



CDLI



DESIGN CHALLENGE

Aradhana Kund

Mentors : Samarth Sharma and Amaan Iqbal

Google Summer of Code 2021

Contents

1. [Curriculum Vitae of the Student](#)
2. [Overview of the Project](#)
3. [Goals and Deliverables](#)
 1. [Planning and Implementation of Admin Dashboard.](#)
 2. [Merging Admin and Public Templates.](#)
 3. [Menu Discrepancy Problem.](#)
 4. [Unifying presentation of all pages.](#)
 5. [View/Edit/Index Pages of all entities should be similar.](#)
 6. [Enhance the display of a single artifact and expanded search results.](#)
 7. [Design edit artifact pages.](#)
 8. [Improve Footer styling](#)
 9. [Enhance the styling of the browse page of users.](#)
 10. [Build a Resource Page](#)
 11. [Restyle Default Template of many entities](#)
4. [Tech Stack Used](#)
5. [Workflow Chart](#)
6. [Timeline](#)
7. [Why I am the right person?](#)
8. [Other Commitments](#)

1. Curriculum Vitae of the Student

Name: Aradhana Kund

Gitlab Handle: [aradhana_kund](#)

Github Handle: [aradhana72](#)

Country of Residence: India

Timezone: IST(India)

Email: aradhanakund@gmail.com

2. Overview of the Project

Cuneiform Digital Library aims at collecting, preserving, and making available the images, text, and metadata of all artifacts inscribed with Cuneiform Script. It has a vast community of users which include scholars, students as well as informal learners.

To converge the vast user base of CDLI, it is important to **re-unify the design** so as to facilitate more interaction between the user and the website. Interactive, clean, and integrated User Interface is very important to increase the **accessibility** of the website, decrease fallbacks and improve user experience. It is also very important to unify the entire design of the website so as to have a **consistent interface**.

This project aims to unify the complete styling of the website using minimal space and complexity. Also, it aims to improve the designing of certain parts of the website to improve the User Interface. It is made sure to deliver a website at the end of this project which will not only provide a better user experience but also help in simplifying the codebase of CDLI.

3. Goals and Deliverables

3.1 Planning and implementation of the Admin Dashboard

- ❑ **Plan and implement the admin dashboard**
 - The wireframe is available [here](#). The way and detailing of implementation have been given [below](#).
- ❑ **Not all logged-in users have access to the admin dashboard**
- ❑ **Not all with access to the dashboard can see and use all admin functionalities.**

IMPLEMENTATION IN DETAIL

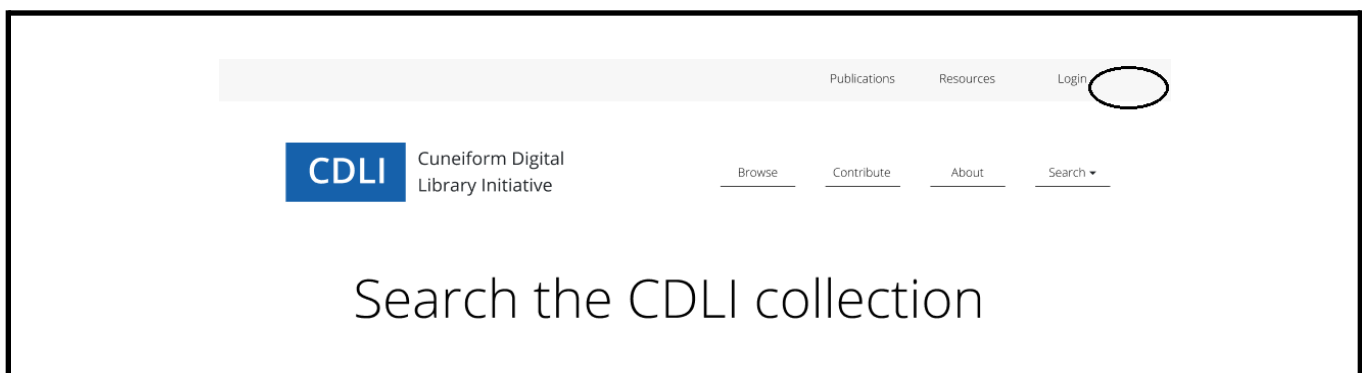
FRONTEND:

- The admin dashboard doesn't exactly resemble other pages(in terms of the header as well as body because admin and public pages use different templates) and has one column of all the entries in alphabetical order(which unnecessarily uses up space and isn't optimal). Refer to the screenshot below:

ADMIN HEADER:



PUBLIC PAGES HEADER:



ADMIN DASHBOARD BODY/MENU:

Dashboard

- Abbreviations
- Agade Mails
- Archives
- Archival Tool
- Articles
- Artifacts
- Artifacts Publications links
- Artifact Types
- Authors
- Authors Publications links
- CDLI tablet
- CdlINotes
- Collections
- Dates
- Dynasties
- Editors Publications links
- ExternalResources
- Genres
- Languages
- MaterialAspects
- MaterialColors
- Materials
- Periods
- Postings
- Proveniences
- Publications

UPDATED PUBLIC PAGE BODY/MENU([link](#)):

Featured

Proveniences

Cuneiform tablets have been found across the wider Middle East, the vast majority are from Iraq, but substantial amounts of material come from neighboring countries.

Periods

Written for more than 3,000 years cuneiform writing covers a wider range of time periods.

