Acknowledgement

I am thankful to MES's Institute of Management and Career Courses, for giving me the opportunity to carry out my post-graduation project.

I am grateful to Dr. V. H. Inamdar, Director IMCC, Dr. Santosh Deshpande, HOD Computer Department, IMCC, Mrs Apurwa Barve Ma'am (Project Guide),Sandeep Nimbalkar sir and Dr. Manasi Bhate, TPO, IMCC for their valuable guidance & expertise suggestions, without which this project would not have been successful.

I would like to thank Mr. Atul Thombre, C.E.O of iSynergy Systems, who gave me an opportunity to develop this system. I would like to thank Mr. Nachiket Karguppikar for guiding me throughout the project. I would also like to thank all colleagues in iSynergy Systems, their support and co-operation made my tasks a lot easier and interesting.

Ruchi Bhagwat

INDEX

CHAPTER 1: INTRODUCTION

- ## <u>1.1 COMPANY PROFILE</u>

<u>OVERVIEW:</u>

iSynergy is a value-driven organization offering Outsourced Product Development and Technology Services in various technology domains. Company has a regional presence in US, India and South Africa with a customer base spread across the geographies. Most of our customers work in cutting-edge technology applications, web solutions, mobile applications, enterprise solutions, integration services and so on. They are innovative technology companies, VC funded companies on growth spree, start-up companies and mid-sized software product and services companies.

iSynergy’s Outsourced Product Development Services are built around your business and technology needs. It covers the entire spectrum of services starting from conceptualization to deployment and migration support. Our expertise and experience in technology architecture, flexibility and adaptability to changes, versatile and dedicated technology culture has an

optimizing effect on your performance in designing, developing and deploying technically critical solutions. Our teams have proven expertise in Java, Microsoft Technologies, Open Source Technologies, Mobile platforms and so on.
iSynergy's Technology Services offer full length consulting in the area of technology decisions, customised solutions and enterprise application integration. It has successfully delivered projects which demanded command over core technology.

iSynergy's technology expertise, understanding of the customers' business, value system, ability to walk with the customer, spirit of ownership, pro-activeness and responsiveness are some of the critical factors in its success story. iSynergy responsibly creates and adds value to the top-line of its customers' businesses to maintain their uniqueness.

another and businesses are turning the opportunity to deliver its advantages to their users and customers.

iSynergy understand the future available with mobile technologies and, therefore, has developed its team to work in the domain. We have successfully developed and deployed mobile applications on Blackberry using J2ME and RIM. We have delivered applications on iPhone, Android, and Windows Mobile using Microsoft .Net CF. These applications communicates with the server, builds interactivity and enables you to run your business when you are the move.

Migration

Change in the technology environment and customers operative scenario demands technology migration at different level. iSynergy provides the technology migration services to ensure such smooth transitions for all the stakeholders. It starts with assessing the technology situation and business requirements, and decides the most effective mechanism. iSynergy Migration Services covers Application Migration, Database Migration & Cross Technology Migration.

Customized Projects

Delivering a software development project is probably a service most of the companies provide today. You have already seen enough of flow charts and diagrams explaining delivery models and methodologies. Still, you had to burn your fingers earlier, which compels you to think ten times before you decide on outsourcing it to someone!

The success solely depends on the outsourcing team and their approach. Finally it’s all about the people. Right people will set right processes to take care of your uniqueness! At iSynergy, this has been the strength since inception. We have successfully delivered customised projects in Web solutions, Enterprise applications and Business solutions.

Enterprise Application Integration

Diverse IT infrastructures, fast paced business and customer needs, limitations of legacy systems architecture creates a need to integrate applications for seamless flow of information between two systems. iSynergy’s EAI delivery framework has helped organizations to crack their information barriers across enterprise. A comprehensive approach towards EAI helps to deal with the challenge of integration. Our expertise in integration of applications and tools, middleware products and expertise in enterprise solutions delivers satisfying experience to you.

1.2 EXISTING SYSTEM

The existing system manages the leads and project details manually. All the information is stored in separate excel sheets.It records details such as-

1. Enquiry details

The very first meeting or enquiry regarding project development is recorded manually excel sheets.

This information is provided to the admin official and he/she enters the information manually.

2. Client details

The client coming up with the requirement ,his details are recorded for future need.

3. Project details

Details of the project such as start date,end date,technology used,requirements,possible schedule,team members working on it is decided and recorded.

4. Keeping track of the payments

The client may decide to pay the full amount at once or in installments.These details are stored.Calculation of amount left to be paid in case of advance
payment,installments is done manually and stored.

5. Generating invoices-domestic and international

Company handles domestic as well as international clients.Calucation and generation of invoices is diferent for each type of client.These calculations are also
done manually.

6. Generating payment reminders

It the task of the account section to keep track of payments,keep in check which client's payment has been completed and which client's payment is pending.

Calculating amount left and sending reminders to respective clients for payment is another task.

7. Reports having different views for different compliances

Existing system requires generation of reports having different views for different compliances.These reports are generated manually referering to multiple excel sheets.

DISADVANTAGES OF EXISTING SYSTEM:

Maintaining and recording these process is tedious and time consuming. This process becomes difficult to handle when the number of clients and projects increase.

There are many limitations for the existing systems.

In manual monitoring, all the work done in the existing scenario is by human intervention which may increase the chance of errors.

Generation of reports is yet another tedious task as it requires going through a lot of excel sheets and filtering of information.

Inconsistency

> Manual data recording is inconsistent because the devices used need precision and it is not possible to that extent for human being. E.g. In order to get exact or accurate customization for product the data has to be recorded in a precise manner and order status has to be managed properly in order to record accurate data.

Error prone

Human based systems are always prone to errors due to miscalculations and carelessness.

Expensive

Human beings need to be paid for their work which in turn is not beneficial to the organisation due to its inaccuracy. This is a bad investment for an organisation and demands for some other alternative for the same.

NEED FOR THE SYSTEM

Lead management system is an automated version of the existing system.This system overcomes the drawbacks of the existing system by maintaining and recording the data and generating reports as and when needed in the form required
by the personnel.
The above mentioned process will be automated. This will not only save time but also make it easier to maintain and retrieve information.

• 1.3 SCOPE OF WORK

Lead management system is an application which focuses on managing leads and generation of invoice receipt.

A project starts from an idea/enquiry which is further elaborated and converted into a project.

The project goes through the development life cycle and generation of milestones and ultimately an invoice in pdf printable format.

This software records every step of the project which includes-

1. Enquiry details
2. Client details
3. Project details
4. Keeping track of the payments
5. Generating invoices-domestic and international
6. Generating payment reminders
7. Reports having different views for different compliances

What view of the report will be generated depends on the user accessing the system. Reports will be generated in the form of graphs and sheets. There will be categories eg: client wise reports, monthly reports, revenue wise reports etc.

This software automates the process.

• 1.4 OPERATING ENVIRONMENT

(Hardware and Software)

Hardware required

- Processor: 1 gigahertz (GHz) or faster

RAM: 1 GB

Hard disk: 40 GB

Internet: Modem with speed of 1MB per second.

