

**AMPLIAR - OPEN SOURCE SOA BASED  
MIDDLEWARE FRAMEWORK FOR CLASSIFIED  
BASED WEB DEVELOPMENT**

Ishani Hansika Liyanaarachchi

(IT13137342)

Bachelor of Science Special (Hons) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

October 2017

**AMPLIAR - OPEN SOURCE SOA BASED  
MIDDLEWARE FRAMEWORK FOR CLASSIFIED  
BASED WEB DEVELOPMENT**

Ishani Hansika Liyanaarachchi

(IT13137342)

Dissertation submitted in partial fulfillment of the requirements for the Bachelor of  
Science Special (Hons) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

October 2017

## DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also I hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute my dissertation in whole or part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as article or books).

.....

Signature

.....

Date

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Name of supervisor: Mr. Nuwan Kodagoda

.....

Signature of supervisor

.....

Date

## ABSTRACT

Online Classified Advertising has been dominating the advertising industry over traditional newspaper advertising since the emergence of web technologies. The rapid growth of mobile technologies and the massive usage of mobile devices has further increased the boundaries of web content advertising over the years. The adoption of online web technologies for advertising has improved the credibility of the advertisers and also improved the speed of transactions, where the traditional way had the interested buyer having to go through dozens of magazine advertisements before making a sound decision on his / her purchase order. This rapid growth in the classified industry has put many companies seeking to gain a share in this ever-growing market, which has led to a number of new classified websites being popping up every year. This has also brought up huge competition between existing classified websites and newly emerging ones, forcing the organizations to update their web functionalities or even expand to new emerging technologies before the competitive rivals do. The time taken for adapting to the competition has directly affected these web based advertising businesses. Web Developers creating these web contents have not only been able to make changes to their existing websites, but also to adapt to new technologies that is continuously and inevitably changing. This raise in the learning curve for the developers has a direct effect on the time taken to bring the application or changes into the domain. This research project proposes a Middleware Framework that focuses on reducing the learning curve the developer has to face when developing classified based web applications. It will support the developers by having a developer friendly API that would reduce the learning curve of the developer for efficiently building classified based websites with core functionalities. This Framework supports many of the latest technologies needed to build a classified website from the scratch. It also facilitates the developers to integrate new technologies extending the framework, requiring only minor changes. This will highly reduce the development time of classified based web sites which will also directly reduce the cost of bringing the application to market. This will also advance their chances to sustainability in a market where the time to adapt to technological changes has been a factor of survival in internet businesses.

**Keywords** - advertising; classifieds; web development; service oriented architecture; advertising framework

## **ACKNOWLEDGEMENT**

There are some individuals who have given their fullest support throughout our research project. It would not have been possible without the kind support and help of those individuals. We would like to extend our sincere thanks to our project supervisor Mr. Nuwan Kodagoda and our external supervisor Mr. Tharindu Edirisinghe for the kind guidance, inspiration and constant supervision as well as encouragement which support us in completion of this research.

We express our gratitude to Mr. Jayantha Amarakuchi for providing final years students with the knowledge that is necessary for writing and presenting effective project progress reports. Our grateful thank goes to all the personnel of the Sri Lanka Institute of Information Technology who contributed essential information. We are highly indebted to our examiner Mr. Indraka Udayakumara whose contribution in simulating suggestions and knowledge, helped us to coordinate our research.

We would like to express our special thanks to our parents, for their support and encouragement which help us in completion of this research.

The completion of this research could not have been possible without the participation and assistance of so many people whose names may not all be enumerated. Their contributions are sincerely appreciated and gratefully acknowledged.

## TABLE OF CONTENTS

DECLARATION .....	i
ABSTRACT.....	ii
ACKNOWLEDGEMENT .....	iii
TABLE OF CONTENTS .....	iv
LIST OF FIGURES .....	vi
LIST OF TABLES .....	vi
LIST OF ABBREVIATION .....	vii
1 INTRODUCTION .....	1
1.1 Background Context.....	1
1.1.1 Background .....	1
1.1.2 Literature Survey.....	2
1.2 Research Gap.....	3
1.3 Research Problem.....	5
1.3.1 Difficulties in providing third party identity provider support .....	6
1.3.2 Difficulties in identifying suspicious logins .....	6
1.4 Research Objectives .....	6
1.4.1 General Objective.....	6
1.4.2 Specific Objectives.....	7
2 METHODOLOGY.....	8
2.1 Methodology .....	8
2.1.1 Background Study .....	9
2.1.2 Requirement Analysis .....	10
2.1.3 Design .....	11
2.2 Testing and Implementation .....	11
2.2.1 Implementation .....	11
2.2.2 Testing.....	23
2.3 Research Findings .....	27
3 RESULTS AND DISCUSSION .....	29
3.1 Results .....	29
3.2 Discussion .....	32
4 CONCLUSION.....	33

REFERENCES.....	34
APPENDIX A: Use case diagram.....	35
APPENDIX B: Use case scenarios .....	36
APPENDIX C: Class diagram .....	38

## LIST OF FIGURES

Figure 2.1: Work Breakdown Structure .....	9
Figure 2.2: High level diagram of authentication module .....	12
Figure 2.3: High level diagram of OAuth process flow .....	13
Figure 2.4: OAuth process in login with Facebook .....	14
Figure 2.5: High level diagram of extensible federated authentication .....	15
Figure 2.6: Code snippet of authentication abstract class .....	15
Figure 2.7: Facebook connector implementation .....	16
Figure 2.8: Linkedin connector implementation .....	16
Figure 2.9: Google connector implementation .....	17
Figure 2.10: Twitter connector implementation .....	17
Figure 2.11: Authentication configuration file .....	18
Figure 2.12: Code snippet used to calculate the risk score .....	22
Figure 3.1: Login interface .....	30
Figure 3.2: Developer settings interface .....	30
Figure 3.3: OTP verification interface .....	31
Figure 3.4: Security Question verification .....	31

## LIST OF TABLES

Table 1.1: Comparison of similar frameworks .....	5
Table 2.1: Login with Facebook with correct fields .....	24
Table 2.2: Login with twitter using correct fields .....	24
Table 2.3: Login with Linkedin using correct fields .....	25
Table 2.4: Login with google using correct fields .....	25
Table 2.5: Local authentication using previous IP, device, browser, location and date/time .....	26
Table 2.6: Local authentication using different IP address .....	26
Table 2.7: Login with local authentication using different IP and invalid credentials .....	27



## LIST OF ABBREVIATION

Abbreviation	Description
MVC	Model, View, Controller
API	Application Program Interface
SDLC	Software Development Life Cycle
UML	Unified Modeling Language
OAuth	Open Authentication
XML	Extensible Markup Language
OTP	One-Time Password

# **1 INTRODUCTION**

## **1.1 Background Context**

### **1.1.1 Background**

The origin of classified base web development is classified advertising. Classified advertising is a form of advertising which is particularly common in newspapers, periodicals and online. Even today printed classified are exists although the online web format decreases the profitability of those printed media [1]. Online web format of classified service provides the wide range of features comparatively to printed media. Many web development frameworks exist today to provide the development of those features. Authentication is a mandatory feature to a website as it gives authority to access website's features. Modern world uses many type of authentication techniques to identify the user. The "Ampliar" framework also has a module called authentication to provide many features as well as secure authentication to the website. Authentication module consists with two component, which are local authentication and federated authentication. In addition to that, a component called adaptive authentication is integrated with the local authentication.

Nowadays users are fond of using social logins to login to other websites rather than registering. Social login is a form of single sign-on using existing information from a social networking service such as Facebook, Twitter or Google to sign into a third-party website instead of creating a new login account specifically for that website [2].

Adaptive authentication provides world-class security without impacting usability. That's because risk checks are done without users even being aware of it- and multi factor authentication is applied only risks are detected [3].

