P-3262

PERSONAL FINANCIAL ADVISOR

PROJECT REPORT

Submitted By

A. SOBIA

Register No.: 0720300045

in partial fulfillment for the award of the degree

of

MASTER OF COMPUTER APPLICATIONS

in

COMPUTER APPLICATIONS

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Coimbatore)

May, 2010

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Coimbatore)

COIMBATORE – 641 006.

Department of Computer Applications

PROJECT WORK

MAY 2010

This is to certify that the project entitled

PERSONAL FINANCIAL ADVISOR

is the bonafide record of project work done by

A. SOBIA

Register No: 0720300045

of MCA (Computer Applications) during the year 2009-2010.

[illegible]

Project Guide

[illegible]

Head of the Department

Submitted for the Project Viva-Voce examination held on 17.05.2010

[illegible] 17/5/10

Internal Examiner

[illegible] 17/5/10

External Examiner

DECLARATION

I affirm that the project work titled **PERSONAL FINANCIAL ADVISOR** being submitted in partial fulfilment for the award of **MASTER OF COMPUTER APPLICATIONS** is the original work carried out by me. It has not formed the part of any other project work submitted for award of any degree or diploma, either in this or any other University.

[illegible]

A. SOBIA

Register No. : 0720300045

I certify that the declaration made above by the candidate is true.

[illegible] 10/05/10

Signature of the Guide

S. GANESH BABU

SL/MCA

KCT

Cognizant Technology Solutions India Private Ltd.
Mountain Mist Campus, Unit -1, STPI - IT Park,
No.363, KGISL Campus, Thudiyalur Road,
Coimbatore - 641035
Phone : (+91-0422) 3923000
Fax : (+91-0422) 3923060

www.cognizant.com

Cognizant

Dear Sir/Madam

This is to certify that **Sobia A**, a **MCA** student of **Kumaraguru College of Technology** has done project work in the company on Personal Financial Advisor under the guidance of Ramesh Kalyan, as part of the college requirement, between the period January 2010 and April 2010.

Yours sincerely,

for **Cognizant Technology Solutions India Pvt. Ltd.**.

[illegible]

Sriram Iyer
Manager – Human Resources

I accept the terms and conditions of the offer as mentioned above.

Signature: [illegible]
Date: 07.05.2010

ACKNOWLEDGEMENT

First of all, I wish to thank the almighty who have blessed me with good health and wealth to carry out this project successfully.

I wish to express my thanks to **Dr. S. Ramachandran, Principal,** Kumaraguru College of Technology, Coimbatore, for permitting me to undertake this project.

I wish to express earnest thanks to **Dr. A. Muthukumar, Course Co-ordinator,** Master of Computer Applications, Kumaraguru College of Technology, Coimbatore, for his support and encouragement.

My deepest acknowledgement to the project co-ordinator **Mr. S. Hameed Ibrahim, Senior Lecturer**, Master of Computer Applications, Kumaraguru College of Technology, Coimbatore, for his timely remarks and comments about the project, which helped me to emend my project and knowledge.

I want to acknowledge my project guide, **Mr. S. Ganesh Babu, Senior Lecturer**, Master of Computer Applications, Kumaraguru College of Technology, Coimbatore, for his support and encouragement in making this project a success. His excellent suggestions and timely encouragement helped me to complete the project and the project report.

I wish to express my gratitude to my CTS Project guide, **Mr. Ramesh Kalyan,** Delivery Manager, Cognizant Technology Solutions, Coimbatore, for his useful suggestions and support given throughout the project.

I wish to thank all my teaching and supporting staff members for their timely help to complete the project successfully.

My heartfelt thanks to my lovable family members and friends, who have provided the mental support for me throughout the course of this project work. Finally, I would like to thank all the people around me, who helped directly or indirectly to complete this project.

ABSTRACT

This system **"PERSONAL FINANCIAL ADVISOR"** mainly aims at creation and maintenance of details about the Financial Agents and their Clients. It also focuses on the communication between the agents and clients.

This system is maintained by the Administrator. They can view the details of each and every agent and client. They can also communicate with the agents and clients if some necessity occurs.

This system is mainly for the financial agents under various financial organizations. The agent must first register them in this system. Then agent can add their membership details by logging into their page. The agent can add their client's personal and policy details. The agent can have any number of clients under them. The agent can communicate with their client via mail or group mail. The agent can mail about premium payment to clients, when payment date arrives. The same way, clients can be informed about policy maturity.

The client receives the client id and password via mail when the agent registers the client account. Using that id, client can login to their page and view his details. If client has done the payment, they can add the payment details through his page. If clients need to know details about any new policies or if they want to change the personal or policy details, they can communicate to agent through mail.

This System is developed in PHP, Flex 3, MySQL and WampServer2.0i.

TABLE OF CONTENTS

List of Tables

List of Abbreviations

Abbreviation	Expansion
PFA	Personal Financial Advisor
PHP	Personal Home Pages/PHP Hypertext Preprocessor
IDE	Integrated Development Environment

List of Figures

CHAPTER 1

INTRODUCTION

This chapter is organized into two parts. The first part deals with the organization profile. It provides a brief insight into the history of the organization and the products. The second part gives an introduction about the project.

1.1 ORGANIZATION PROFILE

Cognizant (NASDAQ: CTSH) is a global IT services and business process outsourcing solutions provider. By leveraging highly flexible business processes, a seamless global delivery network and deep domain expertise, Cognizant delivers a better "return on outsourcing."

Cognizant was one of the first IT services companies to organize around key industry verticals and horizontals. This enables Cognizant to establish extremely close partnerships that foster continuous operational improvements and better bottom-line results for clients.

Cognizant delivers a trusted partnership, cost reductions and business results. To meet the specialized needs of each client, Cognizant has continued to invest in deepening the industry-specific organizational capabilities and delivery excellence. It continuously add experienced team members with distinguished track records in key sectors, such as banking, capital markets, insurance, life sciences, healthcare, manufacturing, logistics, retail, utilities, hospitality, communications, information services, media, and entertainment, who serve as subject matter experts and provide clients with valid insights into and viable solutions to particular industry issues.

The unique Two-in-a-Box™ client-relationship model offers greater customer intimacy, speed of delivery, local decision-making, and responsiveness, which has helped Cognizant build deep, fast-growing partnerships with clients.

At a time when companies across the globe are relentlessly striving to compete better, move faster, and fight harder, leading organizations across various industries —from financial services and healthcare through manufacturing, media, retail, utilities, and telecommunications — are partnering with Cognizant to continuously elevate the business value of their IT assets and to make their businesses stronger.

1.2 PROJECT OVERVIEW

This project "**Personal Financial Advisor**" serves as a website for financial agents to maintain their client's personal and financial details. This system also maintains the personal and membership details of agents. The agent is allotted an agent id. They can logon in to the system using that agent id. They can view his membership details, client policy details from their page. They can add a client from their page. There are reports to know about clients under certain policy, client's policy maturity date, etc.

The client is allotted a client id and password. The client id and password will be sent to the client through mail. The client can logon in to the system using this client id and password. The client can view their policy details in their page. If they have made the payment, it can be added through the client page. If the client needs to change their personal and policy details, provision is given to report it to the agent. The same way if client has some doubt regarding policies, they can contact the agent through mail.

Entire system is maintained by the administrator. Admin is able to view the details about agents and their clients. He can also communicate with the agents and clients if some necessity occurs.

CHAPTER 2

SYSTEM STUDY AND ANALYSIS

A complete understanding of the requirement is essential for the success of software development. The software scope, initially established by the system engineer and refined during the project planning, is refined in detail. Model of the required data, information and control flow, and operational behaviour are created. Alternative solution are analyzed and allocated to various software elements. The feasibility study evaluates the viability of the project and presents the recommended strategy adopted for the development.

2.1 EXISTING SYSTEM

In existing system, if the agent has membership in various organizations, each organization have individual website and agents should maintain details in each website.

2.1.1 Drawbacks of the Existing System

- Agent have to logon into each website to know the status of clients in each organization.
- Communication to the clients is not efficient.
- Queries from the clients are harder to retrieve.
- Finding out the premium collection of particular time period is difficult.

2.2 PROPOSED SYSTEM

This project aims at making the work for the financial agents very simpler. The details about various organization memberships can be maintained at one place. This makes to add and browse every detail at one stop. The agent can use these facilities by just registering in this site.

2.2.1 Advantages of Proposed System

The expected benefits of the proposed system are

- Makes agents work very simpler.
- Makes communication between agents and clients very efficient.
- Helps to compare the collections made at different range of period.
- Helps to notify the clients about the date of maturity and date of premium payment.

CHAPTER 3

DEVELOPMENT ENVIRONMENT

3.1 HARDWARE REQUIREMENTS

The hardware support required for deploying the application

Processor	:	Intel Pentium IV
Speed	:	3.1 GHZ
Memory	:	1 GB RAM
Hard Disk Capacity	:	80 GB
Monitor	:	15" inch SVGA
Mouse	:	Logitech Mouse (Scroll)
Keyboard	:	108 Keys

3.2 SOFTWARE REQUIREMENTS

The software support required for deployment is

Operating System	:	Windows XP
Designing Tool	:	Zend Studio 6.1.2, Flex Builder 3
Scripting Language	:	PHP 5.3.0
Web Server	:	WampServer2.0i
Database	:	MySQL 5.1

3.3 SOFTWARE OVERVIEW

PHP:

PHP: Hypertext Preprocessor is a widely used, general-purpose scripting language that was originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. Originally designed to create dynamic web pages, PHP now focuses mainly on server-side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's Active Server Pages, Sun Microsystems' JavaServer Pages, and mod_perl.

PHP only parses code within its delimiters. Anything outside its delimiters is sent directly to the output and is not processed by PHP (although non-PHP text is still subject to control structures described within PHP code). The most common delimiters are <?php to open and ?> to close PHP sections. The first form of delimiters, <?php and ?>, in XHTML and other XML documents, creates correctly formed XML 'processing instructions'. This means that the resulting mixture of PHP code and other markup in the server-side file is itself well-formed XML.

Variables are prefixed with a dollar symbol and a type does not need to be specified in advance. Unlike function and class names, variable names are case sensitive. Both double-quoted ("") and heredoc strings allow the ability to embed a variable's value into the string. PHP treats newlines as whitespace in the manner of a free-form language (except when inside string quotes), and statements are terminated by a semicolon. PHP has three types of comment syntax: /* */ marks block and inline comments; // as well as # are used for one-line comments. The echo statement is one of several facilities PHP provides to output text (e.g. to a web browser).

In terms of keywords and language syntax, PHP is similar to most high level languages that follow the C style syntax. If conditions, for and while loops, and function returns are similar in syntax to languages such as C, C++, Java and Perl.

MYSQL:

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Sun Microsystems, Inc.

MySQL is a database management system.

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

MySQL is a relational database management system.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases and is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, "SQL-92" refers to the standard released in 1992, "SQL:1999" refers to the standard released in 1999, and "SQL:2003" refers to the current version of the standard. We use the phrase "the SQL standard" to mean the current version of the SQL Standard at any time.

