Government of the People's Republic of Bangladesh

Access to Information (a2i) Programme

Prime Minister's Office

"Office Information and Service Framework (OISF)"

Documentation of Single Sign on (SSO) Integration

28 March 2018

Submitted By:

REVE Systems

Table of Contents

What is Single-Sign-On (SSO)?

Single sign-on (SSO) is a property of access control where a user logs in with a single ID and password to gain access to a connected system or systems without using different usernames or passwords. It is best to refer to systems requiring authentication for each application but using the same credentials from a directory server as Directory Server Authentication and systems where a single authentication provides access to multiple applications by passing the authentication token seamlessly to configured applications as Single Sign-On. As different applications and resources support different authentication mechanisms, single sign-on must internally store the credentials used for initial authentication and translate them to the credentials required for the different mechanisms.

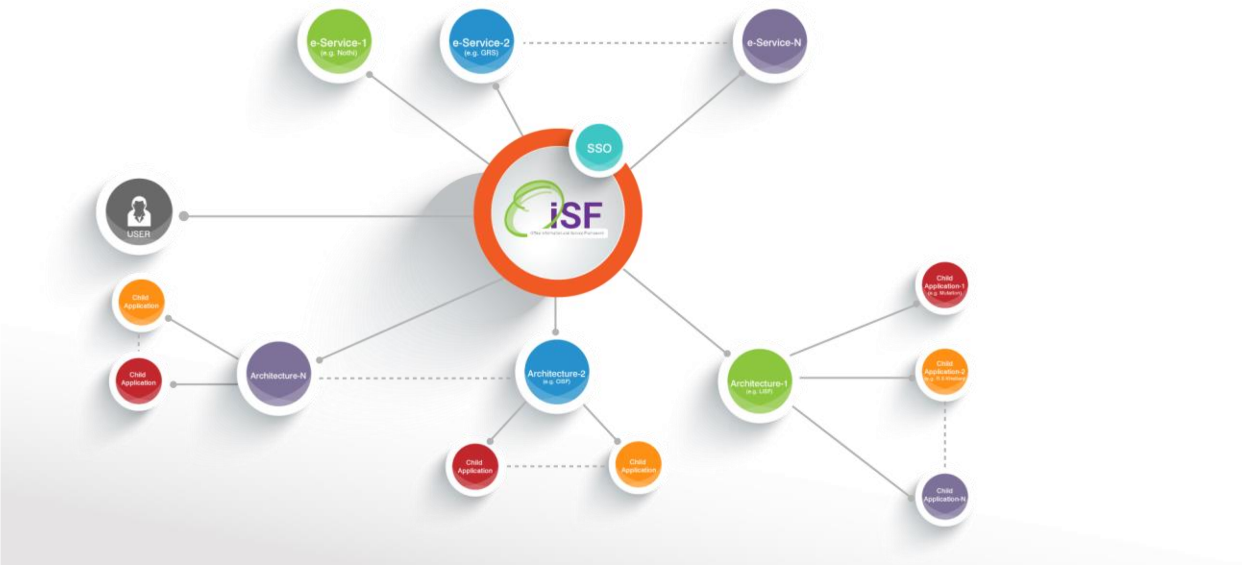


Fig: Parent child Architecture Communication

Among different protocols of SSO, we will use OpenID Connect (OIDC) is an authentication layer on top of OAuth 2.0, an authorization framework. The standard is controlled by the OpenID Foundation. OpenID Connect is a simple identity layer on top of the OAuth 2.0 protocol, which allows computing clients to verify the identity of an end-user based on the authentication performed by an authorization server, as well as to obtain basic profile information about the end-user in an interoperable and REST-like manner. In technical terms, OpenID Connect specifies a RESTful HTTP API, using JSON as a data format. Organizations that have started to use OpenID Connect includes Amazon, Google, IBM, Microsoft, Salesforce, VMWare etc.

Benefits of Single-Sign-On (SSO)?

Benefits of using Single-Sign-On (SSO) includes

1. The responsibility of authentication is shifted from application to identity server. Particular application does not need to implement authentication again.
2. Reduce password fatigue from different user name and password combinations.
3. Reduce time spent re-entering passwords for the same identity.
4. Reduce IT costs due to lower number of IT help desk calls about passwords
5. Removing redundancy as users will only be in one place.

SSO shares centralized authentication servers that all other applications and systems use for authentication purposes and combines this with techniques to ensure that users do not have to actively enter their credentials more than once.

Basic flow of SSO adapted in OISF:

In OISF, we have implemented two (2) use cases of single-sign-on (SSO). In Use case-1, user will put http//:doptor.gov.bd in browser, it will redirect to identity server (central authentication system of government employee), and user will put username and password. After successfully authentication, user will redirect to landing page of OISF. Here user will see configurable dashboard with all permitted s/w systems. By clicking on s/w systems icon, user will able go landing page of corresponding software system.

