OPEN SOURCE SOA BASED MIDDLEWARE FRAMEWORK FOR CLASSIFIED BASED WEB DEVELOPMENT

Project ID: 17-072

Final Report(Individual)

Bachelor of Science Special (Hons) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

October 2017

OPEN SOURCE SOA BASED MIDDLEWARE FRAMEWORK FOR CLASSIFIED BASED WEB DEVELOPMENT

Project ID: 17-072

Final Report(Individual)

W.M.N. Radith – IT14011030

Supervisor: Mr.Nuwan Kodagoda

Bachelor of Science Special (Hons) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

October 2017

DECLARATION

I declare that this is my own work and this report 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 our 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.

	STUDENT NAME	STUDENT NO.	SIGNATURE
2	W.M.N Radith	IT 14 0110 30	

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.

ACKNOWLEDGEMENT

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Table 1: Abbrevitions List

Database	a structured set of data held in a computer, especially one that is accessible in various ways.
API	a set of functions and procedures that allow the creation of applications which access the features or data of an operating system, application, or another service.
Database Abstraction	A database abstraction layer is an application programming interface which unifies the communication between a computer application and databases such as SQL Server, DB2, MySQL, PostgreSQL, Oracle or SQLite
RESTful	A way of providing interoperability between computer systems on the Internet
Module	each of a set of standardized parts or independent units that can be used to construct a more complex structure, such as an item of furniture or a building.
Caching	Hardware or software sub-module that stores data so future requests for that data can be served faster
SPA	Single Page Application
EDAM	Extensible Database Abstraction Module
CRUD	Create read update and delete
DBMS	Database Management System
MVC	Model view controller

1. INTRODUCTION

1.1. Background Context

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 decrease the profitability of those printed media. Online web format of classified service provide wide range of features comparatively to printed media. Advertisement can be longer, searchable even some companies offer free advertising facilities. Due to this flexibility, online web format classified market has become heavily fragmented. Today international range, domestic range even hometown range online classified advertising companies provide their services. Furthermore, there is an increasing emphasis on developing specialized classified websites over general classified websites for vertical markets and niche markets **Invalid source specified.**.

Solid classified website is a valuable service for different perspectives. Find or sell a product or service, a great profitable business respectively to consumer and company perspectives. Building such site from the scratch is much more complex than its sounds where developers can get frustrated and eventually product might be failed or cost overrun. Since this is a very competitive marketing segment it's very important to build the application with possible minimum time with competitive features with other competitors.

The main goal of this project provides a 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.

1.2. Research Gap

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 have 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 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

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 solution to the defined below:

- **Development of a Developer Friendly Middleware framework**

One problem faced by developers is the number of different technologies needed to build a web/mobile application. This is also accompanied by the core differences between these technologies and the differences that arise by the organization that have defined the way their technologies have to be used.

This knowledge gap of the developer not only increase the time to develop a system but also bring on changes to existing systems.

- **Developing Middleware Framework that can Integrate new technologies easily**

One of the main problems faced by developers is how to integrate new technologies to existing system without having to make changes to existing version of the system. For Example: how to Integrate a new social authentication service to yahoo while already catering to Facebook, google and local authentication

1.4. Research Objectives

Main objective of this research is to develop an open source middleware framework for classified based web development. This framework should minimize the workload and the learning curve which a developer has to face throughout the development phase of classified websites. For the developers who are in the field of classified web development, should be able to

- Reduce developer burden, complexity and knowledge gap during the development process.
- Reduce development time and hence cost of a classified based web development project.
- Get sophisticated features as well as wide range of flexibility.

Specific Objectives

- Design and implement a component which supports multiple database manage systems (MySQL]1[, MSSQL,]2[Oracle]3[etc.) and has the capability to extend the database abstraction layer based on the developer's preference. For an example, if the developer needs to connects his/her website to a different database management system apart from the DBMSs listed in the framework, he/she will be provided an easy way of customizing the connection.
- Design and implement the architecture component to extend federated authentication. By default, federated authentication supports Facebook, twitter, yahoo authentications. If developer needs to provide authentication from a different

website, he/she will be provided a feature to add that authentication facility to the framework.

