OPEN SOURCE SOA BASED MIDDLEWARE FRAMEWORK FOR CLASSIFIED BASED WEB DEVELOPMENT

Project ID: 17-072

Software Requirements Specification (SRS)

Bachelor of Science Special (Hons) Degree in Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

02nd May 2017

OPEN SOURCE SOA BASED MIDDLEWARE FRAMEWORK FOR CLASSIFIED BASED WEB DEVELOPMENT

Project ID : 17-072

Software Requirements Specification (SRS)

Author: W.M.N Radith

Student ID: IT 14 0110 30

Supervisor: Mr.Nuwan Kodagoda

Bachelor of Science Special (Hons) Degree in Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

02nd May 2017

DECLARATION

I declare that this is my own work and this SRS 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.

W.M N Radith Signature:

ABBREVIATIONS

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 Purpose

The purpose of this document is to present a detailed description of the process and requirements needed for design and develop of **Extensible Database Abstraction module**. It is one of the core modules in “Ampliar” which is an open source classified base web development middleware framework. The Document contains required requirements and the processors follow that are used to capture and create them. The document also explains the functional and non-functional requirements, purpose, features, interfaces and design constraints of the module as well as project approach module core goals and tasks, operation constraints and module interactions with other external modules. The information organization of this document takes a specific approach which leads developers and team to clearly understand the boundaries and functionalities that they need to design and develop in proper order. The target audience of this document is a customer for approval and designer, developers and other stakeholders for successful project goal achievement.

1.2 Scope

The module to be produce called “**Extensible Database Abstraction module**”. It's a database abstraction layer which can be extensible and modify by the developer or programmer according to their DBMS requirements as well as database structure. Technology that will be used to develop the module is Java(Web) and the main objective of this module is reduce the time and effort when there is an extension or a modification.

In Amplier middleware framework all database related task are handled by this **Extensible Database Abstraction module.** Basically **database connection open, querying to data caching** are handled by this module. Therefore module is a combination of two sub-modules, namely data caching sub-module and data abstraction sub-module. Using exposed program centric API, other modules can interact with this module.

1.3 Definitions, Acronyms, and Abbreviations

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

1.4 Overview

The remainder of this SRS document contains three sections, references, and appendices. Second section objective is to give an overall view of the system to the reader. This section contains six sub-sections. Product perspective sub-section give a basic idea about the system interfaces, user interfaces, hardware interfaces, software interfaces, communication interfaces, memory constraints, special operations that need to done by the user and adoption requirements. Second sub-section which is product functions summarize the major functions of the sub-module and third sub-section describes the characteristics or nature of users like to be. All limitations and constraints are described under forth-subsection and descriptions about the future versions and order of requirement implementation are respectively described under sub-section five and six.

The third section takes a detailed description approach of the system under six sub-sections. First sub-section describes user interfaces, Hardware interfaces, Software interfaces and communication interfaces in more detail. Second sun-section contains describe the relevant classes for this project and third sub-section describes the performance requirements including required speeds or time to complete a task. Fourth and fifth subsections are respectively described restriction on design and software system attributes such that reliability, availability.

References and appendices sections help to the reader as supportive materials.

2. Overall Descriptions

The purpose middleware Ampliar is an open source classified base web development framework. The main objective of this project provides a flexible and customizable middleware framework for the classified base web development projects. The team identifies for main modules to build this middleware framework. Integration of following modules creates ampliar middleware.

- Core framework
- Extensible Database Abstraction
- Extensible Federated Authentication
- Web Analytics Generator

In brief core framework module is the heart of the system, which acts as integration point of other modules, Extensible database abstraction modules handle everything related to data accessing and database operations, all authentication, and authorization process is handled by the Extensible federated authentication module and web analytics takes all web analytics responsibilities.

