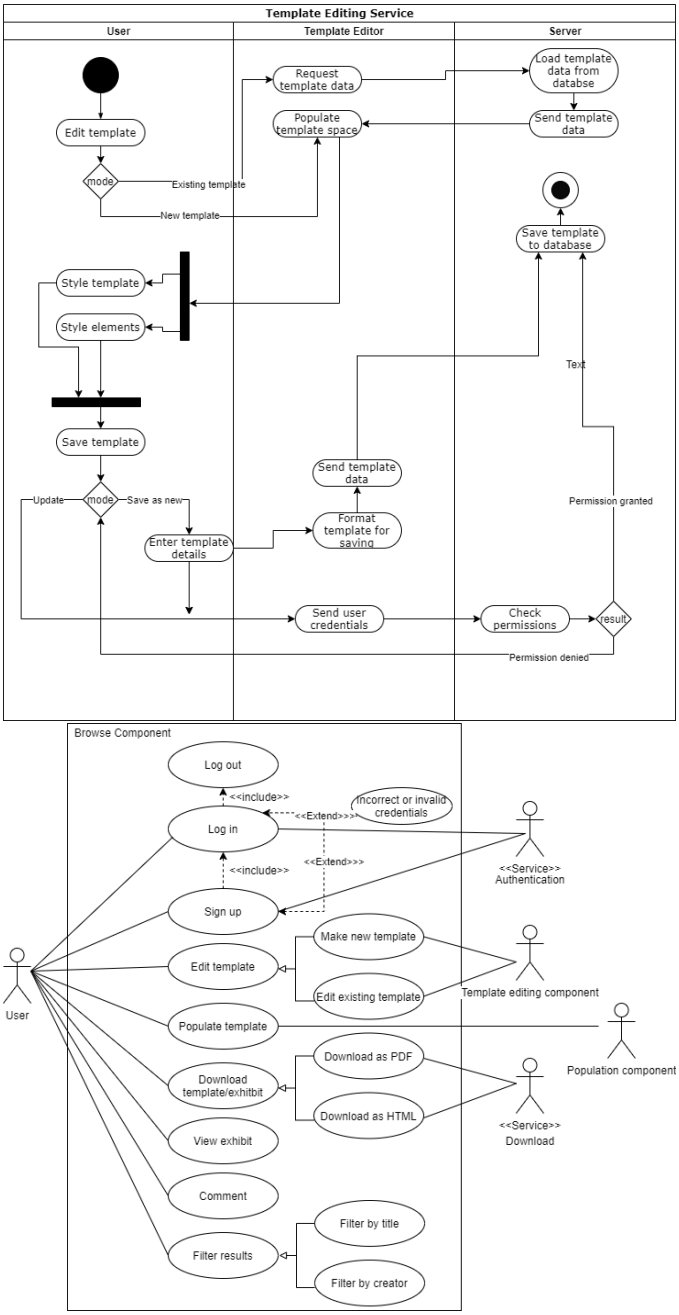

Figure 2: UML activity diagram (top) and use case diagram (bottom) for relevant system components

template in the carousel. The prototypes were evaluated by three computer science honours students. Since a traditional in-person "Wizard of Oz" study was not possible, the participants were shown a live feed of the interface and relayed actions they would have taken to the interviewer, who then carried out the actions. Verbal descriptions of system actions were provided if the video feed was not clear or there was a delay.

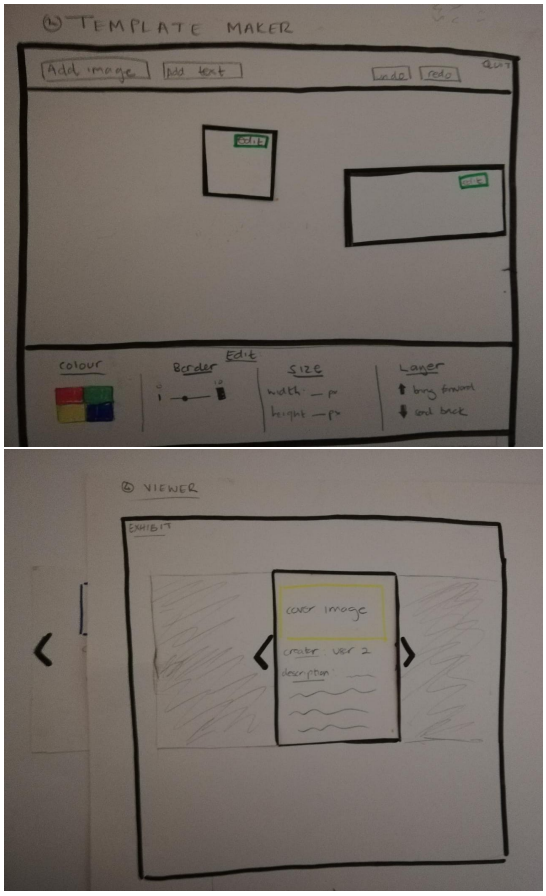

Figure 3: Screenshots of template editor (top) and browse page (bottom) prototype