This module makes sure that developer will have the full functionality with less development time in developing the authentication mechanisms.

### **1.1.2 Literature Survey**

Since the Emergence of the Internet, classified ads have moved on from the old-fashioned way of newspapers to the internet, where the competition is now higher than ever for classified web sites. More than 20+ classified websites are created and used within Sri Lanka itself. These Classified websites generate a huge amount of profit merely from the ads displayed on their websites. New classified websites are still being created in Sri Lanka to potentially capture or part take in the existing online market. These websites require certain web technologies to be implemented in order to create a successful classified web application for the end users. The continuous changes in existing technologies and arrivals of new modern technologies brings a burden to the developers of these classified websites.

A classified web developer has to face the problem of having to learn all the latest technologies from database all the way to authenticating the application needed to create a classified website. This learning curve [4] bring up the time required to build the classified website as well as the cost of the project. This raise in time required to bring the project to market and the increase in the cost that's brought with it is a down side the developer or the organization has to face when carrying on such projects.

If an authentication module has to be built from the scratch, developer will have to spend more time and effort to develop a more reliable authentication mechanism. Even to configure some APIs related to authentication, require previous or additional knowledge. And also, he/she has to spend some more time in testing the module to make sure it's a secure mechanism. Because authentication is a critical component to the website, which act as a gate to user's personal data and information.

## 1.2 Research Gap

The Research gap is defined as the area or topic for which missing or insufficient information limits the ability to reach a conclusion for a question. This research is directly adopted with the development online classified ad domain. When considering this domain not much other researches done in implementing or identifying the relationship between the development ease and time to market the product. Although tools existing exists today in the market for developers to get started on developing classified websites, the tools available have limited support to new emerging technologies and thus making it hard for the developer to maintain or even adopt to new technologies. In the software development field reducing the development time is the secret of reducing the time to market and reducing the project development cost.

There are numerous products available in classified domain for web development. Most of these products present themselves as web platforms, where the application developers can create classified based advertising websites consuming the features of that platform. After the creation of the application, the developer of the classifieds website needs to deploy the application on the same vendor's servers because these products are platform dependent. In addition to that, customizability is at a very poor level in many of the available products in the market.

Following are some existing frameworks available in the classified domain:

- Yclas

Yclas is a popular tool which enables the users who have little or no knowledge in web development, to create classified web sites. Deployment should be done on a platform that is provided by Yclas. Yclas can be used to create a website for real-estate, auto or jobs marketplace. It supports social logins such as facebook, google, twitter. But it doesn't have the facility to extend the authentication point and connect a preferred identity provider. And also it doesn't provide adaptive authentication to local logins. Initially users can get a free trial, but in order to get the full featured product, users has to select and pay for a pricing plan [5].

- Flynax

Flynax is a PHP script developed based on MVC architecture. They offer six PHP classified scripts for classifieds, auto classifieds, real estate agency, boat classifieds, pet classifieds and escort agency. It has 80 powerful plugins to boost basic functionality of the software. But users have to pay to get most of the plugins. Facebook connectivity is integrated with flynax. But doesn't have the facility to connect with other social logins. And also it doesn't provide secure authentication to the local authentication. But it's expensive than other softwares in the classified domain [6].

- Oxyclassifieds

PHP scripts based product that helps to build classifieds and this product is customizable up to some extent. Oxy classifieds define a number of classified ads templates such as General Classifieds, Auto Classifieds, Realty Classifieds and Boats Classifieds. Responsive templates, Location filter, email alerts, user groups and many functionalities are included with oxy classifieds. But it only integrated for the login with facebook and google. It doesn't provide any extensibility facility to extend the federated authentication. And it doesn't have a risk identification mechanism [7].

- Titan classifieds

This can create and cleanly manage classifieds professionally. Customizations can be done through vendor's developers. This product is now part of yclas product [8].

- Os class

This is a PHP script that allows you to quickly create and manage your own classifieds site free of charge. This has dozens of templates, themes and plugins to create classified web sites. Os class is fully customizable. It has various features such as adding categories and subcategories, custom fields, static pages and many more. Even though it is customizable, it only has the connectivity for Facebook.

And it doesn't have a facility to integrate another identity provider for federated authentication. It doesn't have a mechanism to detect suspicious logins [9].

- Classipress

This is a word press plugin which helps to create and manage WordPress based classified websites. It's a theme used for classified ads for word press [10].

Considering the authentication features we're going to provide, we compared the existing products with our framework.

Table 1.1: Comparison of similar frameworks

Features	Yclas	Flynax	Oxy classifieds	Os class	ClassiPress	Proposed framework
<b>Open source</b>	Yes	No	No	Yes	No	Yes
<b>Free</b>	Yes	No	No	Yes	Yes	Yes
<b>Social login</b>	Yes	Yes (only facebook)	Yes	Yes (only facebook)	Yes	Yes
<b>Extensible federated authentication</b>	No	No	No	No	No	Yes

### 1.3 Research Problem

Developers face problems of having to learn new emerging technologies that come out and changes to existing technologies already implemented in their current web applications. This brings on a learning curve which increase the time that the developer takes to implement the changes or build a new version of the system. The main requirement of this research is to find solutions for those problems defined below:

### **1.3.1 Difficulties in providing third party identity provider support**

Web development frameworks which are quite popular nowadays provide the facility of getting the development library externally to implement the identity provider support and integrate with the framework. But most of those external libraries don't provide the functionalities to many identity providers. Rather it provides just one or two popular social identity providers. For the developers who prefer to extend the identity provider support, could not be fulfill with the existing frameworks.

In the classified domain, identity provider support has become an essential part for a website. So, in the development process, if developer has to go through an external library to integrate with the framework in order to get the facility, it will consume more time and more effort.

### **1.3.2 Difficulties in identifying suspicious logins**

Identifying suspicious or risky logins is important in the classified website domain as they have a huge audience of users using the website every day. Frameworks don't come up with a mechanism to identify suspicious logins. If the developer has to implement the functionality from the scratch, it will be a complex and heavy task. The other option is to use an external product to identify the suspicious logins. Even for that it costs time and effort to integrate with the framework and if it's a new product, there will be a learning curve to learn the functionality beforehand.

## **1.4 Research Objectives**

### **1.4.1 General Objective**

General objective of this research is to overcome the issues mentioned under the research problem and develop an extendable, customizable and secure authentication component for the classified middleware framework. This middleware should handle the complexity of different technology layers, that are involve in the development process.

This component should minimize the workload and the learning curve which a developer has to face throughout the development of an authentication component. For the developers who are in the field of classified web development, should be able to

1. Reduce developer burden, complexity and knowledge gap during the development process.
2. Reduce development time and cost of a classified base web development project.
3. Get sophisticated features as well as wide range of flexibility.
4. Reduce the maintainability cost and time.

#### **1.4.2 Specific Objectives**

- Design and implement the architecture component to extend federated authentication. By default, federated authentication supports facebook, twitter, google, linkedin authentications. If developer needs to provide authentication from a different identity provider, he/she will be provided a feature to add that authentication facility to the framework. So that developer can easily implement the functions according to the particular identity provider by going through certain configurations. Developers should be able to easily enable or disable a connector, not going through a source code, but going through a configuration file. Main goal is to reduce the time and effort takes to customize the authentication point and implement the functionality more effectively.
- Design and implement a component to be integrated with the local authentication to identify suspicious and risk logins. This can be called as risk based authentication or adaptive authentication which considers certain risk levels to check whether the login attempt is risky or not. Developer should be able to customize the component according to his/her preference and get the output as the way he/she wanted. Mainly the component will consist with the risk level IP address, location, browser, device and time. If the calculated risk level is high, developer can customize different verification levels to verify the user.



## **2 METHODOLOGY**

### **2.1 Methodology**

In industry, there are many software development life cycle approaches which are used in the development process of the software. The system development life cycle (SDLC) also referred to as the application development life-cycle, is a term used in systems engineering, information systems and software engineering to describe a process for planning, creating, testing and deploying an information system. Software development life cycle consists with 6 phases.