Display: 17’ Monitor

Software Required-

Front-End-

Web browser such as Internet Explorer, Chrome or Firefox.

ReST API

Back-End-

MySQL Server 4.0

OS – windows,linux

• 1.5 DETAILED DESCRIPTION OF TECHNOLOGY USED

- **Java:**

Writing in the Java programming language is the primary way to produce code that will be deployed as byte code in a Java Virtual Machine (JVM); byte code compilers are also available for other languages, including Ada, JavaScript, Python, and Ruby. In addition, several languages have been designed to run natively on the JVM, including Scala, Clojure and Groovy.

Java syntax borrows heavily from C and C++, but object-oriented features are modelled after Smalltalk and Objective-C. Java eschews certain low-level constructs such as pointers and has a very simple memory model where every object is allocated on the heap and all variables of object types are references. Memory management is handled through integrated automatic garbage collection performed by the JVM.

- **Spring Framework:**

The Spring Framework is an open source application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform. Although the framework does not impose any specific programming model, it has become popular in the Java community as an alternative to, replacement for, or even addition to the Enterprise JavaBean (EJB) model.

- **jQuery:**

jQuery is a cross-platform JavaScript library designed to simplify the client-sidescripting of HTML. jQuery is the most popular JavaScript library in use today. jQuery is free, open-source software licensed under the MIT License.

jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and web applications.

- **<u>ReST API</u>:**

REST stands for Representational State Transfer. It is sometimes spelled "ReST". It relies on a stateless, client-server, cacheable communications protocol -- and in virtually all cases, the HTTP protocol is used. REST is an architecture style for designing networked applications.

- **<u>JSON (JavaScript Object Notation)</u>:**

JSON is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others.

CHAPTER 2:

PROPOSED SYSTEM

• <u>2.1 PROPOSED SYSTEM</u>

-Proposed system focuses on automating the entire process of lead management right from enquiry stage to delivery of the product.

-It makes use of latest technology and overcomes the drawbacks of manual work.

-The system being a web based application,can be updated from any place,so location is not a barrier.

-Reports can be easily generated saving lot of time and effort.

- **<u>Middleware</u>:**

  Middleware Platform will be developed using Java. Communication will be done using ReST APIs. Data will be sent in JSON format. Database used for storing the data will be SQL Server. It will have the following component

- ## 2.2 OBJECTIVES OF THE SYSTEM

-automate system

-save time

-save efforts

-increase efficiency by reducing errors in generation of result as everything will be done by the machine.

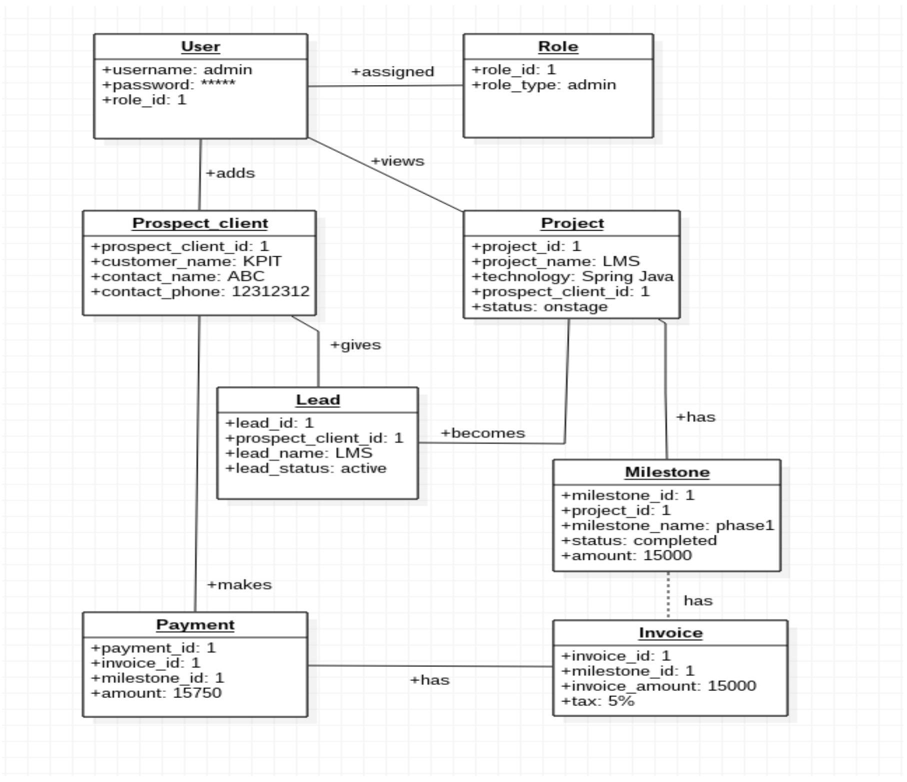
2.3 USER REQUIREMENTS

- Users of the system can be able to use the system from any place.
- Automation of the lead management process.
- Maintaining,sorting,searching data should become easier.
- Report generation should not be a tedious task.
- Using of the application should be easy and user friendly.
- Result generated by the application should be correct and reliable.
- Storing large amounts of data should be easier and faster.