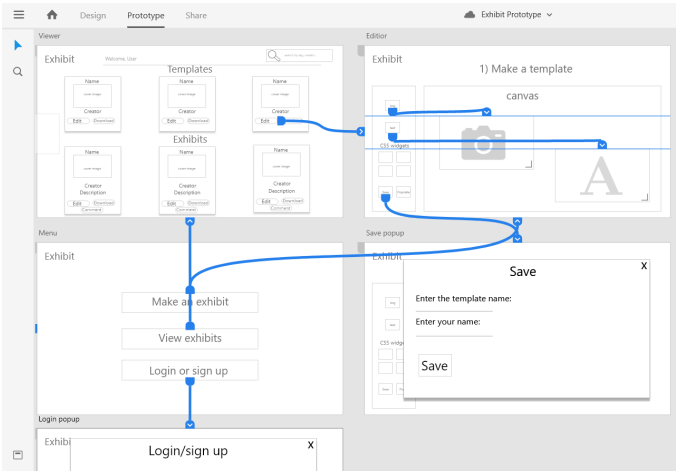

Figure 4: Screenshot of Adobe XD prototype and transitions

Notable feedback on the template editing prototype was lack of clarity about the resize functionality – participants declared styling actions they would take (e.g. "I use the edit menu to change the

background colour to blue"), and indicated that they would drag the elements to change their position, but none assumed that this direct manipulation extended to resize by drag. Users were also not clear on the distinction between image and textbox placeholders after they had been added. The default shapes upon adding were different (square for image vs rectangle for textbox), but this was not sufficiently clear – as one participant pointed out, this would be irrelevant if they styled the elements to be the same size.

The main feedback for the browse page was that participants expressed a desire for a filtering tool, and they considered browsing carousels to be cumbersome.

3.5.2 Digital prototyping. A digital prototype for the entire system was created using Adobe XD[4], which allows transitions to be incorporated as seen in Figure 4. The purpose of this prototype was to provide a blueprint for initial software implementation. Adjustments were made to the system functionality and interface based on feedback from the paper prototype evaluation. For the template editor, to clearly communicate that elements can be resized by dragging, handles were added to resizable elements. For the browse page, the display model was changed from the carousel to a card system, with each template and exhibit represented by a card. Filtering features were also added.

4 IMPLEMENTATION

4.1 Software development methodology

4.1.1 Agile. As in the prototyping stage of design, an iterative approach to development was adopted. In particular, the Agile development methodology[6] informed our work in this stage. We decided to forego a specific Agile framework such as Scrum, to avoid strict roles and procedures that were unsuitable for a small team and the short duration of the project. However, we adhered to the following core Agile principles:

- Prioritisation of individuals and interactions over processes and tools: Users were prioritised from the project's inception by modeling prototypes on requirements gathering and testing feedback. Weekly meetings were held with our supervisor to monitor progress and receive feedback.
- Prioritisation of working software over comprehensive documentation: After the design stage, documentation focused on code commenting and existing documentation adjustments (to flow diagrams, etc.) rather than the production of new documents. To ensure working code, early development included sanity checks through the use of Chrome Developer Tools and print statements, then, as functionality increased, unit tests and more comprehensive software tests.
- Responding to change: Throughout the process, we updated functionality, system models and documentation (such as flow diagrams) based on user evaluation. Concerning the project lifecycle, we adjusted estimates of development time per feature to maximise efficiency. It was also necessary to update our schedules around deliverable due date changes, which were frequent due to the pandemic.
- Iterative and incremental development: Development was split into three iterations, with features added at each iteration. Section 4.1.2 provides further detail.

4.1.2 Feature-driven development. After a minimum feature list was consolidated, a subset was allocated to each group member. The author was assigned the template editing and browse components, as well as commenting and collaboration features. As mentioned, development on these features occurred incrementally.

The first stage of development produced a basic template editor with add features and minimal styling, and a save feature that stored the template by a title and creator entered into a dialog box. The browse page was hardcoded with dummy exhibits to work purely on the user interface and title filtering.

The second stage introduced a control panel, and more template and element styling and Commenting functionality was added to the browse page. The bulk of integration work with the team member in charge of exhibit population occurred in this stage as well.

The last stage focused on collaboration features and drew on feedback after user testing to improve the user interface of both the template editor and browse page.

4.2 Technology, programming languages and environment

HTML and CSS were used for layouts and styling of each component of the system. The Bootstrap CSS framework[40] was used for consistent styling. Client-side scripting uses JavaScript and JQuery, a JavaScript library designed to simplify event-handling and communication with the server via AJAX (Asynchronous JavaScript And XML) calls[24]. JavaScript was chosen as it is an established Web page scripting language that is supported by all target browsers[42].

AJAX calls are used to request and receive information from the server. Conforming to our data structure, the system mostly exchanges data in the JSON (JavaScript Object Notation) format.

Our Web server is an Apache HTTP Server[16], installed on an Amazon Web Services Elastic Cloud instance[5] – a virtual computer running the Ubuntu operating system. The Common Gateway Interface (CGI) protocol[19] was chosen to execute server-side scripts that generate the application's pages. CGI was chosen since it is an established, simple protocol – being a short project, this minimised setup and learning time which would have been required for Web frameworks such as Django and Flask. Server-side scripts were written in Python, chosen mainly for its familiarity to group members. Python also has built-in JSON and CGI modules.

Our data layer consisted of JSON files, bypassing time and server space issues of setting up a traditional database. This structure also suited Agile development as as no strict data model was required upfront. The template and exhibit JSON files have a flattened structure to separate their contents (for example, element styling information for a template) and metadata needed for the browser cards.

For version control, both project developers used Git locally and Github to integrate their components.

4.3 Selected system component and feature details

The subset of the system components and features developed as an outcome of the author's project are detailed below.