Genres

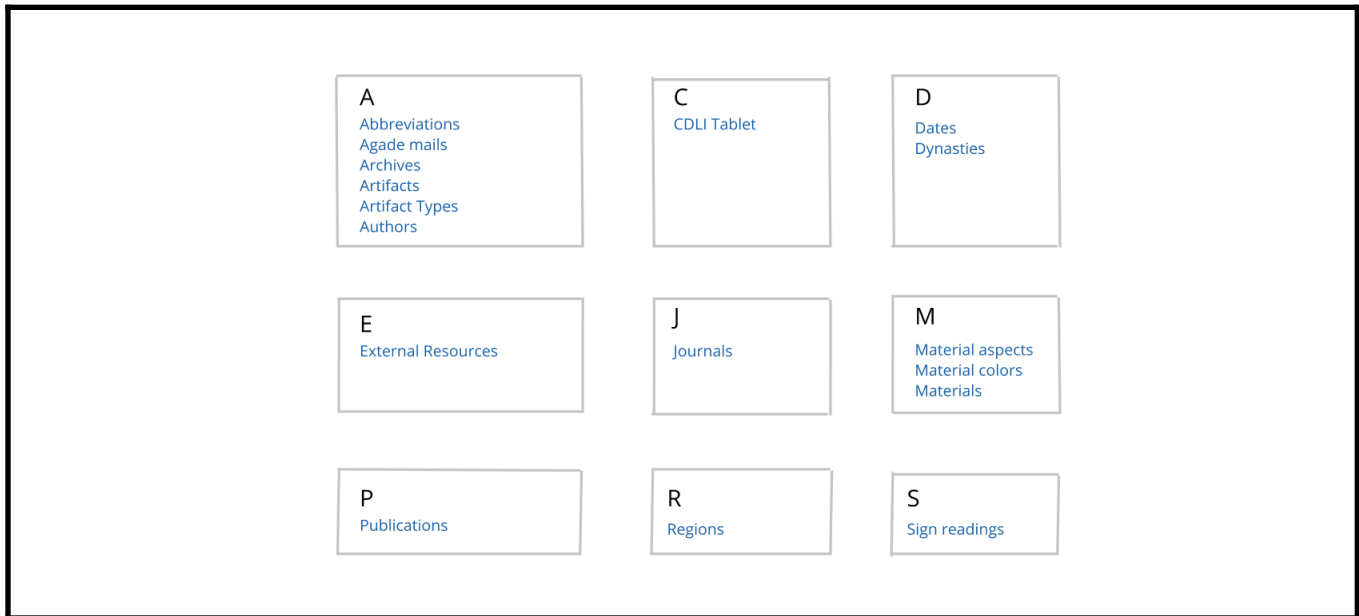
Used to write everything from accounting documents, to epic poetry and scientific texts, the preserved cuneiform texts over many genres.

Languages

Cuneiform was used to write over 10 different languages and many other dialects.

Collections

Cuneiform tablets can be found in modern collections around the world.



- The admin dashboard **frontend** can be achieved by following code snippets, then borders can be added to each alphabet code according to convenience.

framework-phoenix-develop\app\cake\src\Template\Admin\Home\dashboard.ctp

```

1 <?php
2 /**
3  * @var \App\View\AppView $this
4  * @var \App\Model\Entity\Artifact $artifact
5  */
6 $additional=["Abbreviations" => "abbreviations", "Agade mails" => "agade_mails", "Archives" => "archives", "Archival Tool" => "uploads",
7 "Articles" => "add", "Artifacts" => "artifacts", "Artifacts Publications links" => "artifacts_publications", "Artifact types" => "artifactTypes",
8 "Authors" => "authors", "Authors Publications links" => "authors_publications", "CDLI tablet" => "CdliTablet",
9 "CdlNotes" => "cdlNotes", "Collections" => "collections", "Dates" => "dates", "Dynasties" => "dynasties",
10 "Editors Publications links" => "editors_publications", "External resources" => "externalResources", "Genres" => "genres",
11 "Languages" => "languages", "Material aspects" => "materialAspects", "Material colors" => "materialColors", "Materials" => "materials",
12 "Periods" => "periods", "Postings" => "postings", "Proveniences" => "proveniences", "Publications" => "publications",
13 "Regions" => "regions", "Rulers" => "rulers", "Sign readings" => "signReadings", "Staff" => "staff", "Users" => "users"];
14 ?>
15 <h1>Dashboard</h1>
16 <br />
17
18 <div>
19 <ul>
20 <?php $this->Grid->Alphabetical($additional)?>
21 </ul>
22 </div>

```

- The common template can be used for header and body part for both public and admin pages(explained in detail in Task -2: [Merging Admin and Public Templates](#)). We can manage the space of the admin dashboard (body/menu) by the use of **Column Cards**(The same template as for UPDATED browse page of users- [Task 9](#)) for each list of the functionalities(with the respective links provided) as given [here](#)(referring to the admin or editor restricted items).
- It will also be made sure that the designs respond to desktop, tablet as well as mobile views.
- For achieving the above goal, the **existing styling (sass)** and **color palette** will be used so as to maintain the uniformity of the website. Also, the **same** header, footer, and body **templates** will be used(explained in detail in Task -2: [Merging Admin and Public Templates](#)).
- There is extensive use of **HTML, CSS, and Bootstrap** here for achieving the **frontend** part.
- This will ensure that the admin header and footer have necessary **navigation links** as well as for the body/menu part, the contents/menu items will get organized without using unnecessary space. The **common modules** of the admin and user are available at the **merged template**. But the **restricted menu** items can be accessed using the **conditional check**.

BACKEND:

- Database to be created and modules(that to be used in the application-separate modules containing functions of normal logged-in user and admin), roles and rights (assigned to the admin like add, edit, and delete entries) to be added.
- Sample Schema is given below (consists of few functions for example purpose):

```
CREATE DATABASE `admin`;
USE `admin`;

CREATE TABLE IF NOT EXISTS `module` (
  `mod_modulecode` varchar(25) NOT NULL,
  `mod_modulepagename` varchar(255) NOT NULL,
  `mod_modulename` varchar(50) NOT NULL,
  PRIMARY KEY (`mod_modulecode`,`mod_modulepagename`),
  UNIQUE(`mod_modulecode`)
) ENGINE=INNODB DEFAULT CHARSET=utf8;

INSERT INTO module (mod_modulecode, mod_modulename, mod_modulepagename) VALUES
('ADMIN_DASHBOARD', 'Admin_Dashboard', 'dashboard.ctp'),
('ADD_ARTIFACT', 'Add_Artifact', 'add.ctp'),
('EDIT_ARTIFACT', 'Edit_Artifact', 'edit.ctp');
```

- System admin is added. The modules contain the privileges of the admin and the admin is given access to the modules.
- Modules contain the files like basic menu options as well as the restricted options (only accessible by the admins).
- **For example** -EDITING ARTIFACT(exclusive admin functionality)-Admin searches using the main search and an access-based edit button is offered to show the results.
- The next step is **validating user login** using PHP.
 - If user is logged in, the user will be taken to the Admin dashboard(if admin) / Home page(if general user), and entities would be shown as per access.
 - If the user isn't logged in and tries to access any page except the login page, the user will be redirected to the login page.
- On logging in, the admin/user is redirected to a common template where modules are assigned as per the role.
- Again Conditional checking (to ensure visibility) before each button/link to see whether the user has the right to access the particular entry.