- Design and implement a component to use as a web analytic component to monitor and analyses the statistics of the website. Developer may give the opportunity to choose between existing analytical libraries and also he/she will be given the facility to develop an analytic library himself by following certain instructions.
- Design and implementation of Core Framework (Middleware framework core) including the restful Service API used by the web developer for classified web development (eg: ikman.lk, craigslist.org) and End Points to External Application (Mobile/Web), integration with other components of the framework, permissions and roles and security of the core framework, routing, services library that gives the developer the tools they need for modern web development. These endpoint include services the web application/site in its core functionality such as displaying the advertisements, user uploading the advertisements, to even user submitting a form will be provided with a uniform, high level API (Applications Programming Interface) to applications. This middleware framework will facilitate evolution, enhance the reusability and as well improve portability to new platforms. The framework includes other features such as routing and data type conversion, error detection and handling. This middleware framework will save the developers learning curve by providing a common programming abstraction and by hiding low-level details and development time required to complete the project.

2. METHODOLOGY

This section provides in-detail description of one core research objective, in namely design and development of database extension point. In Ampliar middleware all data storing and retrieving related task are performed by this module. A methodology can be defined as a set of guidelines or principles that can be tailored and apply for specific situation, as well as it also includes specific approach, templates, forms, and testing used over the project life cycle.

Amplair is a classified base web development framework primarily focus on to increase the productivity and reliability of classified base web development projects. Primary goal of this product or research is provide efficient, developer friendly, extensible, modular and reliable middleware framework. This primary goal is a combination of four main objectives, those are listed below.

- Highly flexible core framework
- Extensible as well as minimal sql interaction database extension module
- Extensible authentication
- Extensible web analytics generator

From this point onwards, document present the methodology that used to achieve the Database extension point objective. All guidelines, specific approach, templates, forms, and testing are described in follow sub chapters in detail.

2.1. Methodology

This sub-section provides the development life cycle, each procedures, techniques, implementation, best practices and testing in-detail carried out through database extension point.

2.1.1. Background Study

According to marketing statistics as well as new trends, classified web development extends from generic developments to specialized developments. This increase the

number of classified base development projects. As discussed, this development process takes huge amount of time and effort. It's important to have a solution to boost this development process meanwhile reduce the effort.

Currently in this domain we can see multiple frameworks and plugins provide facilities for classified base web development. Almost all these products contain different levels of limitations when we compare those products with classified base web development domain needs and requirements. One of the highlighted limitation is extensibility of those product. Almost all free products are limited in extensibility. Most of the cases those products only support one database vendor, also its very hard to extend without having deep knowledge about product. Those extension leads different issues in production environments. Which is ultimately create developer frustrations as well as waste of valuable resources. Although some products are customizable unfortunately those products are vendor preparatory products. In order to customize the product, customer needs to depends on the vendor. Most cases customization request as well as huge amount of money need to spend on those vendors. At the end of the day this type products make customer totally depends on those vendors.

2.1.2. Requirement Gathering

This research falling under software engineering research category. Then team went deep dive through existing frameworks and plugins to find the problems and weakness of those. Most of the times reach team refer developer forms to get an idea about developer thoughts about those products. To find features and functionalities of those products team refer provided API documentations of products. Throughout this process we are able to find strength of those products as well, which is help us to make our product / middleware as a next step of classified base web development frameworks.

2.1.3. Design

This module defines an interface for common methods each operation performs on the underlying database. Databases follows a generic structure and a common schema for

classified base web applications. Developers can implement the pre-defined interfaces according to DBMS vendor specifications. By default, middleware implements this interface for four major DBMS vendors namely MySQL]1[, MSSQL]2[, ORACLE]3[and PostgreSQL]4[. Developers can consume these implemented methods directly if they are using one of these vendors, otherwise developers are able to implement this interface according to the vendor specifications that they wish to use. If the developer wishes to customize the database structure, it is also possible with the extensibility of the framework. However, in this scenario the developer needs to perform the relevant modifications to the interface as well as the implemented classes.