4.3.1 Landing page and user authentication. This page outlines the main features of the system and provides the basic user authentication features used by all components. The login/sign up button launches a modal dialog box for credential entry, as seen in Figure 5. Minimal user information is needed: only a unique, non-empty username and non-empty password is required to set up an account. A user account is not required to create a template or exhibit – those who choose not to interact with the system via an account have their templates and exhibits attributed to "Anonymous user" to prevent cases of duplicate usernames in the browse stage. User authentication occurs via an AJAX GET request to a Python authentication class on the server, which checks the user credentials and sends back an appropriate response based on the outcome of the attempted verification (un/successful register or in/correct credentials). This response is displayed to the user and the modal dialog box is closed. Once successfully authenticated (valid login/sign up case), an HTTP cookie is used to store the username. The cookie is cleared upon logout or new sign in.

4.3.2 Template Editor and collaboration. The template editor (Figure 6) can be accessed in two modes: a "new template" or "edit" mode. Calls to the Python edit module result in the same basic layout of control panel, edit menu and a blank workspace to be populated with image and textbox placeholders. If the context is determined to be the "new template" mode, a title placeholder is also added. In "edit" mode, the existing template, including last-saved styles for the template and individual elements, are loaded into the workspace after an AJAX call to the Python module that handles access to the JSON templates file.

The left hand side of the control panel contains template-wide style controls such as template background colour. It also has a system status/navigation bar that shows login status and allows the user to navigate to other system pages. The right hand side displays template control options such as saving the template, rendering the template – displaying template sans element background images, handles, etc. – and an option to move into the template population stage. It also includes a help button which brings up an instruction panel of possible actions, illustrated by GIFs.

The edit menu (Figure 7) contains controls for styling a selected textbox or image. Options include element dimensions, border colour and width and background colour (textbox elements only). Where applicable, both direct manipulation options and fields for exact values were provided, with submissions for the fields finalised by buttons or pressing the enter key.

Saving attributes the template to the logged in user, if one exists. Users may save templates by any name they wish, but logged in users are warned if they already have a template of that name. For new templates, users are presented only with the save option as seen in Figure 6. Once a template has been saved, it can be updated by any user with collaboration permissions defined by the original creator, except templates attributed to anonymous users, which may

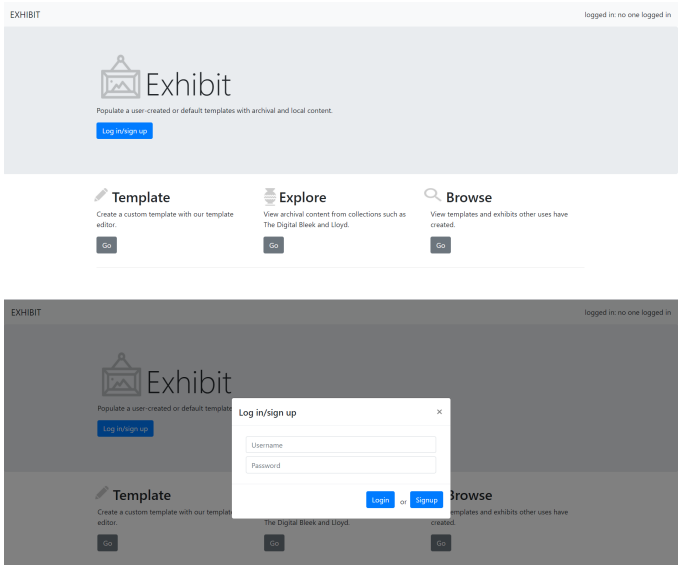

Figure 5: Home page (top) and user authentication dialog (bottom)

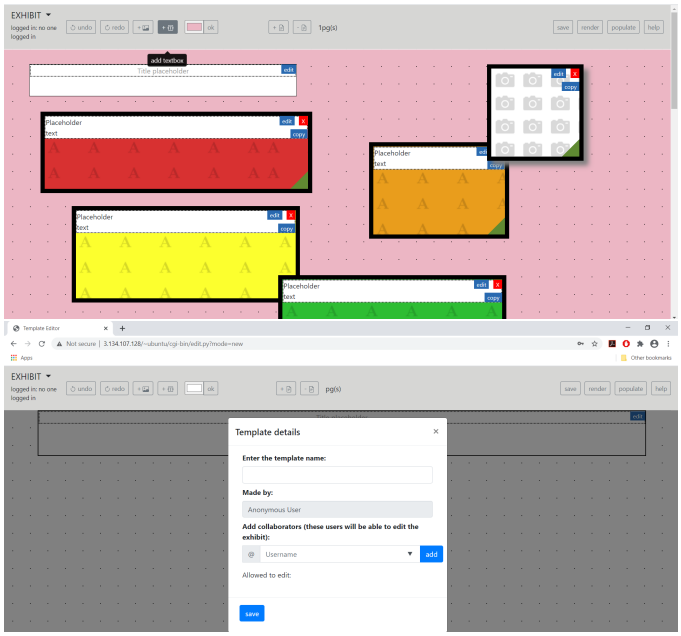

Figure 6: Screenshot of the template editor in "edit" mode (top) and the save dialog box (bottom)

be edited by anyone. An AJAX call to the data access Python module logs the template's contents and metadata (date, cover image, etc.) to the templates JSON file. Cover image metadata is the filepath of where the image is stored, including cache breakers to ensure that they are loaded from the server each time in the browser page – in prototype testing, users expressed frustration with not seeing their updates immediately reflected in cover images for caching reasons.

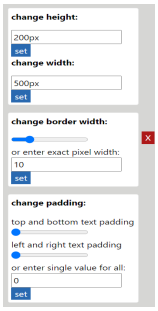

Figure 7: A subset of edit menu options

Collaborators are defined at the save stage – users are presented with a list of registered users to choose from.

