CSC1106 Web Programming

Group 7

2200692 Pang Zi Jian Adrian
2200795 Ashley Tay Yong Jun
2200959 Peter Febrianto Afandy
2201018 Jeffrey Yap Wan Lin
2201052 Nur Hakeem Bin Azman
2201159 Ryan Lai Wei Shao

Table of Contents

01

Overview

02

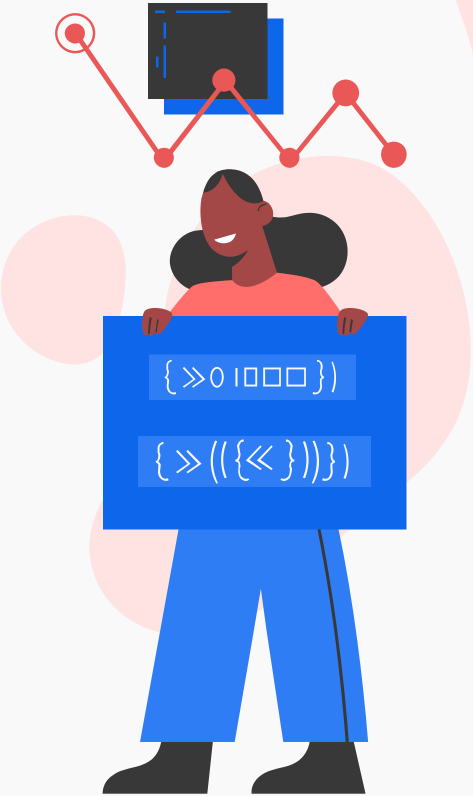
Libraries & Functions

03

Security

04

Demo



01

Overview

Overview

- Development of a CodeIgniter 4 (CI4) Library that enables programmers to create customisable forms for use within their web applications and store the user's respective responses securely.

- Test library against US F1040 form to showcase its features

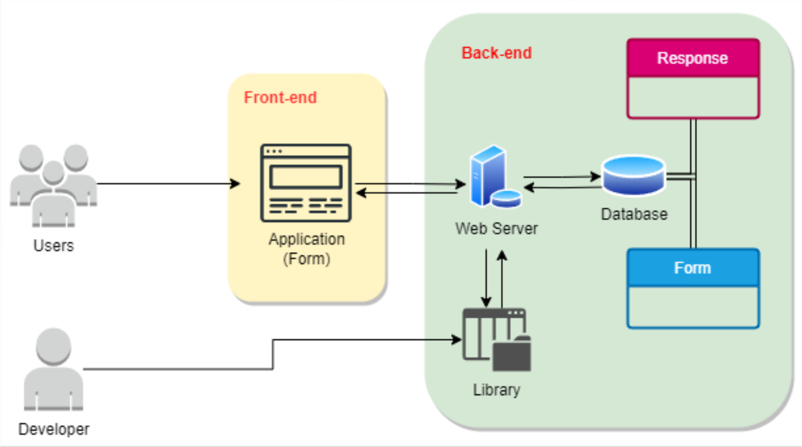
Prepared Deliverables

- Bare-bones web application
  - Only contains essential custom library functions and components

- Administrator Dashboard web application
  - Management of form templates and responses
  - Serve as reference for developers on how to use library
  - Test and showcase library against required F1040 forms

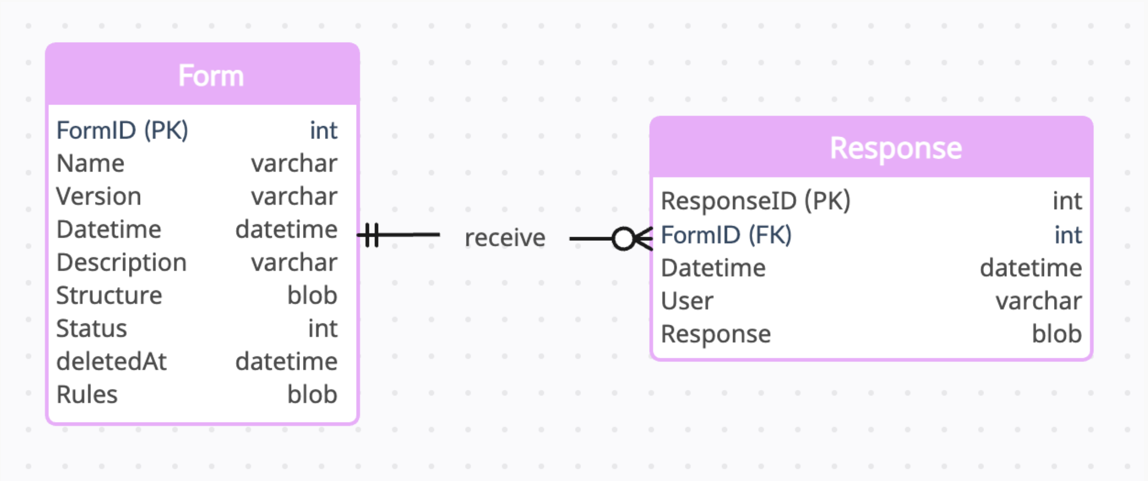
Library Architecture

- CI4 Custom Library (multiple files and components)
- MySQL relational database (for storage and CRUD management of form templates and responses)



Database Design

- Consists of two tables, Form and Responses
- Form table stores – serialized structure of forms, creation date, version number and custom serialized rules
- Response table stores – serialized responses from the end users
- Migration files are included to facilitate fast deployment of the custom library