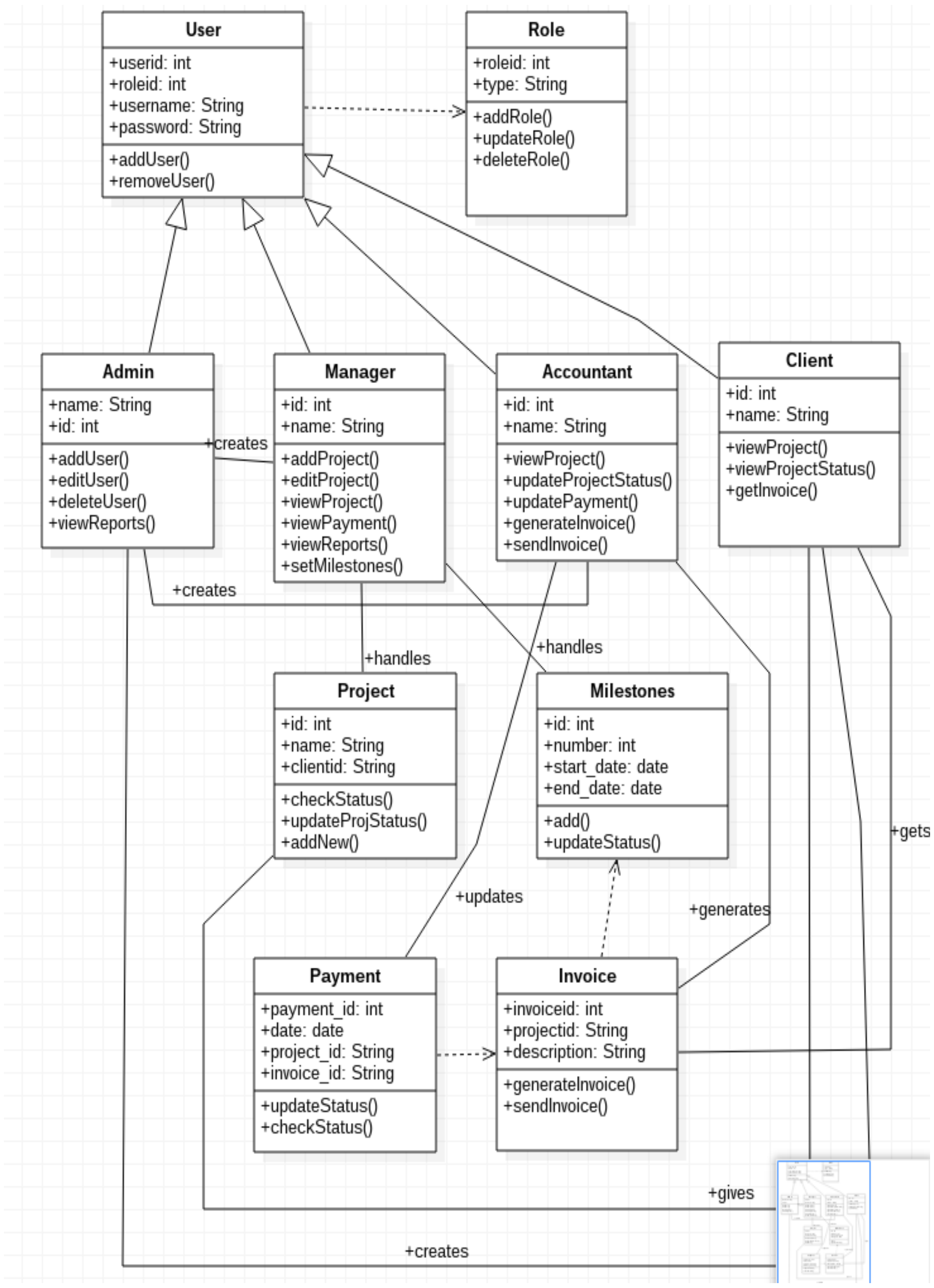
CHAPTER 3

ANALYSIS AND DESIGN

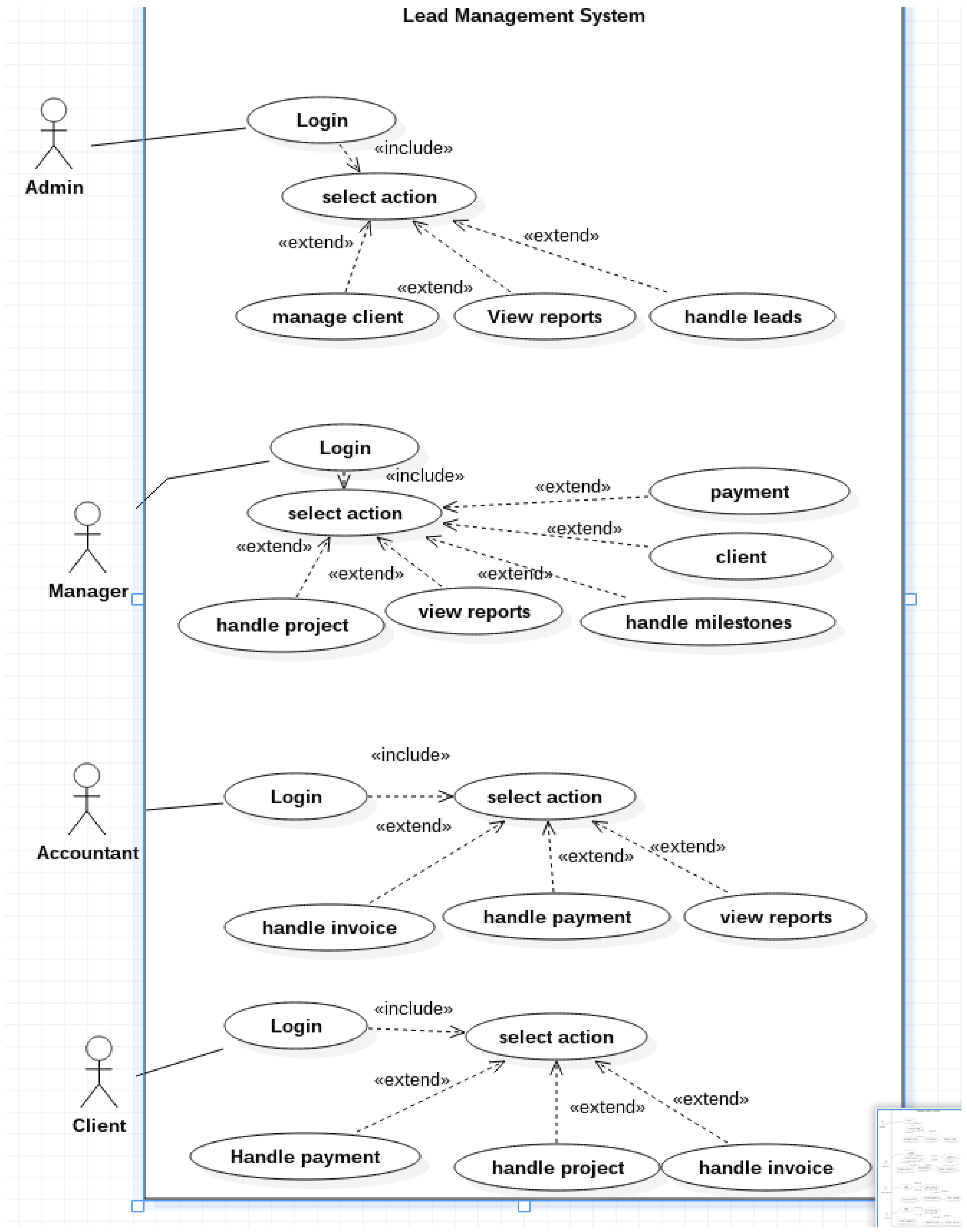
3.1Object Diagram



