Description: (For CDLI Blogs and GSoC website)

CDLI Framework integrates features of the project in a logical infrastructure, prepares new data displays, including machine-readable outputs, to enhance knowledge diffusion. It includes a unified interface for the project website, more powerful search capabilities, more internal links to navigate the catalogue, and intuitive displays for calendars, glossaries, bibliographies, etc.

The Project focuses on :

1) Building an authentication and authorization system for the framework.
2) Integrating Elastic Search for Simple Search.
3) Optimizing queries for Advance Search.
4) Preparing search results (Expanded and Compact format) with data preprocessing.

Proposal Link : Improving_CDLI_Framework.pdf

Objectives and Deliverables:

- Essential Objectives

#	Objective	Deliverable	Issue(s)
1.	Authentication (2FA)	Established more secure login by enforcing 2FA.	#4
2.	Authorization Setup (Role Based)	Successfully setup role based access	#84
3.	Simple Search	Integrating Elastic Search for fast and accurate search results.	#50
4.	Advance Search Optimization.	Optimized advanced search queries.	#48
5.	Search Result display	a. Expanded & Compact result b. Stats for Search Result. c. Search Filters d. Search Setting page	#238
6.	Integrating API's to download search results.	Users will be able to download data in various formats like RDF, XML, etc.	#216

- Additional Objectives

#	Objective	Deliverable	Issue(s)
1.	Search Settings	A separate setting page for	#37

		displaying search page and search result according to configuration.	
2.	Rocket.Chat Setup	Setting up and deploying Rocket.Chat for CDLI Developers.	#95

Notes:

1) Data download (backend) - Lars
2) Frontend - Samarth

Tentative Timeline

Week-1:

Objective	Deliverable
a. Authentication (2FA)	a. Successful implementation of 2FA middleware. b. Testing 2FA Middleware.
b. Password Strength Checker	a. Implemented Password Checker b. Testing on the Register Page.
c. Implement Password Retrieval	a. Implemented Password Retrieval Module b. Testing on the Login Page.

a. Authentication (2FA)

Issue : #4

Objective :

Redirecting users to '*/twoFactor*' if 2FA is not enabled for the user and restricting user access to browse (if logged in) until it is not enabled.

Implementation :

As discussed in Proposal we are going to follow the **Approach 2** i.e creating a middleware layer for 2FA.

Backfall Implementation : **Approach 1**

Checklist (to be added on issue page):

1. Create Middleware for 2FA.
2. Test Login and Registers.
3. Test Pages (when logged in)
4. Test Pages (when browsing as guest)

Resources :

1) https://book.cakephp.org/3/en/controllers/middleware.html
2) https://book.cakephp.org/3/en/development/routing.html#connecting-scoped-middleware
3) https://book.cakephp.org/3/en/tutorials-and-examples/blog-auth-example/auth.html

b. Password Strength Checker

Issue : #107

Objective :

To enhance security, the system should prompt users to create a strong password according to specific requirements.

Implementation :

As discussed in the proposal.

Rules for strong password :

a. At least 8 characters—the more characters, the better
b. At least one uppercase and lowercase letters
c. At least one digit to be present.
d. At least one special character, e.g., ! @ # ?]Note: Avoid '<' & '>' in password, as both can cause problems in Web browsers
e. Passwords should not contain username, first_name, last_name, etc..

Checklist (to be added on issue page):

1. Implement rules.
2. Test on register Page

Resources :

1. https://nakedsecurity.sophos.com/2016/08/18/nists-new-password-rules-what-you-need-to-know/

c. Implement Password Retrieval

This issue is currently handled by @stephenjude. If the issue is not implemented till GSoC time, then we can ask him to hand over the issue.

Branch : https://gitlab.com/cdli/framework/-/tree/phoenix/feature/password_retrival

Issue : #211.

Objective :

Forgot password feature.