MySQL software is Open Source.

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License, to define what you may and may not do with the software in different situations.

The MySQL Database Server is very fast, reliable, and easy to use.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

MySQL Server works in client/server or embedded systems.

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

A large amount of contributed MySQL software is available.

It is very likely that your favorite application or language supports the MySQL Database Server.

WAMP SERVER:

WAMP5 (WAMP means Windows Apache Mysql PHP) is a platform of Web development under Windows. WAMP is a form of mini-server that can run on almost any Windows Operating System. WAMP includes Apache 2, PHP 5 (SMTP ports are disabled), and MySQL (phpMyAdmin and SQLitemanager are installed to manage your databases) preinstalled.

WampServer is a Windows web development environment. It allows you to create web applications with Apache, PHP and the MySQL database. It also comes with PHPMyAdmin and SQLiteManager to easily manage your databases.

An icon on the taskbar tray displays the status of WAMP, letting you know if; a) WAMP is running but no services are opened (the icon will appear red), b) WAMP is running and one service is opened (the icon will appear yellow) or c) WAMP is running with all services opened (the icon will appear white). Apache and MySQL are considered to be services (they can be disabled by left-clicking on the taskbar icon, guiding your cursor over the service you wish to disable and selecting "Stop Service").

The files/web pages that are hosted on your WAMP server can be accessed by typing *http://localhost/* or *http://127.0.0.1/* in the address bar of your web browser. WAMP must be running in order to access either of the above addresses.

FLEX:

Flex is a highly productive, free, open source framework for building expressive web applications that deploy consistently across browsers, desktops, and operating systems by leveraging the Adobe® Flash® Player and Adobe AIR® runtimes. While Flex applications can be built using only the Flex framework, Adobe Flash Builder™ (formerly Adobe Flex® Builder™) software can accelerate development through features like intelligent coding, interactive step-through debugging, and visual design of the user interface layout.

Flex is the way to make rich Internet applications (RIAs) quickly and easily. At its basic level, it's a framework for creating RIAs based on Flash Player. Along with being a framework, Flex is also a new language. At its heart is MXML, a markup language based on Extensible Markup Language (XML) that makes it really easy and efficient to create applications. Unlike developing for some desktop platforms requiring a proprietary binary file format, MXML is just text, so it's easy to read and modify using just a text editor. Therefore, sharing code is as easy as sharing a simple text file.

MXML is an XML language that you use to lay out user interface components for Adobe® Flex® applications. You also use MXML to declaratively define nonvisual aspects of an application, such as access to server-side data sources and data bindings between user interface components and server-side data sources.

MXML development is based on the same iterative process used for other types of web application files such as HTML, JavaServer Pages (JSP), Active Server Pages (ASP), and ColdFusion Markup Language (CFML). Developing a useful Flex application is as easy as opening your favorite text editor, typing some XML tags, saving the file, requesting the file's URL in a web browser, and then repeating the same process.

In the Flex model-view design pattern, user interface components represent the view. The MXML language supports two types of user interface components: controls and containers. Controls are form elements, such as buttons, text fields, and list boxes. Containers are rectangular regions of the screen that contain controls and other containers.

Remote-procedure-call (RPC) services let your application interact with remote servers to provide data to your applications, or for your application to send data to a server. Flex is designed to interact with several types of RPC services that provide access to local and remote server-side logic.

The MXML components that provide data access are called RPC components. MXML includes the following types of RPC components:

- •WebService provides access to SOAP-based web services.
- •HTTPService provides access to HTTP URLs that return data.
- •RemoteObject provides access to Java objects using the AMF protocol (Adobe LiveCycle Data Services ES only).

CHAPTER 4
SYSTEM DESIGN

4.1 DIAGRAMS

4.1.1 USE-CASE DIAGRAMS

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Fig 4.1.1.1: Client Use-Case Diagram

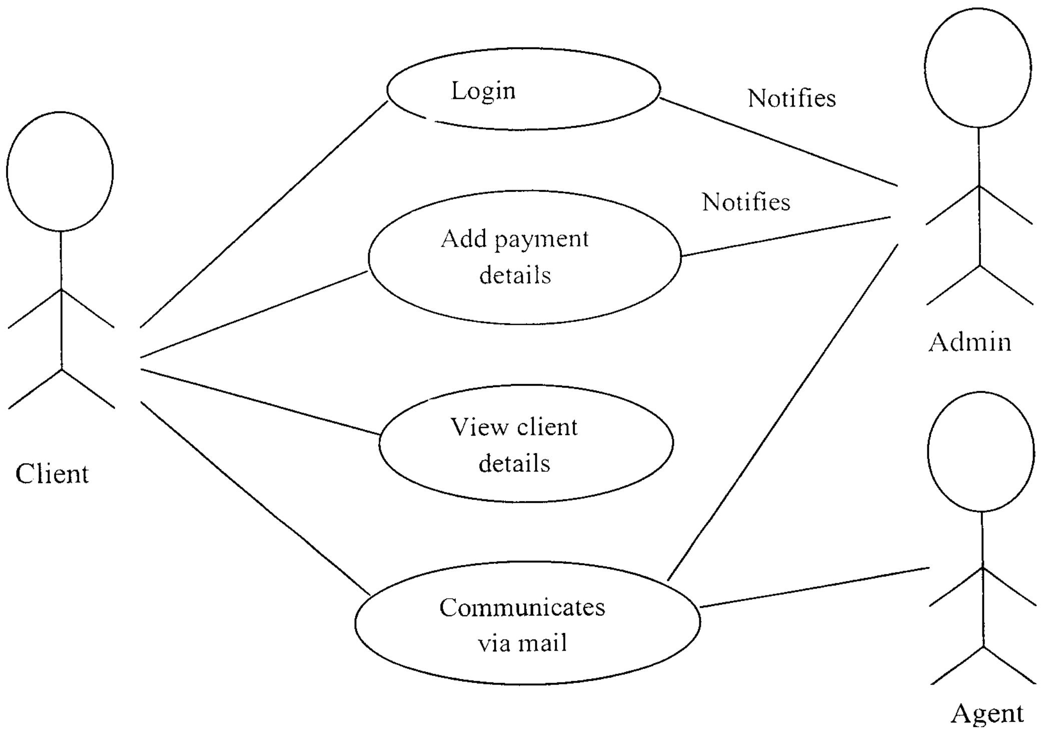
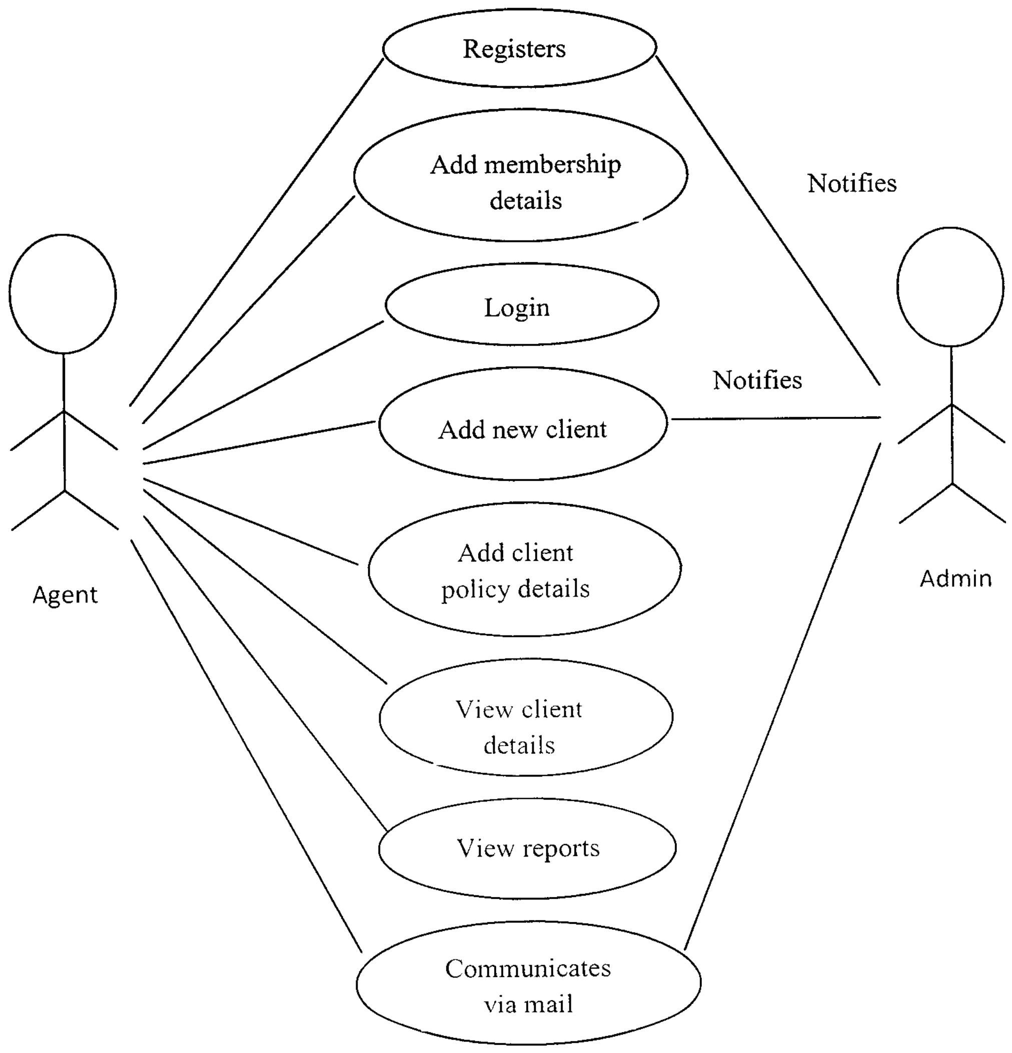


Fig 4.1.1.2: Agent Use-Case Diagram



4.2 ELEMENTS OF DESIGN

System Design is the most creative and challenging phase in the development of a software system. The first step is to determine what input data is needed for the system and then to design a database that will meet the requirements of the proposed system. The next step is to determine what outputs are needed from the system and the format of the output to be produced.

During the design of the proposed system some areas where attention is required are:

- How are the inputs required and the outputs produced?
- How should the data be organized?
- What will be the processes involved in the system?
- How should the screen look?

The steps carried out in the design phase are as follows:

- ✓ Input Design
- ✓ Output Design
- ✓ Database Design

4.2.1 INPUT DESIGN

Input design is a part of the system design and hence must be carefully designed which otherwise lead to serious errors in the later stages of development. Inaccurate input data is the most common cause of errors in data processing. The main objective of designing input focus on

- Controlling the amount of input required
- Avoiding delayed responses
- Keeping process simple
- Controlling and avoiding errors

4.2.2 OUTPUT DESIGN

Output generally refers to the results and information that are generated by the system. For many end-users, output is the main reason for developing the system and the

basis on which they will evaluate the usefulness of the application. Most end-users will not actually operate the information system or enter data through workstations, but they will use the output from the system. When designing output, system analysis must accomplish the following.