SUBTASK 2 AND 3 APPROACH:

- **Conditional check** for each module (separate modules containing functions of normal logged-in user and admin). If the user has access to the particular role, the respective link/button would show up as shown in the pseudocode given below:

```
<?php if (authorize($_SESSION["access"] ["ADMIN_DASHBOARD"] ["Admin_Dashboar
//link or button
<?php } ?>

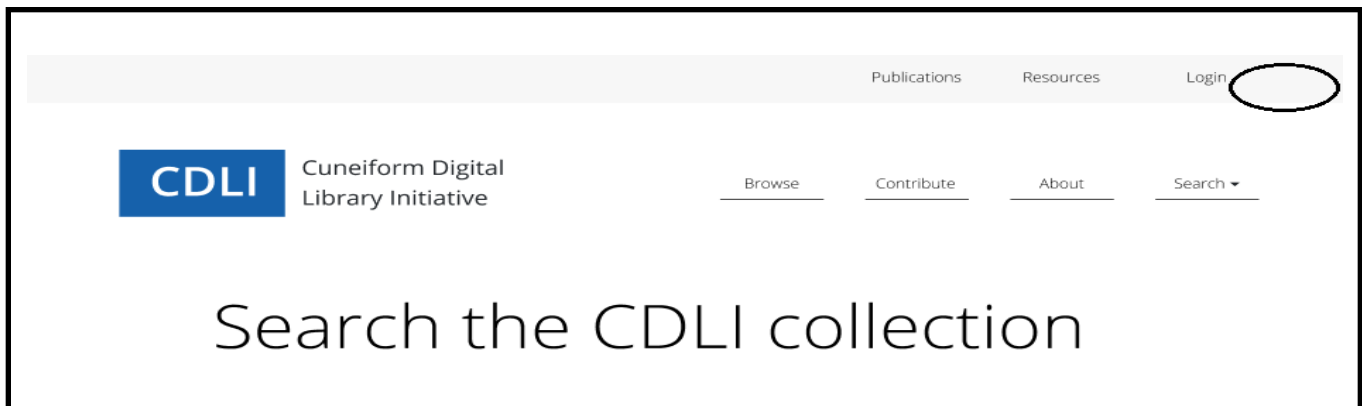
<?php if (authorize($_SESSION["access"] ["ADD_ARTIFACT"] ["Add_Artifact"])))
//link or button
<?php } ?>
```

- If a user tries to access a module using a direct page URL but is not assigned, the user won't be able to pass the security check.
- **EXAMPLE:** Admin or editor searches for the artifacts he is interested in and then we would have an edit option visible as per access.
- From discussions on slack the given solution is observed: On the admin side, most add and edit forms have been generated automatically and require to be audited and changed to the needs of the data. This also includes processing the data entered when needed (especially in the case of inscriptions). This can be done using a security check(mentioned in [SUBTASK 2 and 3](#) of Task 1) and [merging admin and public templates](#).

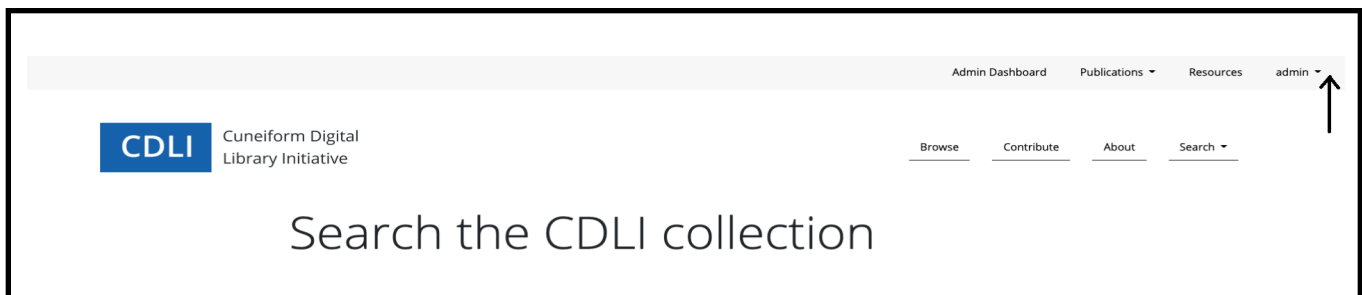
3.2 Merging Admin and public templates

- The admin and public pages use different templates(Issue [#367](#)).
- Both can have common body/menu, header, and footer templates (so as to unify the styling of the website as well as to **avoid the unnecessary use of two templates** which uses extra space and increases complications).
- The modules(containing functionalities or menu items) of the admin and user are available at the common template. But the restricted menu items can be accessed using the conditional check(mentioned in [SUBTASK 2 and 3](#) of Task 1.)
- **EXAMPLE 1:** There is a visible difference between admin and public page styling. The Right top menu of the admin page is too close to the edge and lacks margin, whereas in other pages there is an appropriate margin available. This is because different templates are used for admin and normal logged-in users. It can be resolved by using a common template.