4.3.3 Browse page and associated features. The browse page (Figure 8) uses cards to display templates and exhibits created by all users. Both exhibit and template cards present an overview of the contents –title, creator, last-updated date and an image. For exhibits, it is a cover image chosen by the user and for templates a screenshot taken upon saving, to suggest how it could be used without opening to edit. A user can populate any existing template at this stage via the populate button and open any existing template for editing (as mentioned, saving these edits then depends on permissions). Comments can be left by any system users (no login required) and can be viewed on the "back" of each card – a click on the comments button initiates a horizontal "flip" animation that displays comments left on that template or exhibit, as seen by the two "sides" of the card in Figure 8. This interaction model was used to best leverage limited space for each template/exhibit. The refresh button reloads the page to display any exhibits/templates added since the last load. This reload is achieved by an AJAX GET request to the database access module.

Filtering of templates and exhibits can be done via name tags, implemented via the Taggle.js[10] library, or title search, which uses simple JQuery filtering commands. The name tag field allows users to type directly or delete any tags, and clicking on card name links has the same effect as deleting every tag but that name.

5 TESTING

Post development stage 2, we began user acceptance testing and usability testing. As in the requirements gathering stage, all user interaction occurred via Google surveys and video-conferencing.

5.1 Software testing

5.1.1 Unit testing and ad-hoc testing. To ensure functionality of the system, ad-hoc testing and unit testing was conducted throughout the development cycle. The behaviour-driven testing framework Jasmine[25] was used for JavaScript and the built-in module unittest was used for Python code. For the template editor, unit tests were essential to checking the functionality of the style manipulating functions. Unit tests of the Python code involved checking the CRUD (create, read, update, delete) functionality for our JSON file databases.

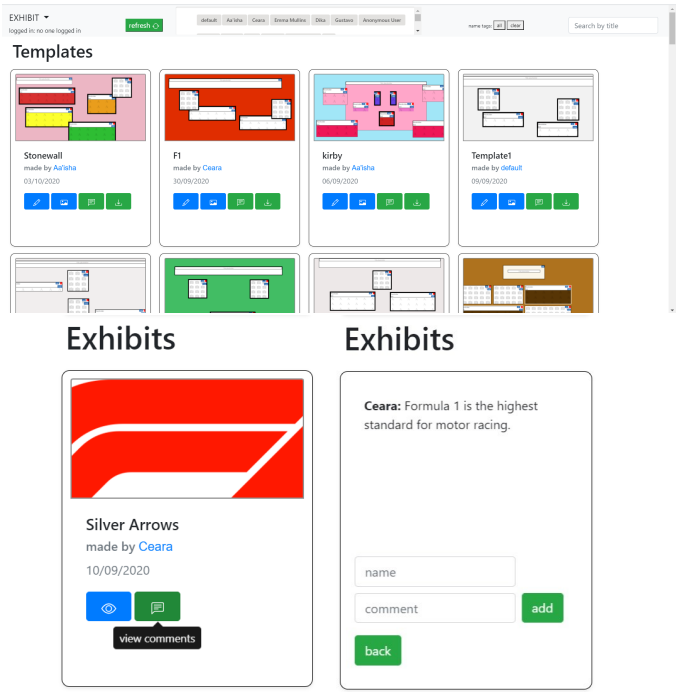

Figure 8: Screenshot of the browse page (top) and commenting functionality (bottom)

5.1.2 Browser compatibility testing. A key concern for Web applications is maintaining functionality across different browsers, as they may conform to different standards regarding markup and styling. Modern browsers all support HTML5[41], but even between the most popular browsers, there are differences in the way Cascading Style Sheets (CSS) – specifically shorthand and transformations – are handled. We used the free tier of online tool LambdaTest[26] to obtain user interface screenshots of the various EXHIBIT pages and simulate real time site access, for each of the target browsers. We did this for the latest two versions per browser. We also enquired about users' browser and version during usability testing to check this against any bugs they reported.

We discovered some bugs in the template editing stage pertaining to the style editing for Firefox: border widths and padding reverted to defaults once rendered or taken to the population stage. This was corrected by using full CSS attribute definitions, for example "border-width-top", "border-width-bottom", etc. as opposed to a single border value as some versions of Firefox do not support shorthand. Using LambdaTest with Firefox and Safari also allowed us to correct an issue with the card flip animation by using vendor specific extensions "-webkit-" and "-ms-" for the transform CSS properties.

5.1.3 System testing. Load and stress testing was not considered a priority for this system. Our system performance tests were therefore limited to latency and average load time. We used the online tool Dareboost[11] to measure performance in these categories. The website gave our load speed an above average rating and allowed us

to identify issues impacting load speed, such as limiting the length of inline scripts.

5.2 Discount usability testing

As there was a delay in obtaining permission to interview UCT staff, we used this time to conduct a heuristic evaluation on the software prototype we had developed.

5.2.1 User selection. The three students who evaluated our paper prototypes were again contacted.

5.2.2 Procedure. The evaluation was carried out via a Google form. Evaluators were provided with a link to the system homepage and a list of Nielsen's 10 Usability Heuristics[32]. Evaluators were asked to consider the components created by the team members separately. A field was provided for each heuristic and they were asked to note any violations they found, as well as classifying these problems with a severity rating according to Table 1. They were also provided with fields to give general comments.