1. Requirement gathering and analysis
2. Design
3. Implementation or coding
4. Testing
5. Deployment
6. Maintenance

As the methodology, this section explains each phase of the SDLC, with using the relevant process descriptions, up to the design phase. Implementation phase and testing phase will be covered in the next sub-section (i.e. 2.2). Work breakdown structure will explain the connection between each step of SDLC.



Figure 2.1: Work Breakdown Structure

### 2.1.1 Background Study

When determining the feasibility of the project, research team determined the research components feasibility too. Identifying the constraints, evaluating them and analyzing the capabilities of the proposed project are the main objectives of the feasibility study. It consists of,

1. Economic feasibility
2. Technical feasibility

#### Economic Feasibility

Determining the positive economic benefits that the proposed system would provide is the main purpose of the economic feasibility. As we were doing the background study, we realized that there was no research conducted in this domain and determined there will be a huge benefit by doing this project. Since it doesn't integrate with payable products to deal with, there were no external cost to determine to evaluate on benefits.

## **Technical Feasibility**

Gathering knowledge about existing technologies and how they will be applicable for our proposed project is technical feasibility. The technical feasibility evaluates whether the project is able to be carried out with the existing equipment, skilled manpower and technologies and make sure the concept is technically possible.

### **2.1.2 Requirement Analysis**

Requirement gathering and analysis is the crucial part of most projects. Before starting to implement the system, research group clarified whether the expected result can be achieved from the project. Since the research team is doing an external project belongs to WSO2 software company, research team met with the external supervisor several times and gathered the requirements.

Followings are used to clarify the requirements.

1. Online Resources:

There were many researches were done related to the authentication of web development. But none of the existing researches deal with extensible mechanism in federated authentication. The research team has found some existing frameworks used for classified web development. But every existing framework has limitations in authentication component

2. Document analysis:

Since there were existing frameworks, we referred their documents to understand the functionalities and limitations of those frameworks. And also we referred the documents regarding the technologies we were expected to use and made sure those were the suitable ones for the implementation.

### **2.1.3 Design**

One of the important phases in SDLC is design phase. The design stage takes approved requirement specification as its initial input. Output which are described in this phase are, features to build, its sub modules, how each module works and relationships between those sub modules. The design phase is started immediately after identifying all the requirements from the end user and to initiate the identification of user interfaces. For this purpose, we have designed set of UML diagrams such as use cases that describe how end users interact with the system.

#### **User interfaces**

Since it's a web development framework, there's no specific user interfaces to be built with the framework. But to make the development process easier, we're giving a sample login and registration interfaces, which user directly interacts with, and also a development configuration screen to make the development tasks easier.

## **2.2 Testing and Implementation**

### **2.2.1 Implementation**

#### **Authentication Module**

Authentication module consists with local authentication and federated authentication components. Federated authentication module deals with the third-party identity providers while local authentication deals with local logins to the website. To enhance the functionality of local authentication component, adaptive authentication functionality is integrated. Figure 2.6 visualize the high-level diagram of the authentication module.

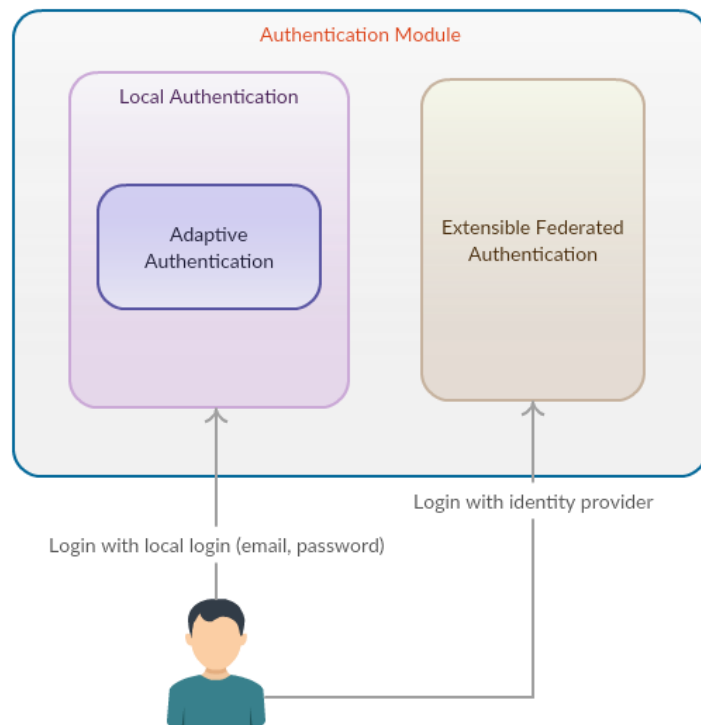


Figure 2.2: High level diagram of authentication module

- **Extensible federated authentication**

Nowadays users are widely using internet to fulfil their needs and requirements. In order to achieve their requirement, users may need to create user accounts in various websites. If you use different user credentials for different websites, you will definitely face the trouble of remembering passwords for the websites you're rarely using. To overcome this situation modern websites, make use of OAuth protocol with the concept of "Identity federation" and "Delegated Authorization" [11]. If a website is providing a facility to use a different identity provider such as Facebook, Twitter, LinkedIn, Google etc. you can simply sign up using that identity provider, rather than wasting your time by filling a lengthy form to create an account. But most of the frameworks doesn't come up with in built APIs for federated authentication. When it does, it will be only for the popular identity providers. There will be no mechanism for the developer to use different identity provider of his/her preference. This component focuses on providing a facility to extend the federated authentication in order to make

the development tasks easier for the developer. Figure 2.7 shows the high-level diagram of OAuth process flow.

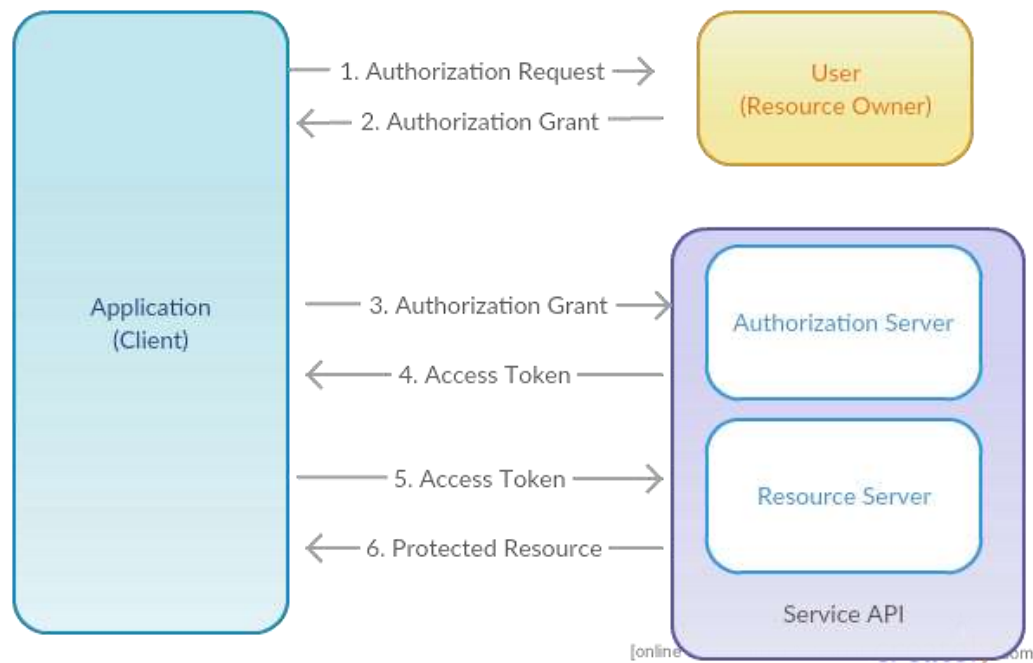


Figure 2.3: High level diagram of OAuth process flow

For the authentication purposes, we use OAuth 2.0 authorization framework. OAuth 2 is an authorization that enables application to obtain limited access to user accounts on an http service, such as Facebook, Github, Linkedin etc.