OTHER PAGES HEADER:





ADMIN PAGE HEADER:



COMMON FOOTER:

NAVIGATE
[Browse collection](#)
[Contribute](#)
[About CDLI](#)
[Search collection](#)

ACKNOWLEDGEMENT
Support for this initiative has been generously provided by the [Mellon Foundation](#), the [NSF](#), the [NEH](#), the [IMLS](#), the [MPS](#), [Oxford University](#) and [UCLA](#), with additional support from [SSHRC](#) and the [DFG](#). Computational resources and network services are provided by [UCLA's HumTech](#), [MPIWG](#), and [Compute Canada](#).

CONTACT US
Department of Near Eastern Languages and Cultures
378 Humanities Building, 415 Portola Plaza,
Los Angeles, CA 90095-1511, USA
  [Donate](#)

© 2019-2020 Cuneiform Digital Library Initiative.

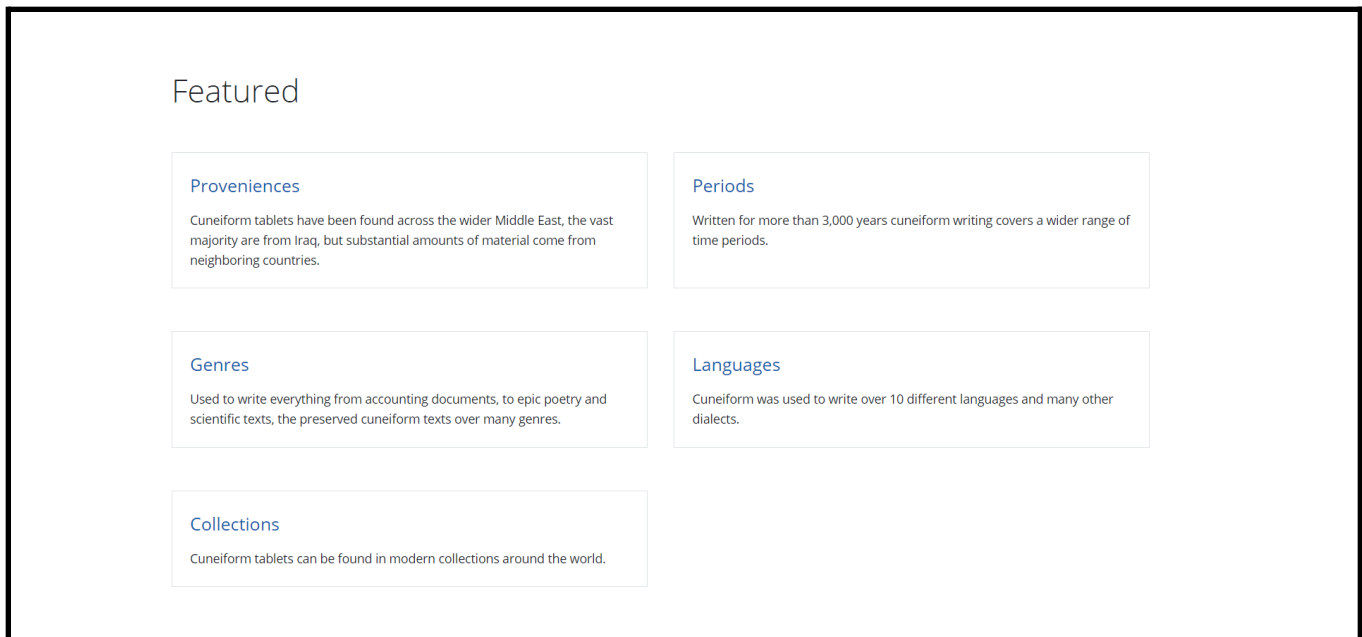
EXAMPLE 2: Different templates used for restricted admin menu items and normal user menu/items. This can be resolved using the methodology mentioned above (explained in detail in Task -2: [Merging Admin and Public Templates](#)).

ADMIN DASHBOARD BODY/MENU:

Dashboard

- [Abbreviations](#)
- [Agade Mails](#)
- [Archives](#)
- [Archival Tool](#)
- [Articles](#)
- [Artifacts](#)
- [Artifacts Publications links](#)
- [Artifact Types](#)
- [Authors](#)
- [Authors Publications links](#)
- [CDLI tablet](#)
- [CdlNotes](#)
- [Collections](#)
- [Dates](#)
- [Dynasties](#)
- [Editors Publications links](#)
- [ExternalResources](#)
- [Genres](#)
- [Languages](#)
- [MaterialAspects](#)
- [MaterialColors](#)
- [Materials](#)
- [Periods](#)
- [Postings](#)
- [Proveniences](#)
- [Publications](#)

PUBLIC PAGE BROWSE BODY/MENU:



3.3 Menu discrepancy problem

- 16px margin to be added to the right top menu of Admin Page.



3.4 Unifying presentation of all pages

- All the pages presentation (in terms of design) can be unified by using the **same template** for body/menu, header, footer as well as using the **same stylesheet and color palette**.
- Though all the pages have a similar presentation there are some exceptions.

- **EXAMPLE 1:** With reference to issue [#457](#), the pagination UI doesn't match with the UI of other search pages of cdli. This can be done by using the same stylesheet for all pages.
- Improper contrast: For pagination, on selecting any page number, the text color should turn to **#FFFFFF**. Along with this, the previous and next to be replaced with arrows as given below.

<div> << Previous <div>1 2 3</div> Next >> last >> </div> <p>Page 1 of 3, showing 20 record(s) out of 46 total</p>	<div> << <div>1 2 3</div> >> </div>
--	---

- **EXAMPLE 2:** The “Publications Page” doesn't have the common header available for other pages. This can be corrected using the same header footer template for all the pages.

Publications Index

PUBLICATION SEARCH

BibTeX Key:

Artifact:

Title:

Entry Type:

3.5 View/Edit/Index pages of all Entities should be similar

This can be done using the **same template**(as specified in [Task-4](#)) for all entity pages ie View, Edit, and Index. (For similar appearance).

VIEW AND EDIT:

- For view and edit, there is pre available template which can be used (**reusability**) for other entities according to the features available for that particular entry of that particular entity.
- The artifact page view and edit options are very neat and clear. So, all other entities can have a common template for view and edit.

