
Web Enabled Service Implementation - Telephone Directory Prototype

Software Design Description

Project Manager:

Chia-Chu Chiang, Ph. D.

Project Team Members:

Benjamin Balogh
David Miller
Matthew Reed

Submitted:

March 24, 2006

Final Product:

May 8, 2006

Change History

<u>Date</u>	<u>Version</u>	<u>Change Author</u>	<u>Change Details</u>
03/19/2006	1.0	Benjamin A. Balogh	Original Document
03/22/2006	1.1	Benjamin A. Balogh	Completed Iteration 1 Draft
03/23/2006	1.2	Matt W. Reed	Added Definitions and References
03/24/2006	1.3	David C. Miller	Added Class Inheritance Diagrams
05/02/2006	1.4	Benjamin A. Balogh	Update Diagrams & Description
05/05/2006	1.5	Benjamin A. Balogh	Final Product

Signature Page

The following signatures are required for approval of this document.

Benjamin A. Balogh
Project Programmer

Date

David C. Miller
Project Programmer

Date

Matthew W. Reed
Project Programmer

Date

Chia-Chu Chiang
Project Manager

Date

Contents

Change History	2
Signature Page	3
1 Introduction	5
1.1 Purpose	5
1.2 Scope	5
1.3 Definitions	6
1.4 References	7
1.5 Overview	7
2 Overall Design	8
2.1 Web Server	8
2.1.1 Apache Server	8
2.1.2 Apache Axis	8
2.2 Backend Application	9
2.2.1 Apache Tomcat.....	9
2.2.2 Access Point Telephone Directory Web Application.....	9
2.2.3 Access Point Telephone Directory Web Service.....	9
2.3 Development Tool.....	9
2.3.1 Eclipse Development Tool	9
2.4 Testing Tool	9
2.4.1 HTTP_Load.....	9
2.5 Database	10
3 Design Description	10
3.1 Access Point Telephone Directory Web Application.....	10
3.1.1 Data Flow Diagram	10
3.2 Access Point Telephone Directory Web Service.....	11
3.2.1 UML Diagram	11
3.2.2 CustomerPortBindingImpl Class.....	11
3.2.3 CustomerPortBindingSkeleton Class	13
3.2.4 CustomerPortBindingStub Class	14
3.3 Database	15
3.3.1 Entity-Relationship Diagram.....	15
3.3.2 Data Dictionary	16
4 Appendix - Software Diagrams	17
4.1 Access Point Software Architecture Diagram	17
4.2 Access Point Data Flow Diagram.....	18
4.3 Access Point UML Diagram	19
4.4 Access Point E-R Diagram.....	19

1 Introduction

1.1 Purpose

The purpose of this document is to provide a comprehensive design for all components of the Access Point product. This document will describe the software architecture and system design for the final Access Point Software Solution. This document is intended to provide all necessary design information to Rock Software Engineering, Inc. [RSE] for development purposes. This document is also intended for our client (C-CC, Limited [C-CCL]), to be used as a reference for software review and approval.

1.2 Scope

This document will describe in detail the software architecture and detailed system design for the Access Point product. This prototype for the Access Point product will encompass the Access Point Web Enabled Service and Access Point Telephone Directory Application.

This document will cover the system components necessary to develop the final Access Point software solution:

- Web Server
- Backend Application
- Database

This document will not cover the end user information necessary to access the Access Point software solution.

1.3 Definitions

Access Point –

Name of the final software produced for this project.

Backend Application –

An application that runs on a server, which accesses a database of information and returns results to the user. The application is invoked when the user logs in to the Web Enabled Service and requests information from the database or issues a command.

C-CC, Limited [C-CCL] –

Fictional Client being used for this prototype. Stands for Chia-Chu Chiang, Limited.

Rock Software Engineering, Incorporated [RSE] –

Fictional company developing the Access Point product. Comprised of the Project Team members listed on the title page.

Server –

A computer accessed remotely by the end-user, used for running applications and storing/retrieving data.