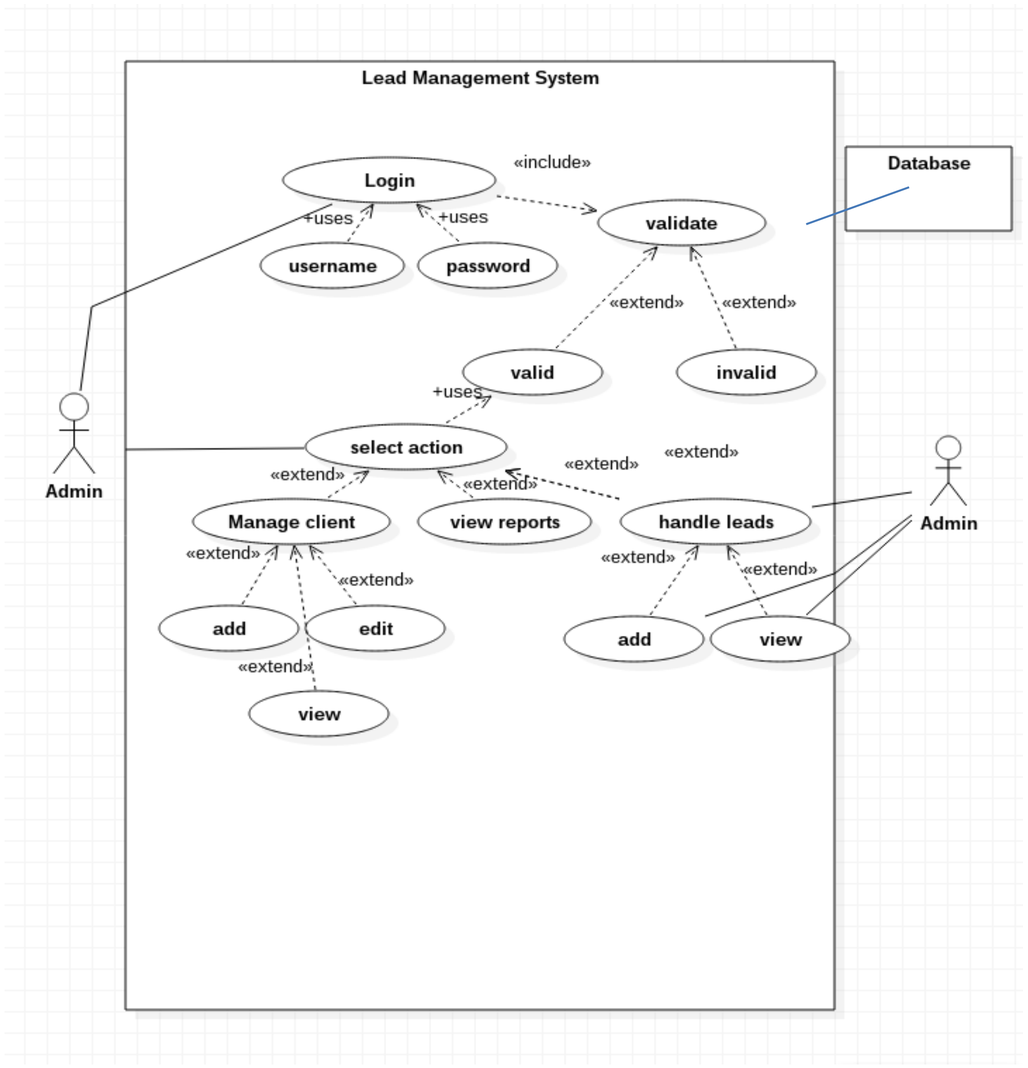
3.2Class diagram



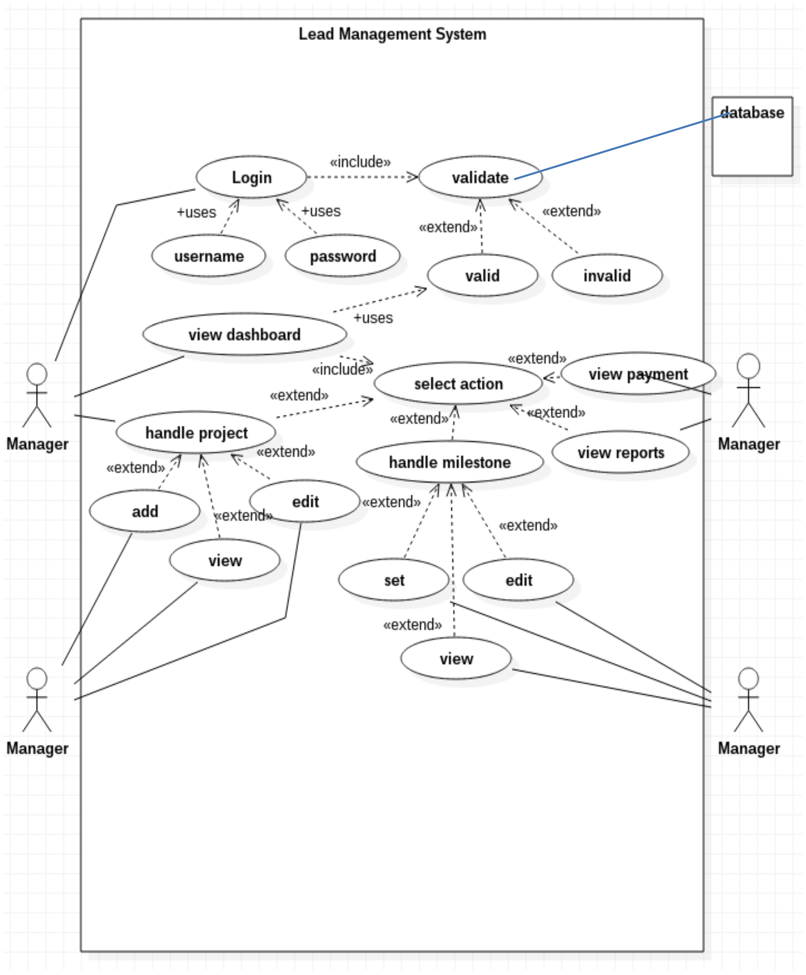
3.3Use Case Diagram (Main Use Case)