There are several roles defined by OAuth.

- Resource owner
- Client
- Resource server
- Authorization server

Following is an example of what exactly happens during the federated authentication process when using any identity provider with OAuth authorization framework.

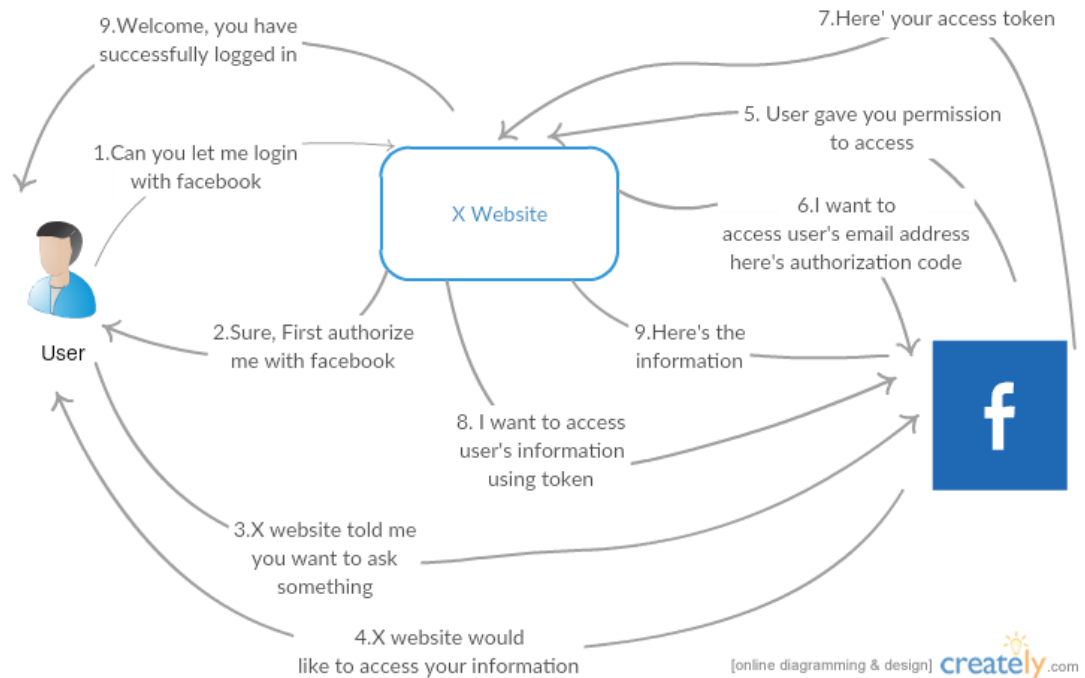


Figure 2.4: OAuth process in login with Facebook

There is an authentication abstraction interface where we define the methods for the OAuth process flow. Those methods will be implemented inside the connectors according to the federated identity provider. We used JIT provisioning to create the users on the fly without creating user accounts in advance [12]. Following figure 2.8 shows the high-level diagram of Extensible federated authentication.

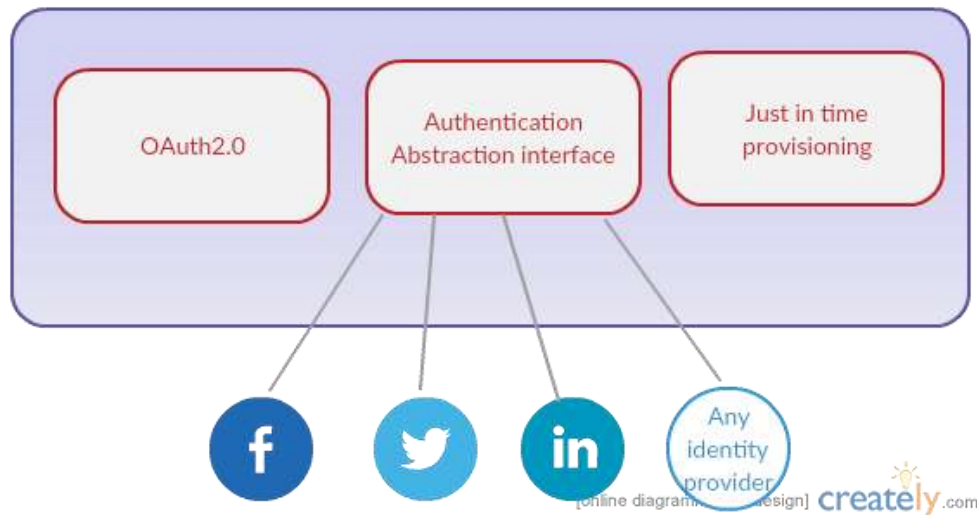


Figure 2.5: High level diagram of extensible federated authentication

Before starting the implementation, I have created OAuth applications in every major identity provider's websites. Then I obtained the relevant information to connect to the API, such as, client Id, client secret, token end-point, client end-point etc.

For the implementation, we created an abstract class called authentication which define the main methods for the OAuth authorization flow. Those methods are to get the authorization code, get access token and get profile details respectively. And also, the common fields which are used in OAuth framework are defined as attributes in the abstract class. Following Figure 2.6 is the code snippet of authentication abstraction class.

```
public abstract class Authentication {
    public static String TOKEN_ENDPOINT;
    public static String AUTH_ENDPOINT;
    public static String CLIENT_ID;
    public static String CLIENT_SECRET;
    public static String REDIRECT_URI;
    public static String ENABLE;

    public abstract String getAuthorizationcode(HttpServletRequest request);
    public abstract String getAccessToken(String output, HttpServletRequest request);
    public abstract void getprofiledetails(HttpServletRequest request, String UserInfoEndpoint, String accessToken);
}
```

Figure 2.6: Code snippet of authentication abstract class

Then if we want to implement a connector, we need to extend from authentication class and implement the methods according to the particular connector. We created the



connectivity for major identity providers, Facebook, Twitter, Google and LinkedIn. Following are the code snippets of those connectors that we're providing.

```

public class Facebook extends Authentication {
    public static String GRANT_TYPE;
    public static String RESPONSE_TYPE;
    public static String SCOPE;

    public Facebook() {
        (...3 lines)
    }
    public static void assignValues() {
        (...53 lines)
    }

    @Override
    public String getAuthorizationcode(HttpServletRequest request) { (...3 lines) }
    public void addAuthorizationHeader(HttpPost httpPost, String CLIENT_ID, String CLIENT_SECRET) {
        (...8 lines)
    }
    public HttpResponse makeAccessTokenRequest(HttpPost httpPost) throws IOException {
        (...6 lines)
    }
    public String handleAccessTokenRequest(HttpResponse httpResponse) throws IOException {
        (...7 lines)
    }
    @Override
    public String getAccessToken(String line, HttpServletRequest request) { (...21 lines) }
    @Override
    public void getprofiledetails(HttpServletRequest request, String FacebookUserInfoEndpoint, String accessToken)
}

```

Figure 2.7: Facebook connector implementation

```

public class LinkedIn extends Authentication {
    public static String GRANT_TYPE;
    public static String RESPONSE_TYPE;
    public static String SCOPE;
    public static String STATE;

    public LinkedIn() {
        (...3 lines)
    }
    public static void assignValues() {
        (...53 lines)
    }

    @Override
    public String getAuthorizationcode(HttpServletRequest request) { (...3 lines) }

    public String getAccessToken(String output, HttpServletRequest request) { (...19 lines) }

    @Override
    public void getprofiledetails(HttpServletRequest request, String linkedInUserInfoEndpoint, String accessToken)
}