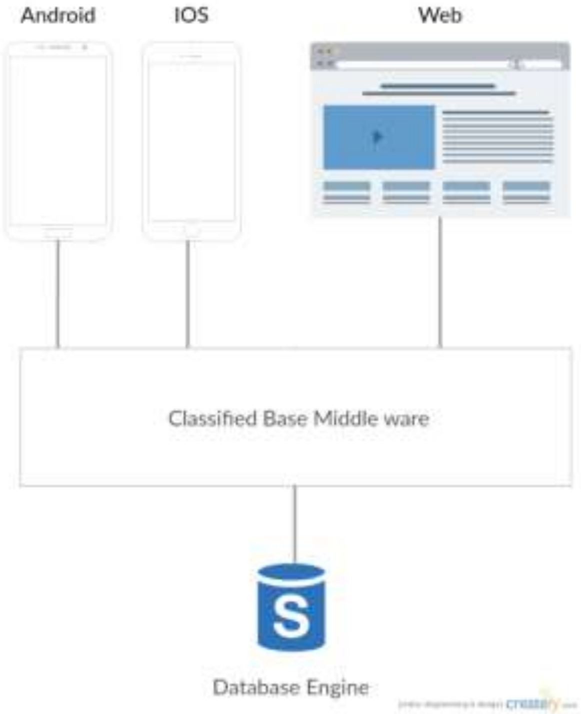


Figure 1: High level view of the complete system

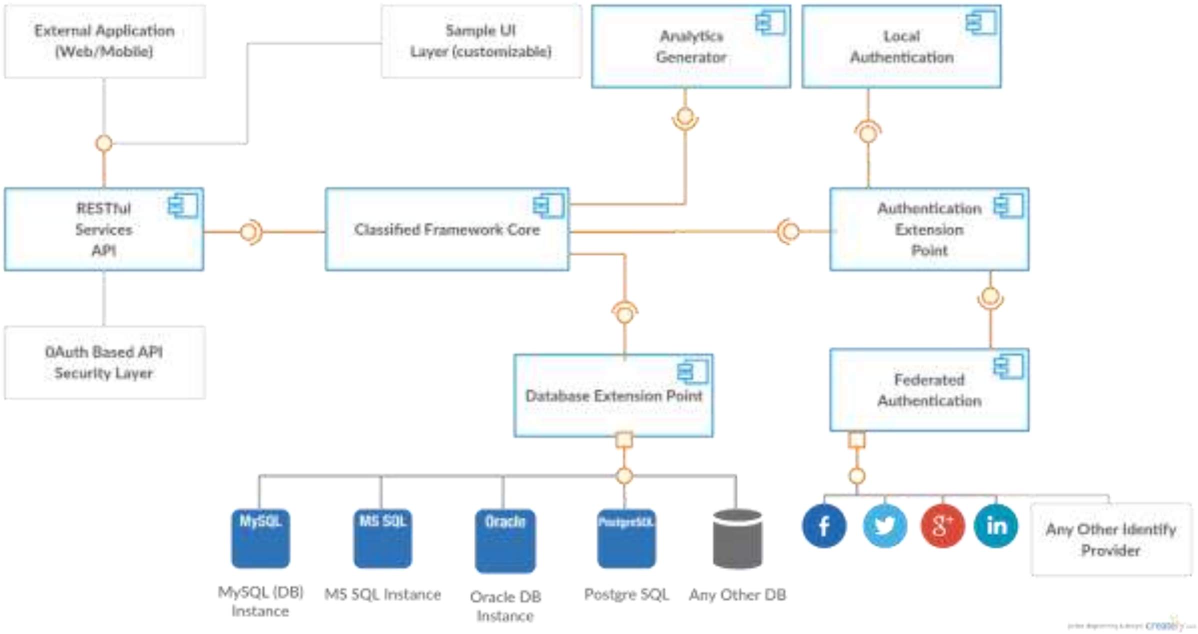


Figure 2: High level architecture diagram of the middleware

This SRS document presents a detailed description of the process and requirements needed for design and develops of **Extensible Database Abstraction module.** This module handles all the database related tasks in middleware framework. Database connection, query execution, caching are few tasks handles by this sub-module. Also, module DBMS vendor support can be extensible by the developer or programmer according to project requirements by consuming less time and effort. Easy extensibility is one of main objectives of this module.