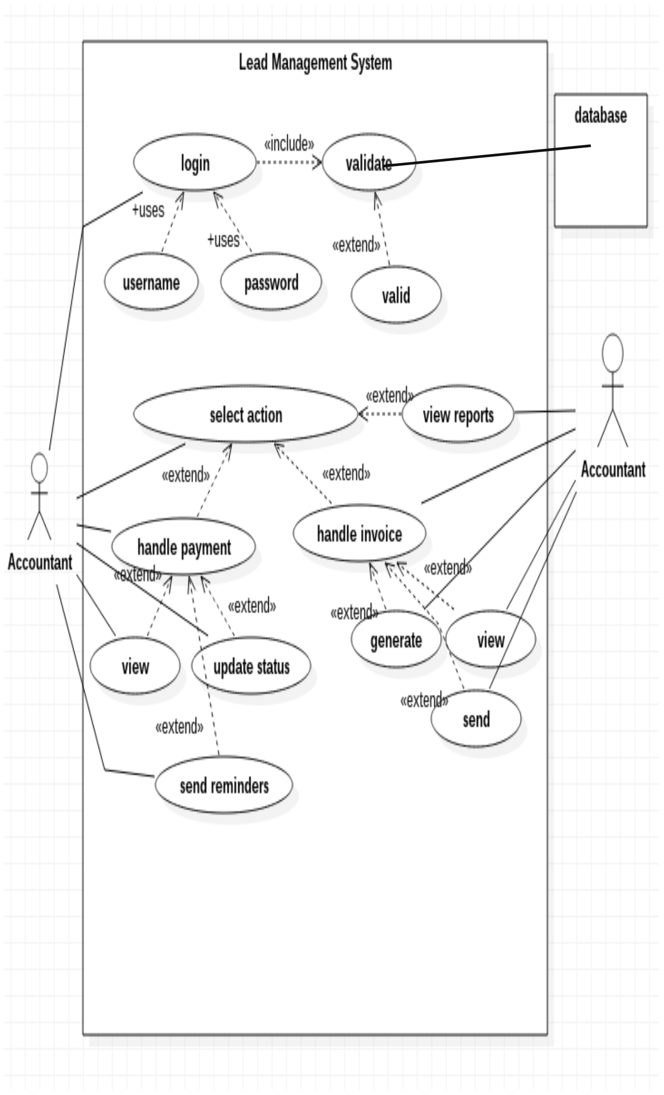
Admin use case



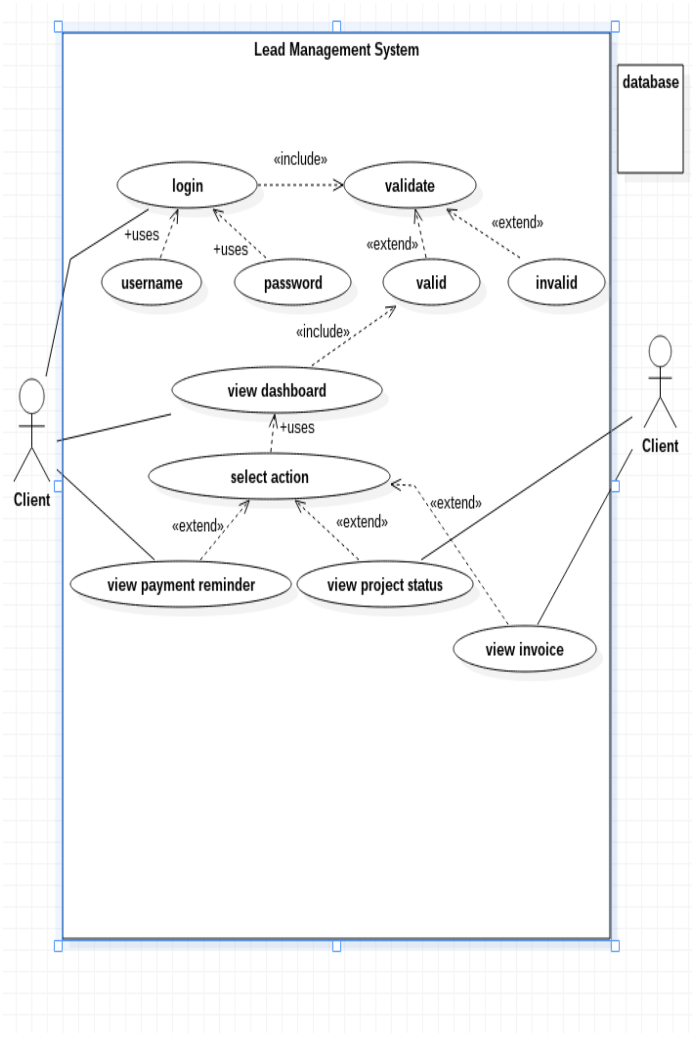
Manager use case



<u>Accountant use case</u>