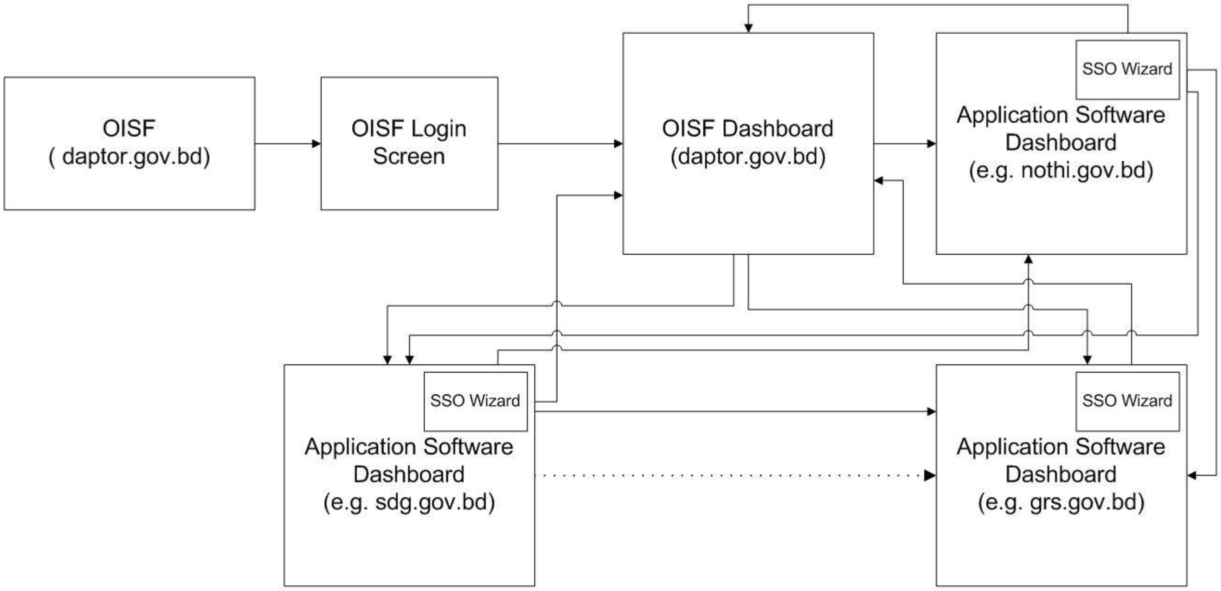

OISF
(daptor.gov.bd)
OISF Login Screen
OISF Dashboard (daptor.gov.bd)
SSO Wizard
Application Software Dashboard (e.g. nothi.gov.bd)
SSO Wizard
Application Software Dashboard (e.g. sdg.gov.bd)
SSO Wizard
Application Software Dashboard (e.g. grs.gov.bd)

Fig: Single-Sign-On (SSO) Use Case-1

In Use case-2, user will put URL of application software (e.g. http//:nothi.gov.bd) in browser, it will redirect to identity server (central authentication system of government employee), and user will put username and password. After successfully authentication, user will redirect to landing page of corresponding software application (e.g. dashboard of nothi). Here user will see wizard with all permitted s/w systems. By clicking on s/w systems icon in wizard, user will able go landing/dashboard page of corresponding software system.

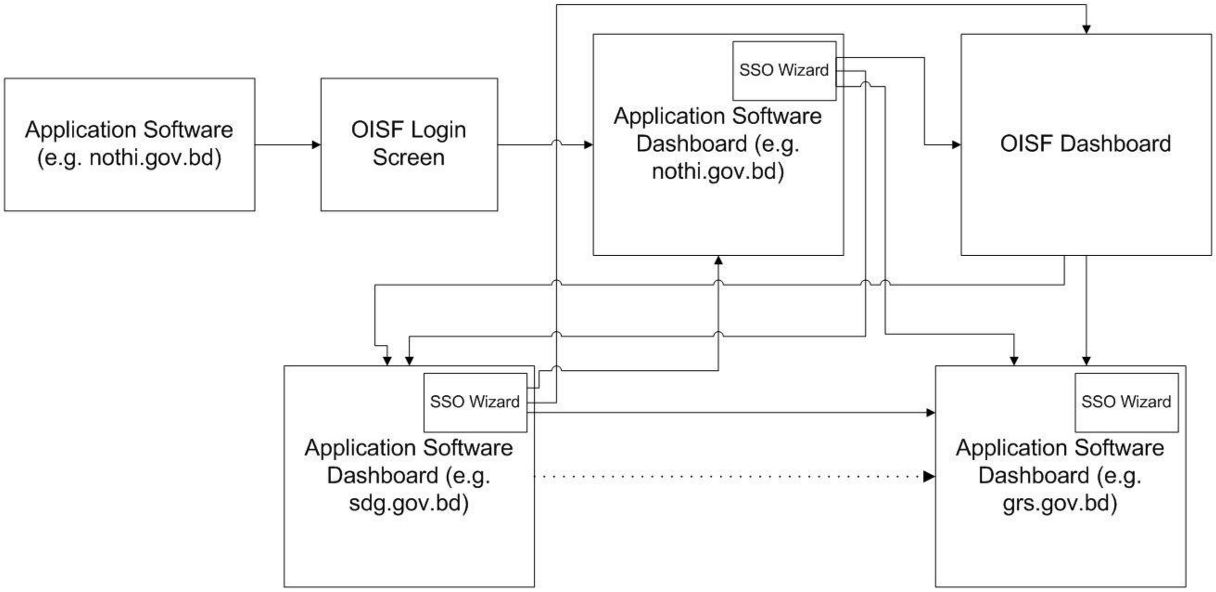

Application Software (e.g. nothi.gov.bd)
OISF Login Screen
SSO Wizard
Application Software Dashboard (e.g. nothi.gov.bd)
OISF Dashboard
SSO Wizard
Application Software Dashboard (e.g. sdg.gov.bd)
SSO Wizard
Application Software Dashboard (e.g. grs.gov.bd)