- ☞ Determine what information to present
- ☞ Decide whether to display, print or speak the information and select the output medium
- ☞ Arrange the presentation of information in an acceptable format.
- ☞ Decide how to distribute the output to intended recipients.

The arrangement of information on a display or printed document is termed as layout. Accomplish the general activities listed above will require specific decisions, such as whether to use pre-printed forms when preparing reports and documents, how many lines to plan on a printed page or whether to use graphics and colour.

The output design is specified on layout performs, sheets that describe the location characteristics, and format of the column headings and pagination.

The output must be provided in a format easily understandable even by a novice user. After analyzing the operations of the system, output information required for each jobs are determined. In addition to this, these outputs may be in format suitable as inputs for subsequent processing.

4.2.3 DATABASE DESIGN

A database is a collection of inter-related data stored with minimum redundancy to serve many users quickly and efficiently. The general objective of database design is to make the data access easy, inexpensive and flexible to the user. An elegantly designed database can play a strong foundation for the whole system.

The details about the relevant data for the system are first identified. According to their relationship, tables are designed through the following method.

- The data type for each data item in the table is decided.
- The tables are then normalized.

The tables are normalized so that they can provide better response time, have data integrity, avoid redundancy and be secure.

DATABASE STRUCTURE:

This system uses many numbers of tables to store the details of agents and clients. It also contains the table about transactions.

4.2.3.1: Table Name: pfa_orgn

This table contains the list of financial organizations where the agents are the members.

Field Name	Data type	Size	Constraint
Orgn_ID	Varchar	10	Primary
Orgn_Name	Varchar	25	
Orgn_type	Varchar	15	
Website	Varchar	50	

4.2.3.2: Table Name: pfa_branch

This table contains the lists of branches under each financial organization.

Field Name	Data type	Size	Constraint
Branch_ID	Varchar	10	Primary
Orgn_id	Varchar	10	Foreign key referenced from pfa_orgn
Branch_name	Varchar	25	
Address	Varchar	75	
Phoneno	Number	13	
Mailid	Varchar	50	

4.2.3.3: Table name: pfa_policy

This table contains the list of financial policies available in each financial organization.

Field Name	Data type	Size	Constraint
Policy_id	Varchar	15	Primary Key
Policy_name	Varchar	30	
Orgn_id	Varchar	15	Foreign key referenced from pfa_orgn
Mode_id	Varchar	3	Foreign key referenced from pfa_mode
Minage	Int		
Maxage	Int		
Minsum	Double		
Maxsum	Double		

4.2.3.4: Table name: pfa_agentpersonal

This table consists of the agent list registered in the site.

Field Name	Data type	Size	Constraint
Agent_id	Varchar	10	Primary key
Agent_name	Varchar	25	
Agent_dob	Date		
Agent_Nationality	Varchar	25	
Agent_sex	Varchar	6	
Agent_qual	Varchar	30	
Agent_occ	Varchar	50	
Agent_mstat	Varchar	10	
Agent_Address	Varchar	75	
Agent_Mobno	Number	13	
Agent_Mailid	Varchar	50	
Agent_pword	Varchar	15	
Agent_lastdate	Date		

4.2.3.5: Table Name: pfa_agentorgn

This table contains the agent membership details in various organization.

Field Name	Data type	Size	Constraint
Agent_id	Varchar	15	Foreign key referenced from pfa_agentpers
Orgn_id	Varchar	15	Foreign key referenced from pfa_orgn
Branch_code	Varchar	15	Foreign key referenced from pfa_branch
Agent_doj	Date		

4.2.3.6: Table Name: pfa_mode

This table contains the list of premium payment mode.

Field Name	Data type	Size	Constraint
Mode_id	Varchar	3	Primary key
Mode_name	Varchar	25	

4.2.3.7: Table Name: pfa_clientpersonal

This table contains the personal details of the clients of financial agents.

Field Name	Data type	Size	Constraint
Client_id	Varchar	10	Primary key
Agent_id	Varchar	10	Foreign key referenced from pfa_agentpersonal
Client_name	Varchar	30	
Client_Dob	Date		
Client_nationality	Varchar		
Client_sex	Varchar	6	
Client_qual	Varchar	25	
Client_occ	Varchar	25	
Client_Addr	Varchar	150	
Client_mobno	Varchar	13	
Client_email	Varchar	50	
Client_pword	Varchar	15	
Client_aincome	Double		
Client_nodep	Int		

4.2.3.8: Table Name: pfa_clientpolicy

This table contains the policy details of each client who owned the policies.

Field Name	**Data type**	**Size**	**Constraint**
Client_id	Varchar	15	Foreign key referenced from pfa_clientpersonal
Policy_id	Varchar	15	Foreign key referenced from pfa_policy
Mode_id	Varchar	3	Foreign key referenced from pfa_mode
Sumassured	Double		
Nominee_id	Varchar	15	
Policy_undertakendate	Date		

4.2.3.9: Table Name: pfa_clipayment

This table contain the list of payments made by the clients.

Field Name	**Data type**	**Size**	**Constraint**
Client_id	Varchar	15	Foreign key referenced from pfa_clientpers
Policy_id	Varchar	10	Foreign key referenced from pfa_policy
Mode_id	Varchar	5	Foreign key referenced from pfa_mode
Paid_date	Date		
Paid_premium	double		

4.2.4 MODULAR DESIGN

Modular design — or "modularity in design" — is an approach that subdivides a system into smaller parts (modules) that can be independently created and then used in different systems to drive multiple functionalities. Besides reduction in cost (due to lesser customization, and less learning time), and flexibility in design, modularity offers other benefits such as augmentation (adding new solution by merely plugging in a new module), and exclusion.

This system is also modularized to reduce the complexity of the system. This contains various modules.

A) User Modules:

User modules are modelled based on the users of the system. This module contains three users:

i) Admin

- administers the entire system.
- can view the details of all agents and clients.
- can communicate with other two users to notify some details.

ii) Agent

- Registers his membership and adds his/her membership details in each organization
- Adds the new client details and their policy details.
- can view the details of clients under him/her.
- can communicate with other two users to notify/query some details.

iii) Client

- can view his/her details.
- can add the payment details that have been made.
- can communicate with other two users to notify/query some details.

B) Reports Module

Reports forms the main output of any system. This system too has many reports which will help admin and agent to take many important decisions. This system contains many reports based on agent id, client id, dates, etc.

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 IMPLEMENTATION

The system is implemented using Zend Studio- Flex builder, PHP and MYSQL.

Implementation of Business Logic

The business logic is implemented using Flex Builder. Flex builder contains many controls which make the website very user friendly. The data are entered using the controls and the output is displayed very effectively. This flex builder is very helpful in the way that it doesn't need to navigate through various pages. The loading of different pages for every function call is omitted which helps in improving performance. This IDE uses only one HTML page. The controls are made visible and invisible at needed times. The communication between the IDE and Database is through the Scripting language named PHP. The communication is in the form of XML data. This IDE uses an HTTP Object to send the data to the Scripting language. After processing of PHP file, the results are returned to IDE in the XML format which can be displayed using various controls and containers.

Implementation of Database Communication

For database communication, MYSQL is used along with PHP. MYSQL is the most popular Open Source SQL database management system. MYSQL has many inbuilt functions to carry out the operations with database. These functions are used as a part of PHP file. Using these functions the queries are executed and the operations on database are carried out.

The inputs are received from action script file of IDE. The operations are performed according to the command received from IDE. Those operations are performed and the results are returned to the IDE using XML files.

CHAPTER 6
SYSTEM TESTING

6.1 SYSTEM VERIFICATION

System Verification is the process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. Verification is ensuring that the product has been built according to the requirements and design specifications- i.e., you built it right. Verification is the assurance that the products of a particular development phase are consistent with the requirements of that phase and preceding phase(s).

In this website, review of interim work steps is done to ensure they are acceptable. In data access, it verifies whether the right data is being accessed in terms of the right place and in the right way.

6.2 SYSTEM VALIDATION

System Validation is the process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements. Validation checks that the product design satisfies or fits the intended usage (high-level checking) — i.e., you built the right product. This is done through dynamic testing and other forms of review. Validation ensures that the product actually meets the user's needs, and that the specifications were correct in the first place.

In this project, validation checks whether the developer is moving towards the right product. Validation coding is written using pre-defined validators available in Flex 3. Each field in registration form are validated such that the right username, password, date etc., is added. Any wrong entry display error messages or warnings. The login form is validated such that the valid registered user only can login to new page. Fields such as e-mail id and website are checked for its format. Validation also determines if this project complies with the requirements and performs functions for which it is intended and meets the organization's goal and user needs.

6.3 TESTING

Testing is a critical element of software quality and assurance and represents the ultimate review of specification design and coding. It is a vital activity that has to be enforced in the development of any system. This could be done in parallel during all the phases of system development. The feedback received from these tests can be used for further enhancement of the system under consideration. The main type of test carried out in Personal Financial Advisor is Unit Testing and Integration Testing.

6.3.1 Unit Testing

A series of stand-alone tests are conducted during Unit Testing. Each test examines an individual component that is new or has been modified. A unit test is also called a module test because it tests the individual units of code that comprise the application. Unit tests focus on functionality and reliability, and the entry and exit criteria can be the same for each module or specific to a particular module. Unit testing is done in a test environment prior to system integration. If a defect is discovered during a unit test, the severity of the defect will dictate whether or not it will be fixed before the module is approved.

In Personal Financial Advisor each component i.e. each form is tested individually to verify that the detailed design for unit has been correctly implemented. Initially the flow of control and data through that page is checked. In a page, each control is further tested in unit testing. The process is done in all the forms of the system.

6.3.2 Integration testing

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit. In a realistic scenario, many units are combined into components, which are in turn aggregated into even larger parts of the program. The idea is to test combinations of pieces and eventually expand the process to test your modules with those of other groups. Eventually all the modules making up a process are tested together. Beyond that, if the program is composed of more than one process, they should be tested in pairs rather than all at once. Integration testing identifies problems that occur when units are combined.

Many forms in the system have communication between each other. This helps in testing integration testing. For ex: if the agent logins, first system should check whether he is a valid user and then it should move to the agent page and display some details of that agent. Here more than a single process is involved and so it needs integration testing.

6.3.3 TEST CASES:

S. No	TEST CASE	EXPECTED RESULT	ACTUAL RESULT
1	User id: admin Password: abcd	Invalid Login	Invalid Login
2	User id: admin Password: admin	Valid Login	Valid Login
3	User id: Axxx Password: xxxxxx	Login to Particular Agent Page	Logins to the Agent home page
4	User id: Cxxx Password: xxxxxx	Login to Particular Client Page	Logins to the Client home page
5	User id: sxxx Password: xxxxxx	Invalid Login	Invalid Login
6	In admin page, If agent is selected in the datagrid	List of clients under him should be displayed	List of clients under him are displayed in a datagrid. If no clients, alert message 'No clients' is displayed
7	In admin reports, if an organization ID is selected	List of agents under that organization membership should be displayed.	List of agents under that organization membership are displayed.