Simple Object Access Protocol [SOAP] –

SOAP is a standard for exchanging XML-based messages over a computer network, normally using HTTP.

User/End User –

An individual accessing the Access Point product, usually an employee of the client company.

Web Enabled Service [Web Service] –

Software based solution that allows the end user to run applications and store or retrieve information remotely via the Internet, often through a standard web browser.

Web Services Description Language [WSDL] –

A format for describing a Web Services interface. It is a way to describe services and how they should be bound to specific network addresses.

1.4 References

“The Apache Software Foundation”. <www.apache.org>, 2006.

“Eclipse Project”, <http://www.eclipse.org/>, May 1st, 2006.

“[http_load - multiprocessing http test client](http://www.acme.com/software/http_load/)”,
http://www.acme.com/software/http_load/, May 1st, 2006.

IEEE Guide to Software Requirements Specifications, ANSI/IEEE Std 830-1984,
IEEE Computer Society, 1984.

IEEE Guide to Software Requirements Specifications, ANSI/IEEE Std 1016-1998,
IEEE Computer Society, 1998.

Interview with Chris Morris, IBM WebSphere Developer, Ascendant Technologies,
April 20th, 2006.

“SOAP”. <http://en.wikipedia.org/wiki/Simple_Object_Access_Protocol>, 2006.

“Web Services And Services-Oriented Architectures”. <<http://www.service-architecture.com>>, 2006.

1.5 Overview

This document is divided into three Document Sections and one Appendix to describe the specifications and requirements for the development of the Access Point product.

Introduction:

- Overview of the Design Documentation

Design Entity:

- Entity Description

Design Description:

- Detailed Diagram
- Detailed Description

Appendix

- Overall Design Diagrams

2 Overall Design

2.1 Web Server

2.1.1 Apache Server

Apache Server or Apache HTTP Server will provide access to the Web Service via HTTP protocol. It is responsible for providing the actual web platform that the service will run on.

Figure 4.1 in the Appendix contains a Software Architecture Diagram showing where Apache Server will be placed within the Access Point product.

2.1.2 Apache Axis

Apache Axis is an implement the Simple Object Access Protocol (SOAP), the protocol for exchanging data in a distributed environment. Axis will listen for client interaction and control movement of data between the client and the Web Enabled Services as necessary. Axis also handles the Web Services Description Language (WSDL), an XML-based language for describing Web Services and how the data is to be structured while transported via SOAP.

Figure 4.1 in Appendix A contains a Software Architecture Diagram showing where Apache Axis will be placed within the Access Point product.

Figure 5.1 in Appendix B contains a copy of the actual WSDL used during Access Point Telephone Directory Service processing.

2.2 Backend Application

2.2.1 Apache Tomcat

Apache Tomcat is the OpenSource software product being implemented as the Application Server in the Access Point product.

2.2.2 Access Point Telephone Directory Web Application

The Access Point Telephone Directory Web Application is comprised of 3 Java Server Pages that work together to display each component available within of the Access Point Telephone Directory Web Service.

2.2.3 Access Point Telephone Directory Web Service

The Access Point Telephone Directory is the Web Enabled Service for the Access Point product prototype. This service is designed to validate a User using the User Name, Password and Business Name fields. After this Validation occurs, the Web Service will then Take a Last Name, with or without a First Name from the End User to return the Phone Number stored in the database.

Figure 4.1 in the Appendix contains a Software Architecture Diagram showing where the Access Point Telephone Directory Web Service will be placed within the Access Point product.

2.3 Development Tool

2.3.1 Eclipse Development Tool

For development purposes an Open Source TomCat Application tool will be utilized to implement the Web Service and to provide a testing environment for deployment. This tool is called Eclipse ,and can be found on the Website listed in the References.

2.4 Testing Tool

2.4.1 HTTP Load

For the purposes of load and performance testing, an Open Source command line tool called HTTP_Load will be utilized. This tool allows for a user to test the number of connections a Web Server and Web Service can handle before performance begins to suffer. A link to this tool can be found in References.