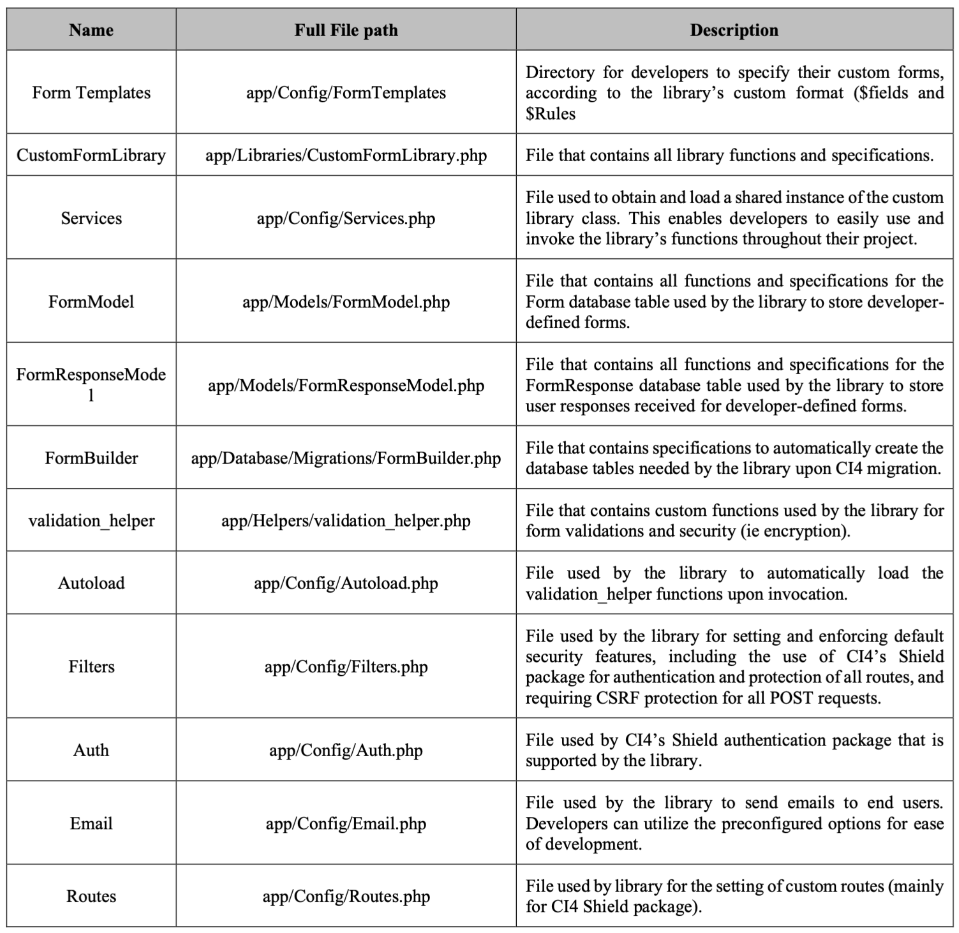


Form	
FormID (PK)	int
Name	varchar
Version	varchar
Datetime	datetime
Description	varchar
Structure	blob
Status	int
deletedAt	datetime
Rules	blob

Response	
ResponseID (PK)	int
FormID (FK)	int
Datetime	datetime
User	varchar
Response	blob

Library Design

- Consists of various files including models, custom library, custom helper functions and configurations



Name	Full File path	Description
Form Templates	app/Config/FormTemplates	Directory for developers to specify their custom forms, according to the library's custom format ($fields and $Rules
CustomFormLibrary	app/Libraries/CustomFormLibrary.php	File that contains all library functions and specifications.
Services	app/Config/Services.php	File used to obtain and load a shared instance of the custom library class. This enables developers to easily use and invoke the library's functions throughout their project.
FormModel	app/Models/FormModel.php	File that contains all functions and specifications for the Form database table used by the library to store developer-defined forms.
FormResponseModel	app/Models/FormResponseModel.php	File that contains all functions and specifications for the FormResponse database table used by the library to store user responses received for developer-defined forms.
FormBuilder	app/Database/Migrations/FormBuilder.php	File that contains specifications to automatically create the database tables needed by the library upon CI4 migration.
validation_helper	app/Helpers/validation_helper.php	File that contains custom functions used by the library for form validations and security (ie encryption).
Autoload	app/Config/Autoload.php	File used by the library to automatically load the validation_helper functions upon invocation.
Filters	app/Config/Filters.php	File used by the library for setting and enforcing default security features, including the use of CI4's Shield package for authentication and protection of all routes, and requiring CSRF protection for all POST requests.
Auth	app/Config/Auth.php	File used by CI4's Shield authentication package that is supported by the library.
Email	app/Config/Email.php	File used by the library to send emails to end users. Developers can utilize the preconfigured options for ease of development.
Routes	app/Config/Routes.php	File used by library for the setting of custom routes (mainly for CI4 Shield package).

Features

1. Custom Form Library
   a. Form Helper & Functions
   b. Library Design
   c. Form Validation Rules

2. Data Management
   a. Database Design
   b. CRUD – Form Responses
   c. CRUD – Form Templates

3. Security
   a. Input Validation
   b. Sanitization
   c. Encryption

4. Additional Features
   a. PDF
   b. Dashboarding
   c. User Register / Log in (Shield)

5. Library Demo

02

Library & Functions

Custom Form Library – Form Functions / Helpers

- The library offers flexibility and simplicity for creating customised forms and elements.

- Seamless integration with Bootstrap and its classes for customisation

- Using MySQL relational database to store the form structure and responses

- Form creation functions take parameters to generate the HTML markup
  - ➢ Eg. new_input($name, $value, $attributes)

- Have the capabilities to handle form validation and input sanitisation
  - ➢ Prevent security threats

Library Design

- Integration with CI4's Form Helper
  - ➢ Referencing CI Form Helper function to replicate some of their key form creation functions
  - ➢ Leverage on familiar syntax and functionalities while utilizing the custom library

- Custom HTML Form Tag Creation
  - ➢ Suite of functions for developers to create customizable HTML tags dynamically
  - ➢ Provide developers with flexibility and simplicity in utilizing the library

- Integration with Bootstrap 5
  - ➢ Leverage powerful grid systems and responsive design capabilities
  - ➢ Consistent styling and compatibility with Bootstrap's predefined styles

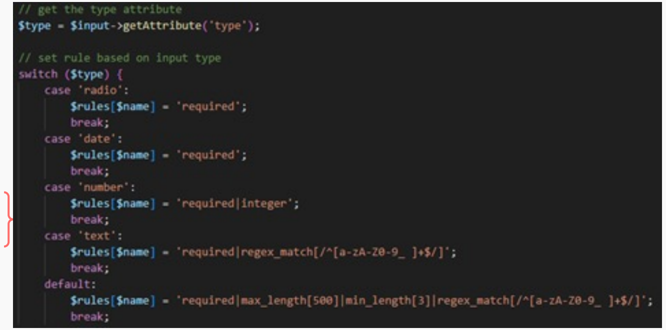
Library Design (cont.)