CHAPTER 7
CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

Personal Financial Advisor is developed to help the financial agents with the idea of reducing their work. Agent can collectively maintain the details of the clients under their in various organizations. This system helps in easy communication between the agents and clients. This system gives many provisions to know the status of actions of clients.

This system also helps the client to know the next premium date, maturity, etc, in advance through mail. Client can also communicate with agent through mail to clarify his doubts, change some details, etc.

It is designed to be highly flexible, so that any future modification and requirements can easily be incorporated.

7.2 FUTURE ENHANCEMENT

This project has been developed as a Master's project and is constrained by time. There is scope for extending the system as per the need.

More reports can be generated, if the need arises. If possible, this system can tie up with the individual financial organization's website and update some details needed. A functionality can be designed that will help clients to plan their investments for future such as child education plans, retirement plans, etc.

Provision to download forms such as maturity claim, change of address, etc, can be provided in future. As this is a website project, we can include some useful links in each page of the site.

CHAPTER 8

APPENDIX

8.1 SAMPLE CODING

MXML File:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="*"
layout="absolute" backgroundColor="#FFFFFF" borderStyle="none" color="#FFFCFC"
applicationComplete="discomm()" backgroundGradientAlphas="[1.0, 1.0]"
backgroundGradientColors="[#98FB9F, #2592DE]">
<mx:Style source="style.css"/>
<mx:Script source="admin.as" />
<mx:Script source="agent_reg.as" />
<mx:Script source="agent_rep.as" />
<mx:Script source="client_add.as" />
<mx:Script source="comment.as" />
<mx:Script source="login.as" />
<mx:Script source="client.as" />
<mx:Script source="agent.as" />
<mx:Script source="plans.as" />
<mx:Script source="agent_addmemb.as" />
<mx:VBox id="vb1" x="175" y="0" horizontalAlign="center"
backgroundColor="0x0099FF" width="900" fontSize="12" color="#560303">
<mx:Text fontFamily="Colonna MT" text="Personal Financial Advisor" selectable="false"
fontSize="60" color="#F9FBF9" fontWeight="normal"/>
<mx:ApplicationControlBar id="acb" backgroundColor="#480606" dock="true"
color="#FFFEFE" width="900" horizontalAlign="right"  fontSize="12">
<mx:LinkBar id="lb" dataProvider="viewstack1" horizontalAlign="center"
separatorColor="#49FD19" color="#5BFD1B" verticalAlign="top" fontSize="11"
enabled="true"/>
</mx:ApplicationControlBar>
```

```
<mx:Image id="img" source="../b1.jpg" height="220" width="900"
horizontalAlign="center"/>

<mx:ApplicationControlBar id="acb2" backgroundColor="#480606" height="40"
dock="true" color="#FFFFFF" width="900" horizontalAlign="center">
<mx:Text fontSize="20" fontFamily="Pristina" color="#FCFDFC" selectable="false"
fontWeight="bold" textAlign="center">
<mx:htmlText>
<![CDATA[Sometimes it's the smallest decisions that can change your life forever]]>
</mx:htmlText>
</mx:Text>
</mx:ApplicationControlBar>

<mx:ViewStack id="viewstack1" color="#6E0707" fontSize="13" fontFamily="Book
Antiqua" change="planchange()">
<mx:Canvas label="Home" id="home">
<mx:HBox id="hbhome" x="0" y="10" width="890" horizontalAlign="center"
color="#0F1151" height="472">
<mx:VBox height="100%" width="440">
<mx:Panel layout="absolute" width="440" id="welcome" title="Hi... Welcome to the Site..."
color="#FFFFFF" borderColor="#0E96FF" backgroundSize="1" height="197">
<mx:Text y="22" selectable="false" width="402" x="28" height="151">
<mx:htmlText>
<![CDATA[This site is mainly meant for the agents of all
financial organizations. This helps agents to maintain
the details about the clients.]]>
</mx:htmlText>
</mx:Text>
</mx:Panel >
<mx:Panel id="comm" width="436" layout="absolute" title="Comments..." height="253">
<mx:VBox x="26" y="38" width="400" height="193">
<mx:Label id="lblc1"/>
<mx:Label id="lblc2"/>
</mx:VBox>
```

```
</mx:Panel>
</mx:VBox>
<mx:VBox y="0" height="470" width="440">
<mx:Panel id="login" x="0" y="0" layout="absolute" title="User Login" color="#F9FBF9"
width="440" backgroundSize="1" height="199">
<mx:Label x="56.95" y="43" text="Login ID:" />
<mx:Label text="Password:" x="52.95" y="82" />
<mx:TextInput x="121.95" y="41" id="txt_name" color="#0F1151"/>
<mx:TextInput x="121.95" y="80" id="txt_pword" displayAsPassword="true"
color="#0F1151" keyDown="k(event)"/>
<mx:Button id="bt1"  buttonMode="true" useHandCursor="true" x="161.95" y="119"
label="Login" color="#0F1151" click="log()" textAlign="center"/>
<mx:LinkButton x="102.95" y="155" label="New Agent??? Click Here..." click="reg()"/>
</mx:Panel>
<mx:Panel id="comments" layout="absolute" x="0" borderColor="#2687FF"  width="440"
title="Post Ur Comments!!!" color="#FEFEFE" height="250">
<mx:Label x="77" y="49" text="Enter Name:"/>
<mx:Label x="53" y="91" text="Enter Comment:"/>
<mx:TextInput x="168" y="50" id="txt_cname" color="#0F1151"/>
<mx:TextArea x="168" y="93" height="88" id="txt_comment" color="#0F1151"/>
<mx:Button x="155" y="203" label="Submit Comment" click="subcom()"
color="#0F1151"/>
</mx:Panel>
</mx:VBox>
</mx:HBox>
</mx:Canvas>
<mx:Canvas id="cp" width="890" label="Plans" >
<mx:VBox id="vplans">
<mx:Panel id="planspan" title="Plans..." width="890" borderColor="#256EC9"
layout="absolute">
<mx:VBox x="30" y="30" width="100%">
<mx:Text width="358" textAlign="left" selectable="false">
<mx:htmlText>
```

```
<![CDATA[Hi..<br>In each organization there are variety of plans offered.<br>Every plan
wont suit every person.<br>For every person the need varies and different plans suit those
needs.<br>The Organizations are   <br>]]>
</mx:htmlText>
</mx:Text>
<mx:LinkButton click="liccan()" label="LIC" useHandCursor="true" buttonMode="true"/>
<mx:LinkButton click="aviva()" label="Aviva" useHandCursor="true" buttonMode="true"/>
</mx:VBox>
</mx:Panel>
</mx:VBox>
</mx:Canvas>
</mx:ViewStack>
</mx:VBox>
</mx:Application>
```

.as File:

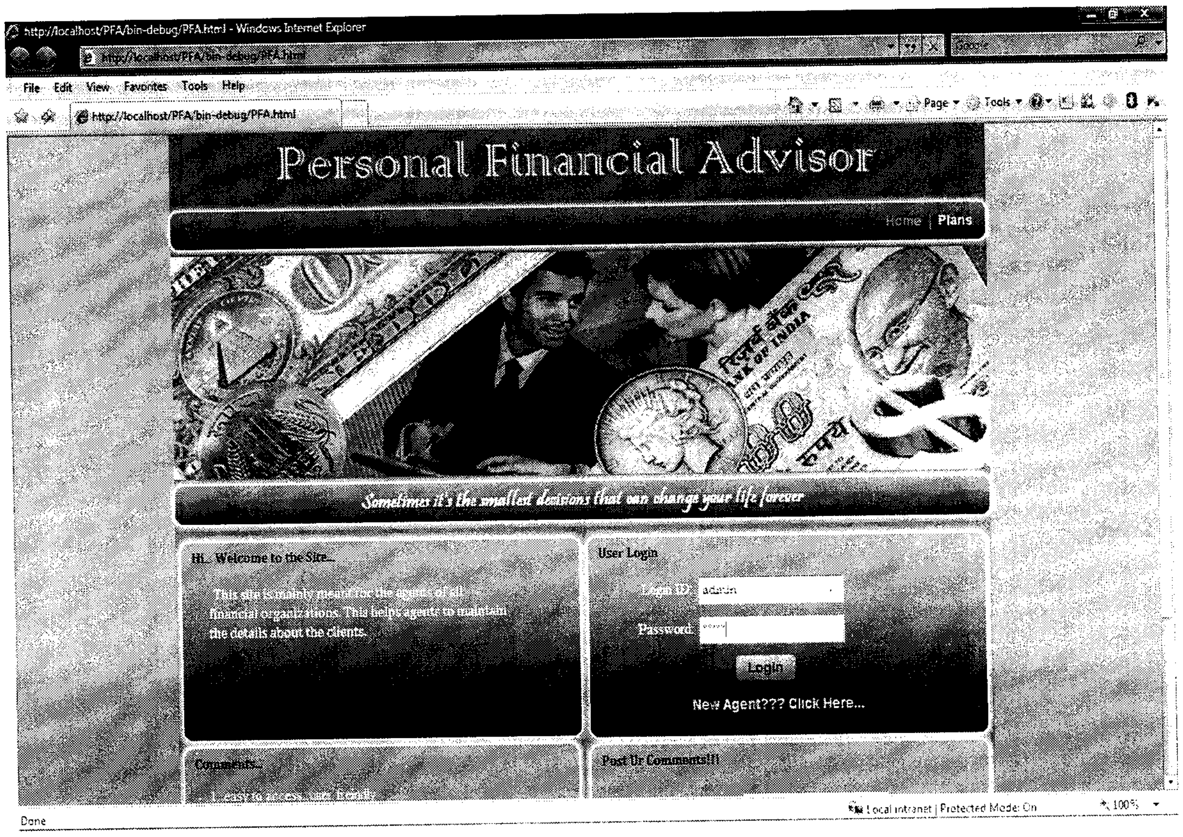
```
// ActionScript file

import mx.controls.*;
import mx.rpc.events.ResultEvent;
import mx.rpc.http.HTTPService;

public var commhttp:HTTPService=new HTTPService();
public function subcom():void
        {
                commhttp.method="POST";
                commhttp.resultFormat="e4x";
                commhttp.useProxy=false;
                commhttp.url="http://localhost/PFA/comm.php";
                commhttp.addEventListener("result",commres);
                var obj:Object=new Object();
                obj.commname=txt_cname.text;
                obj.comm=txt_comment.text;
                commhttp.send(obj);  }

public function commres(e:ResultEvent):void
        {
                Alert.show(String(e.result));
                txt_cname.text="";
                txt_comment.text="";
                discomm();
        }

public var discommhttp:HTTPService=new HTTPService();
public function discomm():void
        {
                discommhttp.method="POST";
                discommhttp.resultFormat="e4x";
                discommhttp.useProxy=false;
                discommhttp.url="http://localhost/PFA/discomm.php";
                discommhttp.addEventListener("result",discommres);
                discommhttp.send();
        }

public function discommres(e:ResultEvent):void
        {
                lblc1.text="1. "+e.result.policy[0].pname;
                lblc2.text="2. "+e.result.policy[1].pname;
        }
```

PHP File:

```
<?php
$con = mysql_connect("localhost","root","");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("pfa_db", $con);

$a=$_POST['agentid'];
$sql = "SELECT DISTINCT(orgn_id) FROM pfa_agentorgn WHERE agent_id='$a'";
$result = mysql_query($sql);
$Return = "<Agent>";

while($row = mysql_fetch_array($result))
  {
        $o=$row['orgn_id'];
        $sql1="SELECT orgn_name FROM pfa_orgn WHERE orgn_id='$o'";
        $result1=mysql_query($sql1);
        $row1= mysql_fetch_array($result1);
        $Return .= "<orgn><OrgName>".$row1['orgn_name']."</OrgName></orgn>";
  }

$Return .= "</Agent>";
mysql_close($con);
print($Return);
?>
```

8.2 SCREEN SHOTS

Fig: 8.2.1 Home Page


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer

Personal Financial Advisor

Home | Plans

Sometimes it's the smallest decisions that can change your life forever

Hi.. Welcome to the Site...

This site is mainly meant for the agents of all financial organizations. This helps agents to maintain the details about the clients.

User Login

Login ID: admin

Password: *****

Login

New Agent??? Click Here...

Comments...

1. easy to access, user friendly

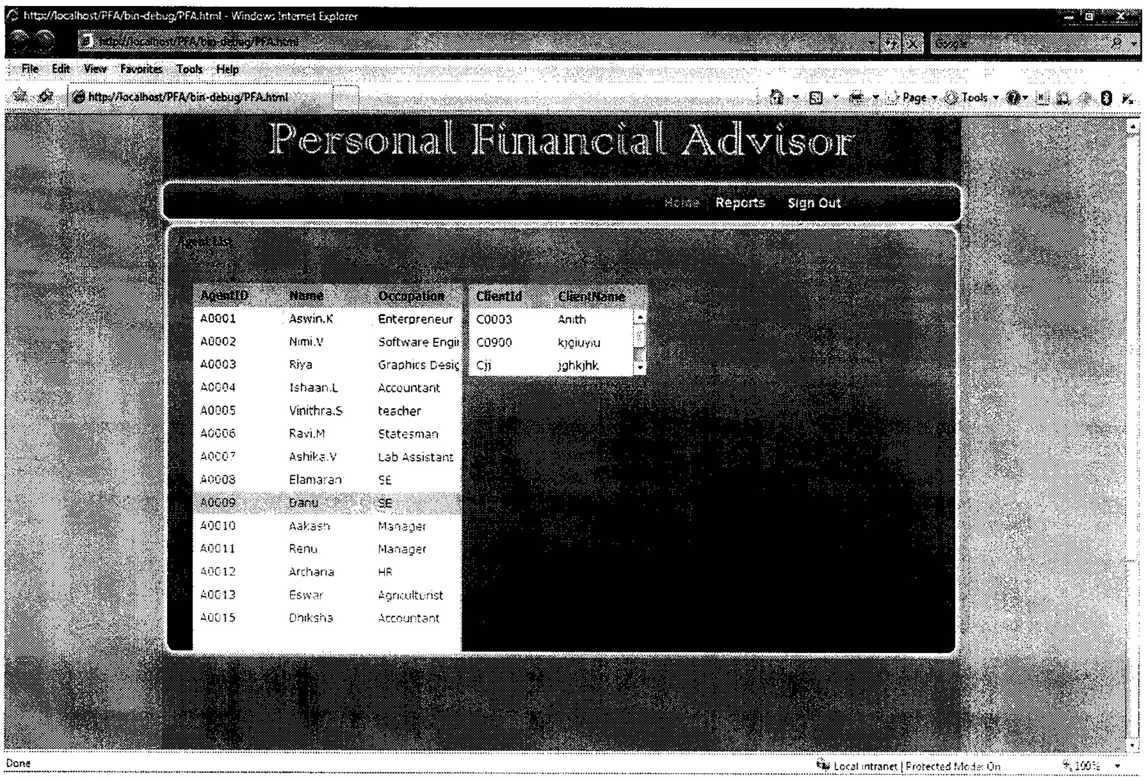
Post Ur Comments!!!

Done

Local intranet | Protected Mode: On

100%

Fig:8.2.2 Admin Page

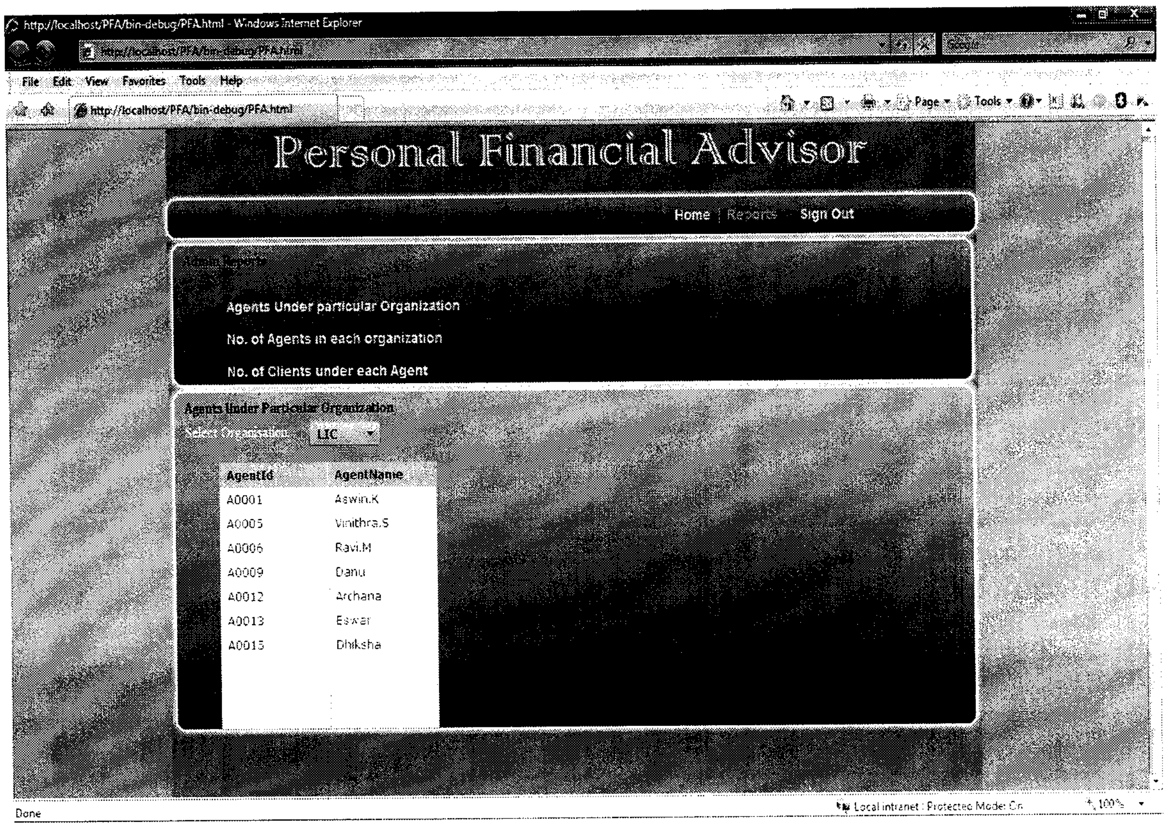

http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Home
Reports
Sign Out
Agent List
Done
Local intranet | Protected Mode: On
100%

Personal Financial Advisor

AgentID	Name	Occupation
A0001	Aswin.K	Enterpreneur
A0002	Nimi.V	Software Engir
A0003	Riya	Graphics Desig
A0004	Ishaan.L	Accountant
A0005	Vinithra.S	teacher
A0006	Ravi.M	Statesman
A0007	Ashika.V	Lab Assistant
A0008	Elamaran	SE
A0009	Danu	SE
A0010	Aakash	Manager
A0011	Renu	Manager
A0012	Archana	HR
A0013	Eswar	Agriculturist
A0015	Dhiksha	Accountant

ClientId	ClientName
C0003	Anith
C0900	kjgiuyiu
Cjj	jghkjhk

Fig: 8.2.3 Admin Reports


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Page Tools
Home
Reports
Sign Out
Done
Local intranet | Protected Mode: On
100%

Personal Financial Advisor

Admin Reports

Agents Under particular Organization