```

Figure 2.8: LinkedIn connector implementation

```

public class Google extends Authentication {
    public static String GRANT_TYPE;
    public static String RESPONSE_TYPE;
    public static String SCOPE;
    public static String APPROVAL_PROMPT;

    public Google()
    [...3 lines]
    public static void assignvalues()
    [...55 lines]
    @Override
    public String getAuthorizationcode(HttpServletRequest request) [...4 lines]

    public String makeAccessTokenRequest(String urlParameters,String tokenEndPoint) throws IOException
    [...15 lines]

    @Override
    public String getAccessToken(String line,HttpServletRequest request) [...5 lines]

    @Override
    public void getprofiledetails(HttpServletRequest request, String GoogleUserInfoEndpoint, String accessToken)
}

```

Figure 2.9: Google connector implementation

```

public class TwitterClass extends Authentication {
    public static String REQUEST_TOKEN_ENDPOINT;

    public TwitterClass()
    [...3 lines]
    public static void assignvalues()
    [...45 lines]
    @Override
    public String getAuthorizationcode(HttpServletRequest request) [...3 lines]
    public RequestToken getRequestToken(HttpServletRequest request, Twitter twitter) throws ServletException [...16 lines]

    public String getVerifier(HttpServletRequest request)
    [...4 lines]

    @Override
    public String getAccessToken(String verifier,HttpServletRequest request) [...14 lines]

    @Override
    public void getprofiledetails(HttpServletRequest request, String UserInfoEndpoint, String accessToken) [...3 lines]
}

```

Figure 2.10: Twitter connector implementation

Similar to these connector implementations, developer can extend and create a new connector by implementing all the abstract methods according to the way that particular connector works. Apart from the common fields, if there are attributes specific to that particular connector, developer can state them under attributes. Same applies to the specific methods as well. Developer will be given a document to guide through the new connector creation. For the configurations related to connectors, developer will be given a configuration file, where he/she needs to modify the configurations according to the app they're creating on the developer website of a particular identity provider. This configuration file includes the information developer

is obtained by creating the developer application in identity provider's developer website and also the fields related to the OAuth process flow. Figure 2.11 is a screenshot of part of a XML configuration file which used in the extensible federated authentication.



```
<?xml version="1.0" encoding="windows-1252"?>
<connectors>
  <connector name="LinkedIn">
    <enable>true</enable>
    <clientId>812j558k7kkqgx</clientId>
    <clientsecret>KsahjjVz47p6lzcX</clientsecret>
    <authorization_endpoint>https://www.linkedin.com/uas/oauth2/authorization</authorization_endpoint>
    <token_endpoint>https://www.linkedin.com/uas/oauth2/accessToken</token_endpoint>
    <client_endpoint>http://localhost:8080/TestProject/Linkedincallback</client_endpoint>
    <grant_type>authorization_code</grant_type>
    <response_type>code</response_type>
    <scope>r_basicprofile r_emailaddress</scope>
    <state>123456</state>
  </connector>
  <connector name="Facebook">
    <enable>true</enable>
    <clientId>1741364576163534</clientId>
    <clientsecret>d23e5754aa74dad41e8eebfel6f3b764</clientsecret>
    <authorization_endpoint>https://www.facebook.com/dialog/oauth</authorization_endpoint>
    <token_endpoint>https://graph.facebook.com/oauth/access_token</token_endpoint>
    <client_endpoint>http://localhost:8080/TestProject/Facebookcallback</client_endpoint>
    <grant_type>authorization_code</grant_type>
    <response_type>code</response_type>
    <scope>public_profile email</scope>
  </connector>
  <connector name="Google">
    <enable>true</enable>
    <clientId>642110263906-sotsrsaeinlsudtfig5dhuvq97247ohe.apps.googleusercontent.com</clientId>
    <clientsecret>75fAn-mynWV6BAhRRCIHNPc5</clientsecret>
    <authorization_endpoint>https://accounts.google.com/o/oauth2/auth</authorization_endpoint>
  </connector>
</connectors>
```

Figure 2.11: Authentication configuration file

Developer only has to modify the configuration file, using his/her app configurations to get the functionality. If developer doesn't want any of identity providers he/she can disable that connector by making the enable configuration false. So that it will dynamically load the login page by reading the configuration file.

Once user logged in for the first time using an identity provider, an identification field will be selected (mostly email address) and user account will be created using the fields we're getting from the identity provider. A field called authenticator will be used to identify the users who are logged in using federated authentication.

- **Adaptive Authentication**

Adaptive authentication provides world-class security without impacting usability. That's because risk checks are done without users even being aware of it- and multi factor authentication is applied only risks are detected. Most of the popular websites who have huge user domain are using this technique to secure their user's information. For the classified web development framework local authentication component, adaptive authentication is integrated considering following risk factors:

1. IP address
2. Geographical location
3. Device
4. Browser
5. Date/Time

To calculate the risk levels more effectively, a weighted scoring model was implemented under adaptive authentication component. Weight is divided among the risk factors and developer can decide which factors to give more priority and assign weights. By default, IP address, location and date/time risk factors will be given more priority.

#### **IP address risk calculation**

IP address will be obtained from Http Request which is coming from the user and check whether that IP address exist in the most previous successful login attempt. If that IP address exist in the previous login attempt, that IP address considers as a no risk IP address. Therefore, the risk score will be 0 for the IP address. If that IP address doesn't exist in the previous successful login attempt, but appears in other successful login attempts, an average risk score will be calculated. If the IP address doesn't exist at all in previous login attempts, it will take as a higher risk. So, for the IP address, risk score will be 1.

#### **Browser risk calculation**

Browser details will be obtained from the Request header and check whether that device type exists in the most previous login attempt. Here the device type will be

considered as computer or mobile. If that device type exists in the most previous login attempt, it will be considered as a no risk. Therefore, the risk score will be 0. If it doesn't exist at all in previous login attempts, it will be considered as a higher risk and risk score will be 1. If that device type doesn't exist in the most previous login attempt, but exist in other login attempts, an average risk score will be calculated.

### **Browser risk calculation**

Browser will also be obtained from the request header and check whether that browser type exists in the most previous successful login attempt. If it exists, then there's no risk at all in browser's point of view. So, the risk will be 0. If it doesn't exist in the most previous login attempt, but exist in other records, an average risk will be calculated. If that browser type doesn't exist at all, the risk is taken as a higher risk and risk score will be 1.

### **Location risk calculation**

Location will be obtained using the IP address by using a geo look up. Latitude and longitude degrees will be considered as the location attributes. Developer can configure the ranges to be considered as higher risk values. In default, latitude range is 3 degrees and longitude range is 1 degree. These degrees were decided based on Sri Lanka's latitude and longitude range. Within the range means, within the country and out of the range means, out of the country. First the latitude and longitude values will be checked with the most previous success login's latitude and longitude values. If they match, that means there's no risk at all. Therefore, the risk score will be 0. If it doesn't match it checks with the ranges configured by the developer. If the latitude and longitude values are within the range that means the user moves within the country after the previous successful login attempt. So, the risk score will be 0. If the last location isn't in the defined range, each previous success attempts will be checked to make sure that user has visited that range of location. Then an average risk will be calculated.

### **Date/Time risk calculation**

To calculate the risk score, logged in date time will be checked with the last successful logged in time. If the difference is more than 30 days, it will consider as a higher risk value. Therefore, the risk score will be 1. If the difference is less than 30 days, it checks with the time range configured by the developer. By default, the time range is configured as 3 hours. That means logged in time +/- 3 hours range will be considered as, user has logged in his usual time of the day. An average risk score will be calculated based on that.

Every risk score is multiplying by its assigned weight and get the weighted risk score for every factor. Then the total risk score will be calculated by adding all the weighted risk scores. The total risk score comes in the state of percentage.