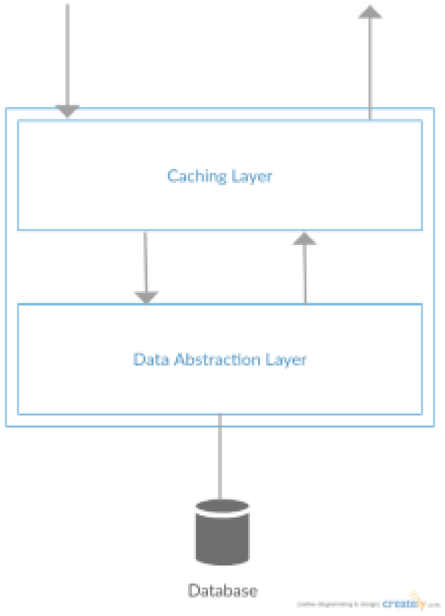


Figure 3 :Extensible Database Abstraction Module

As presented in figure 3, the module contains two sub-modules. First sub-module is the data caching layer. This layer cache frequent accessing data for a given constant time. Developer or programmer can configure this constant time through a configuration file. This layer ensures performance by reducing database access for frequent requesting data.

Data abstraction layer is the second sub-module which directly works with the Database engine. This layer directly performs CRUD operations on the database. By default, layer support MySQL [1], MSSQL [2], Oracle DB [3] and PostgreSQL [4], although layer follows extensible architecture which developer or programmer can extend DBMS support for another vendor with minimum effort and time. Module exposes API for other modules for communication.

2.1 Product perspective

There are a number of products available in classified base web development. Most of those products present as web platforms, the user can create an application using that platform. After the creation user needs to deploy on same vender's servers because these products are platform dependent. Also, customization ability is in very poor level in those products. Almost all products support only MySQL [1]. Therefore, we can see the lack of extensible, customizable and vendor platform in-depended product in this classified based web development segment. Some of the currently available products are listed below

YCLAS [5]

Yclas is a tool which enables the users that are little or no knowledge in web development, to create classified web sites. Deployment should be done on a platform that is provided by Yclas.

FLYNAX [6]

It's a PHP script developed based on MVC architecture.

OXY CLASSIFIDES [7]

PHP script base product that helps to build classifieds, this product can be customizable for some extend.

TITAN CLASSIFIDES [8]

Can create and manage professional level clean classifieds. Customization can be done through vendors' developers. Also can't be used for free.

OS CLASS [9]

It is a PHP script that allows you to quickly create and manage your own free classifieds site.

CLASSIPRESS

It is a word press plugin which helps to create and manage WordPress base classified websites

(See appendix 1 for existing products and purposed product comparison table)

2.1.1 System Interfaces

Purposed system or module does not interact with or it's not a part of any existing system.

2.1.2 User Interfaces

Since this is a module of a middleware framework it's impossible to derive direct user interfaces. Although arguably possible to include total user experience of the module for this section. As developer or programmer perspective effort that consumes for customization, extension, code understandability, maintainability, and configurability can be included. From users' perspective response time, latency, accuracy, feedback can be included.

2.1.3 Hardware Interfaces

The module does have any hardware interfaces, therefore module does not require any specific hardware interfaces. Database server hardware connections are managed by the operating system of the web server. Although module involving different files and data storing, retrieving, caching and processing it's important to have at least 4GB RAM, 500GB hard with drive 2GHz processor for smooth operation. For the deployment of the complete middleware, Glassfish server is recommend.

2.1.4 Software Interfaces

Module interacts with several software interfaces.

- EhCache
- JDBC API
- MYSQL Server
- MSSQL Server
- Oracle Database Server
- PostgreSQL Database Server

EhCache is a famous and open source java caching library. Purposed caching module going be implement using this library. The module uses JDBC API for database connection and interactions. JDBC is an industry stranded database connection API for java. Purposed module by default support for four major DBMS, therefore software interface for each vendor need to be included. The module does not use any ORMS because ORMS in java reduce the performance as well as increase the complexity [1]of code which ultimately reduces the usability of developer or programmer.

(See appendix 2 Ehcache and JDBC comparison and more details)

2.1.5 Communication Interfaces

Module exposed program centric API for database operations. Other modules can use this API to perform database related tasks. Communication between those sub-modules is handled by JVM.

2.1.6 Memory Constrains

Web server minimum RAM requirement is 4GB for smooth operations. Because there is some memory consuming functions are available on the module such as caching, query processing.