No. of Agents in each organization

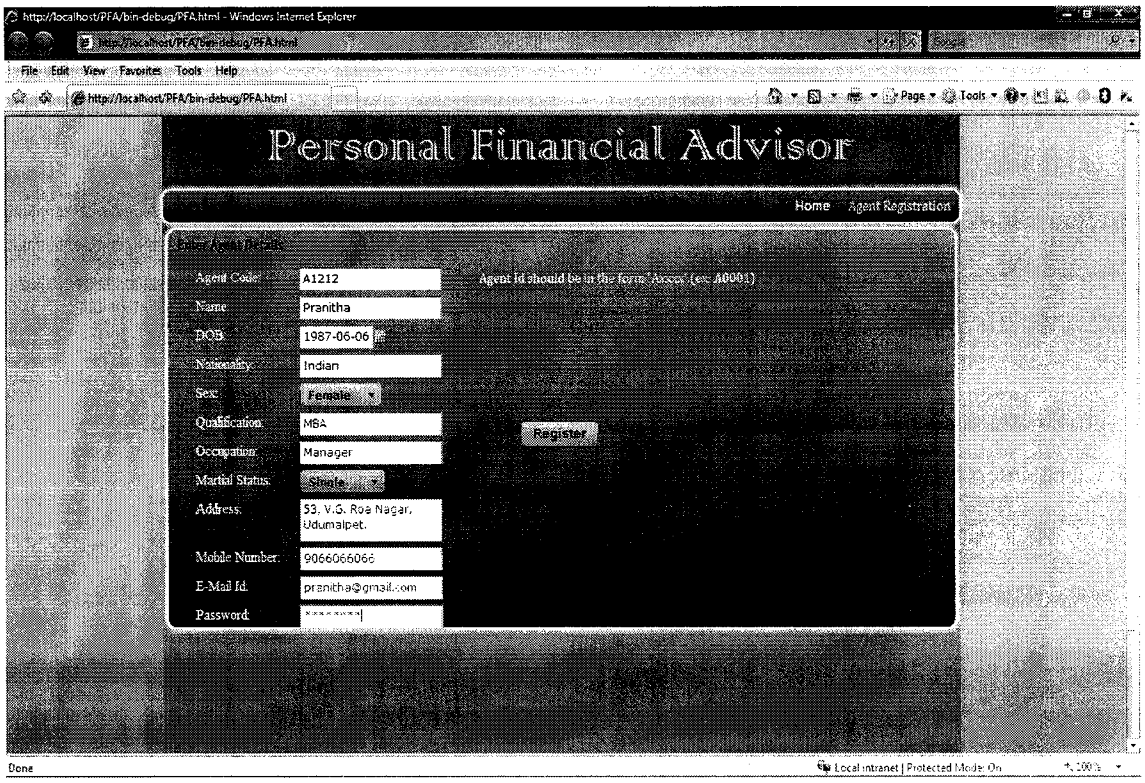
No. of Clients under each Agent

Agents Under Particular Organization

Select Organisation: LIC

AgentId	AgentName
A0001	Aswin.K
A0005	Vinithra.S
A0006	Ravi.M
A0009	Danu
A0012	Archana
A0013	Eswar
A0015	Dhiksha

Fig: 8.2.4 Agent Registration

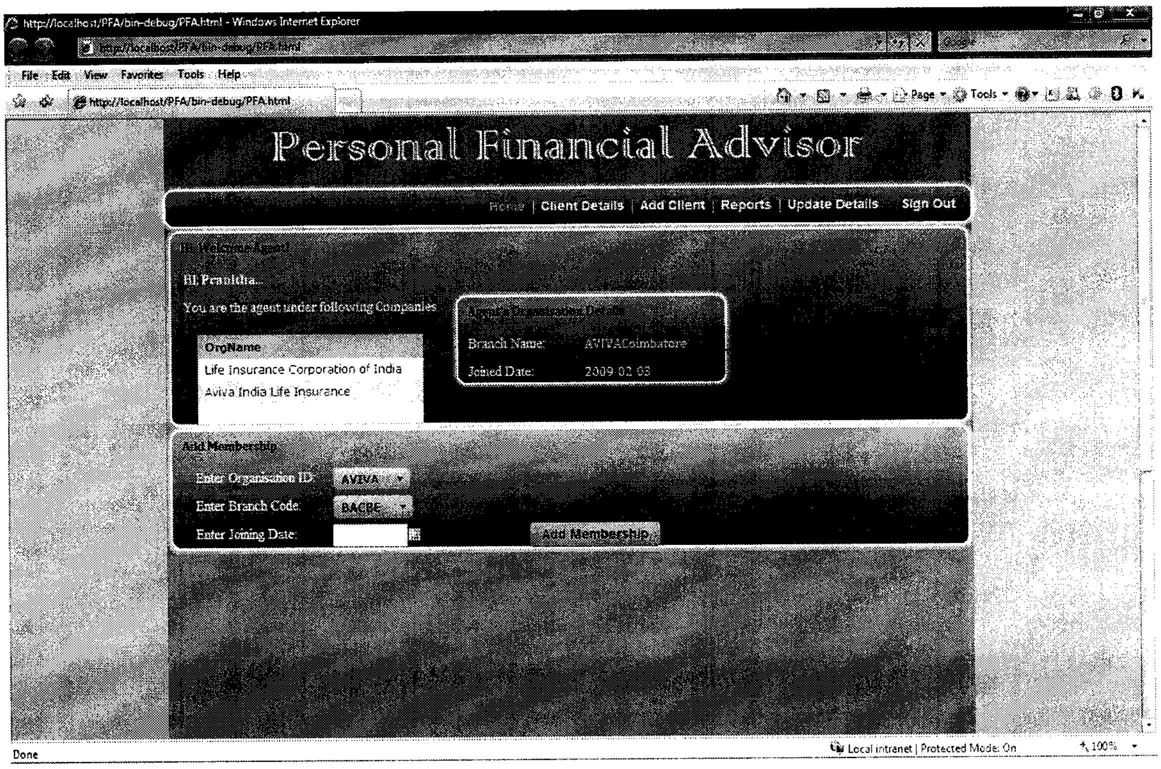

http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home
Agent Registration
Register
Done
Local intranet | Protected Mode: On

Enter Agent Details

Agent Code:	A1212
Name	Pranitha
DOB	1987-06-06
Nationality	Indian
Sex	Female
Qualification	MBA
Occupation	Manager
Martial Status	Single
Address	53, V.G. Roa Nagar, Udumalpet.
Mobile Number	9066066066
E-Mail Id	pranitha@gmail.com
Password	********

Agent id should be in the form 'Axxxx' (ex: A0001)

Fig: 8.2.5 Agent Home Page


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home | Client Details | Add Client | Reports | Update Details Sign Out
Add Membership
Done
Local intranet | Protected Mode: On
100%

Hi Welcome Agent!

HI, Pranitha..

You are the agent under following Companies

OrgName
Life Insurance Corporation of India
Aviva India Life Insurance

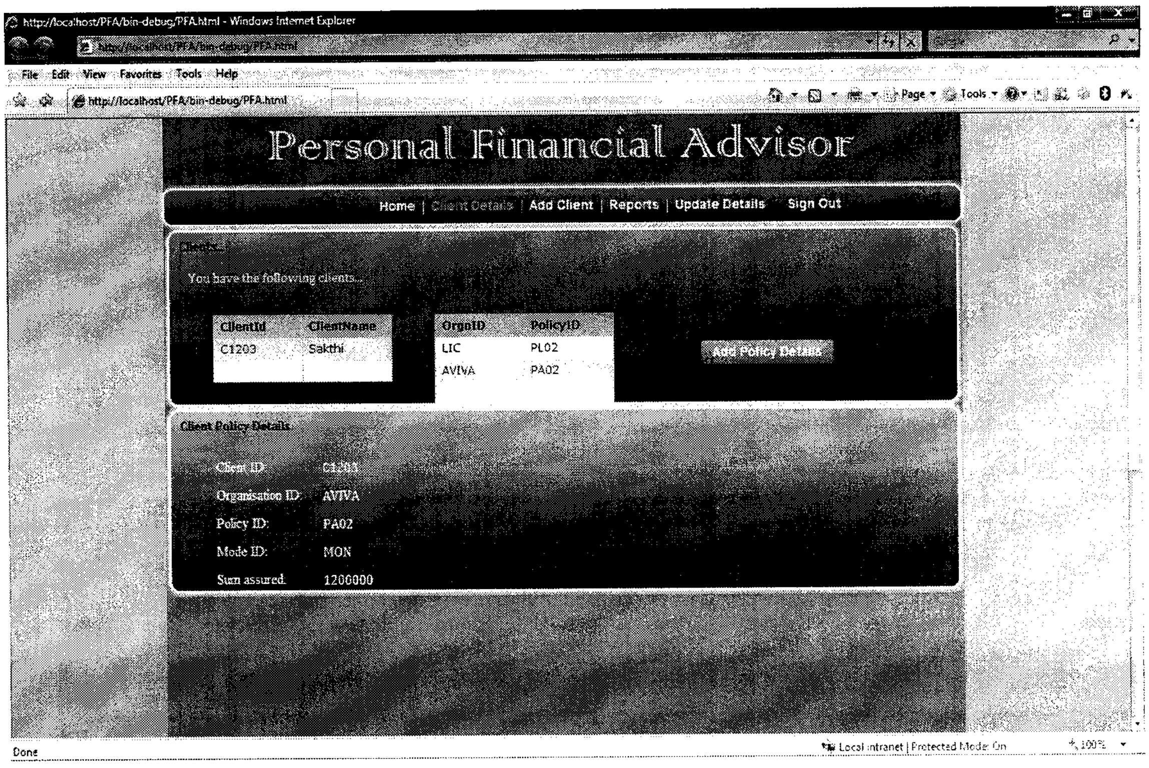
Agent's Organisation Details

Branch Name:	AVIVACoimbatore
Joined Date:	2009-02-03

Add Membership

Enter Organisation ID:	AVIVA
Enter Branch Code:	BACBE
Enter Joining Date:	

Fig: 8.2.6 Agent's Client Details Page


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
http://localhost/PFA/bin-debug/PFA.html
Page Tools
Personal Financial Advisor
Home | Client Details | Add Client | Reports | Update Details Sign Out
Add Policy Details
Done
Local intranet | Protected Mode: On
100%

Clients...

You have the following clients...

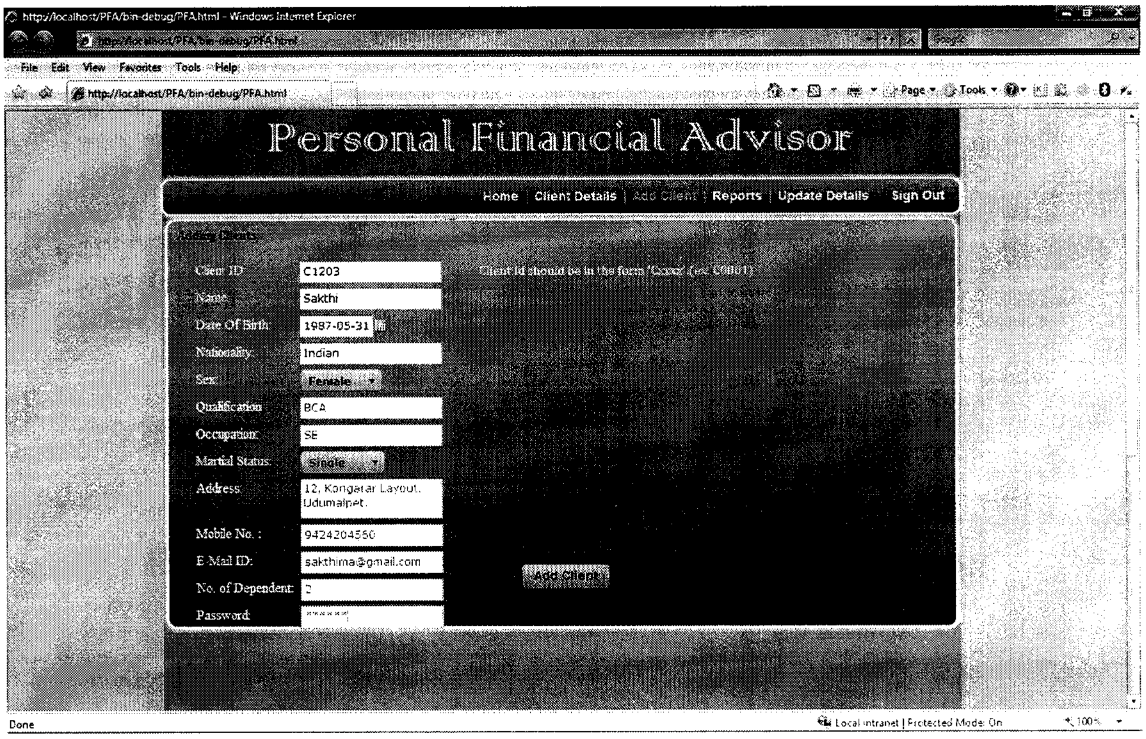
ClientId	ClientName
C1203	Sakthi

OrgnID	PolicyID
LIC	PL02
AVIVA	PA02

Client Policy Details

Client ID:	C1203
Organisation ID:	AVIVA
Policy ID:	PA02
Mode ID:	MON
Sum assured:	1200000

Fig: 8.2.7 Agent – Adding Client Page

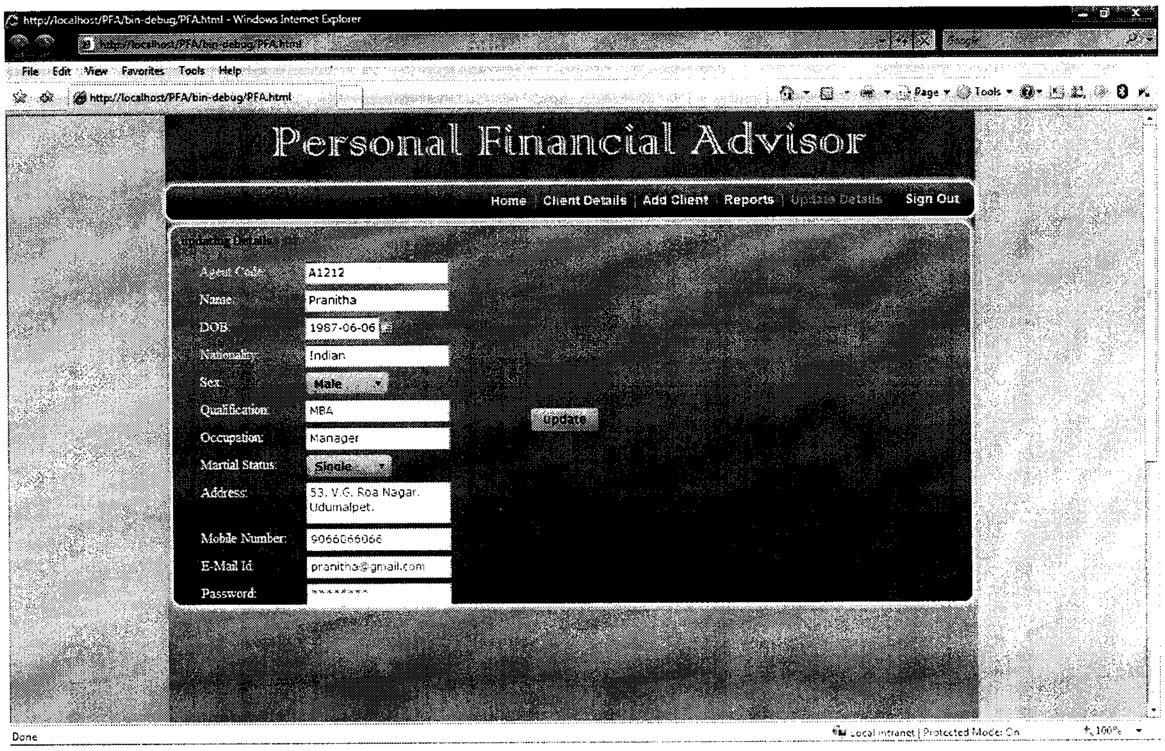