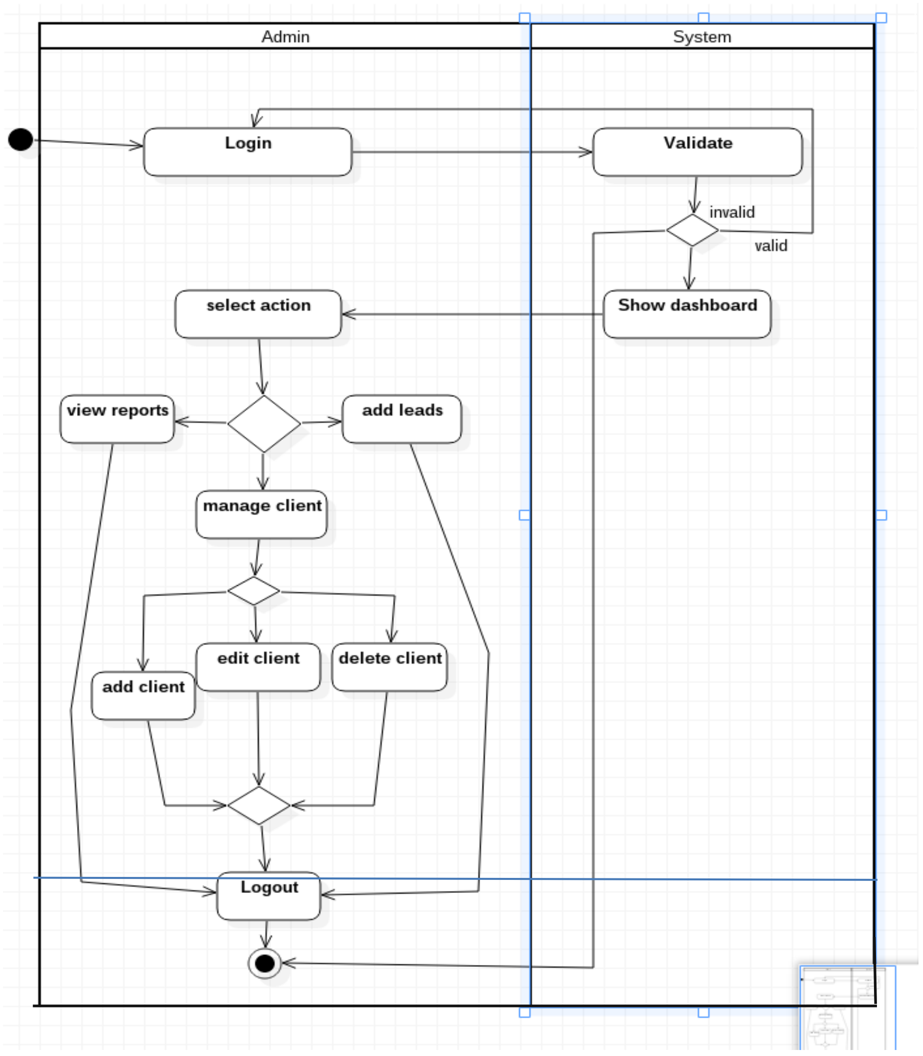


Client use case

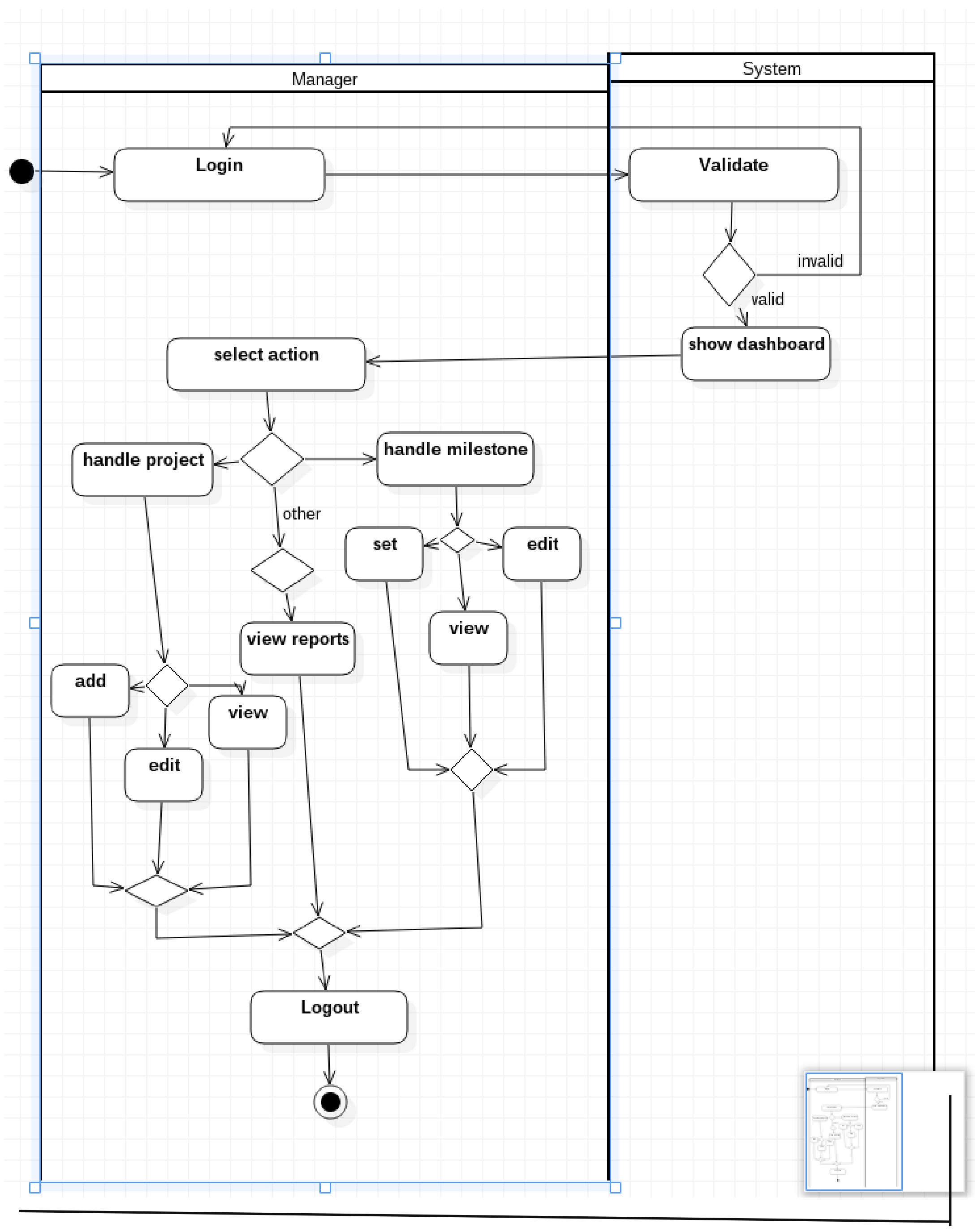


3.4 ACTIVITY DIAGRAM

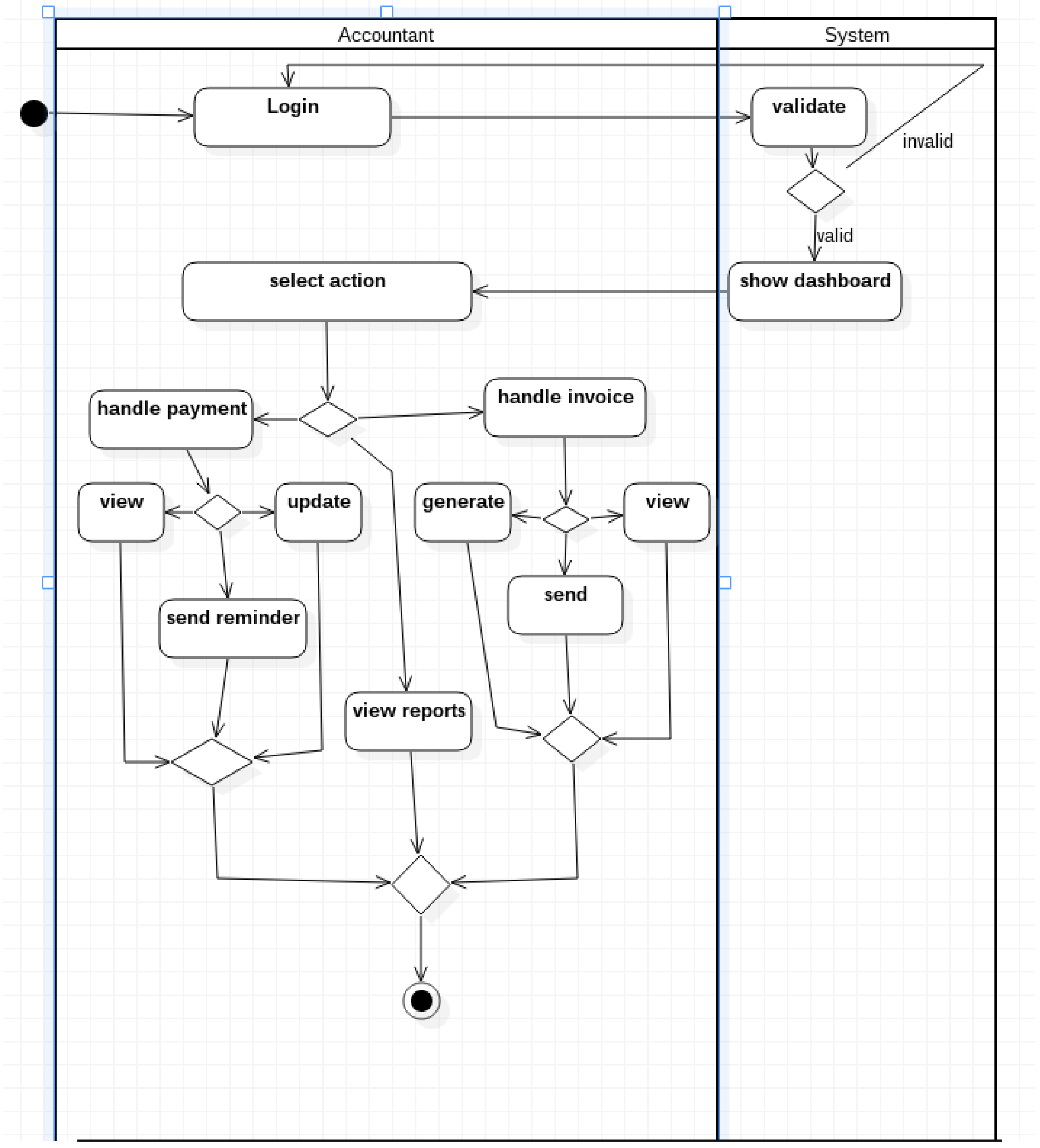