VIEW:

MSVO 1, 010 (P005077)

Translation & transliteration

obverse
1. 2(N45) 5(N14)@, SZE~a
en: 150 N1 units of barley.
2.a., JGISZ.TE]# DU ME~a
en: for ...
2.b., SZUBUR E2~a# SANGA~a#
en: pigsty accountant;

reverse
1. 1(N50), GAN2
en: 10 bur3 field.
calculation: $150N1 + 10 = 15N1$ barley
per bur3
following later tradition and assuming
this is a seeding account, it would
support an identification of the base
unit in grain metrology: N1, with ca.
20-25 liters

[Download image](#)

[Back to top](#) ↑

FULL VIEW OPTIONS

Add/remove elements in full view

Show/hide filter sidebar (desktop)

- ☒ Show filter sidebar
- ☐ Hide filter sidebar

Add/remove fields from search results

Object data:

- ☐ Museum collections
- ☐ Period
- ☐ Provenience
- ☐ Object type
- ☐ Material

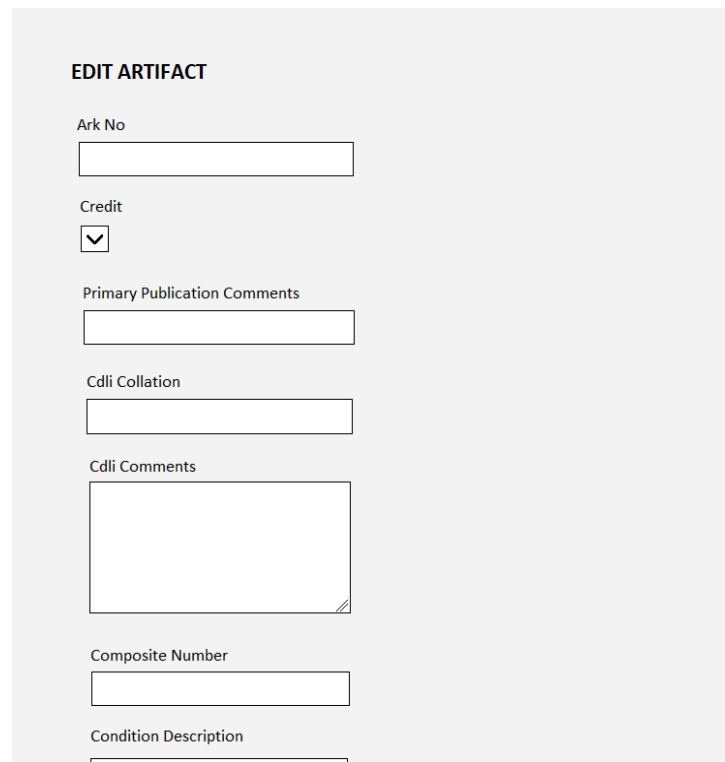
Textual data:

- ☐ Genre/sub-genre
- ☐ Language

Publication Data:

- ☐ Authors
- ☐ Date of publication

EDIT(Modified one-[Task 7](#)):



- Though this is difficult to implement because every entity has its own set of functions and features available but can be implemented using **dynamic common templates**.
 - ❖ The entity details are retrieved from the database.
 - ❖ Template content is fetched from the database based on the entity.
 - ❖ Template variables are replaced by dynamic values.

INDEX :

- Since there are 3,35,000 artifacts, it is quite impossible to index each one of them. So, the search option (already there) would allow the admin or editor to search the artifact he or she is interested in and then the edit button is visible as per access.
- The index page of other entities can be implemented like the wireframe given [here](#) (this is bootstrapy and can be implemented using the existing templates because it used cards and basic styling which has extensively been used throughout the webpage):

3.6 Enhance the display of a single artifact and expanded search results :

- The first and foremost issue that needs to be fixed in both Single Artifact, as well as Expanded Search Results, is [#358](#). The existence of the image has to be checked before displaying it. This can be done by the pseudo code given below:

```
$file = 'url';
$file_headers = @get_headers($file);
if(!$file_headers || $file_headers[0] == 'HTTP/1.1 404 Not Found') {
    $exists = false;
}
else {
    $exists = true;
}
```

- Given below are the individual enhancements required for a Single artifact and expanded search results.

DISPLAY OF SINGLE ARTIFACT:

- Currently, any single artifact looks like this(which is a pretty plane) :
- The design can be improved in the way given below: The below view is easy to implement because this border template is already available for artifact view (on search).