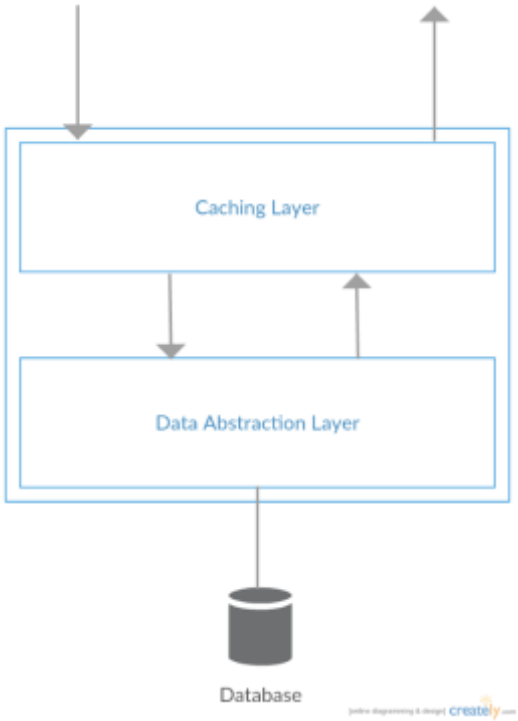

Figure 1: High level architecture digram

As figure describes this module contains two major modules. Those are follows

- Database Abstraction Module
- Caching Module

2.2. Testing and Implementation

2.2.1. Implementation

2.2.1.1. Database Abstraction Module

Primary goal of this module is to provide convenient data access point to the classified base web application developer. By default, this support all four major Database vendors namely MySQL]1[, MSSQL]2[and Oracle]3[, PostgreSQL]4[. Module developed using two levels, those are core level and implantation level. Core level contains abstract level classes and interfaces, those create the base architecture of the module. Implementation level classes implement those abstract level classes according to specification.

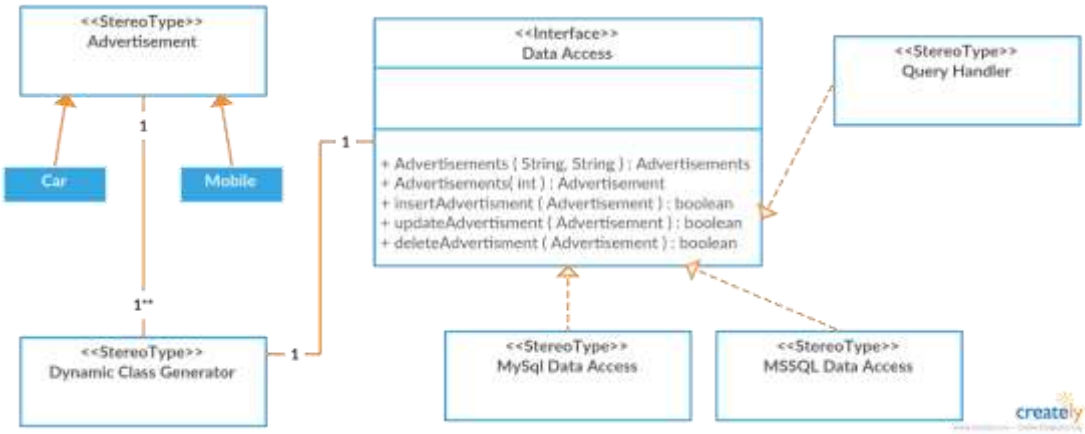

Figure 2: Class Diagram

Figure present high-level class architecture of the module. Data Access interface and Advertisement class belongs to core level. MySQL data access and MySQL data access classes are implement those core level Data Access interface according to Database vendor specifications. Therefore, this module does not depend on vendor, which make extensibility possible. In order to extends Database vendor support, developer only has to implement a Data access class according to particular vendor specifications.

Major issue arise with this component is the management of data structure. Relational Database management systems store data in totally different structure with compare to the document base or NoSQL databases. Most of the cases we can't directly map those

stored data structures into Java object oriented class structure. As a solution team developed a separate class called “Dynamic Class Generator”. Purpose of this class is map non-object relational data structure into object relational data structure.

Team identify a set generic attributes for Advertisement class. Most of the times users request queries using those. Advertisement class provide generic approach for advertisement. Developer can extend this class according to the requirements. For example, if developer want to create a new advertisement category, it can be achieved easily extending this provided advertisement class. This extended class can have any number of attributes according to the requirements. Those attributes are stored in relational database as JSON data type. Major drawback in JSON, is lack of indexing, although the generic Advertisement class attributes can be index, therefor it’s not introduce any critical performance issues.

When inserting data into the database developer only needs to call the insert method at the “query handler” class. This class provide access point for other modules to use this data extension point. Select, insert, update and delete methods can be call with relevant required parameter/s (instance of advertisement). In here developer do not need any database related technological knowledge, method take care of database connection up to query execution. Only thing is developer needs is to configure middleware correctly.

To achieve dynamic behavior according to users’ configurations, team use java reflection API in both “query handler” and “dynamic class generator” classes. Therefore, module able to dynamically identify the database vendor.

```
Class<?> clazz = Class.forName("com.ampliar.dbmodule." + props.getProperty("dbms") + "DataAccess");
Method mInsertAdd = clazz.getDeclaredMethod("insertAdvertisment", Advertisment.class);
Object obj = clazz.newInstance();
mInsertAdd.invoke(obj, add);
```

Figure 3: reflction class creation

As figure describes each method call “query handler” class dynamically identify the relevant database vendor by reading configuration file. Each method call creates relevant database vendor instance dynamically. Main benefit of this, when extending

database vendor support developer only needs to implement Data Access interface according to that vendor specification while following the relevant naming conventions. All underlying task such as identifying the correct vendor handled by the module using this dynamic behavior. All classes that are implement Data Access interface should by the developer use the name suffix with DataAccess(<name> DataAccess). Name must be identical to the “dbms” property value at configuration file.