Fig: Single-Sign-On (SSO) Use Case-2

A snapshot of wizard of single-sign-on(SSO) is given below

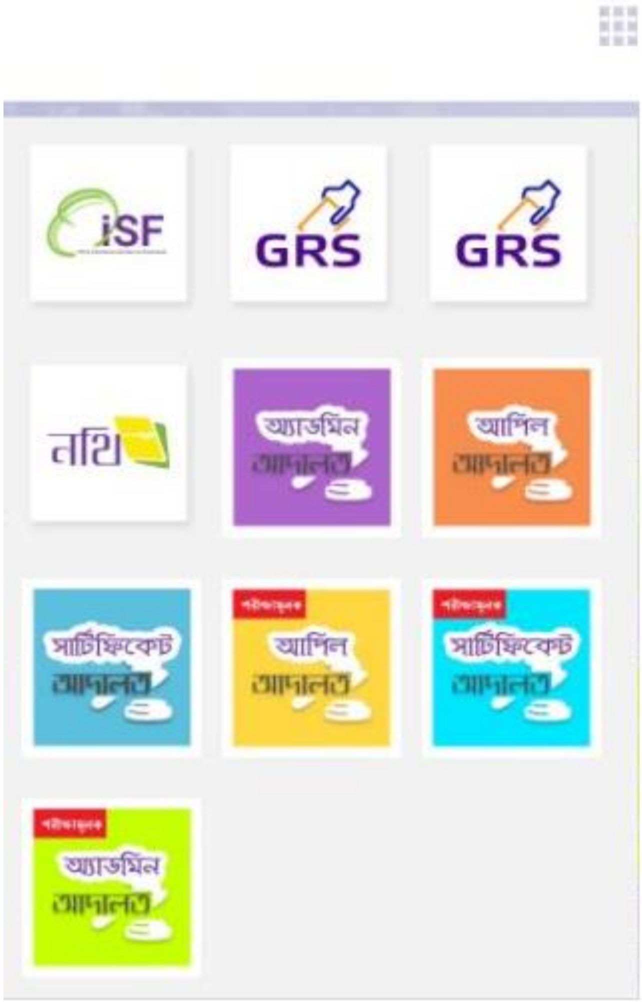


Fig: Snapshot of Wizard of Single-Sign-On(SSO)

An identity provider (abbreviated IdP) offers user authentication as a service. An identity provider is “a trusted provider that lets you use single sign-on (SSO) to access other software systems.

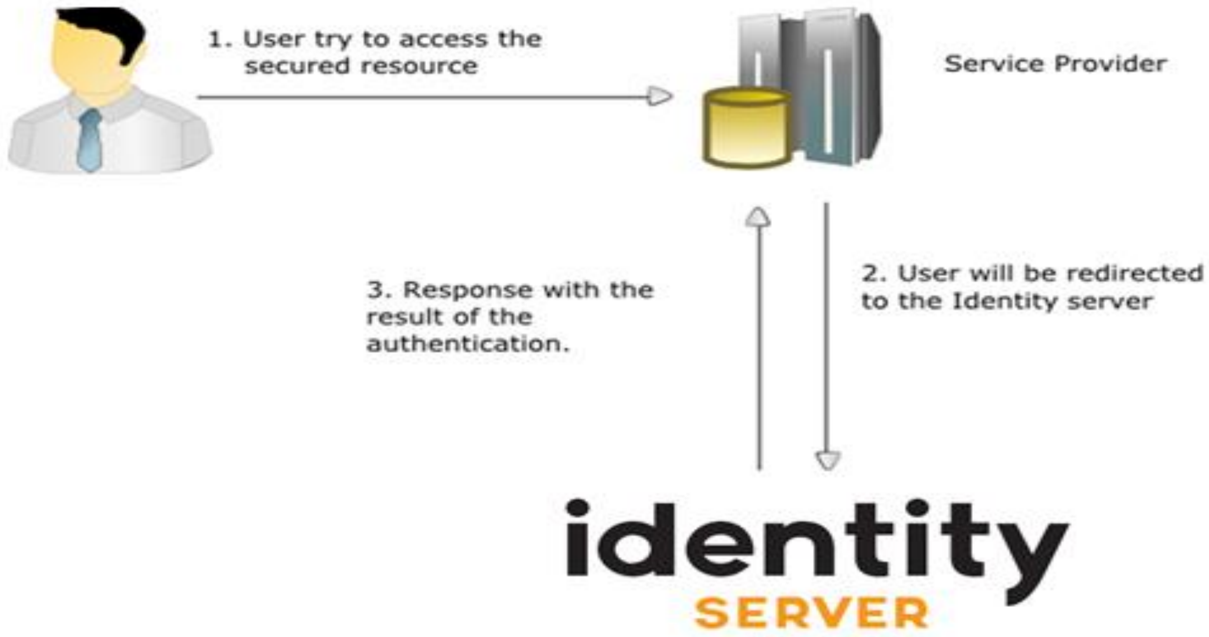
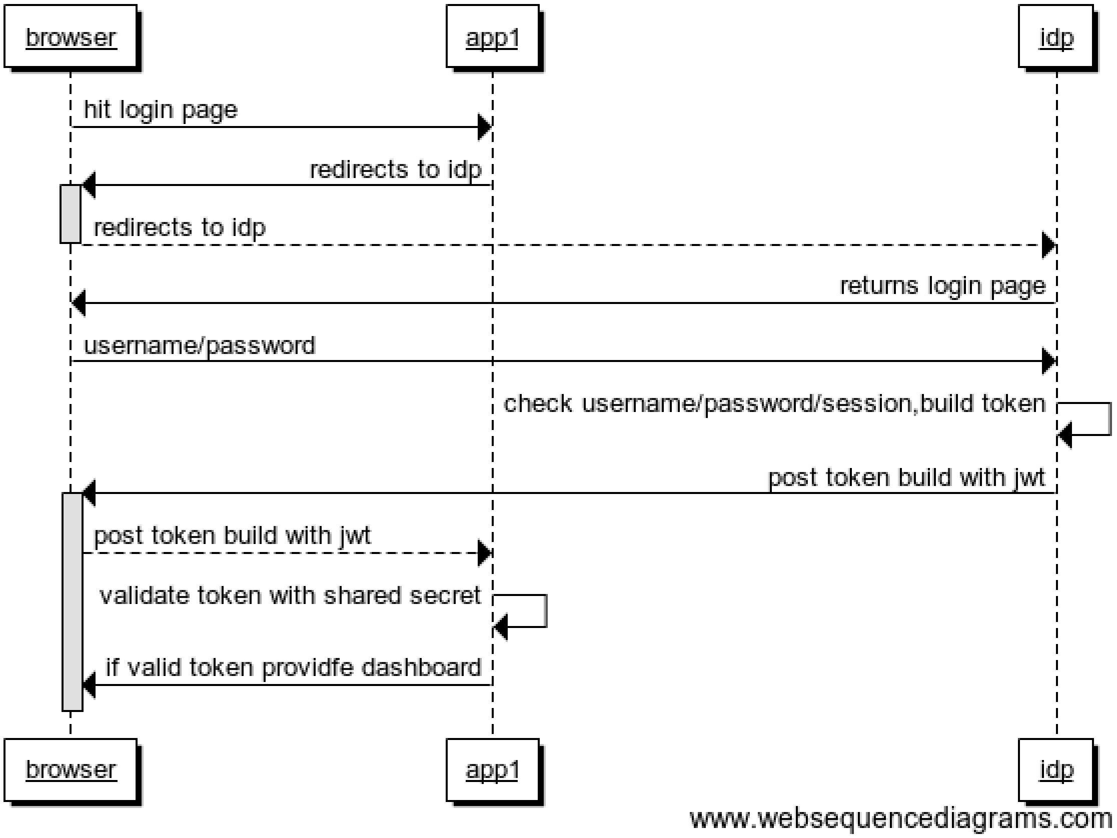