http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home | Client Details | Add Client | Reports | Update Details | Sign Out
Add Client
Done
Local intranet | Protected Mode: On

Adding Clients

Client ID:	C1203
Name:	Sakthi
Date Of Birth:	1987-05-31
Nationality:	Indian
Sex:	Female
Qualification:	BCA
Occupation:	SE
Martial Status:	Single
Address:	12, Kongarar Layout, Udumalpet.
Mobile No. :	9424204550
E-Mail ID:	sakthima@gmail.com
No. of Dependent:	2
Password:	******

Client Id should be in the form 'Cxxxx' (ex: C0001)

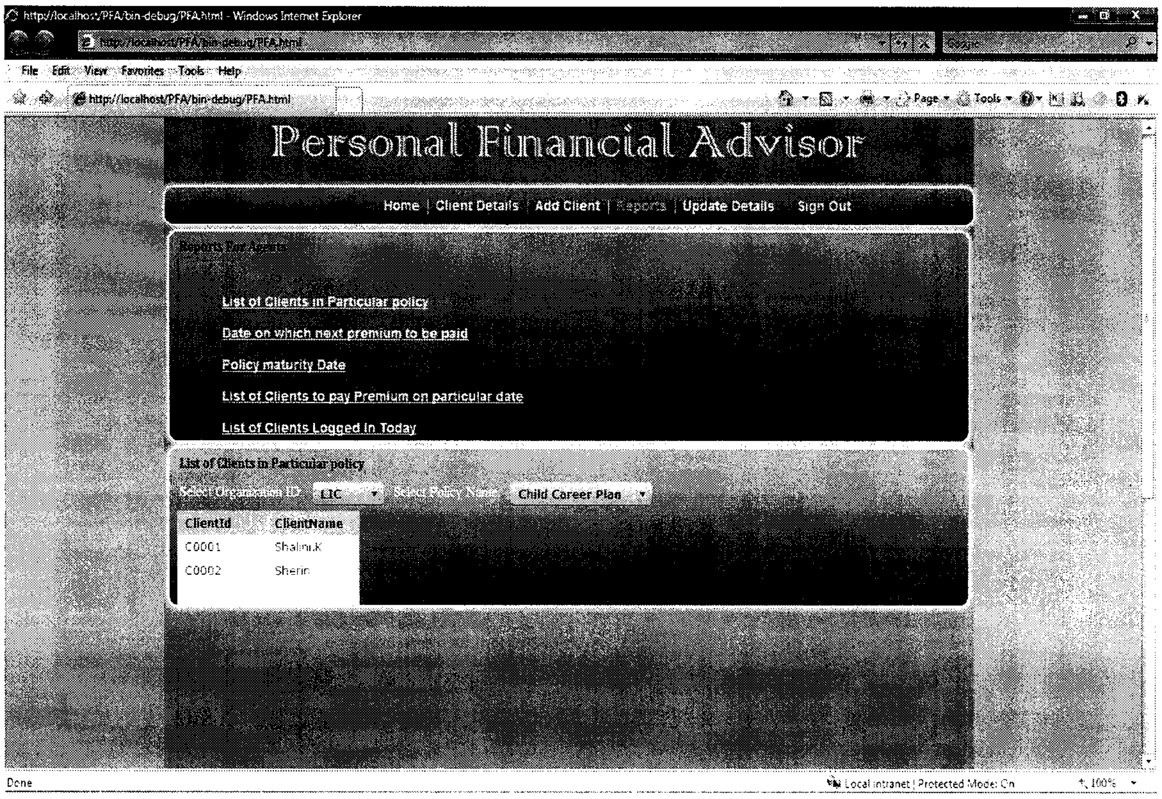
Fig: 8.2.8 Agent's Updating Details


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home | Client Details | Add Client | Reports | Update Details | Sign Out
Update
Done
Local intranet | Protected Mode: On

Updating Details

Agent Code:	A1212
Name:	Pranitha
DOB:	1987-06-06
Nationality:	Indian
Sex:	Male
Qualification:	MBA
Occupation:	Manager
Martial Status:	Single
Address:	53. V.G. Roa Nagar. Udumalpet.
Mobile Number:	9066066066
E-Mail Id:	pranitha@gmail.com
Password:	********

Fig: 8.2.9 Agent Reports – 1


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home | Client Details | Add Client | Reports | Update Details | Sign Out
Done
Local intranet | Protected Mode: On
100%

Reports For Agents

List of Clients in Particular policy

Date on which next premium to be paid

Policy maturity Date

List of Clients to pay Premium on particular date

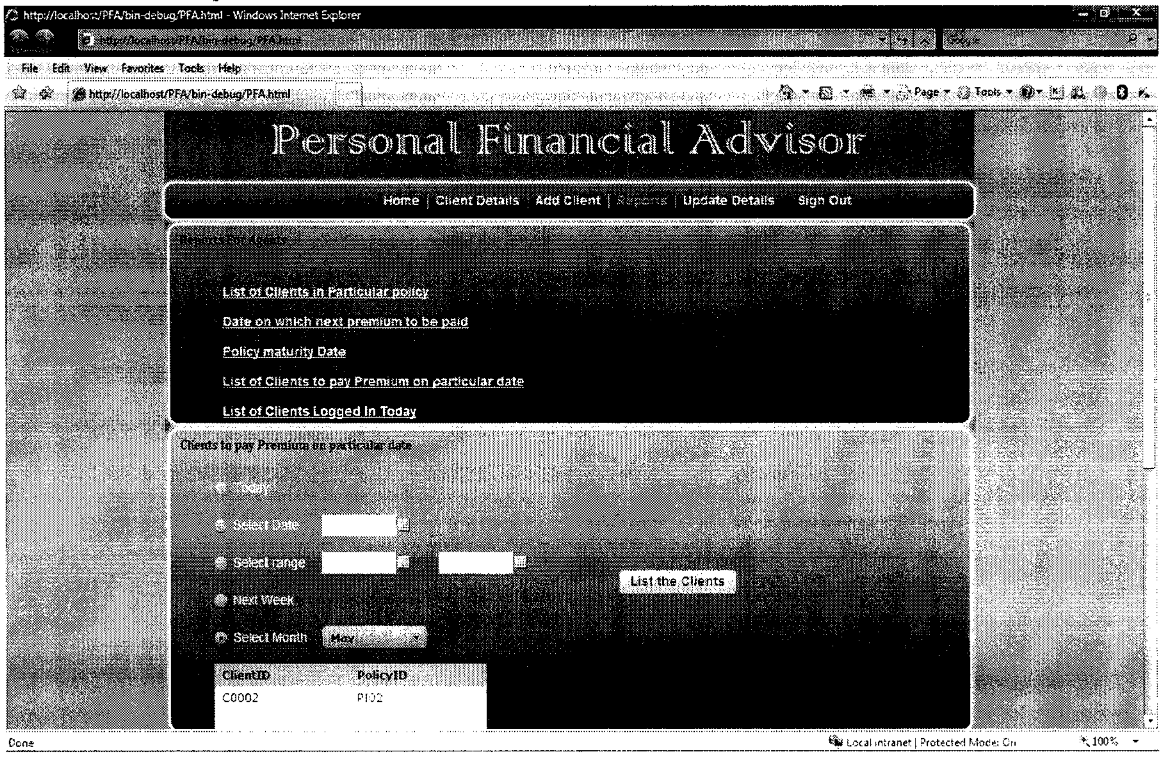
List of Clients Logged In Today

List of Clients in Particular policy

Select Organization ID: LIC Select Policy Name: Child Career Plan

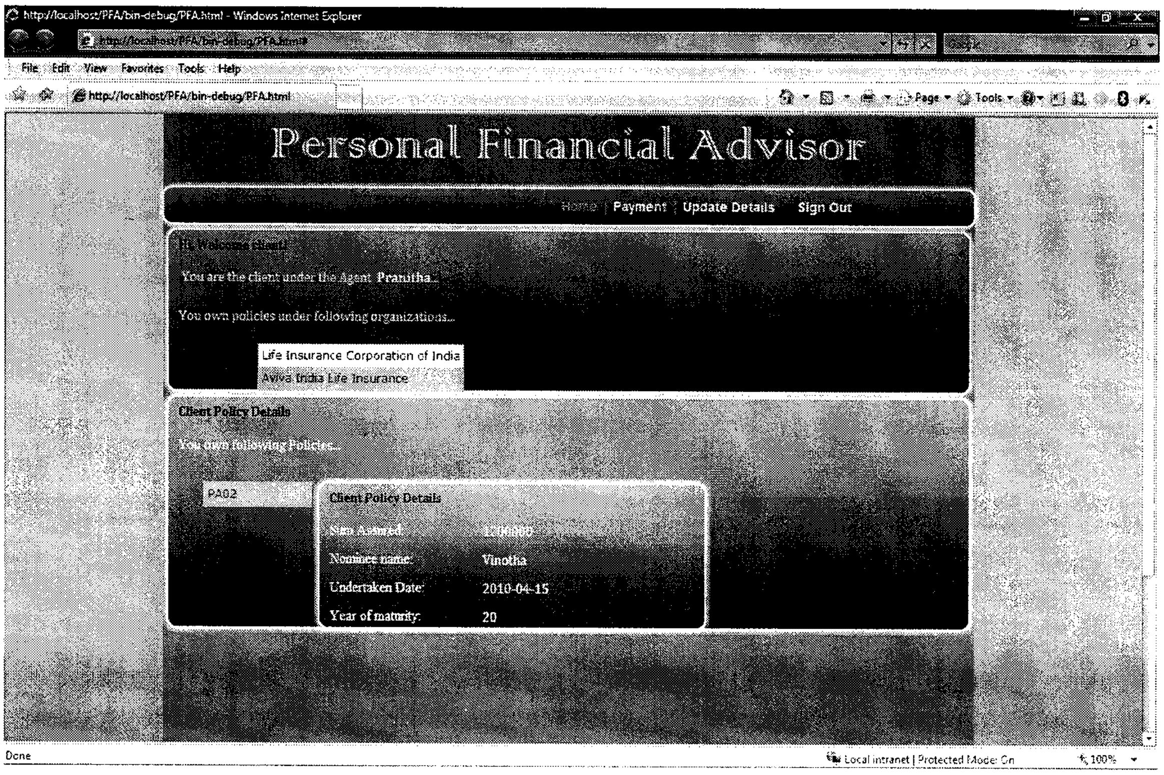
ClientId	ClientName
C0001	Shalini.K
C0002	Sherin

Fig: 8.2.10 Agent Report – 2


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home | Client Details | Add Client | Reports | Update Details | Sign Out
Reports For Agents
List of Clients in Particular policy
Date on which next premium to be paid
Policy maturity Date
List of Clients to pay Premium on particular date
List of Clients Logged In Today
Clients to pay Premium on particular date
Today
Select Date
Select range
Next Week
Select Month
May