Admin activity diagram



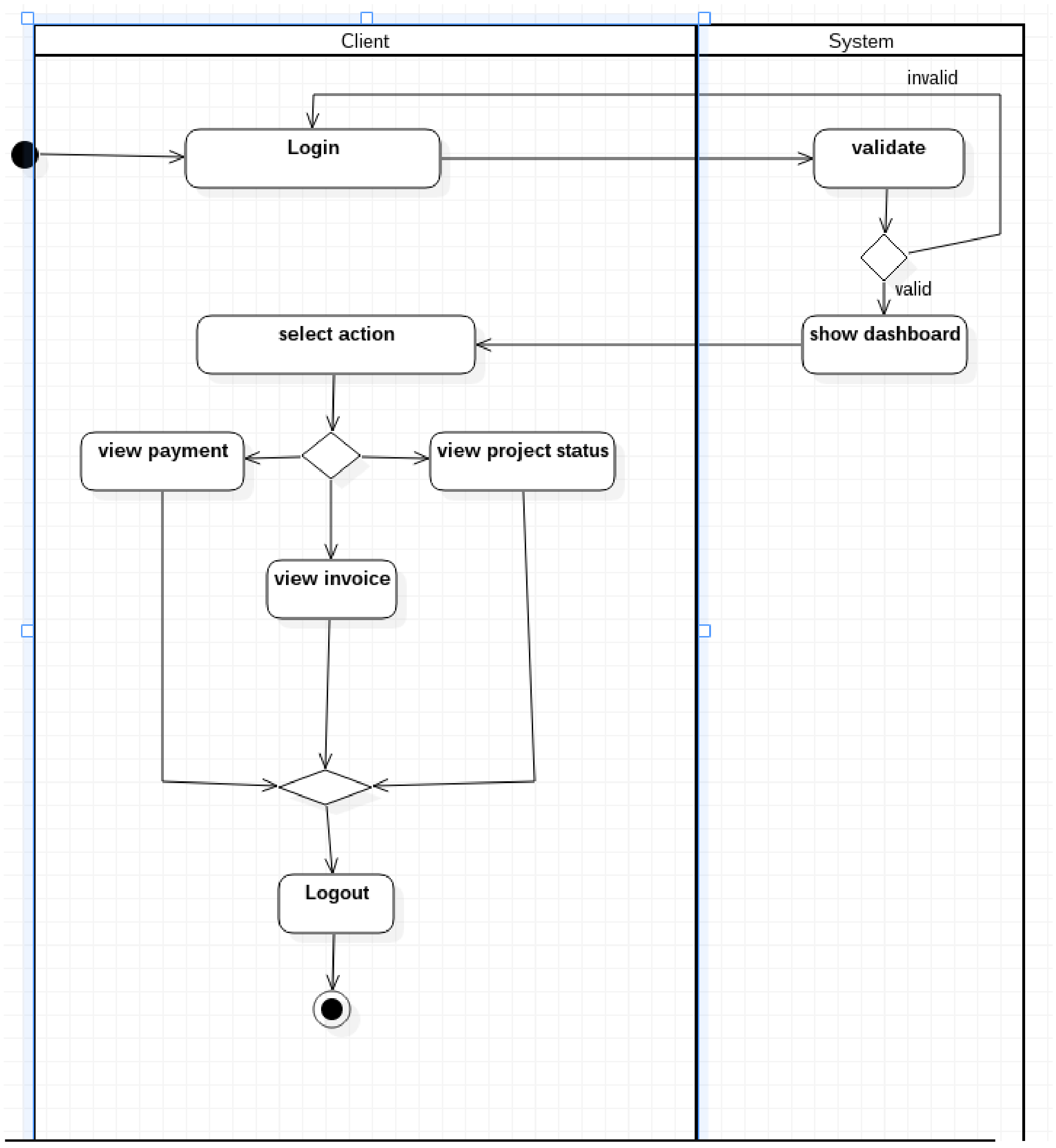
2. Manager Activity Diagram



3 Accountant Activity diagram

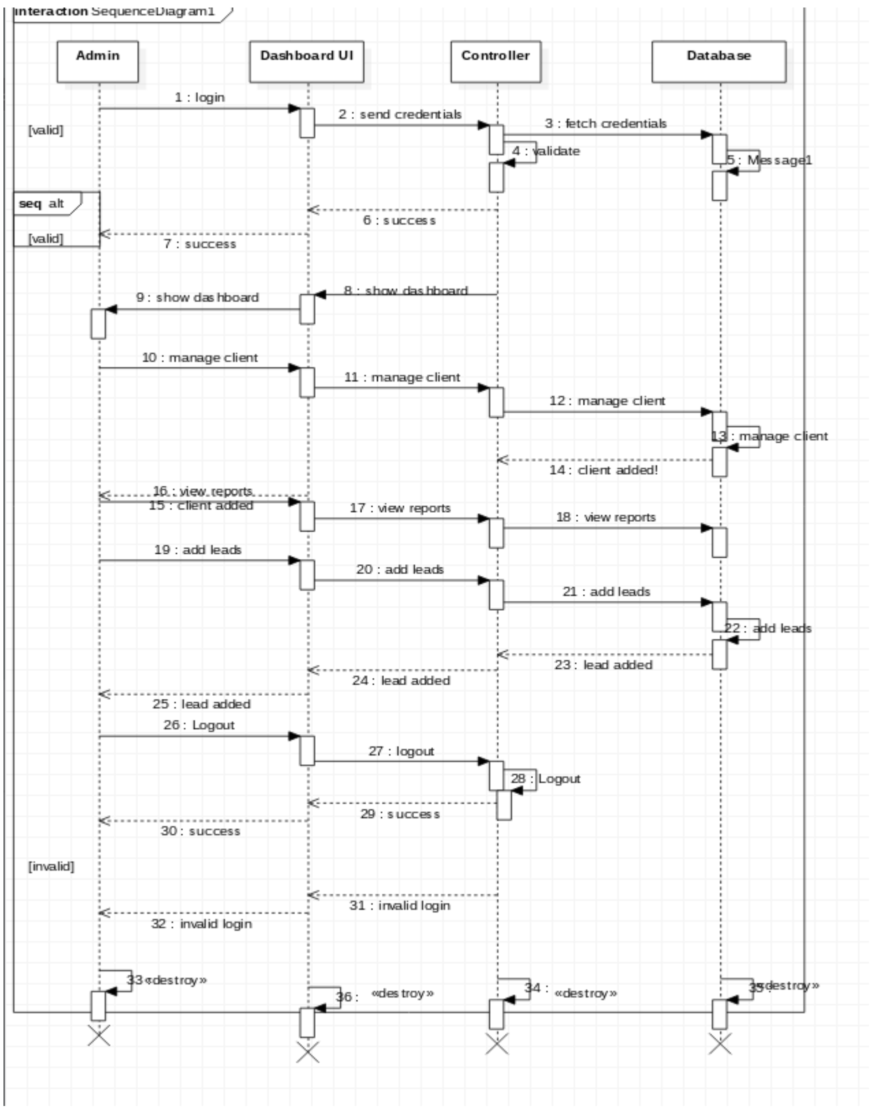