*Total risk score*

$$\begin{aligned} &= (IP \text{ address risk score} \times IP \text{ address weight}) \\ &+ (Browser \text{ risk score} \times Browser \text{ weight}) \\ &+ (Device \text{ risk score} \times Device \text{ weight}) \\ &+ (Location \text{ risk score} \times Location \text{ weight}) \\ &+ (Date/Time \text{ risk score} \times Date/Time \text{ weight}) \end{aligned}$$

Equation 1: Risk Score Calculation

Based on the total risk score, decisions can be taken to verify the user. Following figure 2.12 shows a code snippet of the methods that have used for risk calculation.

```

public class RiskScore {
    private static double ip_weight;
    private static double browser_weight;
    private static double device_weight;
    private static double time_weight;
    private static double location_weight;
    private static double time_range;
    private static double latitude_range;
    private static double longitude_range;

    public static void assignvalues()
    {...33 lines }

    public static double callIpScore(String email,String ip)
    {...43 lines }

    public static double calBrowserScore(String email,String browser)
    {...42 lines }

    public static double calDeviceScore(String email,String device)
    {...42 lines }

    public static double callTimeScore(String email,Date time)
    {...73 lines }

    public static double calLocationScore(String email,double latitude,double longitude)
    {...61 lines }

    public static double totalRiskScore(String email,String ip,String browser,String device,
    Date time,double latitude,double longitude)
    {...13 lines }
}

```

Figure 2.12: Code snippet used to calculate the risk score

If the total risk score is more than 70%, it will consider as a higher risk level and an OTP of 6 digits code will be generated and send to user's email address to verify the user. If the risk score is between 50% and 70%, it will consider as a middle level risk and pops up security questions which were given by the user at the registration stage. If the calculated total risk score is below 50%, it will consider as a minimal risk and redirect user to the home page. Developer can change the decided risk ranges for higher, medium and lower risks. Moreover, if the user credentials are incorrect and risk score is high, an email will be send to the user telling that someone has tried to login to his/her account.

Developer is given a user interface to configure the weight for the risk factors and also for the configurations related to location ranges and time ranges to consider when calculating risk scores. And also developers can enhance this functionality to consider more risk factors and provide customization to users as well.

### **2.2.2 Testing**

Software testing is an important phase in the SDLC. Testing helps in identifying bugs and issues in the system and can take necessary actions to correct to fix those bugs. Testing has happened throughout the development starting from the requirement gathering phase. Testing has conducted in the implementation that always with line to the requirement specification. This can be used to provide information about the quality of the product under tests. There are recognized test levels can be identified under testing. They are unit testing, module testing, integration testing and the system testing.

#### **Unit Testing**

Unit testing is used to test the functionality of each module separately. They are tested alone and isolated. It tests the functionality of a specific section of code, at the function level. Some of the unit testing frameworks are, Junit and Nunit. Unit testing follows white box analysis.

#### **Integration Testing**

Integration testing combines all the units within a program and test them as a group. This testing level is designed to find interface defects between the modules/functions. This tests how each module interacts with each other. Integration testing occurs after module testing and before validation testing. This takes the input as modules, which are unit tested.

#### **System Testing**

System testing tests the whole integrated system to evaluate the system's compliance with its requirements. We perform system testing to ensure that we have completed and achieved up to the expected quality and level of objectives.



## Test Cases

We have tested the functionality of authentication component. Following are some test cases related to the unit testing.

Table 2.1: Login with Facebook with correct fields

Test Case #	TC_01
Description	Login with Facebook
Pre-conditions	There should be an active internet connection and an active Facebook account.
Test Steps	<ol style="list-style-type: none"><li>1. Click on login with Facebook button on Login page.</li><li>2. Enter the email address and password.</li><li>3. Give permission to access your profile</li></ol>
Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a> Password: ishani
Expected Output	Successfully logged in
Actual Output	Successfully logged in
Status	Pass

Table 2.2: Login with twitter using correct fields

Test Case #	TC_02
Description	Login with Twitter
Pre-conditions	There should be an active internet connection and an active Twitter account.
Test Steps	<ol style="list-style-type: none"><li>1. Click on login with Twitter button on Login page.</li><li>2. Enter the email address and password.</li><li>3. Give permission to access your profile</li></ol>
Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a> Password: ishani

Expected Output	Successfully logged in
Actual Output	Successfully logged in
Status	Pass

Table 2.3: Login with Linkedin using correct fields

Test Case #	TC_03
Description	Login with Linkedin
Pre-conditions	There should be an active internet connection and an active Linkedin account.
Test Steps	<ol style="list-style-type: none"> <li>1. Click on login with Linkedin button on Login page.</li> <li>2. Enter the email address and password.</li> <li>3. Give permission to access your profile</li> </ol>
Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a> Password: ishani
Expected Output	Successfully logged in
Actual Output	Successfully logged in
Status	Pass

Table 2.4: Login with google using correct fields

Test Case #	TC_04
Description	Login with Google
Pre-conditions	There should be an active internet connection and an active Google account.
Test Steps	<ol style="list-style-type: none"> <li>1. Click on login with Twitter button on Login page.</li> <li>2. Enter the email address and password.</li> <li>3. Give permission to access your profile</li> </ol>
Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a>

	Password: ishani
Expected Output	Successfully logged in
Actual Output	Successfully logged in
Status	Pass

Table 2.5: Local authentication using previous IP, device, browser, location and date/time

Test Case #	TC_05
Description	Login with local authentication using previous login IP, device, browser, location and date/time
Pre-conditions	There should be an active internet connection and user should be registered with the website.
Test Steps	<ol style="list-style-type: none"> <li>1. Enter the email address and password.</li> <li>2. Click login button</li> </ol>
Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a> Password: ishani
Expected Output	Successfully logged in
Actual Output	Successfully logged in
Status	Pass

Table 2.6: Local authentication using different IP address

Test Case #	TC_06
Description	Login with local authentication using different IP address
Pre-conditions	There should be an active internet connection and user should be registered with the website.
Test Steps	<ol style="list-style-type: none"> <li>1. Enter the email address and password.</li> <li>2. Click login button</li> </ol>

Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a> Password: ishani
Expected Output	Email is sent with the OTP. Redirect to the verify page to enter the OTP
Actual Output	Email is sent with the OTP. Redirect to the verify page to enter the OTP
Status	Pass

Table 2.7: Login with local authentication using different IP and invalid credentials

Test Case #	TC_07
Description	Login with local authentication using different IP address and invalid password
Pre-conditions	There should be an active internet connection and user should be registered with the website.
Test Steps	<ol style="list-style-type: none"> <li>1. Enter the email address and password.</li> <li>2. Click login button</li> </ol>
Input	Email: <a href="mailto:ishanivv@gmail.com">ishanivv@gmail.com</a> Password: ishaniii
Expected Output	Email is sent with the logged in information. Invalid credentials message is displaying
Actual Output	Email is sent with the logged in information. Invalid credentials message is displaying
Status	Pass

## 2.3 Research Findings

During the research, team has identified that there were some classified web development frameworks with lots of limitations and weaknesses which cause low usage. The team found that existing classified web developing frameworks don't provide Extensible database abstraction, extensible federated authentication and built in web analytics generator facilities for the developer within the framework. "Ampliar"

framework focuses more into extendibility of modules and reduce developer frustration.

This section covers the findings that are gained during our research, related to classified web development domain.

- **Extensible Federated Authentication**

Research team has found out that many classified websites use different identity providers for the federated authentication and decided to provide an extensibility to the federated authentication where developer can easily extend the functionality. By going through different classified web sites, we have identified major identity providers and integrated the connectivity to the framework. Abstract class was provided with the methods which are going through the OAuth process flow and made the connectors implement the methods according to their process. Since the goal is to reduce the developer time and effort, configuration file was given to the developer to easily configure the connector configurations.