5.2.3 Results. Feedback from evaluators concerning the author's components are listed by heuristic below.

- Visibility of system status: Two of the evaluators felt that the log in status was not effectively communicated by alerts only. One suggested that a "status message" should be present on all pages, showing the username of the person logged in. This problem was given a severity score of 2 by both evaluators. Progress for saving of templates was a more severe issue, with all three evaluators agreeing that either a progress bar or loading animation should be incorporated. This was marked as a high priority issue to fix – one evaluator commented that they were "frustrated" by the saving system.
- Match between system and the real world: No usability problems were identified under this category. One evaluator remarked that the card flip to show comments was a "decent" match for a real world card, albeit "a bit cute/gimmicky".
- User control and freedom: Evaluators commented positively on the ability to change element styles via multiple methods in the template editing stage. Two of the evaluators felt that the set shapes of image and textboxes was restrictive, and suggested adding additional shapes – this was given a severity rating of 1 and 2 by the evaluators in question. While not identifying it as a usability problem, the third evaluator suggested incorporating more styling options, such as rotation.
- Consistency and standards: One evaluator commented that the lack of symbols in the template editing stage was notable once they visited the browse page. They gave this problem a severity rating of 2. The other evaluators did not identify any usability problems in this category. One commented positively on the "clean" interface and its consistent style, which was achieved by using Bootstrap and adhering to Material Design[18] standards where possible.
- Error prevention: Two of the evaluators found no usability problems here. The third said they would have liked a confirmation dialog before saving or updating a template, as they mistakenly clicked "update" instead of "save new" and this overwrote their old template. The evaluator considered this a serious problem, with a rating of 3.
- Recognition rather than recall: No usability issues were reported in this category. One evaluator commented that the "hover labels" (Bootstrap tooltips) for the buttons were useful for anyone who did not understand the icons chosen, although the icons were "pretty standard". Another evaluator felt that the background images for the elements in the template editor were an effective way to remind the user of the element type they had added.
- Flexibility and efficiency of use: No usability problems were identified. Evaluators commented positively on the key bindings as an alternative to buttons for element styling, although one warned that some users may not appreciate the enter key corresponding to comment submission on the browse page, as it limits comments to single lines. However, they noted that this is "personal choice" and did not consider it a usability issue.
- Aesthetic and minimalist design: None of the evaluators reported usability problems in this category, however one commented that the unauthorised alert when attempting to edit templates was "a bit long".
- Help users recognize, diagnose and recover from errors: Evaluators commented that the undo/redo functionality for the template editor was useful. One evaluator encountered a server error and commented that the alert was not helpful as it merely communicated an error but gave no indication of what exactly went wrong or what to do next. They gave this issue a severity rating of 4.
- Help and documentation: All three evaluators remarked that documentation was lacking, particularly at the template editing stage, and suggested a help button or dedicated instructions page. Two of the evaluators gave this a severity rating of 3 and the other a 4.

Table 1: Usability problem severity ratings

Rating	Meaning
0	Not a usability problem for this system/in this context
1	A cosmetic problem, need only be fixed if there is time at the end of the project
2	A minor usability problem which should be given low priority
3	A major usability problem, high priority to fix
4	A critical usability issue which is imperative to fix this before the product can be released

5.2.4 Discussion. Although not as rigorous as a traditional usability evaluation, this testing exercise gave us the opportunity to fix critical issues before testing with real users. We implemented many of their suggestions, including help and instructions via dialogs with demonstration GIFs, progress animations for saving, and more descriptive error messages.

A notable limitation to the results of this evaluation is that our familiarity with the evaluators may have created bias, leading to more favourable feedback.

5.3 Acceptance testing

5.3.1 Procedure. Developers conducted acceptance testing to check the system's compliance with functional requirements prior to contacting expert users. A functionality checklist functioned as the basis for a cognitive walkthrough of of the system.

5.3.2 Results. Results of the evaluation pertaining to the author's components of the system are summarised in Table 2.

Table 2: Acceptance Testing results

Functionality	Result
Landing page	
Registered user is able to sign in	PASS
New user is able to register	PASS
Template editor	
Adding image	PASS
Adding textbox	PASS
Styling an element (all options tested)	PASS
Changing template background colour	PASS
Changing number of pages	PASS
Saving a template (new template mode)	PASS
Saving a template (as new, from an existing template)	PASS
Only a user with collaboration permissions may update a template	PASS
Browse page	
Filter by adding or removing name tags in input field	PASS
Filter by title search	PASS
Filter by clicking name link	PASS
View comments	PASS
Leave a comment	PASS

5.4 Real-user usability testing

5.4.1 User selection. We contacted 12 participants, from two user groups: university students not affiliated with any GLAM profession and cultural heritage professionals. 6 responses from university students were recorded. Of the 6 in the expert group, there were 2 researchers, a digital content manager, a curator, a historiographer and an administrator for a digital library project. The mix of domain expert and "ordinary" users was intended to reflect target user demographics. A subset of 5 respondents also agreed to video-conferencing interviews (see section 5.4.2), a group of 3 expert users and 2 students.

5.4.2 Procedure. The usability survey was conducted via Google Forms. Potential participants were first presented with a consent form which they were required to "sign" (enter their name in a field) before proceeding. Participants were then given a set of tasks to complete, where each task represented a typical use case devised in the design stage. Tasks were divided into sections for each component of the system. For the template editor, these tasks included adding at least one image and textbox placeholder, styling the elements and saving the template. The browse page tasks were instructions to filter templates and exhibits on arbitrary characteristics to encourage users to explore the various filtering mechanisms.