- CRUD Functions
  - Create, Read, Update, Delete primarily for managing developers' form templates and responses
  - Able to manage and archive form templates and form responses
  - ORM approach (for better security and management)

- Utility Functions
  - Expands the capabilities of the library, providing developers with additional options for managing and sharing their forms

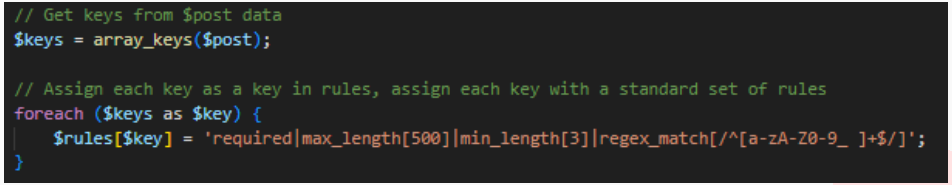
Validation of Forms

- generateRulesFromHTML
  - Dynamically generate rules for individual fields based of HTML structure
  - DOMdocument object allows us to read the HTML structure
  - Setting validation rules according to the individual fields using regex
- generateRulesFromPOST
  - Set validation rules for all POST data keys



```php
// get the type attribute
$type = $input->getAttribute('type');

// set rule based on input type
switch ($type) {
    case 'radio':
        $rules[$name] = 'required';
        break;
    case 'date':
        $rules[$name] = 'required';
        break;
    case 'number':
        $rules[$name] = 'required|integer';
        break;
    case 'text':
        $rules[$name] = 'required|regex_match[/^[a-zA-Z0-9_ ]+$/]';
        break;
    default:
        $rules[$name] = 'required|max_length[500]|min_length[3]|regex_match[/^[a-zA-Z0-9_ ]+$/]';
        break;
```

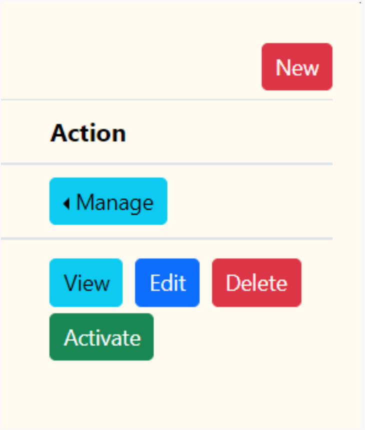


```php
// Get keys from $post data
$keys = array_keys($post);

// Assign each key as a key in rules, assign each key with a standard set of rules
foreach ($keys as $key) {
    $rules[$key] = 'required|max_length[500]|min_length[3]|regex_match[/^[a-zA-Z0-9_ ]+$/]';
}
```

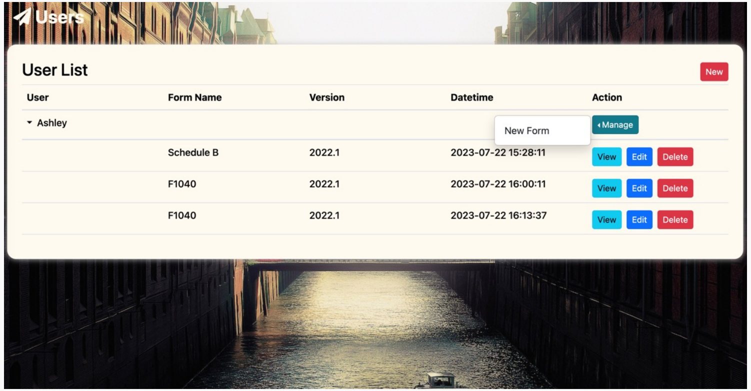
CRUD – Template Dashboard

- New (Create)
  - Creates a new form template, set it as active/inactive, version number, a short description and choose a predefined template
  - Structure of the HTML to show communication to the library
- Manage
  - Users can Soft Delete all the forms in the template dashboard or
  - Users can reverse the effects of soft deleting all the forms
- View (Read)
  - Preview the form created in a PDF format
  - Users can Download or Print the form using the PDF viewer
- Edit (Update)
  - The same as the New (Create) function
  - Modification of form template (versioning)
- Delete
  - Soft Deleting that particular form instead of erasing it from database
- Activate
  - To reverse the effects of Soft Deleting a particular form



Data Management – CRUD (Responses)

- New (Create)
  - ➢ Creates a new user (to submit form response)
  - ➢ User chooses a form they want to fill in and submit
- Manage
  - ➢ Create a new form for that same user
- View (Read)
  - ➢ View the form in a PDF viewer
- Edit (Update)
  - ➢ Users can go back and edit and re-submit their form
- Delete
  - ➢ Deletes the form entirely from the database



User List

User	Form Name	Version	Datetime	Action
▾ Ashley			New Form	◂Manage
	Schedule B	2022.1	2023-07-22 15:28:11	View Edit Delete
	F1040	2022.1	2023-07-22 16:00:11	View Edit Delete
	F1040	2022.1	2023-07-22 16:13:37	View Edit Delete

03

Security

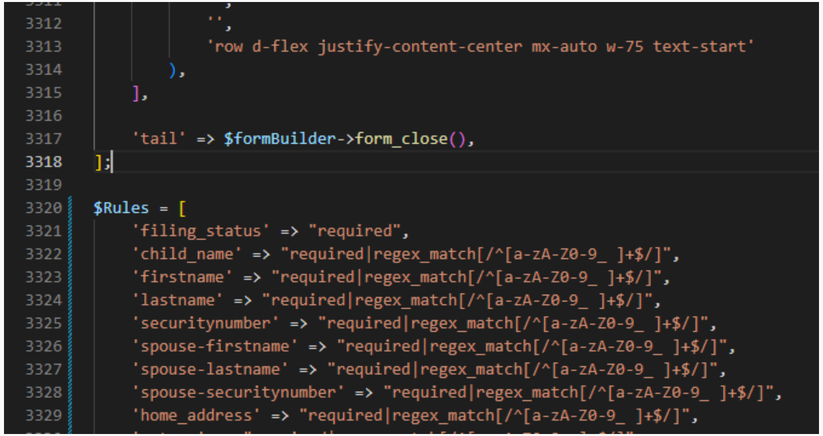
Validation

- Validation helps developers ensure that data acquired is according to his preference
  - ➢ Identify which field is required on the backend
  - ➢ Regex to manage which characters are allowed
  - ➢ Character minimum and maximum limit
  - ➢ Check for correct data type