2.5 Database

A database will be used to house all stored data relating to all aspects of the Access Point product. For the prototype this will be a MySQL database housed on the server provided by CCCL.

Figure 4.1 in the Appendix contains a Software Architecture Diagram showing the role of the MySQL Database will be placed within the Access Point Telephone Directory Service.

3 Design Description

3.1 Access Point Telephone Directory Web Application

3.1.1 Data Flow Diagram

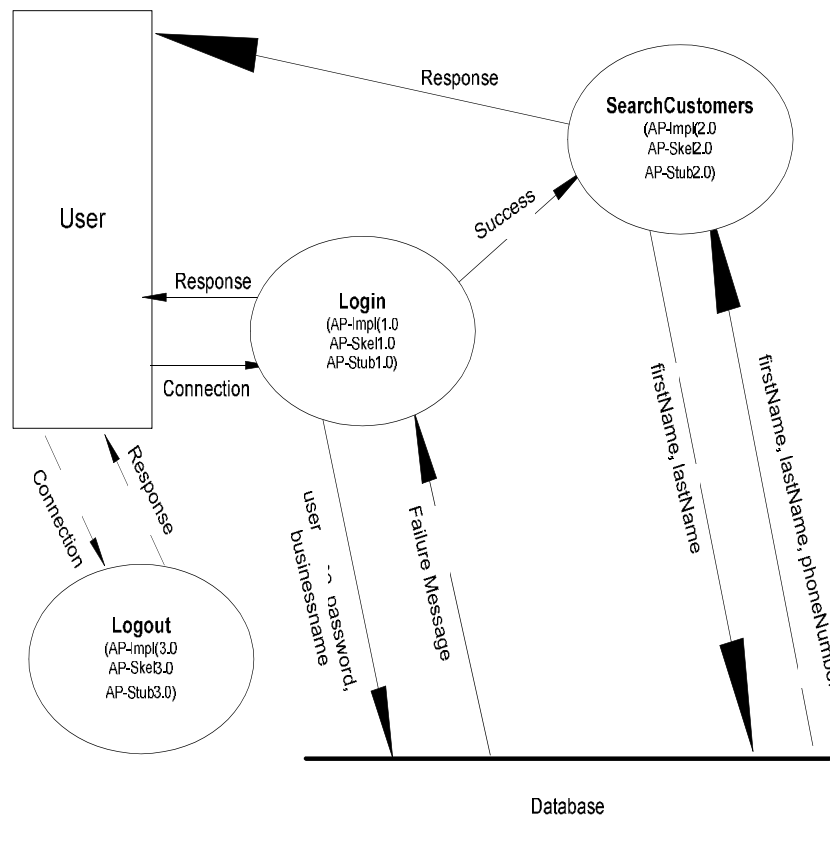


Figure 4.2 in the Appendix contains a copy of this Data Flow Diagram.