BEFORE

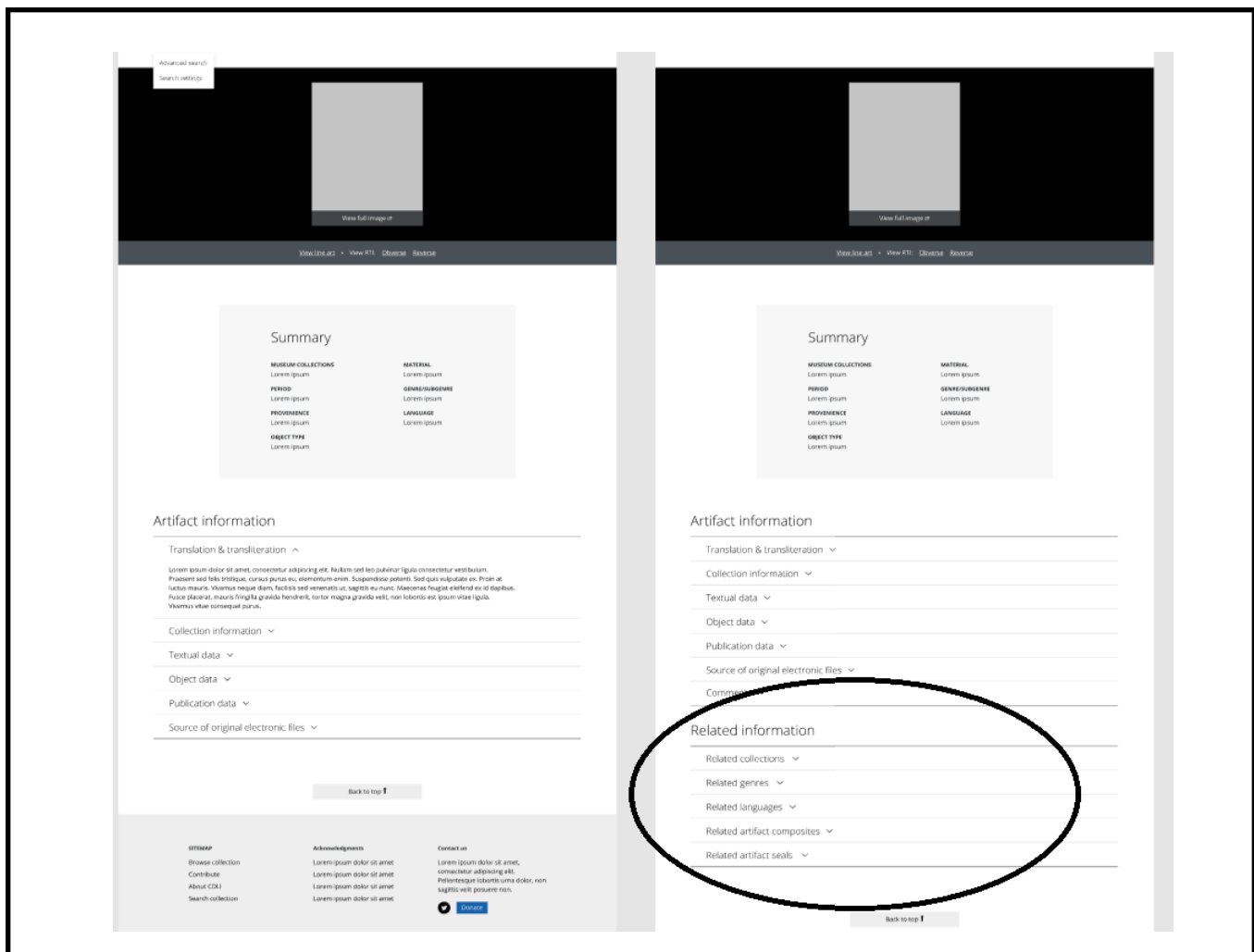


AFTER



DISPLAY OF EXPANDED SEARCH RESULTS:

- On advanced search, **related information** should be displayed for each one of the searched entries. If not available, then in place of related information -" No related information was retrieved. " should be displayed. The implemented wireframe is available [here](#).
- IMPLEMENTATION:**
 - ❖ Dynamic templates to be used having related information list for each search.
 - ❖ The related information details are retrieved from the database.
 - ❖ Template content is fetched from the database based on the search.
 - ❖ Template variables are replaced by dynamic values.
 - ❖ If no related information was retrieved, then the Template variable is replaced with "No related information was retrieved".
- The screenshot given below is how the current web page looks like :



- Unnecessary availability of two buttons. The bottom button can be removed from search and advanced search pages.



Search

Back to top ↑



Advanced Search

Search

3.7 Design edit artifact pages

EDIT ARTIFACT PAGE:

- The “EDIT ARTIFACT” division looks a bit compressed, the division size width can be increased so as to make it clear.
- Also, the edit options do not have space between them and the input forms beside them. The **alignment is also improper**-The input form boxes aren't in the same line as well as the multiline input forms appear first and the fields appear after that which gives an improper look. This can be corrected in the way given below:

<u>BEFORE:</u>	<u>AFTER:</u>
<p>EDIT ARTIFACT</p> <p>Ark No <input type="text" value="21198/zz001q0dtm"/></p> <p>Credit <input type="text" value="v"/></p> <p>Primary Publication Comments <input type="text"/></p> <p>Cdli Collation <input type="text"/></p> <p>Cdli Comments <input type="text"/></p> <p>Composite No <input type="text"/></p> <p>Condition Description <input type="text"/></p> <p>Date Comments <input type="text"/></p>	<p>EDIT ARTIFACT</p> <p>Ark No <input type="text"/></p> <p>Credit <input checked="" type="checkbox"/></p> <p>Primary Publication Comments <input type="text"/></p> <p>Cdli Collation <input type="text"/></p> <p>Cdli Comments <input type="text"/></p> <p>Composite Number <input type="text"/></p> <p>Condition Description <input type="text"/></p>

- Above can be created by the code snippet given below (placeholders can also be added for extra convenience) :

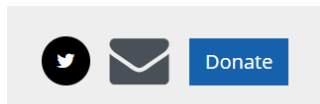
```
framework-phoenix-develop\app\cake\src\Template\Admin\Artifacts\edit.ctp
```

```
13      <?php
14          echo $this->Form->create($row);
15          echo $this->Form->control('ark_no', ['placeholder'=>'Enter Ark Number']);
16          echo $this->Form->control('credit_id', ['placeholder'=>'Enter Credit Id']);
17          echo $this->Form->control('primary_publication_comments', ['placeholder'=>'Enter Primary Publication Comments']);
18          echo $this->Form->control('cdli_collation', ['placeholder'=>'Enter Cdli Collation']);
19          echo $this->Form->control('cdli_comments', ['placeholder'=>'Enter Cdli Comments']);
20          echo $this->Form->control('composite_no', ['placeholder'=>'Enter Composite Number']);
```

- The above design can be improved in future scopes.

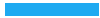
3.8 Improve Footer Styling

- There is **no hover effect** on the contact icons(in the footer) given below. Can be added using the shadowing effect.