When extending vendor support, developer be able to work with the particular vendor. Developer can integrate particular vendor driver through maven, then relevant configurations has to be done, finally implantation of relevant vendor <vendor>DataAccess class. In here developer need to implement all the methods which are defined in the interface according to vendors language specifications.

2.2.1.2. Caching Module

Caching is very important concept when in it comes to distributed systems. Most of the time in deployment environments database server deployed separately. Those situations it’s hard to get the database level caching benefits due to the network traffic introduce in between application server and the database server. Again, if we setup caching at database level, when it comes to database extensions, developers required particular database vendor cache configuration knowledge which is ultimately increase the complexity of extension.

Considering those facts team develop separate application level caching module. Module use ehcache]5[as the caching library. Developer can configure this module according to their requirements. “Query handler” class act like the external interface in this class, therefore we directly apply this caching library to its methods. Each search query cache by this library for the specified time by developer.

```
<ehcache xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="ehcache.xsd"
    updateCheck="true"
    monitoring="autodetect"
    dynamicConfig="true">

    <diskStore path="java.io.tmpdir" />

    <cache name="movieFindCache"
        maxEntriesLocalHeap="10000"
        maxEntriesLocalDisk="1000"
        eternal="false"
        diskSpoolBufferSizeMB="20"
        timeToIdleSeconds="300" timeToLiveSeconds="600"
        memoryStoreEvictionPolicy="LFU"
        transactionalMode="off">
        <persistence strategy="localTempSwap" />
    </cache>

</ehcache>
```

Figure 4: Ehcache config.xml

Figure is the sample screen capture of a ehcache configuration file. This is a XML base configuration file and located in src/resources directory. By default, component provide essential caching features of ehcache. Developers can configure this file the way they want. If they wish to use advance features in ehcache they can include those configuration in this file the way the use ehcache with other frameworks or projects.

(See appendix 1)

2.2.2. Testing

Primary goal of testing is to investigate the quality of the product, in this case quality of each modules. Testing allows to understand the potential risk and reliability of a software or any product. Through the testing finding possible errors and other defects helps to maximize the quality of the product at the production environment. Since we

provide a solution for developers, we must ensure product meet possible best quality to avoid developer frustration as well as productivity.

2.2.2.1. Module Testing

Module testing involve test and verify the functionality of each functions inside a module. In this data extension point module, module testing carried out by performing test on each individual function inside both data abstraction module as well as caching module.

2.2.2.2. Integration Testing

Working system means collection of modules or components. Module testing ensure the functionality of each module at the atomic level or isolated level. Integration with other modules or communicating with other related modules may introduce various defects and errors. This module mainly communicates another three modules, also act like the main data supplier for those modules. Team plans to perform integration testing against for major modules, respect to the behavior impact on one module to another module.

2.2.2.3. System Testing

This is the final level of testing a system. Integration testing verify the communication quality between components. In system testing more focus on testing the whole system, in our case complete middleware framework to verify, system achieve all objectives we expected at the beginning of the project. Team plans to perform system testing to verify the security, performance and usability objectives of the middleware framework.

2.3. Research Findings

This research helps us to explore classified base web development domain in depth. We are able to identify what are the requirements of this domain, what are sub domains of this domain as well as how this domain impact on individual businesses life as well as human life. Basically, web base classified makes human life much easier that paper base classifieds. With all those facts we are able compare and analyses existing

products with the classified base web development domain to find problems as well as weakness of those products.

Major issue we found is all those products are limited to one platform in most cases. These products need to deploy at vendors platform. Next issue was lack of customization facilities. None of those products support customizations at end user level (developer level). Developers need to use products directly the way vendor provide it. Also, none of those products are support easy extensibility.

Identifying those weakness and team use those as strengths of our product. Basically, our product use core java, which means these modules are able use with any existing java frameworks. Support for major database management systems if developer wants extend it's also possible with few amounts of codes. No vendor base deployments, developer can deploy choose environment if they wish to deploy. All source codes are open source; therefore, anyone can modify or extend the frame work according to their preferences, also they can offer their contribution to this framework for further enhancements and improvements. The most impotent thing is except the core framework, other modules are in depended from the framework, if someone wish to use this components in their products it is also possible.