3.2 Access Point Telephone Directory Web Service

3.2.1 UML Diagram

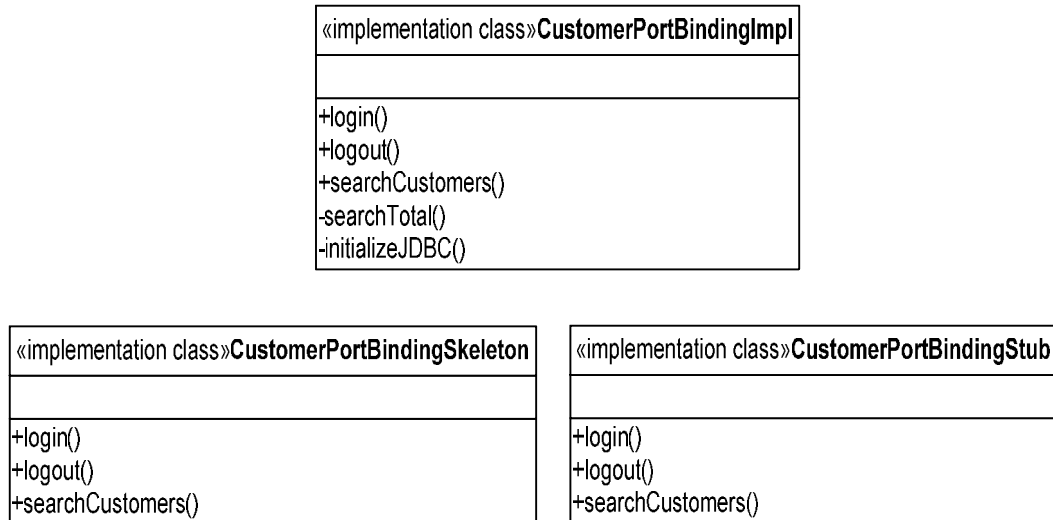


Figure 4.3 in the Appendix contains a copy of this UML Diagram.

3.2.2 CustomerPortBindingImpl Class

3.2.2.1 *Login(string, string, string)*

Method ID	AP-Impl1.0
Method Name	Public CustomerPortBindingImpl::Login (public)
Return Type	boolean
Input Parameters	string username string password string businessname
Output Parameters	boolean
Error Messages	Invalid Login Information
Files Accessed	MySQL Database
Files Changed	None
Method Called	InitializeJDBC()
Narrative	Loads the Database Driver for use in database connections. Takes Input fields User Name, Password, and Business information Compares these entries in a database table of user information Returns valid or invalid result based on comparison.

3.2.2.2 Logout(boolean)

Method ID	AP-Impl3.0
Method Name	Public CustomerPortBindingImpl::Logout (public)
Return Type	boolean
Input Parameters	boolean valid
Output Parameters	boolean
Error Messages	User Not Logged In
Files Accessed	None
Files Changed	None
Method Called	None
Narrative	Checks output from login method and changes that value to false.

3.2.2.3 searchCustomers(string,string)

Method ID	AP-Impl2.0
Method Name	Public CustomerPortBindingImpl::searchCustomers (public)
Return Type	string
Input Parameters	string firstName string lastName
Output Parameters	com.example.www.Customer.xsd.Customer[]
Error Messages	Database Connection Error Name Not Found
Files Accessed	MySQL Database
Files Changed	None
Method Called	searchTotal(string,string)
Narrative	Takes the first and last name strings from customer Connects to MySQL Database Searches Customers Table for Matching Information Returns All Matches or Name not Found Error Message

3.2.2.4 searchTotal(string,string)

Method ID	AP-Impl2.1
Method Name	Private CustomerPortBindingImpl::searchTotal (private)
Return Type	Integer
Input Parameters	string firstName string lastName
Output Parameters	Integer size
Error Messages	Database Connection Error Query Failed Error
Files Accessed	MySQL Database
Files Changed	None
Method Called	None
Narrative	Takes the first and last name strings from searchCustomer method Determines if the first and last name fields are populated or null. If lastName is null, no query is performed. Else if firstName is null, the database is queried to count matching lastName. Else if both names exist, the database is queried to count matches on both names. Returns the Number of Matching Entries to searchCustomers.

3.2.2.5 *InitializeJDBC()*

Method ID	AP-Impl1.1
Method Name	CustomerPortBindingImpl::InitializeJDBC (private)
Return Type	Void
Input Parameters	None
Output Parameters	None
Error Messages	Driver Not Found
Files Accessed	None
Files Changed	None
Method Called	None
Narrative	Load Driver for Database Connection.

3.2.3 CustomerPortBindingSkeleton Class

3.2.3.1 *Login(string, string, string)*

Method ID	AP-Skel1.0
Method Name	CustomerPortBindingSkeleton::Login (public)
Return Type	boolean
Input Parameters	string username, string password, string businessname
Output Parameters	boolean
Error Messages	Login Error Messages
Files Accessed	None
Files Changed	None
Method Called	CustomerPortBindingImpl::Login
Narrative	Calls the Login Method to perform login through the webservice.