2.1.7 Operations

Main targeted user types of this module are Developers or programmers. Following configurations need to be done by the developer,

- DBMS configurations
- cache duration configurations
- Backup and recovery configurations

2.1.8 Site adaptation requirements

To use this module in their own development and productions developers meet below requirements

- JDK 1.7 or higher
- JDBC
- Glassfish server 4
- IDE (eclipse, IDEA)
- DBMS (MySQL, Oracle, PostgreSQL or MSSQL)

Also, the developer needs a web browser to launch available demo application. Middleware documentation and configurations are implemented only in English. This middleware can use as a service provider for SPA, Android, IOS or another web application.

2.2 Product functions

The module is more oriented on developers' functionalities. Although implementing classified web application using purposed middleware, end user also able to perform functions though this module. Those are listed and described below.

End User

- User login
- User registration
- Classified advertisement creation
- Classified advertisement update
- Classified advertisement delete
- Classified advertisement search
- Classified advertisement sort
- Search classified advertisements
- Sort Classified advertisements

Developer

- DBMS configurations
- Configure caching
- Extend Database support
- Customize database structure
- Backup configurations

(See appendix 2 EDAM use case diagram)

User registration

Use Case Number	01
Use Case	Registration
Actor	End-user
Pre-Condition	User should be on the web site
Flow of Event	• User select sign up • User fill relevant details on sign up form • User submit form

Post-Condition	Display success message and redirect to users profile
Alternatives	Display error message if there something went wrong

Table 1: user registration use case

User Login

Use Case Number	02
Use Case	Login
Actor	End-user
Pre-Condition	User should be registered user
Flow of Event	• User select sign in • User enter username and password • User submit form
Post-Condition	Display success message and redirect to users profile
Alternatives	Display error message if there something went wrong

Table 2: User Login Use Case

Classified advisement creation

Use Case Number	03
Use Case	Classified advisement creation
Actor	End-user
Pre-Condition	User should be signed in
Flow of Event	• User select create advertisement button • Fill relevant details • User submit form
Post-Condition	Display success message and redirect to created advertisement
Alternatives	Display error message if there something went wrong

Table 3: Classified advisement creation use case

Classified advisement update

Use Case Number	04
Use Case	Classified advisement update
Actor	End-user
Pre-Condition	User should be signed in
Flow of Event	• User select created advertisement • User select edit button

	• Edit relevant details • User submit form
Post-Condition	Display success message and redirect to updated advertisement
Alternatives	Display error message if there something went wrong

Table 4:Classified advisement update use case

Classified advisement delete

Use Case Number	05
Use Case	Classified advisement delete
Actor	End-user
Pre-Condition	User should be signed in
Flow of Event	• User select created advertisement • User select delete button
Post-Condition	Display success message and redirect to home
Alternatives	Display error message if there something went wrong

Table 5: Classified advisement delete use case

Classified advisement search

Use Case Number	06
Use Case	Classified advisement search
Actor	End-user
Pre-Condition	User should be in the web application
Flow of Event	• User enter keyword • User select search button
Post-Condition	Display relevant search result
Alternatives	Display error message if there something went wrong or search result is unavailable

Table 6: Classified advisement search use case

Classified advisement Sort

Use Case Number	07
Use Case	Classified advisement filter
Actor	End-user
Pre-Condition	User should be in the web application
Flow of Event	• User filtering method in drop down

Post-Condition	Display sorted result
Alternatives	Display error message if there something went wrong

Table 7:Classified advisement filter use case

DBMS configurations

Use Case Number	08
Use Case	DBMS configurations
Actor	Developer
Pre-Condition	User should open the configuration file using a text editor
Flow of Event	• User enter configurations (DB vender, credentials, Database)
Post-Condition	Connect to the database successfully
Alternatives	Console log error if there something went wrong

Table 8: DBMS configurations use case

Caching configurations

Use Case Number	9
Use Case	Caching configurations
Actor	Developer
Pre-Condition	User should open the configuration file using a text editor
Flow of Event	• User enter caching configurations (time)
Post-Condition	Execute configurations successfully
Alternatives	Console log error if there something went wrong

Table 9: Caching configurations use case

Extend database vendor support

Use Case Number	11
Use Case	Extend database vendor support
Actor	Developer
Pre-Condition	User need to develop custom sub-module for particular vendor following the module architecture
Flow of Event	• User plug developed module
Post-Condition	Execute module successfully
Alternatives	Console log error if there something went wrong