After completing the tasks, the evaluation section of the survey asked participants to answer a System Usability Scale questionnaire. We decided on the alternating scales format of the questionnaire, meaning odd-numbered questions were scored with 1 corresponding to "strongly disagree" and 5 corresponding to "strongly agree" and the even questions following the opposite scale. This is intended to limit acquiescence bias, where respondents tend to select a positive response option disproportionately more frequently[14].

The final part of the survey asked the participant to enter their email address if they were willing to be contacted for a follow-up video-conferencing interview about the usability of the system. Excluding the informed consent procedure, the interviews followed a similar structure to the survey. While one team member acted as the instructor reading the tasks and questions, the other recorded the user's comments. The usability questionnaire portion of the interview asked the same questions, on a strictly positive scale this time, with the instructor encouraging them to elaborate on their score for each question. These sessions served mainly to gain more qualitative feedback, such as asking users to elaborate on specific adjectives they used to describe components of the system. The sessions were not recorded.

5.4.3 Results. Results of the Google survey are shown in Figure 9. Each question's average score is shown, with standard deviation represented by black lines. The questions are listed below for reference. The full questionnaires and anonymised raw scores can be found on the project website.

SUS QUESTIONS:

(1) I think that I would like to use this system frequently.

(2) I found the system unnecessarily complex.

(3) I thought the system was easy to use.

(4) I think I would need the support of a technical person to be able to use this system.

(5) I found the various functions in this system were well-integrated.

(6) I thought there was too much inconsistency in this system.

(7) I would imagine that most people would learn to use this system very quickly.

(8) I found the system very cumbersome to use.

(9) I felt very confident using the system.

(10) I needed to learn a lot of things before I could get going with this system.

5.4.4 Discussion. The system received a slightly above average score of 3.25 for usefulness (question 1), with most students responding neutrally (3) and most expert users responding positively with scores of 4 and 5. Questions related to ease of use (questions 2, 3, 4, and 8) were more positive, with all receiving average scores above 3.4, indicating that users agreed with positive statements

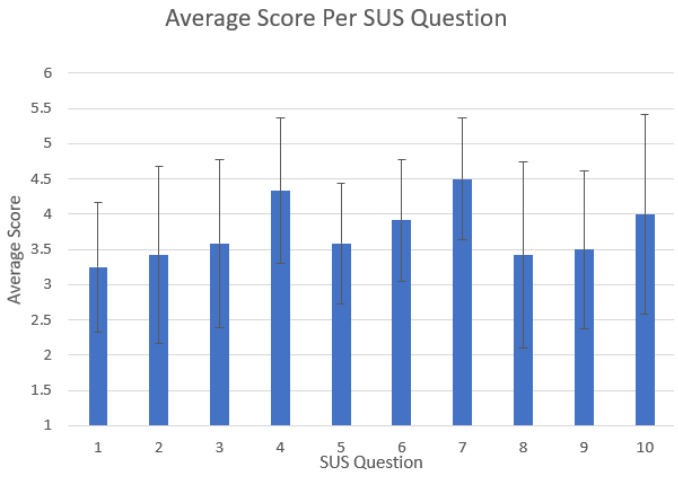

Figure 9: Bar graph showing System Usability Scale questionnaire results

regarding the system's ease of use and disagreed with negative statements. In particular, the average score of 4.3 for question 4 indicated that most users felt the system was not biased towards people with technical experience. The system received high average scores for question 7 and 10 (4.5 and 4, respectively), indicating that the system had a shallow learning curve. Finally, the system received above average scores for questions 5, 6 and 9, relating to satisfaction with the system.

User interviews largely echoed these results. All interviewees elaborated on feedback we singled out for clarification, and some also motivated their SUS scores, such as detailing the parts of the system they felt was "cumbersome" for question 8. Expert users provided clarity on their answers regarding usefulness of the system by describing how it would be used in their daily interaction with archival multimedia objects. Some interviewees also gave feedback they had not mentioned in the survey. For example, one interviewee, during a discussion of the download options, mentioned that they would consider an image download option useful.

Limitations that must be noted for this testing stage pertain to the survey procedure. Firstly, our choice of the SUS questionnaire has well-known drawbacks, including the wording of questions being unsuitable for non-English home language speakers[13]. Additionally, our choice of alternating scales may have been confusing and resulted in misleading results – we attempted to mitigate this by including instructions that informed users of scales, in bold text, but we cannot be sure all users noted this. Video conferencing results suggest that users interpreted it correctly, but only a subset of Google survey respondents were interviewed so this is not guaranteed.

We also note the limitations of our interview procedure: rigorous usability tests are usually conducted in conditions that limit distractions and outside influence, but there was no way to guarantee this via video-conferencing. Although the video-conferencing provided some non-verbal feedback to the system, we also experienced some delays and spells of poor quality due to variable Internet connection strength.

6 CONCLUSIONS

The project aimed to create an online tool for creating digital exhibitions that allowed customisation of styles, integrated archival uploads and a space to view other system users' creations. This paper outlines the development of the template creation and browse components, as well as commenting and collaboration functionality, of a Web application intended to meet these requirements. System testing verified that functional requirements were met. Non-functional requirements such as accessibility to non-technical users were assessed by usability and heuristic evaluations. 15 participants were involved in these evaluations, with representatives from both expert and non-expert groups. Results suggest that the system satisfied non-functional requirements, with generally positive feedback regarding learnability and accessibility in particular.

However, the unique circumstances of the pandemic must be taken into account when considering the results of user testing, as usual conditions to ensure integrity of the evaluations could not be met.