3.2.3.2 *Logout(boolean)*

Method ID	AP-Skel3.0
Method Name	CustomerPortBindingSkeleton::Logout (public)
Return Type	boolean
Input Parameters	boolean valid
Output Parameters	boolean
Error Messages	Logout Error Messages
Files Accessed	None
Files Changed	None
Method Called	CustomerPortBindingSkeleton::Logout
Narrative	Calls the Logout Method to perform logout through the webservice.

3.2.3.3 *searchCustomers(string,string)*

Method ID	AP-Skel2.0
Method Name	CustomerPortBindingSkeleton::searchCustomers (public)
Return Type	string
Input Parameters	string firstName, string lastName
Output Parameters	com.example.www.Customer.xsd.Customer[]
Error Messages	CustomerSearch Error Messages
Files Accessed	None
Files Changed	None
Method Called	CustomerPortBindingImpl::searchCustomers
Narrative	Calls the searchCustomers Method to perform a Customer Search through the Web Service.

3.2.4 CustomerPortBindingStub Class

3.2.4.1 *Login(string, string, string)*

Method ID	AP-Stub1.0
Method Name	CustomerPortBindingStub::Login (public)
Return Type	boolean
Input Parameters	string username string password string businessname
Output Parameters	boolean
Error Messages	None
Files Accessed	None
Files Changed	None
Method Called	None
Narrative	Performs the Client Call to Login using the Web Service.

3.2.4.2 *Logout(boolean)*

Method ID	AP-Stub3.0
Method Name	CustomerPortBindingSkeleton::Logout (public)
Return Type	boolean
Input Parameters	boolean valid
Output Parameters	boolean
Error Messages	None
Files Accessed	None
Files Changed	None
Method Called	None
Narrative	Performs the Client Call to Logout using the Web Service.

3.2.4.3 searchCustomers(string,string)

Method ID	AP-Stub2.0
Method Name	CustomerPortBindingSkeleton::searchCustomers (public)
Return Type	string
Input Parameters	string firstName string lastName
Output Parameters	com.example.www.Customer.xsd.Customer[]
Error Messages	None
Files Accessed	None
Files Changed	None
Method Called	None
Narrative	Performs the Client Call to searchCustomers using the Web Service.