Table 10 : Extend database vendor support use case

Customize database structure

Use Case Number	10
Use Case	Customize database structure
Actor	Developer
Pre-Condition	User need to perform customization following module architecture
Flow of Event	• User plug customized structure
Post-Condition	Execute stature successfully
Alternatives	Console log error if there something went wrong

Table 11: Customize database structure use case

Backup configurations

Use Case Number	12
Use Case	Backup configurations
Actor	Developer
Pre-Condition	User should open the configuration file using a text editor
Flow of Event	• User enter backup configurations (duration, path)
Post-Condition	Execute configurations successfully
Alternatives	Console log error if there something went wrong

Table 12: Backup configurations use case

2.3 User characteristics

This module has two type of user, thus are end-user and developers. The module does not interact with **end user** directly, therefore end user do not need any special skills or requirements, basic computing skills or mobile app competency would enough.

Developers are directly involved with this module, therefore they need to be familiar with java web technology, middleware concepts, SOA concepts, development tools such as IDE, maven, jerkins, EhCahe, Git, SQL. Although basic to the intermediate competency in those areas enough to work and customize the module.

2.4 Constrains

- Server should be able to handle multiple requests
- Server resource should be increased with respective to the user amount increase
- Developers should follow module architecture and proper guidelines when customizing the module.
- For development, all required tool and technologies should be installed on the system
- For production environment, relevant tools and applications should be installed on the system
- Both user types need be confident in the English language
- Caching should be configured and enable to ensure the middleware performance
- Customization may produce performance, security, integration issues if it’s not done in proper manner

2.5 Assumptions and dependencies

Assumptions

- Middleware and it’s all modules work in any operating system independently
- Future java versions, libraries, tools will support for middleware and it’s all sub-modules
- No data losses, corruption during the communication between middleware and modules inside the middleware
- When customizations developers use recommended tools, architecture, best practices and guidelines to ensure the stability of the module.

Dependencies

- Middleware and its modules development and deployment process are done on recommended environment with using recommended tools

2.6 Apportioning of requirements

Data abstraction layer one of the sub- module of this module, will be developed first. This sub-module takes bottom-up development approach, which firstly develop a generic database scheme for classified base web application and following that structure, develop and define data abstraction sub-module. After that the development of caching sub-module will start, it’s the other sub- module and its depends on data abstraction sub-module, therefore its need to follow the data abstraction sub-module.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The module does not have any direct user interfaces.

3.1.2 Hardware Interfaces

Server side there should be minimum space of 10GB, the server should install and run glassfish server. And relevant DBMS should be installed and configured. JDK 1.7 or higher required for executing all modules.

The development team at least have computers, 4GB RAM, 20GB disk space 5400RPM hard drive, 1.6 GHz processing power, and 1366x768 screen resolution monitors.

To test the middleware the development team do not need any specific hardware interfaces, although if they are planning to create a mobile application using this middleware, then a mobile device need for test that application.

To refer documentation, download dependencies and test middleware in hosted environment development team need an internet connection.

3.1.3 Software Interfaces

Name	**Version**	**Purpose**
JDK	1.7 or higher	Execute sub-modules
Glassfish server	4.0	Deploy application
MySQL	5.7 or higher	Database operations
MSSQL	2012 or higher	
Oracle	12c	
PostgreSQL		
GIT	Latest stable version	Version controlling
Source tree	Latest stable version	GIT GUI client
Apache jmeter	3.2	Performance testing
Apache maven	3.3	Dependency Management
Jenkins	2.46	Continuous integration
Google Chrome	Latest stable version	Developments
Postman	Latest stable version	Test APIs
Junit	4.12	Unit testing
Intelij IDEA	16.0 or higher	Integrated development environment