(See appendix 2)

3. CONCLUTION

The middleware framework provides an effective and efficient way of developing classified based website. The web development community at current in Sri Lanka is rapidly growing, and this framework can boost their work allowing them to deliver their product at a much lesser time and adding value to them. Because of the current unavailability of such a customizable framework in Sri Lanka, the market segment can easily be captured.

The middleware framework by the authors will be an opensource project that can be worked by the community to be used by the software developers in Sri Lanka and worldwide. Documentation and guidelines will be made available to the developers and the middleware framework which they can use to build their applications on top of that. With time more features can be added which enlarges the user base of this middleware framework.

4. REFERENCES

]1[M. Commiunity, "MySql," MySql,]Online[. Available: https://www.mysql.com/.

]2[Microsoft, "SQL Server," Microsoft,]Online[. Available: https://www.microsoft.com/en-us/sql-server/sql-server-2016.

]3[Oracle, "Oracle Database," Oracle,]Online[. Available: http://www.oracle.com/technetwork/database/database-technologies/sql/overview/index.html.

]4[P. Commiunity, "PostgreSQL," PostgreSQL,]Online[. Available: https://www.postgresql.org/.

]5[J. Community, "Ehcache,"]Online[. Available: http://www.ehcache.org/.

]6[K. L. Solutions, Java Server Programming Black Book, Dreamtech, 2014.

]7[Pivotal, "Spring," Pivotal ,]Online[. Available: https://spring.io/docs.

]8[Q. Luo, S. Krishnamurthy, C. Mohan, H. Pirahesh, H. Woo, B. G. Lindsay and J. F. Naughton, *Middle-Tier Database Caching for e-Business*, p. 12.

]9[J. Community, "Hibernate,"]Online[. Available: http://hibernate.org/.

]10[T.-H. Chen, W. Shang, J. Yang, A. E. Hassan and M. W. Godfrey, "An Empirical Study on the Practice of Maintaining Object-Relational Mapping Code in Java Systems," p. 12, 2016.

]11[A. Biswal, "Java caching frameworks for enterprise applications," Linkedin, 13 March 2015.]Online[. Available: https://www.linkedin.com/pulse/java-caching-frameworks-enterprise-applications-anshuman-biswal.

5. APPENDICES

Appendix 1 EhCache and JDBC

Out of the box, java offers direct methods for caching. Java Hash Map, Hash Table, and JNDI are some of those methods that contain in Java caching System. The major drawback is none of those default methods are provide a mechanism for removing the cache object from the memory when it's no longer need or automatically creation of the objects when it's accessed after expiration.

JBoss, OSCache, EhCache are some open source fireworks where SpritCache, Coherance are commercial frameworks.

Team consider below listed factors in order to decide the optimum caching framework

- Support general purpose caching
- Minimal dependencies
- Documentation level
- Production performance
- Open source

The team left SpritCahe and Coherance because those are not from open source world, there for we cannot use those for open source development.

EcCache was the best selection according to factors that we considered. It's an open source, well-documented framework with minimal decencies. Also, lot of leading java frameworks used EcCache for caching. Spring , Hibernate are few of those.

Data Abstraction layer contains JDBC sub-module and Data Abstraction interface. JDBC is a specification from Sun Microsystem that provides stranded abstraction(API) for java applications to communicate with different databases. JDBC has 4 type of drivers

- Type 1 driver – it acts like a bridge between JDBC and other DB connectivity mechanisms such as ODBC, therefore its depends on ODBC driver which makes application indirectly depends on ODBC

- Type 2 driver - it's a native API driver, its need native binaries installed in running environment

- Type 3 driver – it's a middleware base driver, it converts JDBC call into database specific calls using middleware, which caused to increase the network traffic

Type 4 driver – it's a database protocol driver, this driver converts JDBC call into Database call using database specific protocol. This protocol is proprietary protocol for DBMS vendor

Appendix 2: Table of comparison between existing products and proposed product

Table 2: Existing product comparison table

Features	Yclas	Flynax	Oxy classifides	Titan classifides	Os class	Classipress	Proposed Framework (amplier)
Open Source	*YES*	*NO*	*NO*	*NO*	*YES*	*NO*	*YES*
Free	*YES*	*NO*	*NO*	*NO*	*YES*	*YES*	*YES*
Extensible Database Integration	*NO*	*NO*	*NO*	*NO*	*NO*	*NO*	*YES*
Social Login	*YES*	*YES* *(only facebook)*	*YES*	*NO*	*YES* *(only facebook*	*YES*	*YES*
Extensible Federated Authentication	*NO*	*NO*	*NO*	*NO*	*NO*	*NO*	*YES*
Web Analytics	*NO*	*NO*	*NO*	*NO*	*NO*	*NO*	*YES*