3.3 Database

3.3.1 Entity-Relationship Diagram

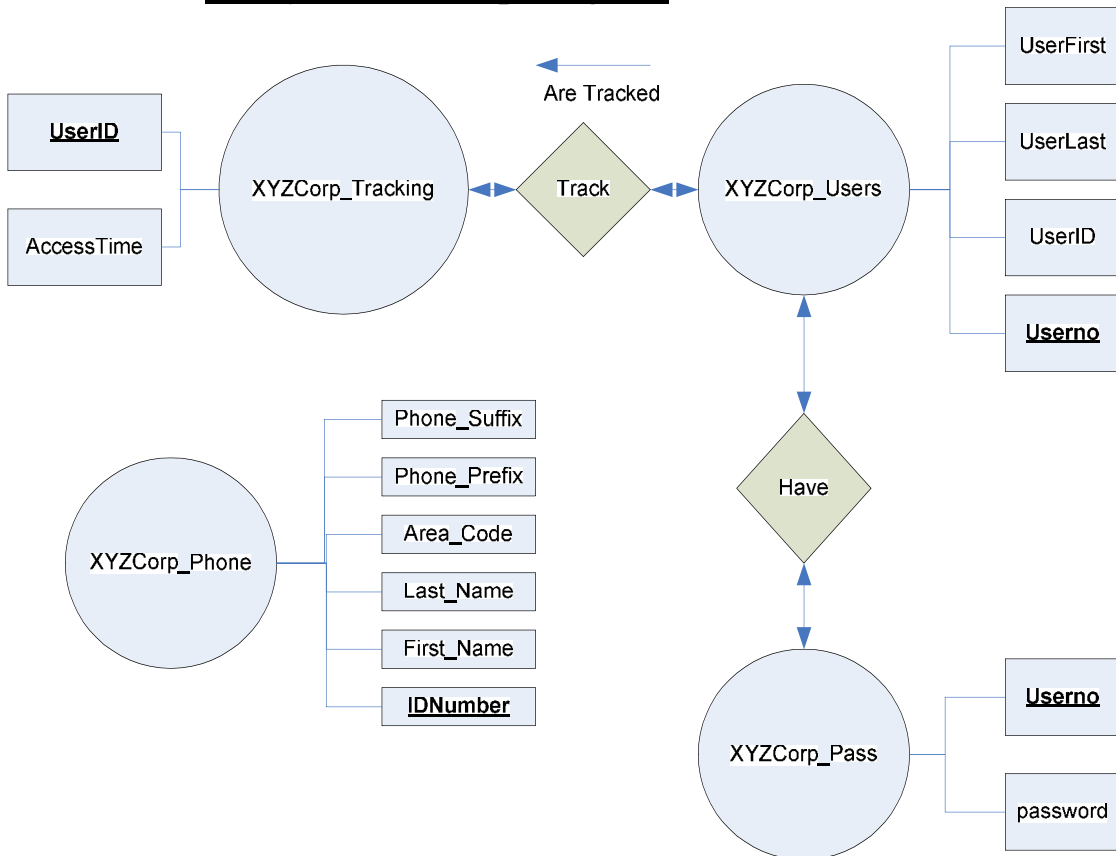


Figure 4.3 in the Appendix contains a copy of this Entity-Relationship Diagram.

3.3.2 Data Dictionary

3.3.2.1 *Phone Directory Table*

Database Name	chiang.compsci.ualr.edu	
Database Schema	XYZCorp	
Table Name	XYZCorp_Phone	
<u>Attribute Name</u>	<u>Data Type</u>	<u>Key</u>
IDNumber	Int(11)	PRI
First_Name	Varchar(50)	
Last_Name	Varchar(50)	
Area_Code	Smallint(6)	
Phone_Prefix	Smallint(6)	
Phone_Suffix	Smallint(6)	

3.3.2.2 *Users Table*

Database Name	chiang.compsci.ualr.edu	
Database Schema	XYZCorp	
Table Name	XYZCorp_Users	
<u>Attribute Name</u>	<u>Data Type</u>	<u>Key</u>
Userno	Int(11)	PRI
Userid	Varchar(7)	
Userlast	Varchar(20)	
Userfirst	Varchar(20)	