- **Weighted Risk Score model to use in adaptive authentication**

Nowadays risk based authentication is popular among websites as it identifies suspicious logins and block users from using the website until the user verifies about his/her identification. Risk scoring model was developed considering the risk factors IP address, device, browser, location and date/time. Since the effect of each factor has to be considered and priorities should be given, weighted concept was introduced. Using the weight developer can configure which risk factors to give more priority and which factors to give less priority. According to the calculated risk, decisions can be taken to verify the user. Developer can configure different verification levels based on the risk score.

## 3 RESULTS AND DISCUSSION

### 3.1 Results

The main goal of this research is to provide solid feature-rich middleware framework for classified base web development. This middleware handles the complexity of different technology layers, that are involve in the development process.

Though the final outcome of the research project, external parties related to the project development can get an idea of the final outcomes and the quality of the final product, which includes verification and validation.

We have presented the concept and the techniques that have been used in the process of the development of the “Ampliar” framework. Through our research paper, we have clearly pointed out the problems which were faced by the developers in developing classified websites and these problems have been addressed by this solution. By accepting the research paper from “**National Information Technology Conference 2017**”, the domain experts and community have accepted this solution as a proper solution which will address major problem areas.

This subsection provides evidence to the implementation results and the solutions provided for the identified research problems. Since it’s a development framework, developers directly should interact with the configuration files. For some cases, developer is provided with a user interface to reduce the complexity of the functionality.

Following is the user interface provided for the login with local authentication and federated authentication.

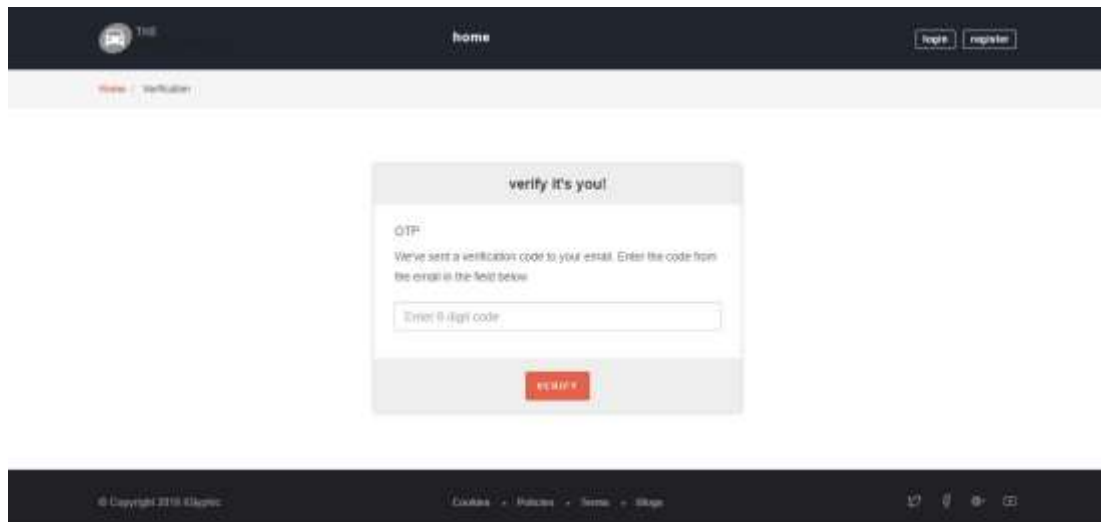
Figure 3.1: Login interface

This login interface can be dynamically load by modifying the enable configuration in the authentication configuration file. If developer doesn't want the google connector to be integrated with the authentication, he/she only should do is modifying the enable configuration under google connector as false.

To configure the adaptive authentication, developer will be given a user interface to easily enter the configurations. So that, he doesn't have to go through the code to make changes in the configuration, this make the developer tasks more flexible.

Figure 3.2: Developer settings interface

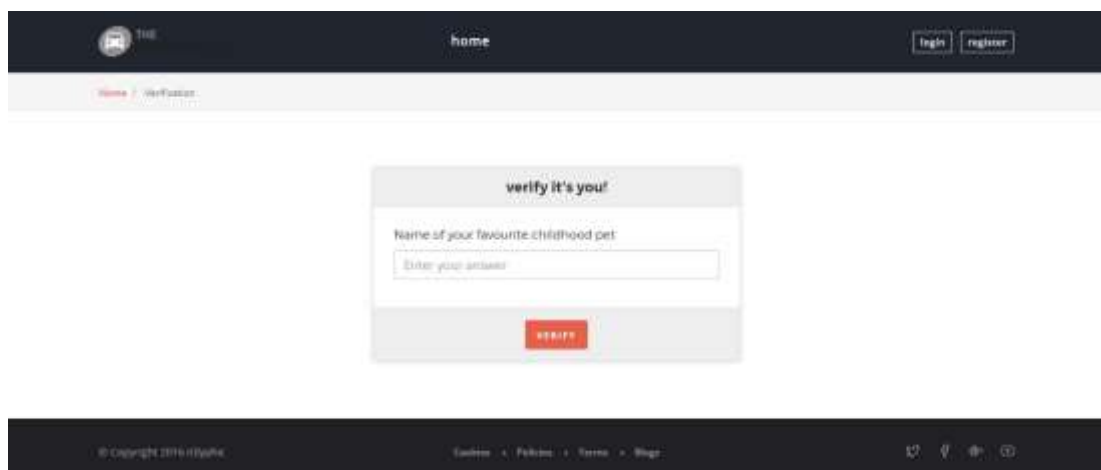
If the risk score calculated for the adaptive authentication is higher, OTP will generate and send to user's email. Following screen appears to enter the 6-digit code to verify the user.



The screenshot shows a web application interface for OTP verification. At the top, there is a dark navigation bar with a logo on the left, the word "home" in the center, and "login" and "register" buttons on the right. Below the navigation bar is a breadcrumb trail showing "Home" and "Verification". The main content area features a light gray box with the heading "verify it's you!". Underneath, it says "OTP" and "We've sent a verification code to your email. Enter the code from the email in the field below:". There is a text input field labeled "Enter 6-digit code" and a red "verify" button at the bottom of the box. The footer contains copyright information, navigation links, and social media icons.

Figure 3.3: OTP verification interface

If the risk score is in the middle level, user will be prompted the security question which was given by the user at the registration stage.



The screenshot shows a web application interface for security question verification. It has the same dark navigation bar and breadcrumb trail as Figure 3.3. The main content area features a light gray box with the heading "verify it's you!". Underneath, it asks "Name of your favourite childhood pet" and provides a text input field labeled "Enter your answer". A red "verify" button is at the bottom of the box. The footer is identical to the one in Figure 3.3.

Figure 3.4: Security Question verification



### **3.2 Discussion**

This chapter includes a discussion regarding the total progress or the results shown in the previous chapter and the accuracy of the work done.

We have developed a classified web development framework which can be used by novice developers as well as professional developers. It consists of advanced features to support multiple database vendors, extensible federated authentication, web analytics generator and core functionalities which are identified by the research team as lack of the existing frameworks. These modules alone are providing the customizable features to customize the component in the way developer wants. This enhances the freedom and the flexibility that a developer can have when using the framework.

In order to use the framework, developers can just download the framework from the website and run in the java platform. To customize the functionalities developer can refer to the developer guides we're providing with the framework. These components can be further developed to enhance the functionalities and make the use of extendibility. Since this is an open source framework, others can also be part of the development and can contribute to the future development.

Framework was developed using object oriented concepts and have used standard web interfaces and configurations during the development. This framework also can be integrated with any other java web development frameworks, so that developers shouldn't worry about working with their preferred web development framework.

Anyone who is having little knowledge about programming languages can use this framework, since it doesn't directly interact with the coding. If a professional developer is using this framework, he/she can enhance the features and directly involve with the development if/she wishes.