1. User try to access the secured resource
Service Provider
3. Response with the result of the authentication.
2. User will be redirected to the Identity server
identity
SERVER

Fig: Identity Server of Single-Sign-On (SSO)

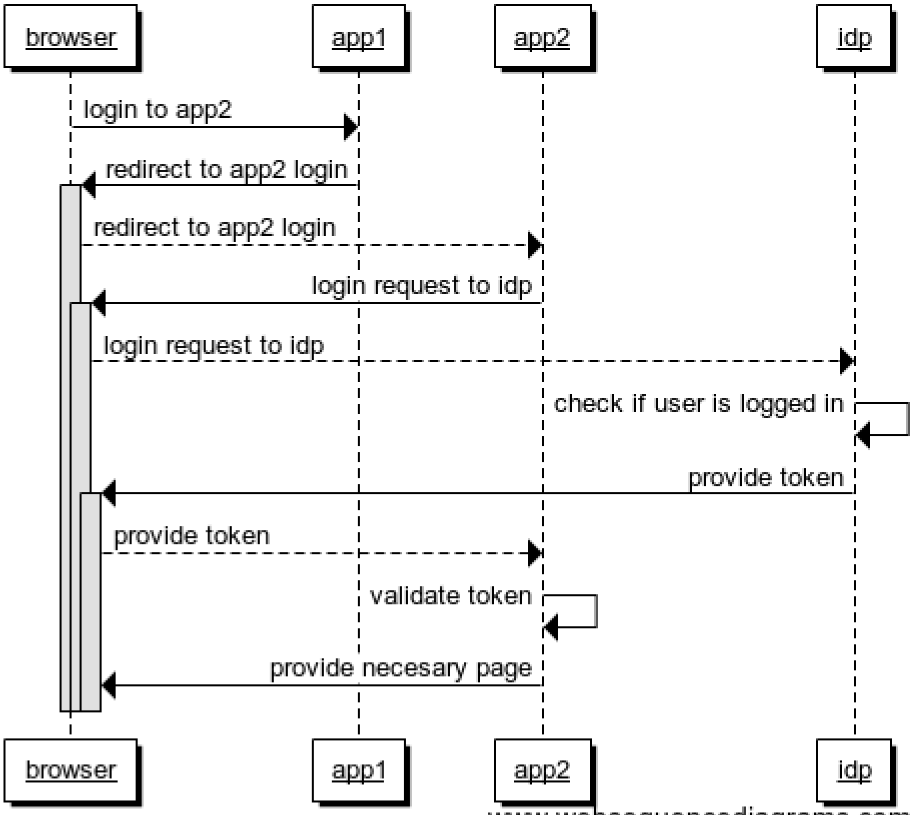
Steps of Single-Sign-On (SSO)