3.9 Enhance the styling of Browse Page of Users

- **Column Cards** to be used for each set of menu options (starting with the same alphabet) so as to give proper styling. Here is the wireframe of the updated browse page.
- Given below is the current browse page :



Dispersed categories

A Abbreviations Agade mails Archives Artifacts Artifact types Authors	C CDLI tablet	D Dates Dynasties
E External resources	J Journals	M Material aspects Material colors Materials
P Publications	R Regions	S Sign readings

- Given below is the modified browse page :

A Abbreviations Agade mails Archives Artifacts Artifact Types Authors	C CDLI Tablet	D Dates Dynasties
E External Resources	J Journals	M Material aspects Material colors Materials
P Publications	R Regions	S Sign readings

- The above can be achieved by following the **pseudo code** given below:

```
framework-phoenix-develop\app\cake\src\Template\Home\browse.ctp
```

```
<div class="bs4-cdli-card-horizontal border-0 card col-md-12 col-lg-4">
  <div class="card-body text-left border">
    <p class="card-title">
      <a>Alphabet</a>
    </p>
    <p class="card-text">Entity 1</p>
    <p class="card-text">Entity 2</p>
    .
    .
    .
  </div>
</div>
```

3.10 Build a Resource Page

- Currently, the Resource Page link is non-functional and doesn't navigate the user to a different page. There is no resource page available and it needs to be created according to the instructions given in issue [#209](#).
- Given [here](#) is the wireframe of the resources page keeping in view the points given in [#209](#).
- The link to an index of each entity is provided using column cards (the same template of [the Browse page](#) can be used). Finally, inbuilding the resource page inside the main framework header.

3.11 Restyle Default Template of many similar entities

- This task deals with restyling the default template used in a lot of similar entities. An example has been provided below:
- Currently, the "Artifact Types" page looks like this :

BARREL

Artifact Type	barrel
Parent Artifact Type	
Id	11

RELATED ACTIONS

[List Artifact Types](#)
[List Parent Artifact Types](#)
[List Child Artifact Types](#)
[List Artifacts](#)

NO RELATED ARTIFACT TYPES

RELATED ARTIFACTS

[Number of related artifacts - 264](#)

- The “Related Actions” division needs to be removed. Also, the “Related Artifacts” division consists of a non-functional link. The default template has been duplicated for many entities which gives rise to such improper styling and functionality as given in issues [#471](#) and [#279](#).
- This can be corrected by preparing a default template for **all entities** (because this inconsistent behavior has been noticed in many entities). The wireframe of the default template is provided [here](#).

**** IT WILL BE TESTED AND MADE SURE THAT ALL THE ABOVE DESIGNS AND IMPROVISATIONS RESPOND TO MOBILE AND TABLET. ****

**** IT WILL BE TESTED AND MADE SURE THAT MOST OF THE FUNCTIONALITIES AND DESIGNS REMAIN SAME FOR BROWSERS WITH JAVASCRIPT ENABLED AND DISABLED. ****

4. Tech Stack Used

1. HTML:

- Supported by all browsers.
- Integrate easily with all languages.
- Lightweight.
- Display changes instantly.
- Platform independent.
- Can easily embed media.
- Can link to any page - external or internal (Hypertext).

2. CSS:


- Helps to improve accessibility.
- High loading speed.
- Easy maintenance.
- Platform independence.
- Style is applied consistently across a variety of sites.
- Form spontaneous and consistent changes.
- Has the power for re-positioning.

3. Bootstrap:

- Responsive
- Speed of development.
- Customizable.
- Consistency.
- Packed JS component.
- Integration.
- Pre-built theme.
- Pre-style components.

4. Javascript:

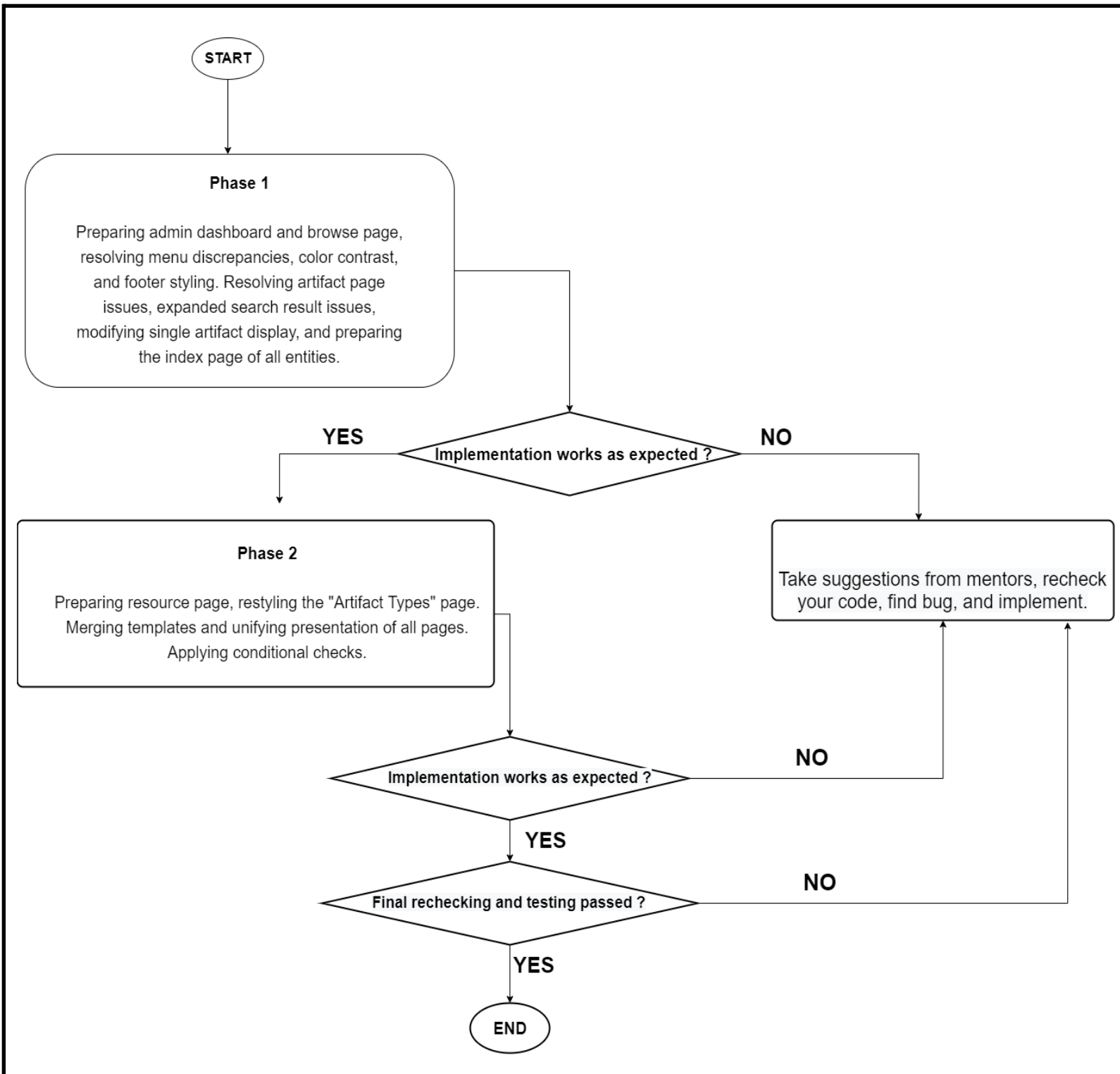
- It is a client-side language.
- Comparatively faster for the end-user.
- Extended functionality to web pages.
- No compilation needed.