List the Clients
Done
Local intranet | Protected Mode: On
100%

ClientID	PolicyID
C0002	P102

Fig: 8.2.11 Client Home Page


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
Personal Financial Advisor
Home | Payment | Update Details Sign Out
Done
Local intranet | Protected Mode: On
100%

Hi, Welcome client!

You are the client under the Agent **Pranitha**...

You own policies under following organizations...

Life Insurance Corporation of India
Aviva India Life Insurance

Client Policy Details

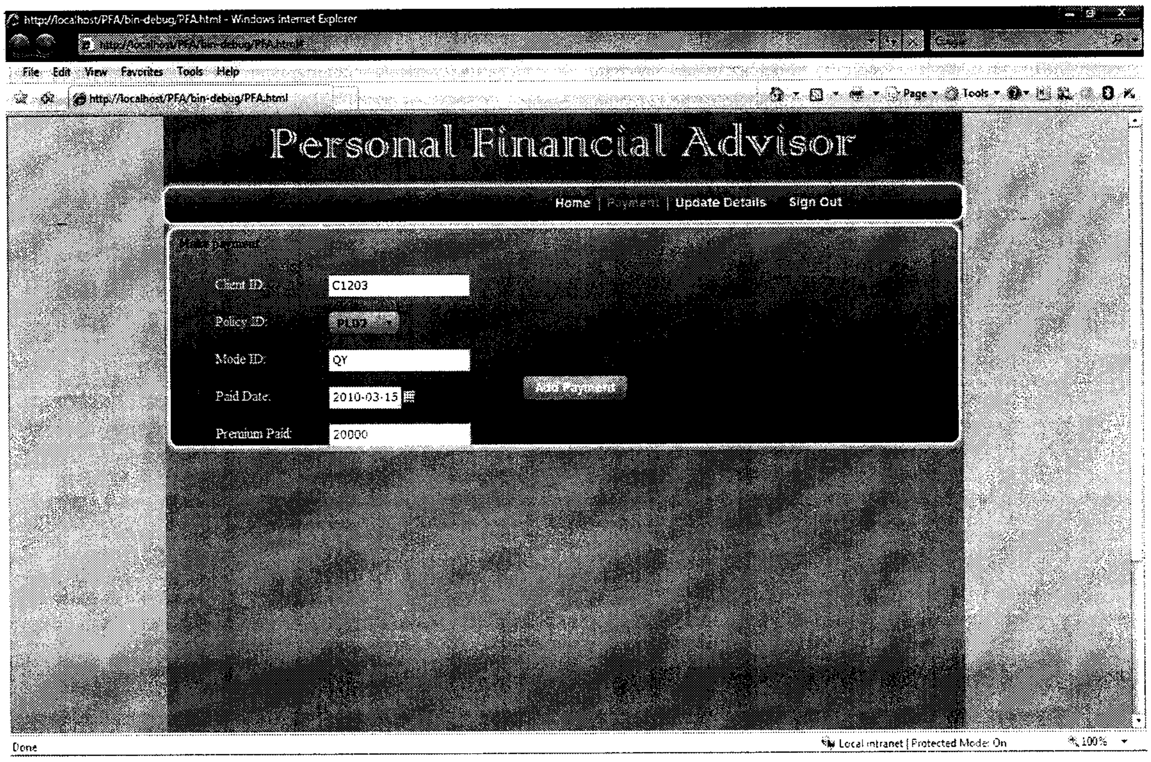
You own following Policies...

PA02

Client Policy Details

Sum Assured:	1200000
Nominee name:	Vinotha
Undertaken Date:	2010-04-15
Year of maturity:	20

Fig: 8.2.12 Client Payment Details Page



http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer

Personal Financial Advisor

Home | Payment | Update Details Sign Out

Make payment

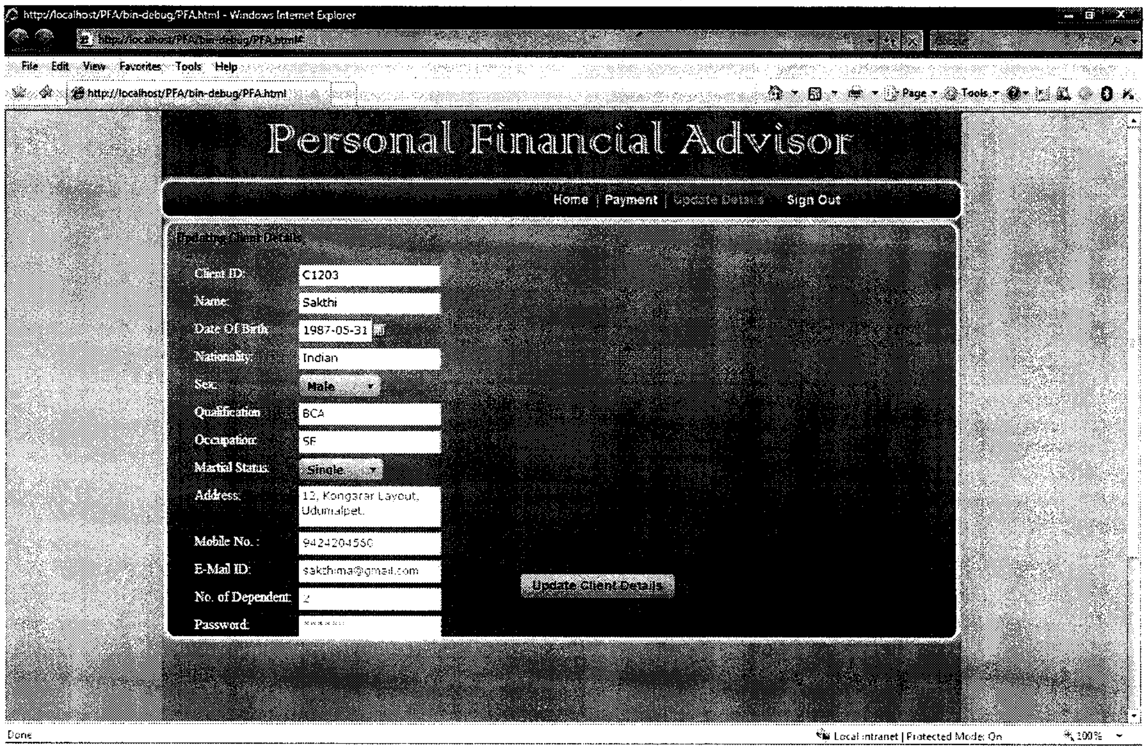
Client ID:	C1203
Policy ID:	PL02
Mode ID:	QY
Paid Date:	2010-03-15
Premium Paid:	20000

Add Payment

Done

Local intranet | Protected Mode: On

Fig 8.2.13 Client updating details


http://localhost/PFA/bin-debug/PFA.html - Windows Internet Explorer
File Edit View Favorites Tools Help
http://localhost/PFA/bin-debug/PFA.html
Personal Financial Advisor
Home | Payment | Update Details | Sign Out
Update Client Details
Done
Local intranet | Protected Mode: On
100%

Updating Client Details

Client ID:	C1203
Name:	Sakthi
Date Of Birth:	1987-05-31
Nationality:	Indian
Sex:	Male
Qualification:	BCA
Occupation:	SE
Marital Status:	Single
Address:	12, Kongarar Layout, Udumalpet.
Mobile No. :	9424204560
E-Mail ID:	sakthima@gmail.com
No. of Dependent:	2
Password:	******

CHAPTER 9
REFERENCES

1. **Learning Flex 3 – Alaric Cole -** O'Reilly Media, Inc., June 2008.

2. **Adobe® Flex® 3 Developer Guide** – Adobe Systems, 2008.

3. **PHP 6/MySQL® Programming for the Absolute Beginner – Andy Harris** – Course Technology, 2009.

4. **http://livedocs.adobe.com/flex/3**

5. **http://www.w3schools.com/PHP**

6. **http://devzone.zend.com**

7. **http://www.php.net/manual**

8. **http://www.actionscript.org**