1. Users hit a particular app. For example : nothi.gov.bd
2. User is redirected to the identity server. idp.doptor.gov.bd
3. User enters username/password
4. idp authenticates user
5. idp sends id token to the app with proper authentication.
6. app returns dashboard to user

Single Sign On


browser
app1
idp
hit login page
redirects to idp
redirects to idp
returns login page
username/password
check username/password/session,build token
post token build with jwt
post token build with jwt
validate token with shared secret
if valid token providfe dashboard
browser
app1
idp
www.websequencediagrams.com

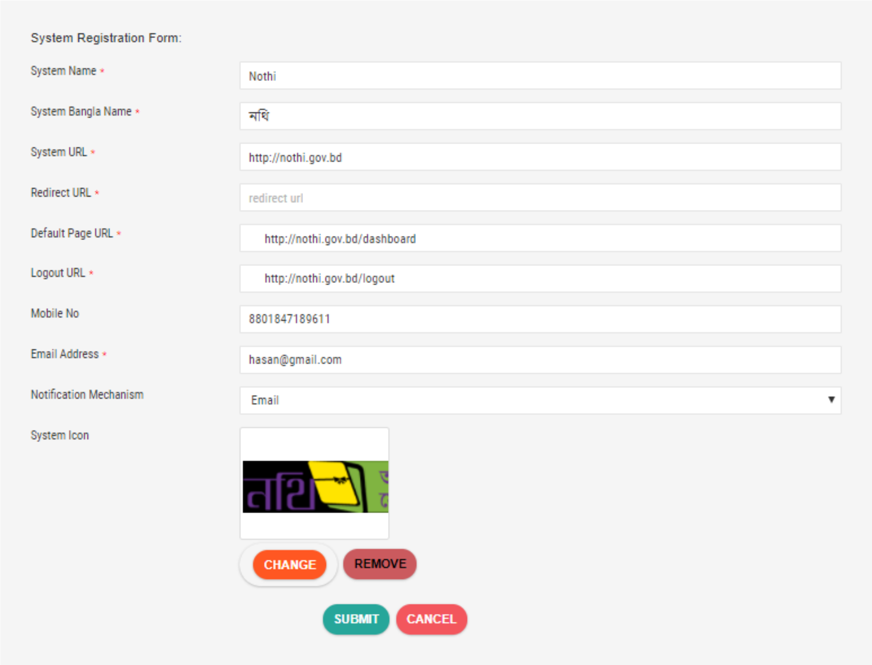
App to App Login



www.websequencediagrams.com

Software Systems Registration Steps

1. Request an admin to register the application or register it from portal
2. Provide the following for registration :
    a. Application url
    b. Application post back url
    c. Application landing page
    d. logo (190px*190px)
    e. Application name (eng bang etc), email, mobile no



System Registration Form:

System Name *	Nothi
System Bangla Name *	নথি
System URL *	http://nothi.gov.bd
Redirect URL *	redirect url
Default Page URL *	http://nothi.gov.bd/dashboard
Logout URL *	http://nothi.gov.bd/logout
Mobile No	8801847189611
Email Address *	hasan@gmail.com
Notification Mechanism	Email
System Icon	

3. SSO library will be provided through email or by an admin
4. Use the provided library to write necessary functions in controller or other places.
5. Add widget by adding javascript and css library

SSO Library structure

1. SSO folder contains all the necessary classes required for SSO integration.
2. Please keep the SSO folder inside App\Model namespace
3. Following classes are important :

a. **SSOConstants**

This class contains following important information :
APP_ID : application id
SHARED_SECRET : application secret/key
LANDING_PAGE_URI : idp will post back to this uri. So, after login, application can load this uri.

b. **AppLoginRequest**

This class will contain necessary function to build request to idp

```
$appLoginRequest = new AppLoginRequest();
$appLoginRequest->setLandingPageUrl($landingPageUrl);
$requestUrl = $appLoginRequest->buildRequest();
```