Implementation :

Checklist (to be added on issue page):

Resources :

1. https://github.com/hunzinker/CakePHP-Auth-Forgot-Password/blob/master/controllers/users_controller.php

Week-2 and Week-3 :

Objective	Deliverable
a. Access control page.	a. To view, add or remove the roles of users of the system.
b. System setup for roles viz. Users, Editors and Granular Access.	a. System should display contents based on roles as described in objectives. b. Testing web pages in the system.
c. System setup for the Admin role.	a. Admin should be able to access all pages. b. Re-testing of web pages according to admin role.
d. Documenting Role based Access.	a. Documentation which developers should follow to add functionality to the framework based on roles.

a. Access control page

Issue : #232

Objective :

Page which can help admin to assign and unassign roles to the users.

Implementation :

Page will consist of a list of users and their respective roles. Admin can edit roles of respective users.

Checklist (to be added on issue page):

1. Defining types of role in details. (apart from 4 mentioned in issues #84)
2. Assigning role id for each type.
3. Fixing Edit and Delete on “/admin/users”.
4. Documentation :
    a. How to restrict access methods and views based on privilige

Resources :

b. System setup for roles viz. Users, Editors and Granular Access

Issue : #84

Objective : framework authorization for roles : Users, Editors and Granular Access.

Implementation :

The implementation approach will be to keep check conditions in each controller and on the view page according to roles. This approach might be naive but can be replaced if some better solution is found.

Checklist (to be added on issue page):

1. Implemented for role User.
2. Implemented for role Editors.
3. Implemented for role Granular Access.
4. Tested for User.
5. Tested for Editors.
6. Tested for Granular Access.
7. Documentation.

Resources :

1. https://github.com/AlessandroMinoccheri/UserPermissions

c. Setting up a system setup for admin role

Issue : #84

Objective : framework authorization for roles : Users, Editors and Granular Access.

Implementation :

Checklist (to be added on issue page):

1. Implemented for role Admin.
2. Tested for Admin.

3. Documentation.

Resources :

d. Documenting role based system

Issue : #84

Objective :

Documenting the page/functionality accessible by role and how to add the functionality according to role.

Implementation :

1. List of all pages with their respective functionality and accessibility description.

Checklist (to be added on issue page):

Resources :

Comments :

@epp > this is really important because not all pages are finished, also some gsoc students will be mingling less, like the mobile app person, but she will need to know this because she will also prepare the cdli tablet management interface for admins and editors
@epp > the mobile app relies on cdli tablet entries, not artifact entries all the cdli tablet material is public and the app basically works by looking up a json feed.
@epp > basically tell developers how to restrict access to their methods and views based on privilege

Week-4:

Objective	Deliverable
a. Triggers for inactive users.	a. For login activity older than 6 months, make account status inactive
b. Create 2FA Google Authenticator Guide	a. Documented 2FA Google Authenticator.
c. Fix '/logout' functionality	a. Restricting logout using GET request and accepting only POST request.
d. Documentation & Report for Phase-1	a. Phase-1: all implementation should be documented. b. Report For Phase-1 evaluation.

a. Triggers for inactive users.

Issue : Not created yet.

Objective : To set users status inactive after showing inactivity for more than 6 months.

Implementation :

As discussed in Proposal, creating a script running every day at specific time to check users last login activity against the current time and checking if the difference between time periods is greater than 6 months or not.

Checklist (to be added on issue page):

Resources :

1. https://blog.securityinnovation.com/blog/2011/09/automatically-lock-inactive-user-accounts.html
2. https://stackoverflow.com/a/9645245/5921680 Something like this

b. Create 2FA Google Authenticator Guide

Issue : Not created yet.

Objective :

A proper guide to be documented to make users aware about the process of 2FA using Google Authenticator.

Implementation :

Create an Guide_2FA.md

Checklist (to be added on issue page):

Resources :

c. Fix '/logout' functionality

Issue : Not created yet.

Objective :

Making logout functionality using POST request and not GET Request.

Implementation :

1. If logout is requested using GET, then the user will be redirected to the logout page form asking the user one more time if he really wanna logout.
2. For POST request, it will execute as per its default functionality.