- Helps improve security by validating for characters allowed with regex
  - ➢ Prevent html tags for XSS attacks
  - ➢ SQL injection detection
  - ➢ URL validation for phishing attacks

Validation of Forms

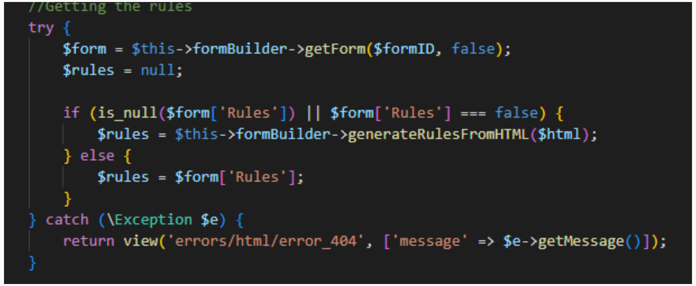
- Rules can be declared under a form template and saved to database
- Rules can be retrieved when required from database when required by developer
- If no rules are declared, default value is set to NULL



```
                '',
                'row d-flex justify-content-center mx-auto w-75 text-start'
            ),
        ],

        'tail' => $formBuilder->form_close(),
];

$Rules = [
        'filing_status' => "required",
        'child_name' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'firstname' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'lastname' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'securitynumber' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'spouse-firstname' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'spouse-lastname' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'spouse-securitynumber' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
        'home_address' => "required|regex_match[/^[a-zA-Z0-9_ ]+$/]",
```



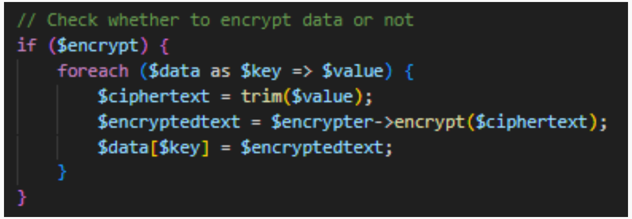
```
//Getting the rules
try {
    $form = $this->formBuilder->getForm($formID, false);
    $rules = null;

    if (is_null($form['Rules']) || $form['Rules'] === false) {
        $rules = $this->formBuilder->generateRulesFromHTML($html);
    } else {
        $rules = $form['Rules'];
    }
} catch (\Exception $e) {
    return view('errors/html/error_404', ['message' => $e->getMessage()]);
}
```

Encryption of Responses

- Enable developers to encrypt or decrypt data using the validate() function in the validation_helper.php

- Encryption process
  - Instantiate 'Encrypter' class from CodeIgniter's config services
  - Determine encryption need from $encrypt value
  - Iterate through user input data and trim white spaces
  - Encrypt data using 'Encrypter' object and specified key

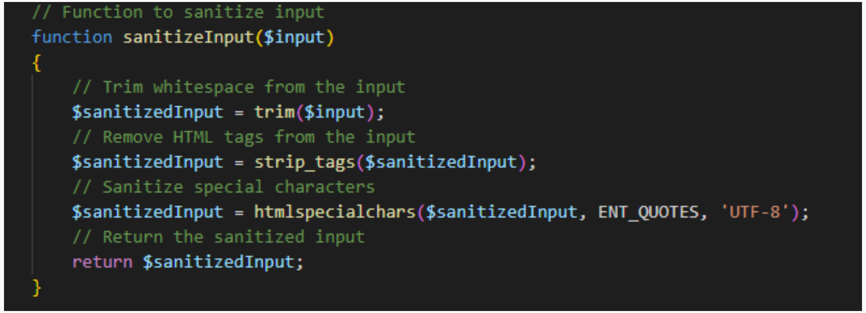


```
// Check whether to encrypt data or not
if ($encrypt) {
    foreach ($data as $key => $value) {
        $ciphertext = trim($value);
        $encryptedtext = $encrypter->encrypt($ciphertext);
        $data[$key] = $encryptedtext;
    }
}
```

- Encryption transforms data into an unreadable form using a key, preventing unauthorized parties from deciphering it in case of a breach

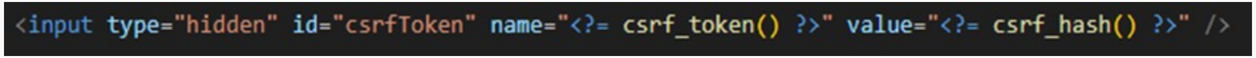
Input Sanitization

- Library provided functions such as sanitizeInput() in the validation_helper.php file to prevention against XSS and SQL injection attacks
- Techniques Implemented
    - Removal of Whitespace, HTML Tags and Special Characters from user inputs
    - Escaping of special characters
    - Stripping HTML Tags
    - Removal of special Characters from user inputs

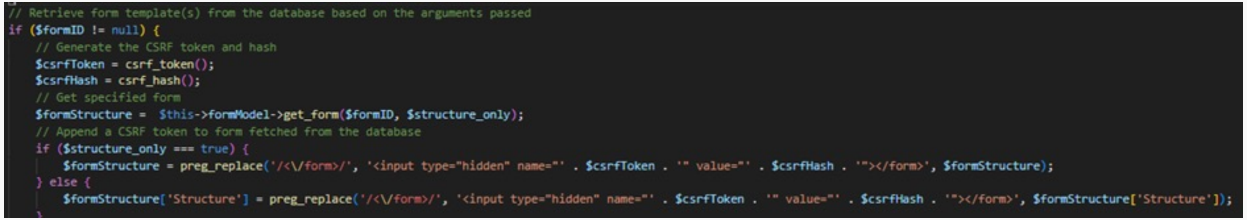


```php
// Function to sanitize input
function sanitizeInput($input)
{
    // Trim whitespace from the input
    $sanitizedInput = trim($input);
    // Remove HTML tags from the input
    $sanitizedInput = strip_tags($sanitizedInput);
    // Sanitize special characters
    $sanitizedInput = htmlspecialchars($sanitizedInput, ENT_QUOTES, 'UTF-8');
    // Return the sanitized input
    return $sanitizedInput;
}
```