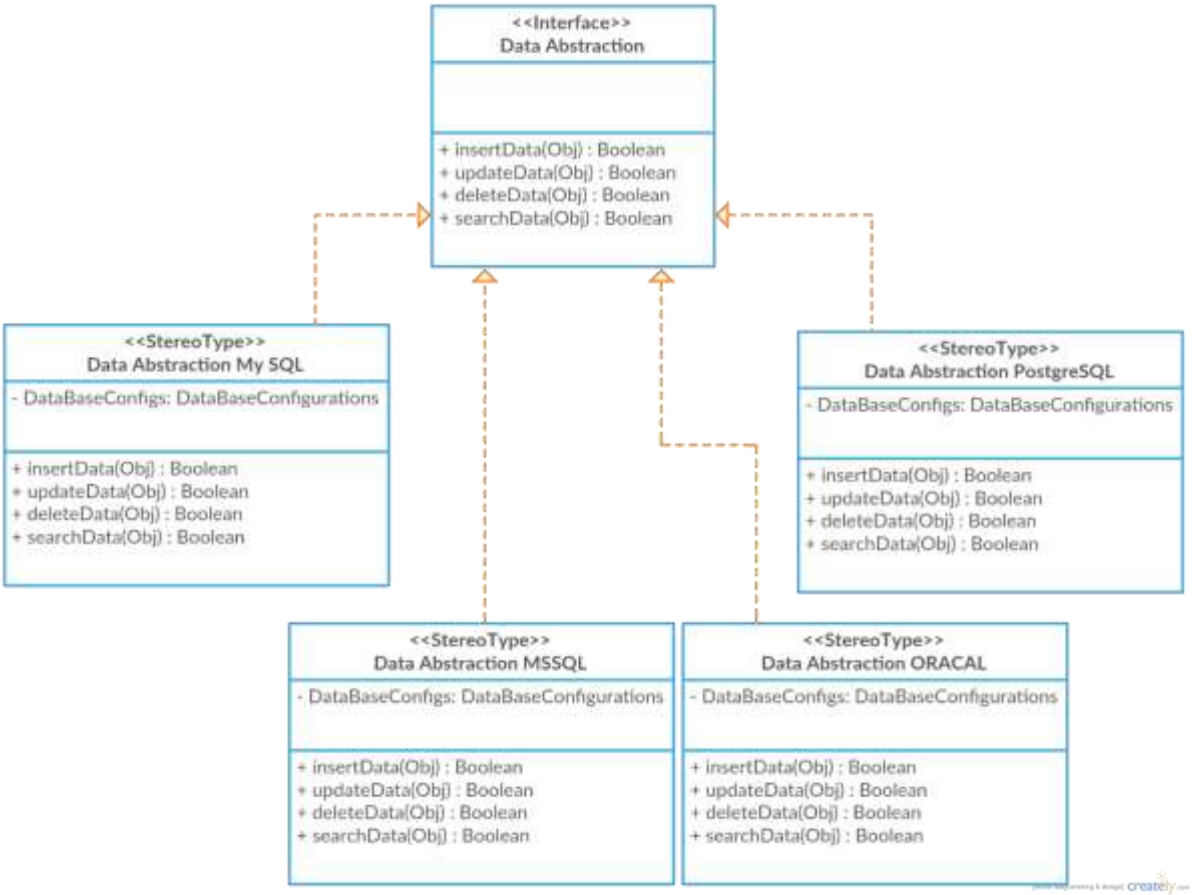
FindSecBugs	Latest stable version	Security Testing
OWASP		
Dependency Check		
OWASP ZAP		
Ehcache	3.0	Caching layer implementations

Table 13 : list of software interfaces

3.1.4 Communication Interfaces

Middleware contains separate RESTful API module which exposes the complete functionality of the middleware for outside world. That module required HTTP and internet connection to communicate between client application. Other modules in inside the middleware including extensible database abstraction module use program centric API to perform communications among them. Therefore, all internal communications in the middleware framework are handled by the JVM.