Checklist (to be added on issue page):

Resources :

1. https://stackoverflow.com/questions/3521290/logout-get-or-post

Week-5 and Week-6:

Objective	Deliverable
a. Setting up ElasticSearch	a. ElasticSearch containers should be ready and basic ElasticSearch API's queries should work.
b. Updating ElasticSearch configuration to setup search queries.	a. ElasticSearch configured according to content to be searched in MySQL.
c. Simple Search with ElasticSearch	a. ElasticSearch will be fully integrated and in working condition.
d. Testing Simple Search.	a. Performing various testing to benchmark the time.
e. Backup indexes	a. Backup and recovery mechanism created.
~~f. Highlighting Search Result~~	~~a. Searched keywords will be highlighted.~~
f. Expanded and Compact View.	a. Users can view the search results in expanded and compact view format.
g. Documentation	a. Documentation about Setup, Implementation of ElasticSearch.

a. Simple Search

Issue : #50

Objective : Making Simple search fast and accurate by integrating ElasticSearch.

Implementation :

1. Setup the ElasticSearch containers and test basic queries with dummy data.

2. Setting up additional configuration according to MySQL DB.
3. The search query requested from the user side will first query the mysql DB then it will update indexed in ElasticSearch.
   There is another way which creates the index directly from MySQL DB using JDBC.
   References : 5 and 6.
4. Once the ElasticSearch is querying on data present from MySQL DB, then it will be tested for various scenarios and time will be recorded.
5. Setting up a backup mechanism for created indexes. Since the indexes are created on Nodes (i.e. Server), when the node fails the data present on that node is lost.
6. Highlighting search results.
7. Documentation

PS: ElasticSearch documentation by CakePhp will be followed for initial implementation.
Reference : 3

Checklist (to be added on issue page):
1. Highlighted Search result only for compact view.

Resources :
1. https://www.youtube.com/watch?v=qG3N2rMOeBE
2. https://github.com/cakephp/elastic-search
3. https://book.cakephp.org/elasticsearch/3/en/index.html
4. https://www.cloudways.com/blog/setup-elasticsearch-with-mysql/
5. https://discuss.elastic.co/t/indexing-mysql-database-with-elasticsearch-newbie/6229/6
6. https://github.com/jprante/elasticsearch-jdbc
7. https://medium.com/hepsiburadatech/how-to-create-faceted-filtered-search-with-elasticsearch-75e2fc9a1ae3
8. https://book.cakephp.org/3/en/controllers/components/cookie.html

Week-7 and Week-8:

Objective	Deliverable
a. Rewriting Advance Search Controller.	a. Refactoring the present Advance Controller
b. Implementing regex for Advance Search.	a. Advance Search with regex support
c. Expanded, Compact View and Statistics	a. Search result with Expanded and Compact View and Statistics.

a. Advance Search

Issue : #48

Objective : To optimize advanced Search.

Implementation :

First approach is to expand the simple search to accommodate the advanced search functionality.

Second Approach will be to write efficient queries.

Checklist (to be added on issue page):

Resources :
https://github.com/cdli-gh/cdli-search
https://github.com/cdli-gh/cdli-gh.github.io/blob/master/_pages/cdli_search.md

b. Expanded, Compact View and Statistics

Issue : #238

Objective :

Prepare search results according to expanded and compact view and statistics for every search.

Implementation :

Checklist (to be added on issue page):

Resources :

Week-9:

Objective	Deliverable
a. Search Setting Page	a. Search result will display according to configuration set by User on Search Setting Page.
b. Integrate Download feature	a. Search results are available for download.
c. Testing Simple and Advance Search with new implemented features.	a. Search Setting and Download should work as expected for both Simple and Advanced Search.

a. Search Setting Page

Issue : #37

Objective : To create a search Setting Page.

Implementation :

A special page to set configuration for search options and search result display.
The default settings will be stored in client's browser cookies.
In the future we might make some provision to store cookies on the server.

Checklist (to be added on issue page):

Resources :

b. Sample

Issue : Not created yet.

Objective :

Implementation :

Checklist (to be added on issue page):

Resources :

Week-10:

Objective	Deliverable
a. RocketChat Docker Setup	a. RocketChat config setup completed.
b. Configuring SSL Reverse Proxy	b. Setting up SSL Reverse Proxy for RocketChat.
c. Testing RocketChat and Documentation	c. Testing RocketChat with backup mechanism and Documentation.

a. Sample

Issue : Not created yet.