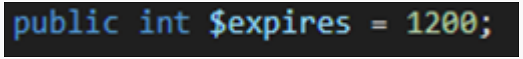
CSRF Token

- Inserting CSRF hidden field into HTML forms to perform verification by the server
  - 

```
<input type="hidden" id="csrfToken" name="<?= csrf_token() ?>" value="<?= csrf_hash() ?>" />
```

- Generation of CSRF token and hash value using whenever getForm() function is called to retrieve form template from database (automatically implemented by library)
  - 

```
// Retrieve form template(s) from the database based on the arguments passed
if ($formID != null) {
    // Generate the CSRF token and hash
    $csrfToken = csrf_token();
    $csrfHash = csrf_hash();
    // Get specified form
    $formStructure =  $this->formModel->get_form($formID, $structure_only);
    // Append a CSRF token to form fetched from the database
    if ($structure_only === true) {
        $formStructure = preg_replace('/<\/form>/', '<input type="hidden" name="' . $csrfToken . '" value="' . $csrfHash . '"></form>', $formStructure);
    } else {
        $formStructure['Structure'] = preg_replace('/<\/form>/', '<input type="hidden" name="' . $csrfToken . '" value="' . $csrfHash . '"></form>', $formStructure['Structure']);
```

- Enhance the security further by reducing the CSRF lifespan in the setting configuration
  - 

```
public int $expires = 1200;
```

04

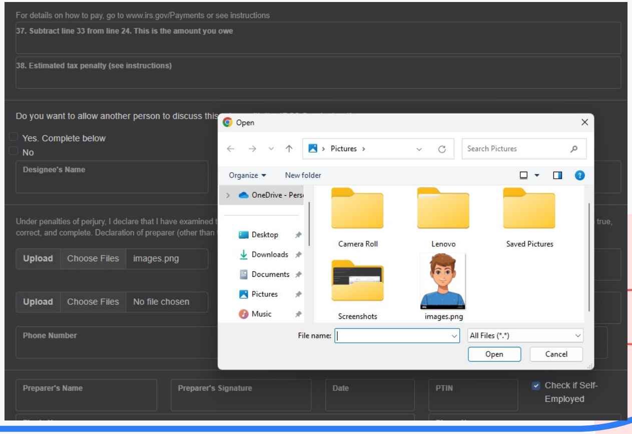
Advanced Features

File Upload

- Enable file upload for taking in signature for F1040 form using the validateUploadedFile() function

- File validation process
  - Check if there is any file uploaded
  - Run the pre-defined rules for validation to check for file size and extension
  - Display error message if validation fail otherwise upload the file into the server with unique numeric names
  - File path will be saved together with other input fields together into database

- Ensure data integrity and consistency by ensuring the data collected meets a specific criteria



For details on how to pay, go to www.irs.gov/Payments or see instructions

37. Subtract line 33 from line 24. This is the amount you owe

38. Estimated tax penalty (see instructions)

Do you want to allow another person to discuss this

□ Yes. Complete below

□ No

Designee's Name

Under penalties of perjury, I declare that I have examined t ... true, correct, and complete. Declaration of preparer (other than