## 4 CONCLUSION

Analyzing the requirements, studying existing technologies to implement this system, gathering ideas from resourceful authorities and conducting literature reviews were done related to extensible federated authentication and adaptive authentication to achieve objectives of this research component.

In this report, I have discussed about the use of authentication component in the classified web development framework which will reduce the time, effort and knowledge gap of developers in developing classified web sites. Moreover, how developers can extend the functionalities and enhance the component functionalities were discussed in the report.

By using our framework, developers can get the flexibility in using any identity provider support to be integrated with the framework. Instead of using third party product to do the adaptive authentication part, developers can use the adaptive authentication component built in the framework for risk calculation. None of the existing frameworks support these functionalities as well as the customizability.

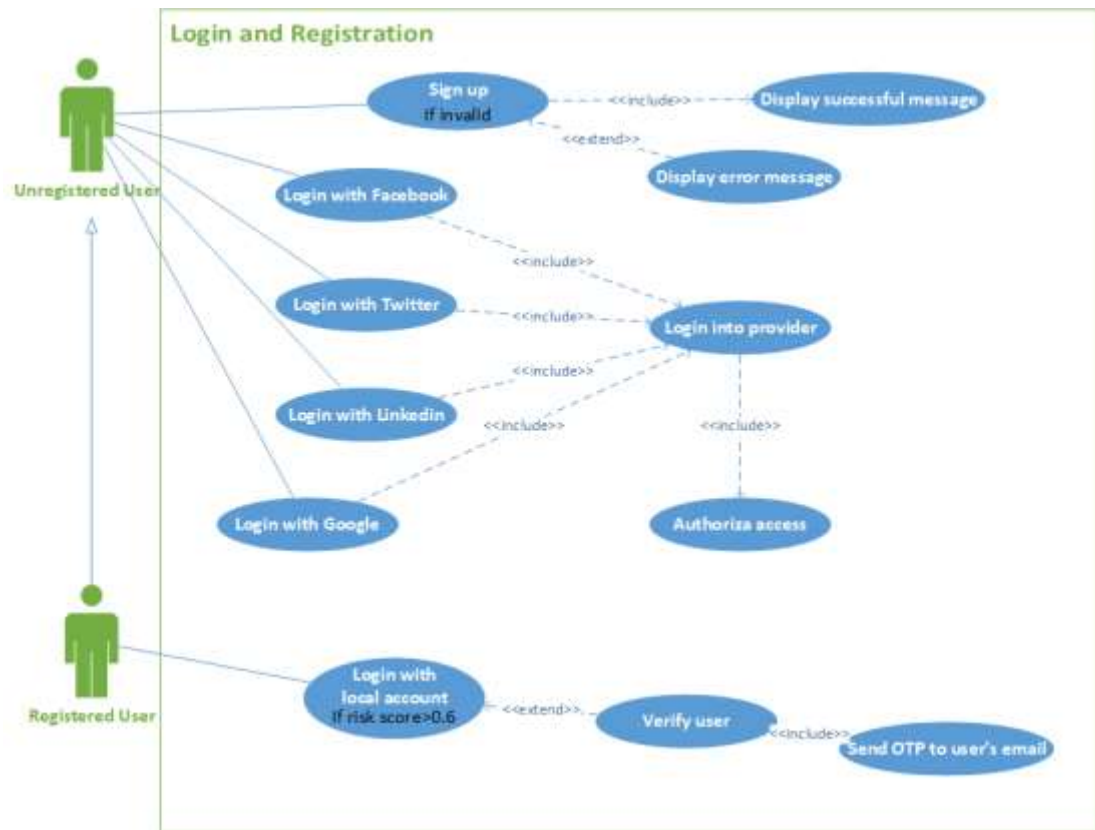
We have achieved the level of sophistication we wanted to achieve in our framework and even surpassed it. Through our framework we want to provide the best possible service to the developer in classified web development and “ampliar” wants to make sure that they’re able to consume the provided services for efficiently developing their projects with a minimum learning curve and lead their projects towards successful delivery.

This research component features can be further developed to provide more advanced functionalities for the developer in later stages.

## REFERENCES

- [1] S. Diaz, "On the Internet, A Tangled Web Of Classified Ads," Washington Post, Friday August 2007. [Online]. Available: <http://www.washingtonpost.com/wp-dyn/content/article/2007/08/30/AR2007083002046.html?hpid=sec-tech>.
- [2] B. Prescott, "Social Sign-On: What is it and How Does It Benefit Your Web Site?," [Online]. Available: <http://www.socialtechnologyreview.com/articles/social-sign-what-it-and-how-does-it-benefit-your-web-site>. [Accessed 15 March 2017].
- [3] "Adaptive Authentication," [Online]. Available: <https://www.secureauth.com/products/secureauth-idp/adaptive-authentication>.
- [4] T. W. Lin and E. J. Blocher, Cost Management: A Strategic Emphasis, Tata McGraw-Hill Education, 2006.
- [5] "Yclas Home," [Online]. Available: <https://yclas.com/>. [Accessed 15 February 2017].
- [6] "Flynax Home," [Online]. Available: <https://www.flynax.com/>. [Accessed 17 February 2017].
- [7] "Oxyclassifieds Home," [Online]. Available: <https://www.oxyclassifieds.com>. [Accessed 17 February 2017].
- [8] "Titan Classifieds Home," [Online]. Available: <http://www.titanclassifieds.com/>. [Accessed 17 February 2017].
- [9] "Osclass Home," [Online]. Available: <https://osclass.org/>. [Accessed 17 February 2017].
- [10] "Classipress Home," [Online]. Available: <https://wordpress.org/plugins/tags/classipress/>. [Accessed 17 February 2017].
- [11] "The OAuth 2.0 Authorization Framework," [Online]. Available: <https://tools.ietf.org/html/rfc6749>. [Accessed 5 February 2017].
- [12] "Design and implement just-in-time provisioning with SAML 2.0," [Online]. Available: <https://www.ibm.com/developerworks/library/se-jitp/>. [Accessed 12 February 2017].

## APPENDIX A: Use case diagram



## APPENDIX B: Use case scenarios

Use case 1	User registration												
Goal	To register user in the classified web site												
Pre-condition	None												
Actor	User												
Success Scenario	<table><tr><th>Step</th><th>Action</th></tr><tr><td>1.0</td><td>Go to registration page</td></tr><tr><td>2.0</td><td>Provide user details</td></tr><tr><td>3.0</td><td>Validate user details</td></tr><tr><td>4.0</td><td>Store details in database</td></tr><tr><td>5.0</td><td>Display successful message</td></tr></table>	Step	Action	1.0	Go to registration page	2.0	Provide user details	3.0	Validate user details	4.0	Store details in database	5.0	Display successful message
Step	Action												
1.0	Go to registration page												
2.0	Provide user details												
3.0	Validate user details												
4.0	Store details in database												
5.0	Display successful message												
Alternative flow	<table><tr><th>Step</th><th>Action</th></tr><tr><td>3.0</td><td>If user details are invalid, system gives an error message</td></tr></table>	Step	Action	3.0	If user details are invalid, system gives an error message								
Step	Action												
3.0	If user details are invalid, system gives an error message												

Use case 2	Login with local authentication				
Goal	To register user in the classified web site				
Pre-condition	User should be pre-registered				
Actor	User				
Success Scenario	<table><tr><th>Step</th><th>Action</th></tr><tr><td>1.0</td><td>Go to Login page</td></tr></table>	Step	Action	1.0	Go to Login page
Step	Action				
1.0	Go to Login page				

	2.0	Provide user details
	3.0	Validate user details
	4.0	Display home page
Alternative flow		
	Step	Action
	3.a.1	If user credentials are invalid, system gives an error message
	3.a.2	If user credentials are invalid and risk score is high, email is sent to user telling suspicious login activity
	3.a.3	If user credentials are valid, but risk score is higher, verification page will be loaded.

## APPENDIX C: Class diagram