c. **AppLoginResponse, AppLoginResponseDTO**

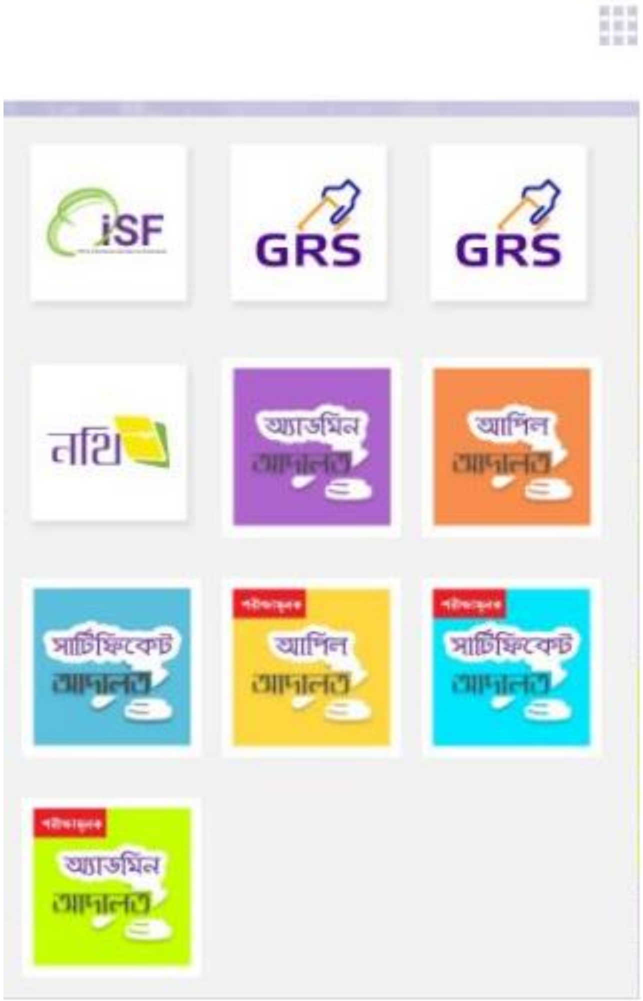
This class will contain necessary functions for parsing idp response:

```
$appLoginResponse = new AppLoginResponse();
$response = $appLoginResponse->parseResponse($token,$nonce);
        // after parsing, we will get response type of class
AppLoginResponseDTO
        //$response contains all necessary user information.`
        //var_dump($response->getUserName());

        //var_dump($response->getDesignation());
        //var_dump("please provide dashboard");
        //var_dump($response->getUserName());
        //var_dump($response->getEmployeeRecordId());
        //var_dump($response->getOfficeId());
        //var_dump($response->getDesignation());
        //var_dump($response->getOfficeUnitId());
        //var_dump($response->getInchargeLabel());
        //var_dump($response->getOfficeUnitOrganogramId());
        //var_dump($response->getOfficeNameEng());
        //var_dump($response->getOfficeNameBng());
        //var_dump($response->getOfficeMinistryId());
        //var_dump($response->getOfficeMinistryNameEng());
        //var_dump($response->getOfficeMinistryNameBng());
        //var_dump($response->getUnitNamEng());
        //var_dump($response->getUnitNameBng());
        //var_dump($response->getLandingPageUrl());
```

Widget Library:

To switch among applications we will use the following widget :



We can easily add the widget in our project by adding the following library :

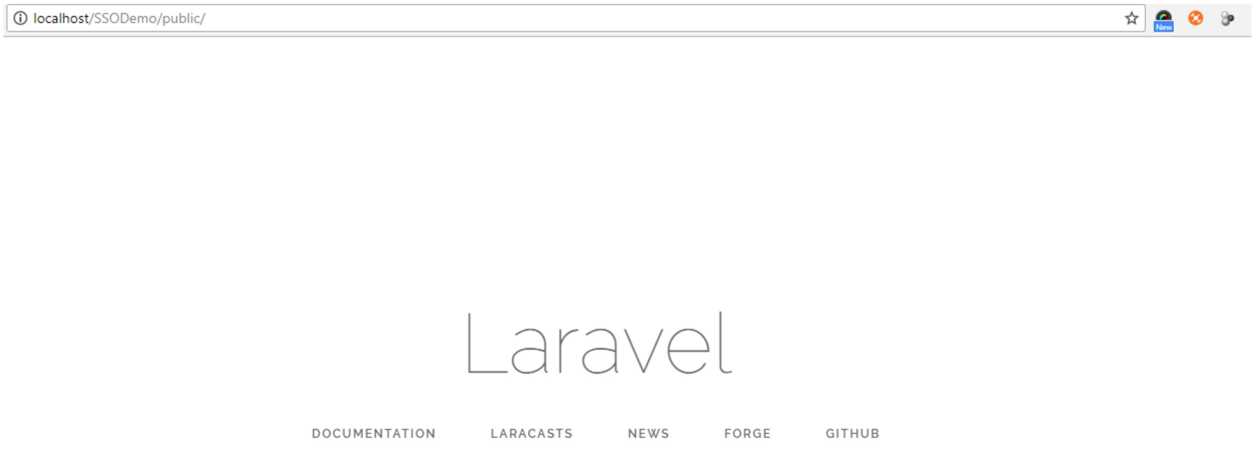
```
<link rel="stylesheet" type="text/css" href="http://103.48.18.28/sso/lib/style.css"/>
<script type="text/javascript" src="http://103.48.18.28/sso/lib/script.js"></script>
```

As we have got basic knowledge of sso, now lets integrate our sso with some popular frameworks :

Integration with common frameworks :

Laravel Integration :

1. Create a new project suppose SSODemo, in laravel with the following composer command :
   laravel new SSODemo.
1. Please keep the SSO folder in **app/model**;
2. http://localhost/SSODemo/public/ in browser will provide the following page. which indicates laravel is installed properly.



3. Add the following library in composer.json :

```
"require": {
    "firebase/php-jwt": "5.0.0"
},
```

4. Inside **routes\web.php** add the following :

```
Route::get('/', 'SSOLoginController@ssologin');
Route::post('applogin','SSOLoginController@applogin');
```

```
Route::get('ssologout','SSOLoginController@ssologout');
```

5. Inside **App\Http\Middleware\VerifyCsrfToken.php** add the following code :

```
protected $except = [
  'ssologin',
  'applogin’,
  'ssologout'
];
```

6. Inside **App\Http\Controller\SSOLoginController.php** add the following code. SSOLoginController.php is attached. The applogin get function will just redirect to the idp login page. Now lets examine the applogin() post function. If we can reach the green text , then our user is authenticated and please take action(give dashboard/session etc) accordingly. But if we reach the brown then user failed to authenticate and redirect to the idp login page.