Objective :

Implementation :

Checklist (to be added on issue page):

Resources :

Week-11:

Objective	Deliverable
● Buffer week for pending deliverables	● Will be utilised to complete the pending deliverables (if any) or fix bugs and testing.

Week-12:

Objective	Deliverable
a. Final Project Report	a. Final Project Report should be ready and reviewed by Mentor
b. Code Submission	a. Successfully submit the clean and well documented code.

a. Final Project Report

Objective : To prepare the final project report.

Implementation :

a. Project Report Outline ?

Resources :

b. Code Submission

Objective : To submit code on GSoC website..

Implementation :

a. Code submission format ?

Resources :

Other Resources :

1. Evaluation

Discussion

@epp >

Search challenges

Current state

Right now, there is a main search available on the front page which displays results in a format that looks a bit like the expanded search results. There is also an advanced search with some regex prepared for some fields, and the search results appear on the same page as advanced search instead of sharing the display with simple search. There is no search settings page and we can't find the filters to filter search results. There is no compact display of search results. There is a stats display on the advanced search page but it should be available only after search. Search queries are slow.

What we need

- A main search that works with elasticsearch
- An advanced search that works with regex
- The display of search results, from both types of search
    - Expanded results
    - Compact view results
    - Stats results
- The filter bar to weed out search results based on additional criteria
- The Search option page and the application of its settings to search results

What to do and how

Main search

- Read about elastic search cakephp implementation and install the plugin
- Generate the elasticsearch indexes and start testing search
- Implement a system for refreshing the indexes based on changes in the db

@karna98 >

Simple Search

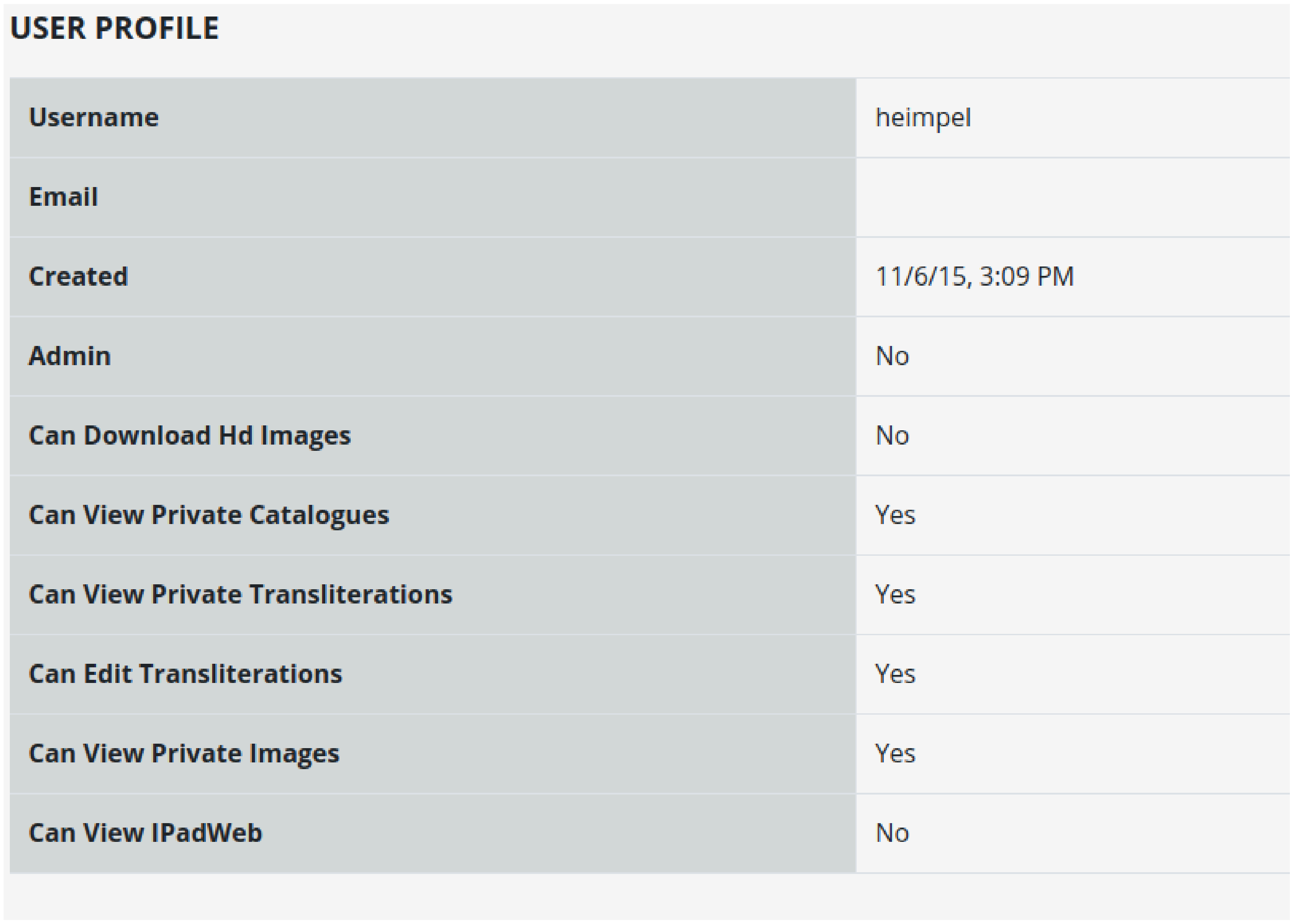
Compact Display	Expanded Display
Highlight transliteration in the compact view	