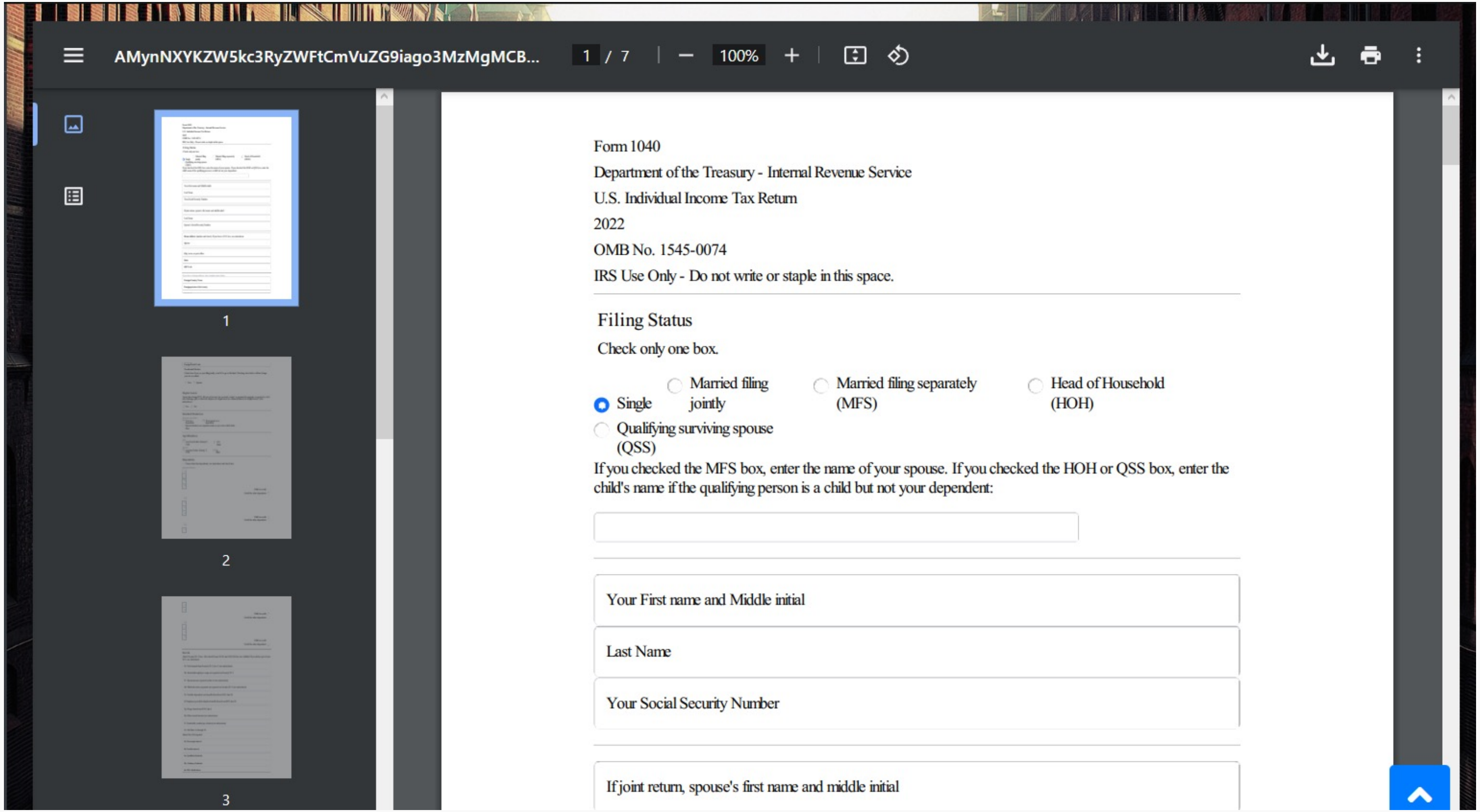
Upload | Choose Files | images.png

Upload | Choose Files | No file chosen

Phone Number

Preparer's Name | Preparer's Signature | Date | PTIN | ☑ Check if Self-Employed

PDF Viewer



Form 1040
Department of the Treasury - Internal Revenue Service
U.S. Individual Income Tax Return
2022
OMB No. 1545-0074
IRS Use Only - Do not write or staple in this space.

Filing Status

Check only one box.

● Single ○ Married filing jointly ○ Married filing separately (MFS) ○ Head of Household (HOH)
○ Qualifying surviving spouse (QSS)

If you checked the MFS box, enter the name of your spouse. If you checked the HOH or QSS box, enter the child's name if the qualifying person is a child but not your dependent:

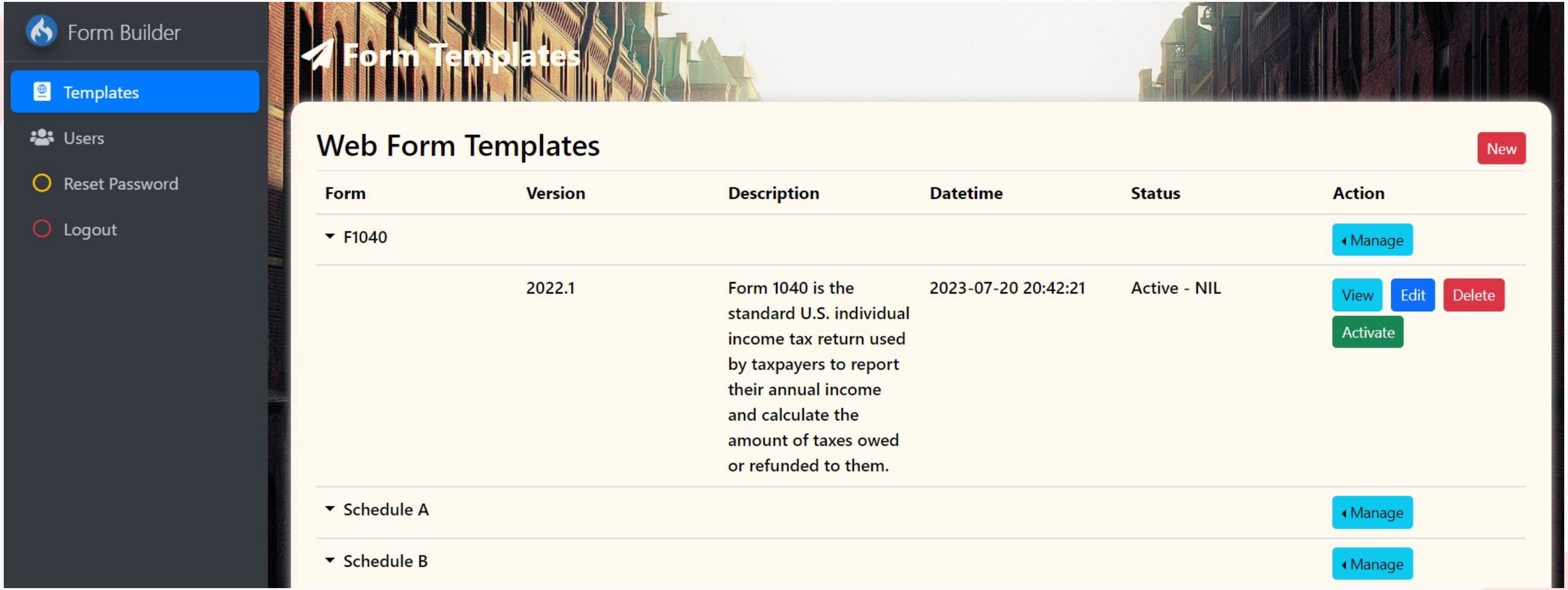
Your First name and Middle initial

Last Name

Your Social Security Number

If joint return, spouse's first name and middle initial

Dashboarding – Form Templates

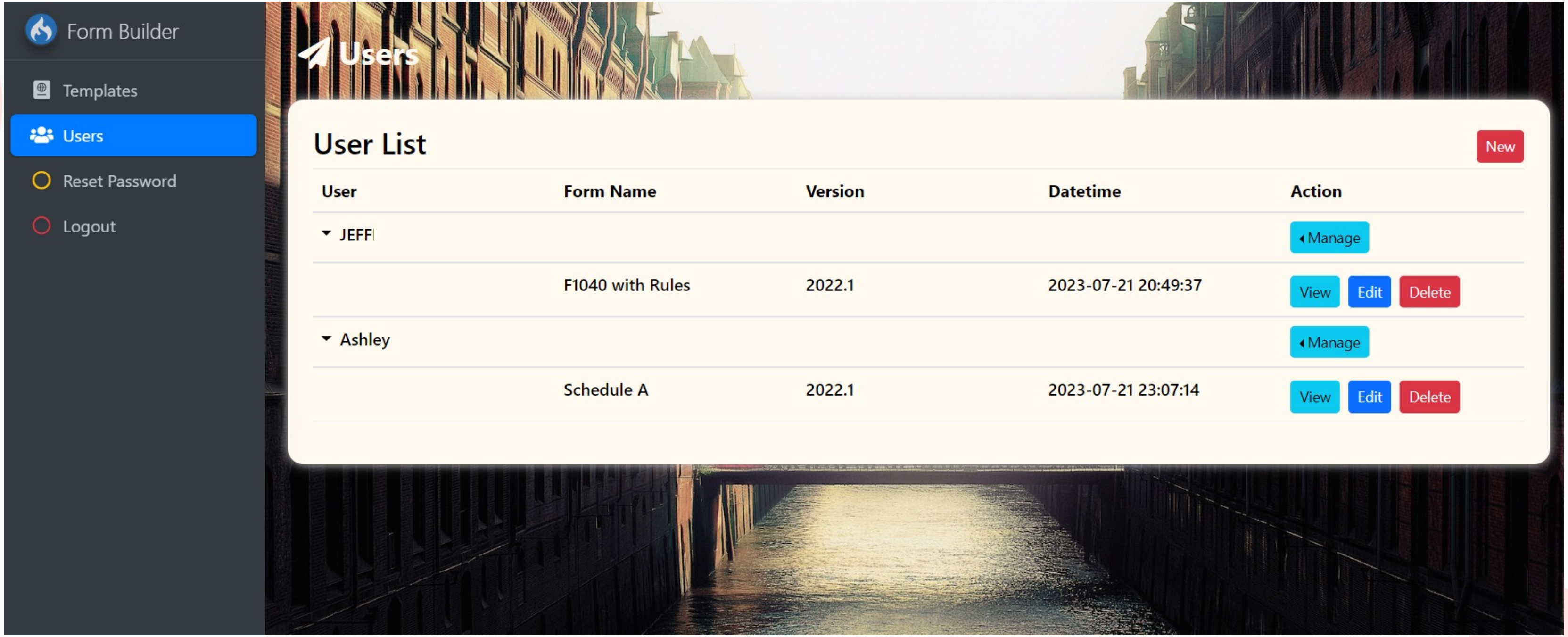


Web Form Templates

New

Form	Version	Description	Datetime	Status	Action
▾ F1040					◂ Manage
	2022.1	Form 1040 is the standard U.S. individual income tax return used by taxpayers to report their annual income and calculate the amount of taxes owed or refunded to them.	2023-07-20 20:42:21	Active - NIL	View Edit Delete Activate
▾ Schedule A					◂ Manage
▾ Schedule B					◂ Manage

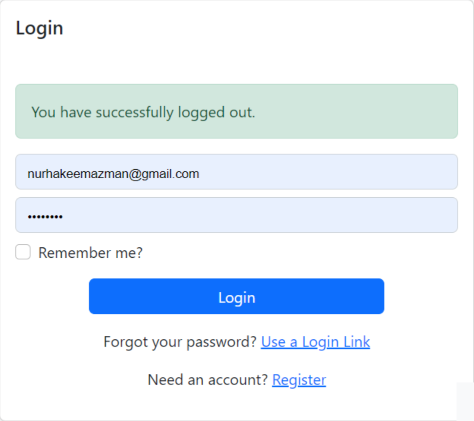
Dashboarding – Users



User List

User	Form Name	Version	Datetime	Action
▾ JEFF				◂Manage
	F1040 with Rules	2022.1	2023-07-21 20:49:37	View Edit Delete
▾ Ashley				◂Manage
	Schedule A	2022.1	2023-07-21 23:07:14	View Edit Delete

CI4 Shield



Login

You have successfully logged out.

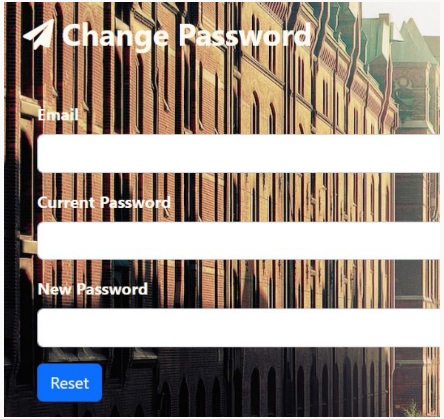
nurhakeemazman@gmail.com

••••••••

□ Remember me?

Forgot your password? Use a Login Link

Need an account? Register

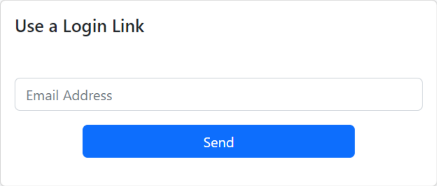


Change Password

Email

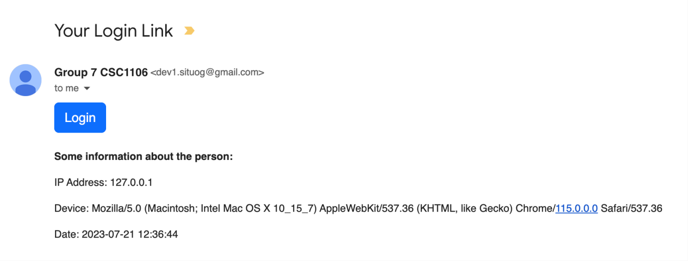
Current Password

New Password



Use a Login Link

Email Address



Your Login Link

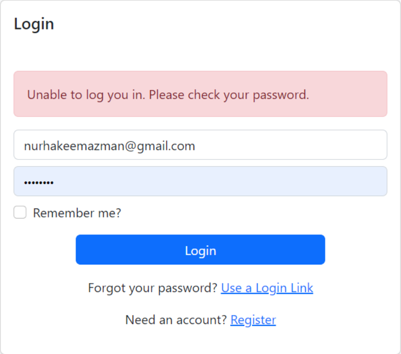
Group 7 CSC1106 <dev1.situog@gmail.com>
to me

Some information about the person:

IP Address: 127.0.0.1

Device: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.0.0 Safari/537.36

Date: 2023-07-21 12:36:44



Login

Unable to log you in. Please check your password.