**** Example below are for explanation. Please do not copy paste from here. Rather a controller will be attached with library as an example.**

```
<?php

namespace App\Http\Controllers;

use App\User;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Redirect;
use \Firebase\JWT\JWT;

use App\Model\SSO\AppLoginRequest;
use App\Model\SSO\AppLoginResponse;
use App\Model\SSO\AppLogoutRequest;


class SSOLoginController extends Controller
{
        public function ssologin() {
                $appLoginRequest = new AppLoginRequest();
                $requestUrl = $appLoginRequest->buildRequest();
                $nonce = $appLoginRequest->getReqNonce();
                $request->session()->put("nonce",$nonce);
                return Redirect::to($requestUrl);
        }

        public function applogin(Request $request) {
                try{
                        $token = $request->input('token');
                        $nonce = $request->session()->get("nonce");

                        $appLoginResponse = new AppLoginResponse();
        }


            $response = $appLoginResponse->parseResponse($token,$nonce);

            $request->session()->put('username', $response->getUserName());
            $request->session()->forget("nonce");

             //  User is authenticated at this point. Please provide him session.

             // Class AppLoginResponse() will contain all the necessary information.
            //
            //var_dump($response->getUserName());
            //var_dump($response->getDesignation());
            //var_dump("please provide dashboard");
            //var_dump($response->getUserName());
            //var_dump($response->getEmployeeRecordId());
            //var_dump($response->getOfficeId());
            //var_dump($response->getDesignation());
            //var_dump($response->getOfficeUnitId());
            //var_dump($response->getInchargeLabel());
            //var_dump($response->getOfficeUnitOrganogramId());
            //var_dump($response->getOfficeNameEng());
            //var_dump($response->getOfficeNameBng());
            //var_dump($response->getOfficeMinistryId());
            //var_dump($response->getOfficeMinistryNameEng());
            //var_dump($response->getOfficeMinistryNameBng());
            //var_dump($response->getUnitNamEng());
            //var_dump($response->getUnitNameBng());
            //var_dump($response->getLandingPageUrl());
            //  User is authenticated give session/dashboard/default page.
            //  Redirect to $response->getLandingPageUrl();
            return view('widget');
        }
        catch(\Exception $ex){
            // User failed to authenticate.
            // Because of invalid shared secret , nonce etc.
            // Please show an error page.
            // Please do not redirect to login page again.
            // Please do not give users sessions.
            dd($ex);
        }
    }
    public function ssologout(Request $request){
        //Please invalidate your sessions
        $appLogoutRequest = new AppLogoutRequest();
        $requestUrl = $appLogoutRequest->buildRequest();
        return Redirect::to($requestUrl);
    }

}
```

If we have come upto this, then we have successfully completed login to our application with identity provider. We are half done. Now, we have to sso to different applications with widget.

7. **Adding widget :**

Please add the following :

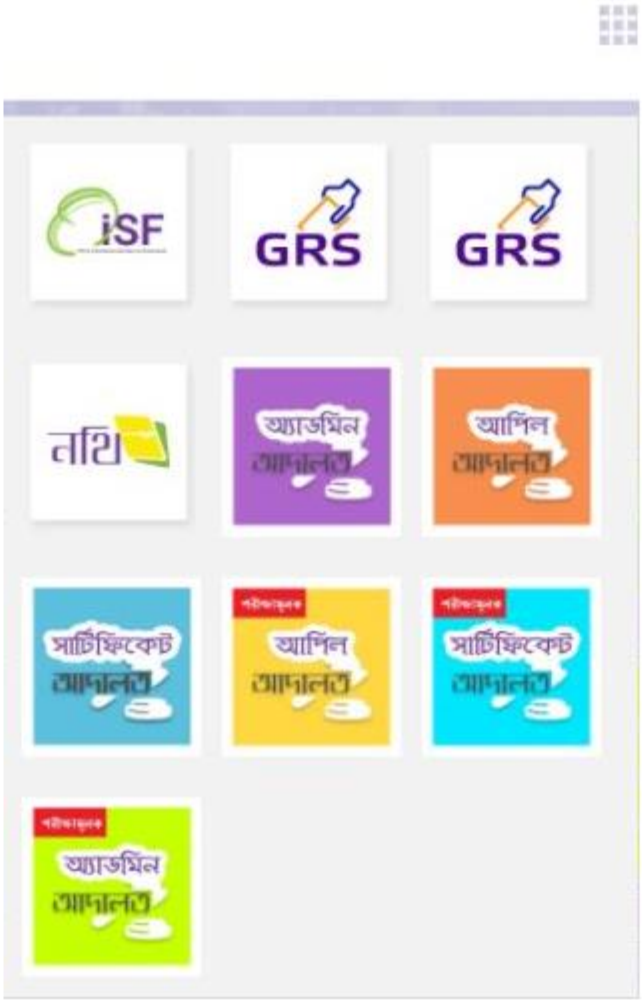
```
<link rel="stylesheet" type="text/css" href="http://103.48.18.28/sso/lib/style.css"/>
<script type="text/javascript" src="http://103.48.18.28/sso/lib/script.js"></script>
```

Initialize widget :

widgetColor: currently support two values : light and dark.
widgetSize: please set the widget size according to your need.

```
<script>
            widget.init(
                        {
                            "widgetColor" : "light",
                            "widgetSize" : "20px"
                         }
            );
  </script>
```

You will find the following widget in your page.