3.3.2.3 *Password Table*

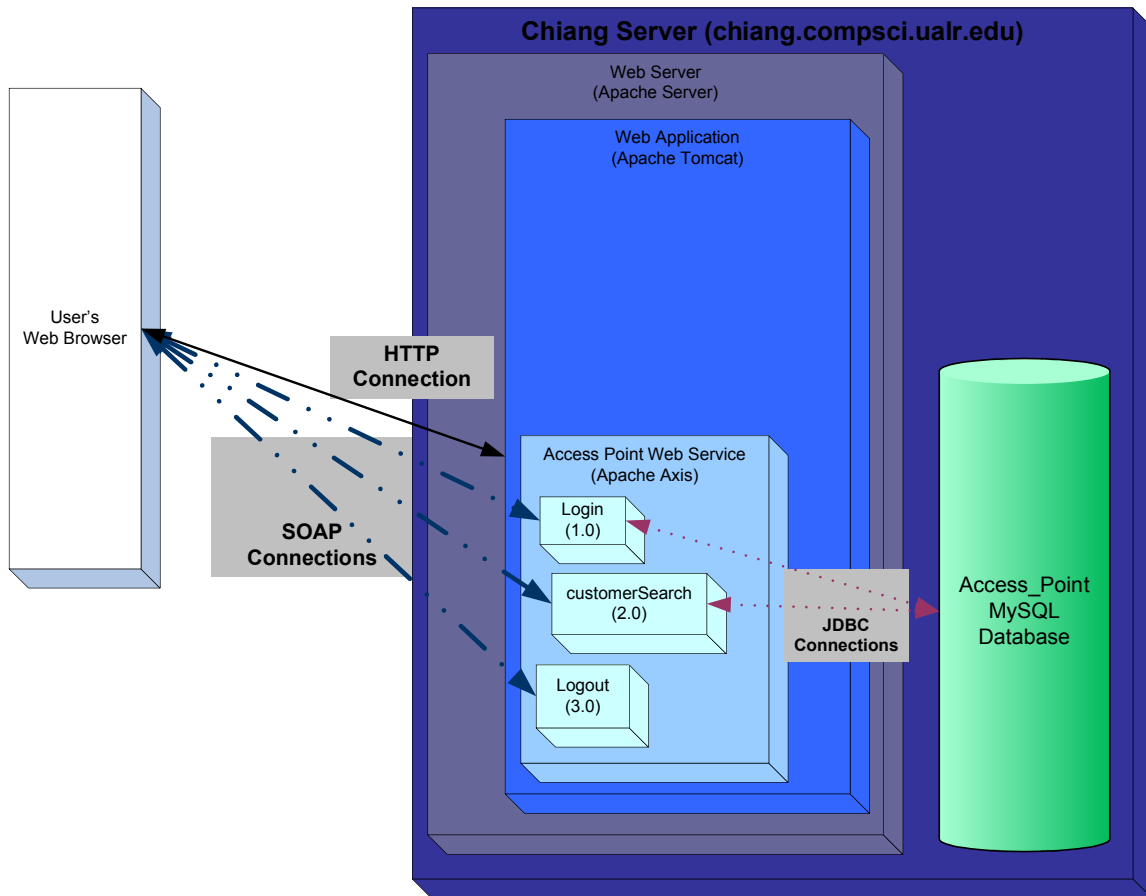
Database Name	chiang.compsci.ualr.edu	
Database Schema	XYZCorp	
Table Name	XYZCorp_Pass	
<u>Attribute Name</u>	<u>Data Type</u>	<u>Key</u>
Userno	Int(11)	PRI
Password	Varchar(10)	

