Sprint 2

Group Members
161109 Hammad Asif
161112Hamza Aslam
161117 Ahmad Razi
161135 Bilal Fareed

Group Number 3
Date: 19-April-2019

Test Cases of Life Scribble

Here are test cases of Life Scribble. Although there would be thousands of test cases for LifeScribble but we are limiting the tests to common features like - profile setting, comments and post in timeline etc. We will leave features like creating page creation, groups creation, events etc.

Test Cases for Registration Page

1. Verify that all the specified fields are present on the registration page
2. Verify that the required/mandatory fields are marked with * against the field
3. Verify that for better user interface dropdowns, radio buttons and checkboxes etc fields are displayed wherever possible instead of just textboxes
4. Verify the page has both submit and cancel/reset buttons at the end
5. Verify that clicking submit button after entering all the required fields, submits the data to the server
6. Verify that clicking cancel/reset button after entering all the required fields, cancels the submit request and resets all the fields
7. Verify that whenever possible validation should take place at client side
8. Verify that not filling the mandatory fields and clicking submit button will lead to validation error
9. Verify that not filling the optional fields and clicking submit button will still send data to server without any validation error
10. Check the upper limit of the textboxes
11. Check validation on date and email fields (only valid dates and valid email Ids should be allowed
12. Check validation on numeric fields by entering alphabets and special characters
13. Verify that leading and trailing spaces are trimmed
14. Verify that entering blank spaces on mandatory fields lead to validation error
15. Verify that after making a request to the server and then sending the same request again with the same unique key will lead to server side validation error

Test Casesfor Login Page

1. Verify that the login screen is having option to enter username and password with submit button and option of forgot password
2. Verify that user is able to login with valid username and password
3. Verify that user is not able to login with invalid username and password
4. Verify that validation message gets displayed in case user leaves username or password field as blank
5. Verify that validation message is displayed in case user exceeds the character limit of the user name and password fields
6. Verify that there is reset button to clear the field's text
7. Verify if there is checkbox with label "remember password" in the login page
8. Verify that the password is in encrypted form when entered
9. Verify that there is limit on the total number of unsuccessful attempts
10. For security point of view, in case of in correct credentials user is displayed the message like "incorrect username or password" instead of exact message pointing at the field that is incorrect. As message like "incorrect username" will aid hacker in bruteforcing the fields one by one
11. Verify the timeout of the login session
12. Verify if the password can be copy-pasted or not
13. Verify that once logged in, clicking back button doesn't logout user
14. Verify if SQL Injection attacks works on login page
15. Verify if XSS vulnerability work on login page

Life Scribble User Timeline Cases

1. Verify that user can set profile pic uploaded from his or her computer.
2. Verify that user can set profile pic uploaded from mobile.
3. Verify that uer can set profile pic from photos present on his facbook account's photo section.
4. Verify that user can set profile from webcam or mobile camera.
5. Verify that user can set cover pic uploaded from his or her computer.
6. Verify that user can set cover pic uploaded from mobile.
7. Verify that user can set cover pic from photos present on his facbook account's photo section.
8. Verify that user can set cover from webcam or mobile camera.
9. Verify that uploading image of unsupported type should lead to error message.
10. Verify that uploading image of size exceeding maximum allowed size should lead to error message.
11. Verify that uploading image of size less than the allowed minimum size should lead to error message.
12. Verify that uploading image of larger dimension than permitted should lead to error message.
13. Verify that uploading image of smaller dimension than permitted should lead to error message.
14. Verify that change in profile pic should get reflected in each post/comment of the user's timeline.
15. Verify that user can add/edit their account information displayed to other users.
16. Verify that users can post text in their timeline and the same gets displyed to their friends.
17. Verify that users can post images in their timeline and the same gets displyed to their friends.
18. Verify that users can post links with or without preview in their timeline and the same gets displayed to their friends.
19. Verify that user can tag friends in their posts.
20. Verify that users can see the all the post in their timeline.
21. Verify that users can see comments, likes and reactions in the posts present in their timeline.
22. Verify that users can post comments, like and react to the posts present in their timeline.

Friend and their Timelines Cases

1. Verify that user can search for friends in Life Scribble's 'Find friends' search functionality.
2. Verify that user can send friend request to any user by visting their page.
3. Verify that user can navigate through their Friend's friend and send friend request to them.
4. Verify that user can approve or decline received friend request.
5. Verify that user can unfriend any existing friend.
6. Verify that users can see timeline of their friends.
7. Verify that users can post text in their friend's timeline.
8. Verify that users can post images in their timeline and the same gets displyed to their friends.
9. Verify that users can post links with or without preview in their friend's timeline.
10. Verify that user can tag friends in their posts on friend's timeline.
11. Verify that users can see the all the post in their friend's timeline.
12. Verify that users can see comments, likes and reactions in the posts present in their friend's timeline.
13. Verify that users can post comments, like and react to the posts present in their friend's timeline.

Life Scribble Notification Test Scenarios

1. Verify that users receive different notifications on Life Scribble 'Notifications' icon.
2. Verify that users receive different notifications on email or cell phone based on the settings chosen when not logged in to Life Scribble.
3. Verify that users receive notifcation when their friend request gets approved.
4. Verify that users receive notifcation when they get friend request.
5. Verify that users recieve notification when they get tagged by someone on posts or comments.
6. Verify that users receive notification when they get comments, like or reactions on their posts.
7. Verfiy that users receive notifcation when someone posts on their timeline.

Here is one simple test for inputs validation

```
<?php


class testSimple_add extends PHPUnit_Framework_TestCase {


  public function test_form_validation($formInput, $formInputValue)
{
   $response = $this->json('POST', route('client.store', [
      $formInput => $formInputValue,
   ]));


   $response->assertStatus(422);
   $response->assertJsonValidationErrors($formInput);
}


public function nameInputValidation()
{
```

```
    return [
        'Name is required' => ['name', ''],
    ];
}

public function emailInputValidation()
{
    return [
        'Email is required' => ['email', ''],
        'Email must be valid' => ['email', 'not-an-email'],
    ];
}

}

?>
```

Test for email validation

```
<?php

declare(strict_types=1);
final class Email
{
   private $email;
   private function __construct(string $email)
   {
```

```
        $this->ensureIsValidEmail($email);
        $this->email = $email;
    }
    public static function fromString(string $email): self
    {
        return new self($email);
    }
    public function __toString(): string
    {
        return $this->email;
    }
    public function ensureIsValidEmail(string $email): void
    {
        if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
            throw new InvalidArgumentException(
                sprintf(
                    '"%s" is not a valid email address',
                    $email
                )
            );
        }
    }
}

?>
```