Integration with Cake Php :

1. Register application from portal or request super admin to do registration.
2. Keep sso library in App\Model namespace
3. Add **firebase/php-jwt** with composer
4. Add the following codes in controller

```
public function login()
{
  if($this->request->is('get')){
      $landingPageUrl = $request->input("landing_page_uri");

      $appLoginRequest = new AppLoginRequest();
      $appLoginRequest->setLandingPageUrl($landingPageUrl);
      $requestUrl = $appLoginRequest->buildRequest();
```

```
    $nonce = $appLoginRequest->getReqNonce();

    $session = $this->request->session();
    $session->write('nonce', $nonce);
    return $this->redirect($requestUrl);
}
 if($this->request->is('post') && isset($this->request->data['token']) ){
   $token = $this->request->data['token'];
   $this->request->data['username'] = ' ';
   $this->request->data['password'] = ' ';

    try{
            $session = $this->request->session();
            $nonce = $session->read(“nonce”);

            $appLoginResponse = new AppLoginResponse();
            $response = $appLoginResponse->parseResponse($token,$nonce);

            $session()->write('username', $response->getUserName());
            $session->delete("nonce");

            // User successfully authenticated
            // Please provide him session
            // $response contains all the necessary user information.
            //var_dump($response->getUserName());
            //var_dump($response->getDesignation());
            //var_dump("please provide dashboard");
            //var_dump($response->getUserName());
            //var_dump($response->getEmployeeRecordId());
            //var_dump($response->getOfficeId());
            //var_dump($response->getDesignation());
            //var_dump($response->getOfficeUnitId());
            //var_dump($response->getInchargeLabel());
            //var_dump($response->getOfficeUnitOrganogramId());
            //var_dump($response->getOfficeNameEng());
            //var_dump($response->getOfficeNameBng());
            //var_dump($response->getOfficeMinistryId());
            //var_dump($response->getOfficeMinistryNameEng());
            //var_dump($response->getOfficeMinistryNameBng());
            //var_dump($response->getUnitNamEng());
            //var_dump($response->getUnitNameBng());
            //var_dump($response->getLandingPageUrl());
```

```
                return $this->redirect($response->getLandingPageUrl());
            }
            catch(\Exception $ex){
                // var_dump($ex);
                // User failed to authenticate. May be because of invalid nonce, shared secret.
                // Very important
                // Show a page with error
            }
        }
    }

public function ssologout(Request $request){
            //Please invalidate your sessions
            $appLogoutRequest = new AppLogoutRequest();
            $requestUrl = $appLogoutRequest->buildRequest();
            return $this->redirect($requestUrl);
}
```

5. Add widget library according to need.

   Please add the following :

   ```
   <link rel="stylesheet" type="text/css" href="http://103.48.18.28/sso/lib/style.css"/>
   <script type="text/javascript" src="http://103.48.18.28/sso/lib/script.js"></script>
   ```

   Initialize widget :

   widgetColor: currently support two values : light and dark.

   widgetSize: please set the widget size according to your need.

   ```
   <script>
           widget.init(
           {
                   "widgetColor" : "light",
                   "widgetSize" : "20px"
                 }
                );
            </script>
   ```

Integration with spring boot application:

1. Please register application from portal or request an admin to do it.
2. Please keep the provided library in com.revesoft.sso folder
3. Add maven repository
4. <!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->

   ```
   <dependency>
     <groupId>io.jsonwebtoken</groupId>
     <artifactId>jjwt</artifactId>
     <version>0.7.0</version>
   </dependency>
   ```
5. Add the following code in controller :

```
@Controller
public class SSOLoginController {

  @RequestMapping(value="/", method = RequestMethod.GET)
  public void showIDPLoginPage(HttpServletRequest request,
HttpServletResponse response){
    try{
      String landingPageUrl = request.getParameter("landing_page_url");

      AppLoginRequest appLoginRequest = new AppLoginRequest();
      appLoginRequest.setLandingPageUrl(landingPageUrl);

      String authnRequest = appLoginRequest.buildAuthnRequest();

request.getSession().setAttribute("nonce",appLoginRequest.getNonce());

      response.sendRedirect(authnRequest);
    }
    catch (Exception e) {
      e.printStackTrace();
    }
  }


@RequestMapping(value = "/applogin", method = RequestMethod.POST)
public void jwtSSO(HttpServletRequest request, HttpServletResponse response,
Principal principal) {
  try{
    String token = request.getParameter("token");
    LoginResponse loginResponse = new AppLoginResponse();

loginResponse.setSessionNonce((String)request.getSession().getAttribute("nonce"
));
    SSOResponseDTO ssoResponseDTO = loginResponse.parseResponse(token);
     // User successfully authenticated please provide him
session
     // ssoResponseDTO contains all the necessary user data
    response.sendRedirect(ssoResponseDTO.getLandingPageUrl());

  }
  catch (Exception ex){
     ex.printStackTrace();
     //User failed to authenticate
     // Please do not provide session
```

```
         // Show an error page

    }

       @RequestMapping(value = "/ssologout", method = RequestMethod.GET)
       public ModelAndView ssologout(HttpServletRequest request){
           // Please invalidate your sessions.
           try{
               request.getSession().invalidate();
               AppLogoutRequest appLogoutRequest = new AppLogoutRequest();
               return new ModelAndView("redirect:" +
       appLogoutRequest.buildLogoutRequest());
           }
           catch (Exception e){
               e.printStackTrace();
               return null;
           }
       }


}
```