7 FUTURE WORK

7.1 Template editor

As mentioned by users in the testing stage, more editing options can be included, such as different shapes for image and text placeholders, custom textures and border images.

An alternative design of the template editing stage, for example relegating the control panel to a hamburger menu only available on demand, would result in a better mobile experience that maximises the space for template elements.

7.2 Browse page

Support for different languages or translate features would be helpful to assist non-English language speakers, and could be achieved via integration of open source translation software such as OmegaT[34].

7.3 General

The account system could be more robust, perhaps integrating external user authentication such as the Google Identity Toolkit[12] instead of custom accounts. An additional "My account" page would then offer a logged in user view of the browse page, with only their templates and exhibits, and viewing statistics could also be shown here.

8 ACKNOWLEDGEMENTS

The author thanks their supervisor, Dr Hussein Suleman, second reader, Dr Hafeni Mthoko, and project partner, Ceara Mullins for their feedback and assistance throughout this project. The contributions of students and archival professionals who assessed our system is also greatly appreciated.

REFERENCES

[1] [n.d.]. *The Five Hundred Year Archive.* Retrieved May 16, 2020 from http://www.apc.uct.ac.za/apc/research/projects/five-hundred-year-archive

[2] [n.d.]. *THE DIGITAL BLEEK AND LLOYD.* Retrieved May 16, 2020 from http://lloydbleekcollection.cs.uct.ac.za/

[3] 2020. *About slideshare.* Retrieved May 16, 2020 from https://www.slideshare.net/about

[4] Adobe. 2020. *Adobe XD Features.* Retrieved September 20, 2020 from https://www.adobe.com/products/xd/features.html

[5] Inc. Amazon Web Services. 2020. *What is Amazon EC2?* Retrieved September 28, 2020 from https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

[6] Kent Beck, Mike Beedle, Arie Van Bennekum, and Alistair Cockburn. 2001. Manifesto for agile software development. (2001).

[7] Daniel Biella, Wolfram Luther, and Nelson Baloian. 2010. Beyond the ARCO standard. In *Proceedings of the 16th International Conference on Virtual Systems and Multimedia.* IEEE, 184–191. https://ieeexplore.ieee.org/document/5665985

[8] Kelli Bogan. 2011. Creating digital archives with WordPress. *Library Technology Reports* 47, 3 (April 2011), 47–55. https://link-gale-com.ezproxy.uct.ac.za/apps/doc/A255559918/AONE?u=unict&sid=AONE&xid=fe1b8261

[9] Gobinda Chowdhury. 2009. From digital libraries to digital preservation research: the importance of users and context. *Journal of Documentation* 66, 2 (July 2009), 207–223. https://tefkos.comminfo.rutgers.edu/Courses/Zadar/Readings/Chowdhury%20dig%20pres%20J%20Doc%202009.pdf

[10] Sean Coker. 2013. *Taggle.js.* Retrieved September 28, 2020 from https://sean.is/poppin/tags

[11] Dareboost. 2020. *Website Speed Test and Website Analysis.* Retrieved September 10, 2020 from https://www.dareboost.com/en

[12] Google Developers. 2020. *Introduction to the Toolkit.* Retrieved May 16, 2020 from https://developers.google.com/identity/toolkit

[13] Kraig Finstad. 2006. The System Usability Scale and Non-Native English Speakers. *Journal of Usability Studies* 1 (August 2006), 185–188. https://uxpajournal.org/wp-content/uploads/sites/8/pdf/JUS_Finstad_Aug2006.pdf

[14] Kraig Finstad. 2020. Is It Time to Go Positive? Assessing the Positively Worded System Usability Scale (SUS). *Human Factors: The Journal of the Human Factors and Ergonomics Society* (January 2020). https://doi.org/10.1177%2F0018720819881556

[15] Schubert Foo, Yin-Leng Theng, Dion Hoe-Lian Goh, and Jin-Cheon Na. 2009. From Digital Archives to Virtual Exhibitions. In *Handbook of Research on Digital Libraries: Design, Development, and Impact.* IGI Global, Hershey, PA, USA, Chapter 9.

[16] The Apache Software Foundation. 2020. *What is the Apache HTTP Server Project?* Retrieved September 28, 2020 from https://httpd.apache.org/ABOUT_APACHE.html

[17] Efstratios Geronikolakis, Paul Zikas, Steve Kateros, Nick Lydatakis, Stelios Georgiou, Mike Kentros, and George Papagiannakis. 2020. A True AR Authoring Tool for Interactive Virtual Museums. In *Visual Computing for Cultural Heritage.* Springer, New York, NY, USA, 225–242. https://arxiv.org/pdf/1909.09429

[18] LLC Google. 2020. *Material Design Guidelines.* Retrieved September 28, 2020 from https://material.io/design/guidelines-overview

[19] Shishir Gundavaram. 2002. *CGI Programming on the World Wide Web.* O'Reilly Open Books Project.

[20] Monika Hagedorn-Saupe, Werner Schweibenz, Giuliana De Francesco, and Maria Teresa Natale. 2015. Digital Exhibitions, a Powerful Tool for Cultural Institutions Audience Development. In *Proceedings of the Digital Heritage International Congress*, Vol. 2. IEEE, 203–204. https://ieeexplore-ieee-org.ezproxy.uct.ac.za/document/7419493/

[21] Juliet Hardesty. 2014. Exhibiting library collections online: Omeka in context. *New Library World* 115, 3 (March 2014), 75–86. http://dx.doi.org/10.1108/NLW-01-2014-0013

[22] Ideum. 2017. *Omeka Goes Everywhere.* Retrieved April 26, 2020 from https://ideum.com/portfolio/omeka-goes-everywhere

[23] Kyle Jones and Polly-Alida Farrington. 2012. *Learning from Libraries That Use WordPress: Content-Management System Best Practices and Case Studies.* American Library Association, Chicago, IL, USA.