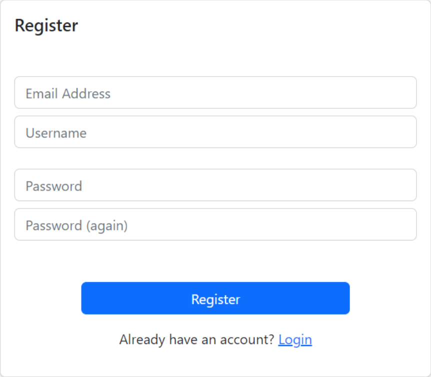
nurhakeemazman@gmail.com

••••••••

□ Remember me?

Forgot your password? Use a Login Link

Need an account? Register



Register

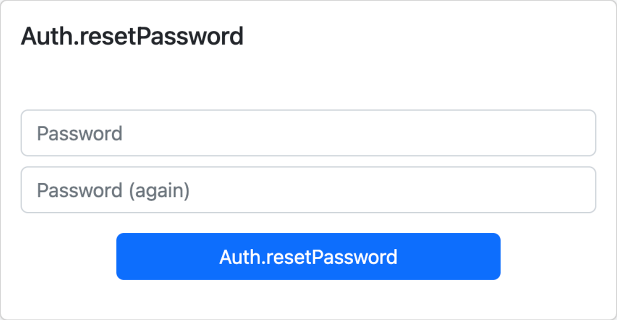
Email Address

Username

Password

Password (again)

Already have an account? Login

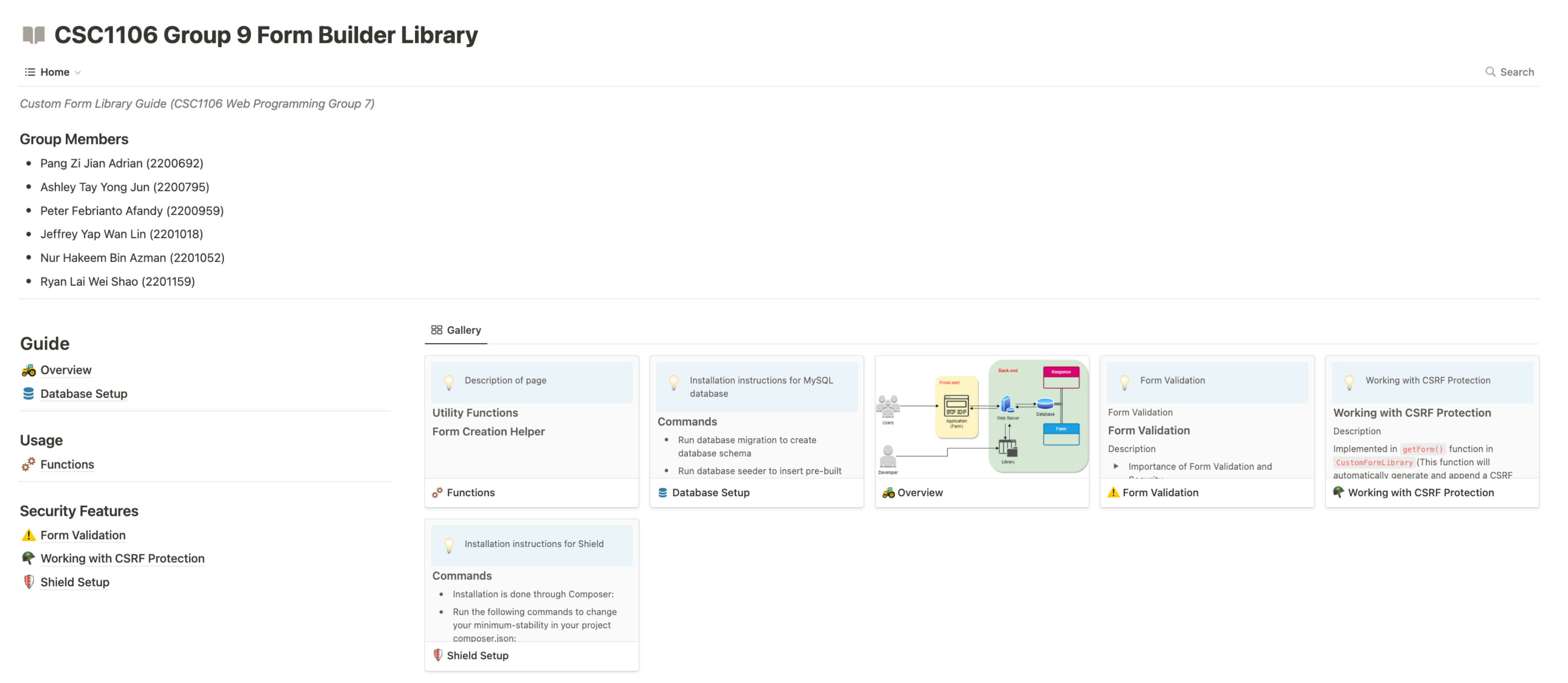


Auth.resetPassword

Password

Password (again)

Notion Documentation



CSC1106 Group 9 Form Builder Library

Custom Form Library Guide (CSC1106 Web Programming Group 7)

Group Members

- Pang Zi Jian Adrian (2200692)
- Ashley Tay Yong Jun (2200795)
- Peter Febrianto Afandy (2200959)
- Jeffrey Yap Wan Lin (2201018)
- Nur Hakeem Bin Azman (2201052)
- Ryan Lai Wei Shao (2201159)

Guide

- Overview
- Database Setup

Usage

- Functions

Security Features

- ⚠ Form Validation
- Working with CSRF Protection
- Shield Setup

Description of page

Utility Functions

Form Creation Helper

Functions

Installation instructions for MySQL database

Commands

- Run database migration to create database schema
- Run database seeder to insert pre-built

Database Setup

Overview

Form Validation

Form Validation

Form Validation

Description

▸ Importance of Form Validation and

⚠ Form Validation

Working with CSRF Protection

Working with CSRF Protection

Description

Implemented in `getForm()` function in `CustomFormLibrary` (This function will automatically generate and append a CSRF

Working with CSRF Protection

Installation instructions for Shield

Commands

- Installation is done through Composer:
- Run the following commands to change your minimum-stability in your project composer.json:

Shield Setup

05

Demo

End

Thank you