3.2 Classes / Objects



<<Interface>>
Data Abstraction
+ insertData(Obj) : Boolean
+ updateData(Obj) : Boolean
+ deleteData(Obj) : Boolean
+ searchData(Obj) : Boolean
<<StereoType>>
Data Abstraction My SQL
- DataBaseConfigs: DataBaseConfigurations
+ insertData(Obj) : Boolean
+ updateData(Obj) : Boolean
+ deleteData(Obj) : Boolean
+ searchData(Obj) : Boolean
<<StereoType>>
Data Abstraction PostgreSQL
- DataBaseConfigs: DataBaseConfigurations
+ insertData(Obj) : Boolean
+ updateData(Obj) : Boolean
+ deleteData(Obj) : Boolean
+ searchData(Obj) : Boolean
<<StereoType>>
Data Abstraction MSSQL
- DataBaseConfigs: DataBaseConfigurations
+ insertData(Obj) : Boolean
+ updateData(Obj) : Boolean
+ deleteData(Obj) : Boolean
+ searchData(Obj) : Boolean
<<StereoType>>
Data Abstraction ORACAL
- DataBaseConfigs: DataBaseConfigurations
+ insertData(Obj) : Boolean
+ updateData(Obj) : Boolean
+ deleteData(Obj) : Boolean
+ searchData(Obj) : Boolean
creately

Figure 4: sample class diagram for data abstraction sub module

3.3 Performance Requirements

Middleware and it’s all modules need to ensure performance at in optimal level. Classified websites contain thousands of users, and performance of the underlying applications are crucial in this domain. Following performance requirements need to be full fill by the module.

Responsiveness: Delay between request and response should be less than 1 seconds. To achieve this caching layer cache frequently requested search quires. This will minimize the database call, which ultimately reduces the response time.

Load Balancing: Since there are huge amount of user, modules need to be capable of handling multiple requests at a time

Internet Connection: its important have proper internet connection because poor internet connection will reduce the responsiveness of client application

Minimized dependencies: if there are lot of dependencies with external libraries, especially with ORMs performance of the module may suffer

3.4 Design constraints

Usability of the application is depending on the client application which is using as this middleware as a service provider.

At this stage, Extensible database abstraction module does not support multiple database connections or users at one time.

At this stage, Extensible database abstraction module does not support network or distributed databases.

Each sub-module in the module need to be develop using the proper coding stand as well as proper architecture.

All functionalities and usage of those need to be clearly documented.

3.5 Software system attributes

3.5.1 Reliability

All data related task in this middleware are handled by this sub-module since data is the most valuable assets in any system this module, this module needs to achieve possible maximum reliability

- The module needs to work 24 hours and make periodic backups of the database in order to recover from database crashes or something goes wrong.
- Any data corruption should not happen
- Module needs to test in each level with different test methods to keep the failure probability at possible lowest level
- If any failure occurs, there should be a mechanism to notify relevant authorities in possible minimum time.

3.5.2 Availability

Availability of this module is very crucial, because without data the complete mindware and it's all other modules functions might be corrupted.

- Cheche pool or layer need to be perform caching at optimal level
- There should not be any caching overflows.
- Frequent accessing data need to be cache to minimizes the query processing. [2]
- Other modules of the middleware should be able to use this module at any given time period

3.5.3 Security

Security needs to be concern at very high level for this module because data is the most protective asset of any system today. All data access only be able with proper authorization.

- Unauthorized access to the data should be permitted
- All backups should be stored in a secure location
- All external exposed API used HTTPS protocol
- User separate API security layer
- Perform various levels and types of security tests

- User proper technics to avoid SQL related security issues

3.5.4 Maintainability

Module all major maintenance is performed by the development team. Since these is an open source project anyone can contribute or customize this middleware. All modules of the framework use a properly modularized approach which reduces the maintaining burden. All development and designing process follow proper coding stranded and documentations for help the other developers to maintain the modifications and customizations in future.

3.6 Other requirements

Proper documentations and help and support

Since this is a development tool proper documentation and guidelines need to be prepared to help developers who use this middleware for their development projects. All functionalities, customizations, pros, cons need to be included in those documentations.

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] "Yclas Home," Yclas, [Online]. Available: https://yclas.com/.

[6] "flynax home," flynax, [Online]. Available: https://www.flynax.com/.

[7] "OXY CLASSIFIDES Home," OXY CLASSIFIDES, [Online]. Available: https://www.oxyclassifieds.com/.

[8] "TITAN CLASSIFIDES Home," TITAN CLASSIFIDES, [Online]. Available: http://www.titanclassifieds.com/.

[9] "OS CLASS Home," OS CLASS, [Online]. Available: https://osclass.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] 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.

[12] 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.