Advance Search

Compact Display	Expanded Display
Highlight transliteration in the compact view	

@karna98 >

Regarding the user roles



USER PROFILE

Username	heimpel
Email	
Created	11/6/15, 3:09 PM
Admin	No
Can Download Hd Images	No
Can View Private Catalogues	Yes
Can View Private Transliterations	Yes
Can Edit Transliterations	Yes
Can View Private Images	Yes
Can View IPadWeb	No

1. Instead of Admin, it will be replaced by User type.
   User Type will have Drop down for roles
   a. Normal User
      (By default)
      (if better name please suggest)
   b. Editor
      (i. By the name, it means he will be editing something . But I have no clue of what he will be editing.) (ii. Will he add and can edit that particular items or any item added by different user)
      > https://github.com/cdli-gh/cdli-gh.github.io/blob/master/_pages/cdli_privileges.md

(iii. Is there any list/doc where we can figure out which page performs what action like add/edit. It will be helpful while coding for permission. PS: Not routes, I need overview of all pages and what are their functionality)

c. Admin
(I guess he will own everything)
(There won't be a thing which I have to restrict from admin. Right?)
> for now **No**
Issue : While visiting **/admin/users/index**. If I click on **'Edit Your profile**, it redirects to the user profile of logged in user and not of the index user. Either the option should not be there or it needs to be fixed.
PS. Edit Your profile is placed quiet oddly. Most of the websites either have at bottom or at top.

2. I guess the (**Can View** *) are the granular roles.
If yes, then I am thinking of assigning them role_id.
As I will be passing this role_id in SESSION variable (role_id => [rid1, rid2, ...]) and will be using it. (Then I don't have to query every time on DB)
>
It also means admin, editor and normal user will have role id.
We will be requiring a role_id table in DB to establish a role_id and its respective name.
And the db schema for uses table can be modified by replacing all the **can view** * columns by a single column where we can store a role array.
> Changes coming in the updated DB version.
Removing **can view** * columns from users_table and creating new role_mapped_user_table to map role to user_id.

One more issue is if i kept a variable in session it should be secure.
> Create issue in future to check security concerns regarding SESSION
https://stackoverflow.com/questions/1181105/how-safe-are-php-session-variables

UI Suggestions by @epp
> https://www.bootstraptoggle.com/
https://codepen.io/jenniferlouie/pen/vEVMwe

One more thing we need to make changes to the URL part, we don't have to disclose the user_id and any user can access the user profile of others by changing user Id.

@karna98 >

At the end of GSoC, this whole document will be made public or I will add this document to GSoC Blogs. If that's okay ?
> Yes