3.3.2.4 *Tracking Table*

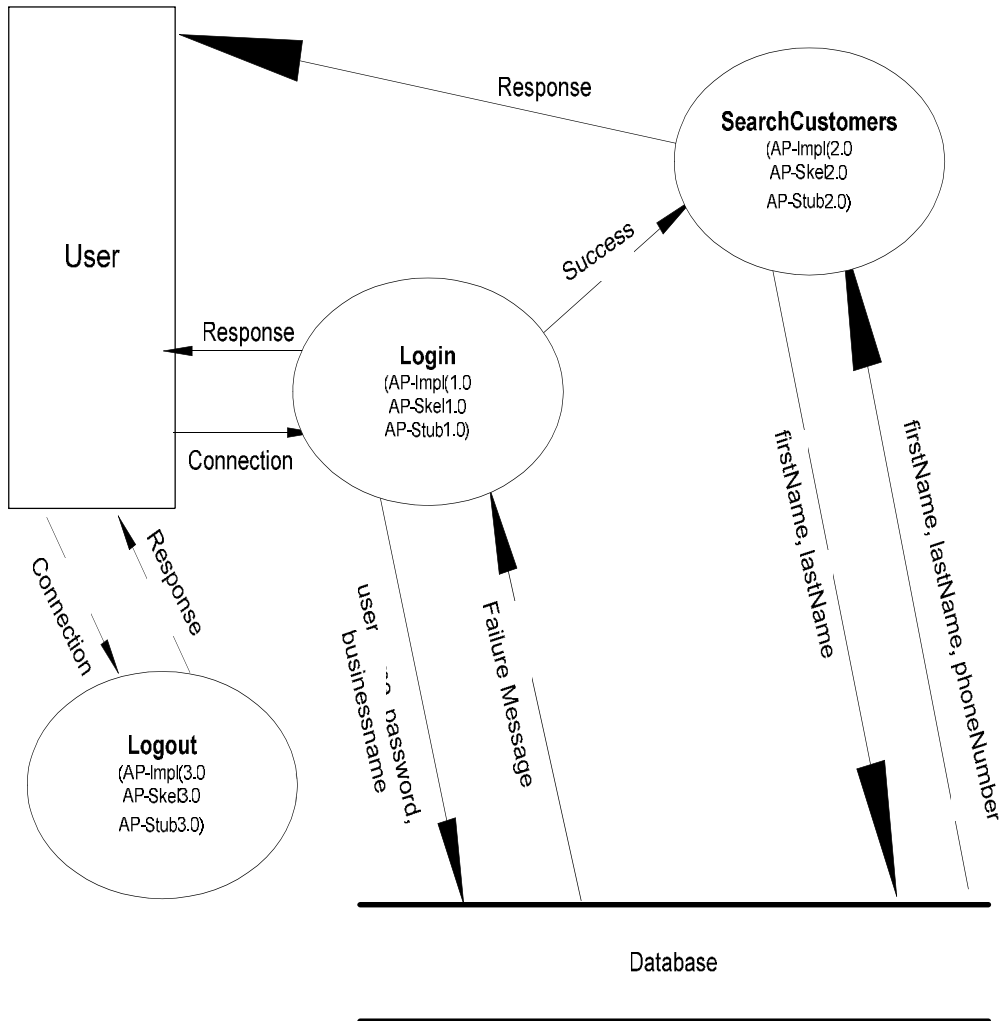
Database Name	chiang.compsci.ualr.edu	
Database Schema	XYZCorp	
Table Name	XYZCorp_Tracking	
<u>Attribute Name</u>	<u>Data Type</u>	<u>Key</u>
Userid	Varchar(7)	FOREIGN KEY
Accesstime	Datetime	

4 Appendix - Software Diagrams

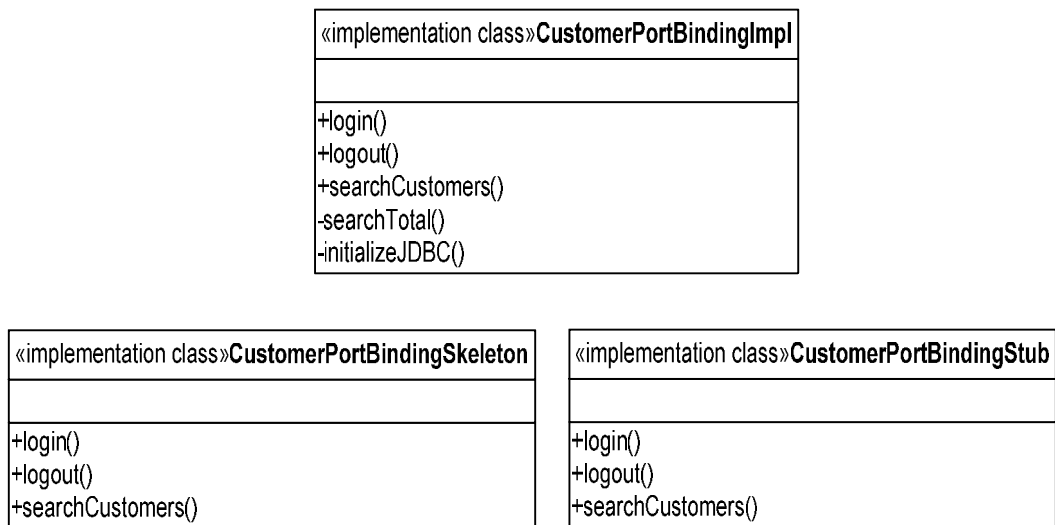
4.1 Access Point Software Architecture Diagram



4.2 Access Point Data Flow Diagram



4.3 Access Point UML Diagram



4.4 Access Point E-R Diagram