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

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

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

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

APPENDICIES

Appendix 1: Table of comparison between existing products and proposed product

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*

Table 14: existing product comparison table

Appendix 2: 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. [12]

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 [13] 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 [13] for caching. Spring [14], Hibernate [15] 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 [16]
- **Type 2 driver -** it's a native API driver, its need native binaries installed in running environment [16]
- **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 [16]

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 [16]

Appendix 3: EDAM use case diagram

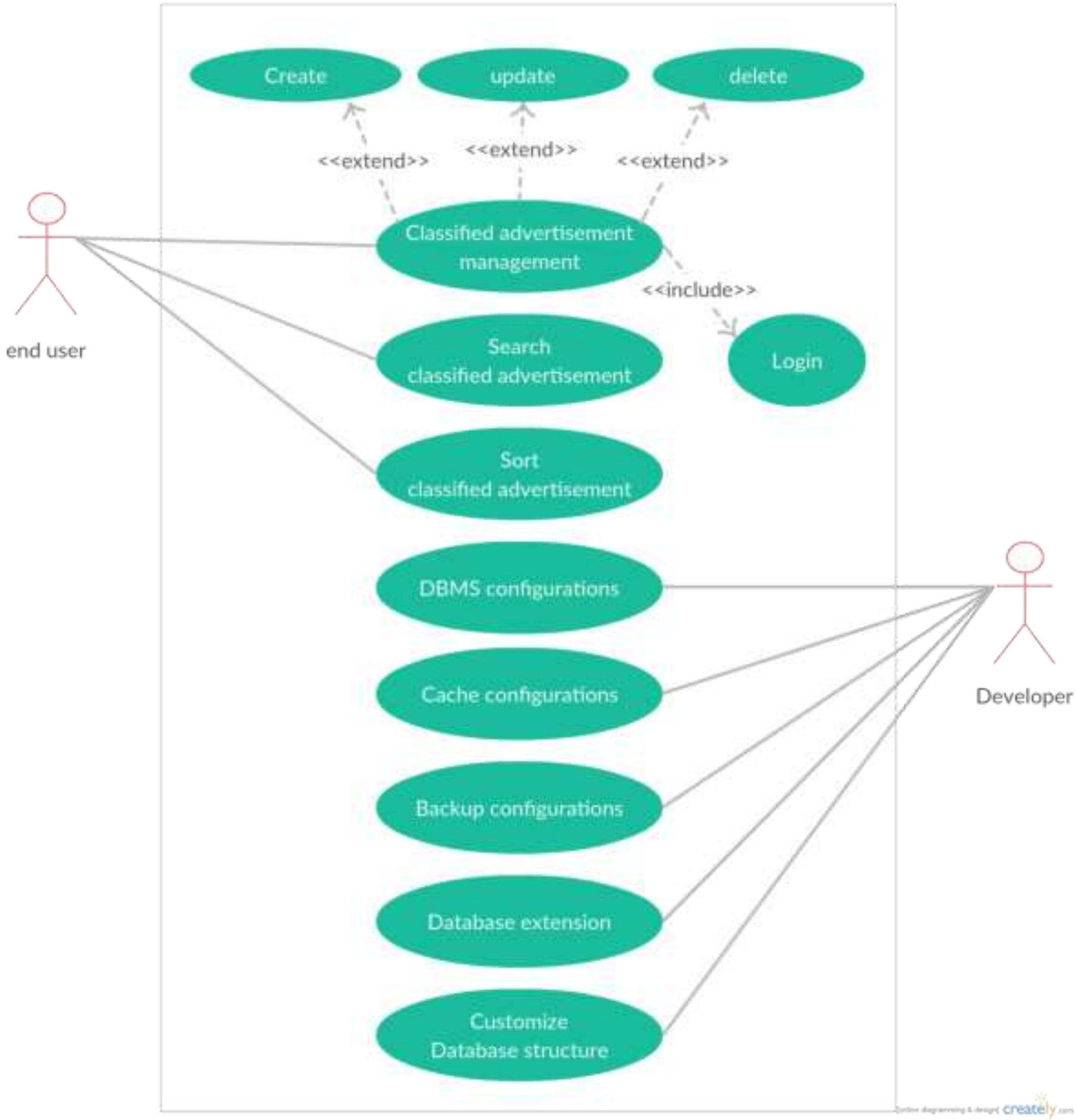


Figure 5: EDAM use case diagram