[24] The jQuery Foundation. 2020. *What is jQuery?* Retrieved September 20, 2020 from https://jquery.com/

[25] Pivotal Labs. 2020. *Jasmine documentation.* Retrieved September 10, 2020 from https://jasmine.github.io/pages/docs_home.html

[26] LambdaTest. 2020. *About LamdaTest.* Retrieved September 10, 2020 from https://www.lambdatest.com/

[27] Clifford Lynch. 2002. Digital collections, digital libraries and the digitization of cultural heritage information. (March 2002). https://firstmonday.org/ojs/index.php/fm/article/download/949/870/6263

[28] Sam Habibi Minelli, Iva Meštrović, Petra Milovac, Orsolya Veress, Donatas Snarskis, Jovita Vilimaitiene, Linnéa Karlberg Lundin, Karin Glasemann, Piot Kożurno, and Marek Wieczorek. 2015. Museums' Experiences in Creating Cultural Narrations Using the AthenaPlus Tool Called MOVIO. *Uncommon Culture* 6, 11 (June 2015), 67–87. http://uncommonculture.org/ojs/index.php/UC/article/viewFile/6086/4635

[29] Sam Habibi Minelli, Maria Teresa Natale, Barbara Dierickx, Paolo Ongaro, Daniele Ugoletti, Rubino Saccoccio, and Marc Aguilar Santiago. 2014. MOVIO: A semantic content management and valorisation approach for archives and cultural institutions. In *Arxius i Indústries Culturals. Girona, de l'11 al 15 d'octubre de 2014. 2a Conferencia Anual d'Arxius· 9a Conferencia Europea d'Arxius.* International Council on Archives. https://www.girona.cat/web/ica2014/ponents/textos/id234.pdf

[30] Sam Habibi Minelli, Maria Teresa Natale, Paolo Ongaro, Marzia Piccininno, Rubino Saccoccio, and Daniele Ugoletti. 2014. MOVIO: A toolkit for creating curated digital exhibitions. *Procedia Computer Science* 38 (January 2014), 28–33. https://www.sciencedirect.com/science/article/pii/S1877050914013660/pdf?md5=ef95993470fa0f6eb980ae581c5d5bce&pid=1-s2.0-S1877050914013660-main.pdf&_valck=1

[31] Maria Teresa Natale. 2012. *Handbook on Virtual Exhibitions and Virtual Performances: Version 1.0.* INDICATE Project. https://www.digitalmeetsculture.net/wp-content/uploads/2013/01/Handbook_on_Virtual_Exhibitions_and_Virtual_Performaces.pdf

[32] Jakob Nielsen. 1994. *10 Usability Heuristics for User Interface Design.* Retrieved April 28, 2020 from https://www.nngroup.com/articles/ten-usability-heuristics/

[33] Jakob Nielsen. 2003. *Paper Prototyping: Getting User Data Before You Code.* Retrieved June 24, 2020 from https://www.nngroup.com/articles/paper-prototyping/

[34] omegat.org. 2019. *Introduction to the Toolkit.* Retrieved May 16, 2020 from https://omegat.org/

[35] ICOMOS International Committee on Cultural Tourism. 2002. *ICOMOS international cultural tourism charter: principles and guidelines for managing tourism at places of cultural and heritage significance.* International Council on Monuments and Sites.

[36] Daniela Petrelli, Luigina Ciolfi, Dick Van Dijk, Eva Hornecker, Elena Not, and Albrecht Schmidt. 2013. Integrating material and digital: a new way for cultural heritage. *Interactions* 20, 4 (August 2013), 58–63. https://dl.acm.org/doi/fullHtml/10.1145/2486227.2486239

[37] Linda Rath. [n.d.]. Omeka.net as a librarian-led digital humanities meeting place. ([n. d.]).

[38] Christine Schein. 2018. *Digital Presentation Tools.* Retrieved April 26, 2020 from https://www.coloradovirtuallibrary.org/technology/digital-presentation-tools/

[39] David Silver. 1997. Review: Interfacing American Culture: The Perils and Potentials of Virtual Exhibitions. *Bibliothek Forschung und Praxis* 49, 4 (1997), 825–850. https://muse.jhu.edu/article/175548

[40] Bootstrap Team. 2020. *About Bootstrap.* Retrieved September 20, 2020 from https://getbootstrap.com/docs/4.5/getting-started/introduction/

[41] W3Schools. 2018. *HTML5 Browser Support.* Retrieved May 16, 2020 from http://w3schools-fa.ir/html/html5_browsers.html

[42] W3Schools. 2020. *JavaScript Versions.* Retrieved September 20, 2020 from https://www.w3schools.com/js/js_versions.asp

[43] Martin White, Nicholaos Mourkoussis, Joe Darcy, Panos Petridis, Fotis Liarokapis, Paul Lister, Krzysztof Walczak, Rafal Wojciechowski, Wojciech Cellary, Jacek Chmielewski, et al. 2004. ARCO-an architecture for digitization, management and presentation of virtual exhibitions. In *Proceedings Computer Graphics International, 2004.* IEEE, 622–625. https://researchportal.bath.ac.uk/files/11203331/ARCO_original_CGI2004.pdf