- 
- Easy debug and test.
 - Platform independent.
 - Event-based programming language.
 - Procedural programming capabilities.

5. CakePHP:

- Open-source platform.
- No pre-configuration required.
- Object Relational Mapping.
- Easily extendable.
- CRUD Scaffolding.
- Ability to create test.
- Proper class inheritance.
- Model View Controller(MVC) Pattern.
- Easily extend with components and plug-ins.

5. Workflow chart



6. Timeline

1. Phase 0 - Pre Community Bonding Period (Present- May 17th, 2021)

- **Fix current issues/bugs** present and **merge pending PRs**.
- **Explore the framework/codebase** and post new issues(if encountered).
- Since cdli uses CakePHP in its backend, I would explore more about CakePHP and its implementation.

2. Phase 1 - Community Bonding Period (May 17th - June 7th, 2021)

- Discuss my plans about implementing project tasks in greater detail with mentors.
- Create a **detailed schedule** for the tasks and deliverables expected to be done.
- **Structure the workflow** for better implementation.
- Check and determine if there are any dependent issues to be solved before getting started with the coding phase.

3. CODING PHASE - PHASE 1:

June 7 to June 13:

- Finish preparing a common template(entities within cards) for [browse page](#) of users as well as [admin dashboard](#).
- Finish implementing the front end of [admin](#) and [browse](#) using this common template.

June 14 to June 20:

- Finish handling the [Menu Discrepancy Problems](#).
- Finish handling color contrast issues(eg: [Pagination](#) UI).
- Finish improvisation of the [footer Styling](#).

June 21 to June 27:

- Finish enhancement of the [edit artifact page](#) (aligation and appearance).
- Finish enhancement of the display of [single artifact](#) by adding bold borders(using pre-available view template).

- Finish adding conditional check before [displaying any image](#) and enhancing display when no “[Related Information](#)” is available.

June 28 to July 4:

- Finish preparing the default [index page template](#) for all entities.
- Finish applying the index template to all entities (except artifacts).

July 5 to July 11:

- Reusing the pre-available [view](#) template for all entities.
- Finish applying the view template to all entities.
- Sending report for Phase 1.

4. CODING PHASE - PHASE 2:

July 12 to July 18

- Finish preparing the [Resource Page](#).
- Finish applying all the links of the entities in the description of the resource page.
- Finish inbuilt the resource page inside the main framework header.

July 18 to July 25

- Finish restyling the “[Artifact Types](#)” page.
- Finish preparing a default template for all the Artifact Types and applying it.

July 26 to August 1

- Finish [Merging](#) the Admin and Public templates for all entities.
- Finish [Unifying](#) the presentation of all admin and public pages by the use of common templates.
- Rechecking if everything works fine.

August 2 to August 8

- For now only there were no conditional checks to access any functions/features/options of any entity. Everything was open to everyone.
- [Conditional checking implemented](#) where admin/user is redirected to a common template on passing the conditional. Again Conditional checking(to **ensure**

visibility) before each button/link to see whether the user has the right to access the particular entry.

August 9 to August 30

- Rechecking and testing every implemented feature.
- Preparing Documentation.
- Submission of Final Report.

7. Why I am the right person to work on this project?

Contributions to CDLI

- I have contributed to CDLI and it has been some weeks till now. I am quite active on slack, GitLab, and Github always coming on the front to resolve issues and follow up on everything to keep myself updated about the codebase of CDLI.
- [Given Below are some of my contributions :](#)

Issues	Merge Requests	Tags
#253	!227	dev:front end, ux & design
#308	!227	dev:front end, ux & design
#334	!243	good first issue, ux & design
#463	!243	dev:front end, ux & design
#103	#15	dev:front end, ux & design, Doing, feature:enhancement, Priority:3, Documentation
#103	#16	dev:front end, ux & design, Doing, feature:enhancement, Priority:3, Documentation

- I have contributed to issue [#214](#) which has helped me to understand a lot about dealing with the fallbacks with browsers having Javascript disabled. It has helped me to figure out ways of implementing my project for browsers with or without Javascript.
- Apart from this I am currently working on [Task 11](#) ([#471](#) and [#279](#))and trying to get rid of unnecessary divisions and lines of code from **159 files**. This will definitely improvise and simplify the codebase and styling of the default template used in a lot of entities.

Other Achievements

I have been working on Web Development for two years with **MERN Stack** and have familiarity with **PHP**.

- **Web Development Instructor** at MakerScientist Powered by ScienceUtsav.
- Open Source Contributor for **GirlScript Summer of Code 2021**.
- **Pre-Finalist** at Myntra Hackathon(**Myntra** Hackerramp).
- **Javascript Instructor** at CampK12.

8. Other Commitments

No prior commitments as of now. If selected for GSOC, I will devote all of my working hours(35-40 hours per week) to the work of GSOC exclusively.