4 Client Activity Diagram

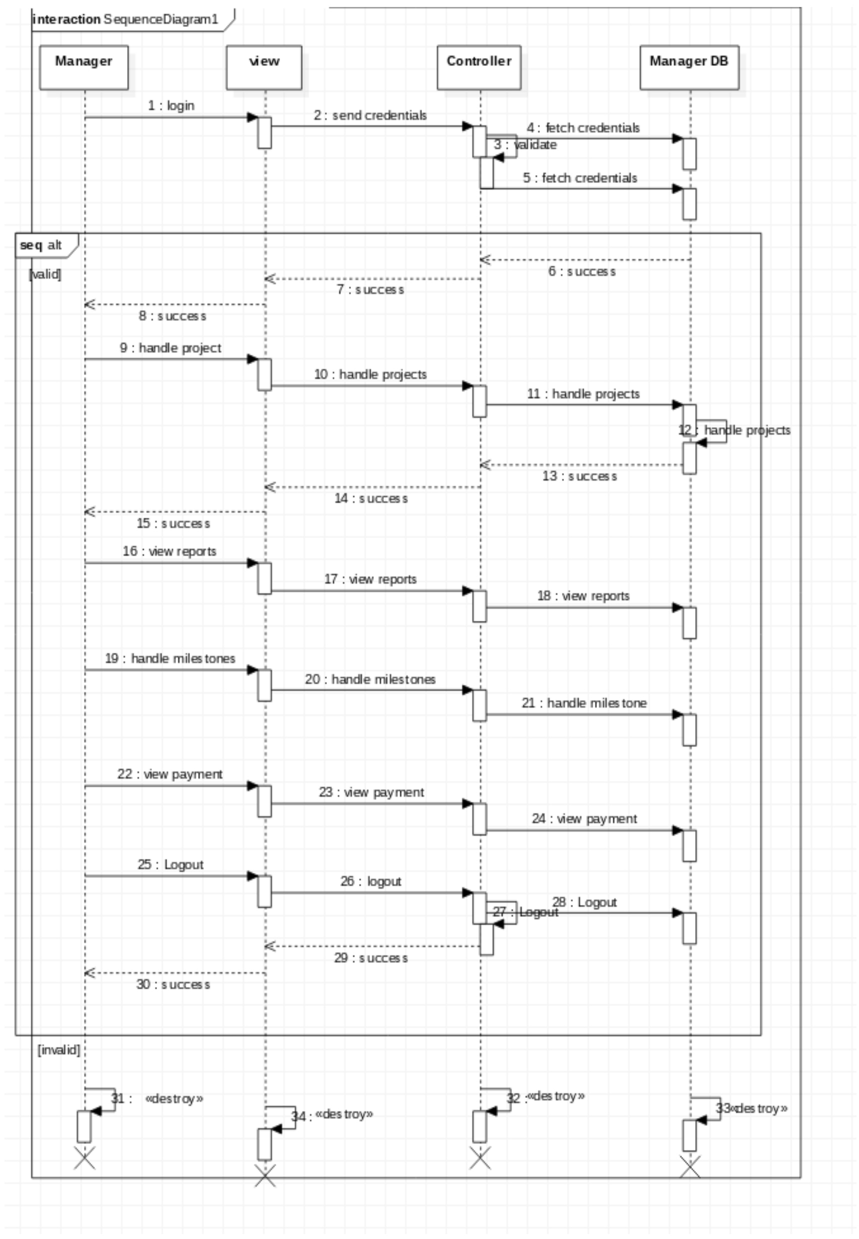


3.5 Sequence Diagram

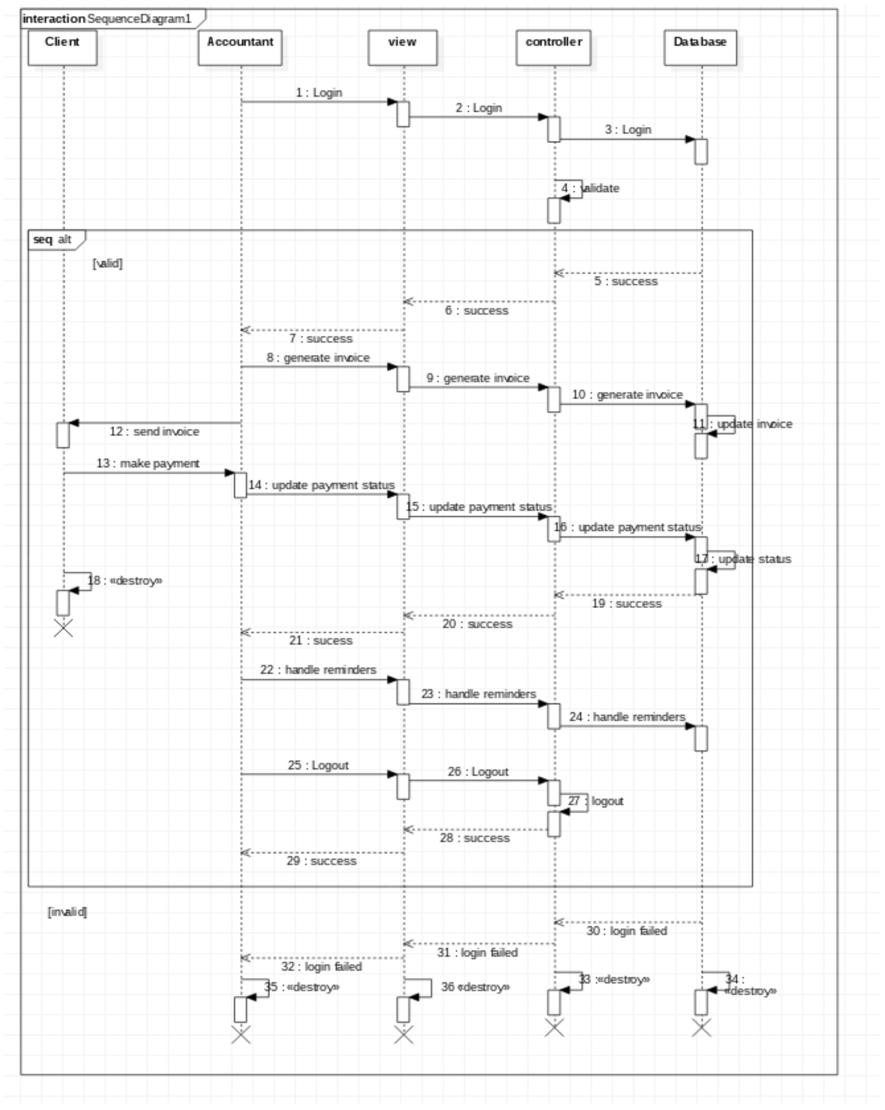
1 Admin Sequence diagram



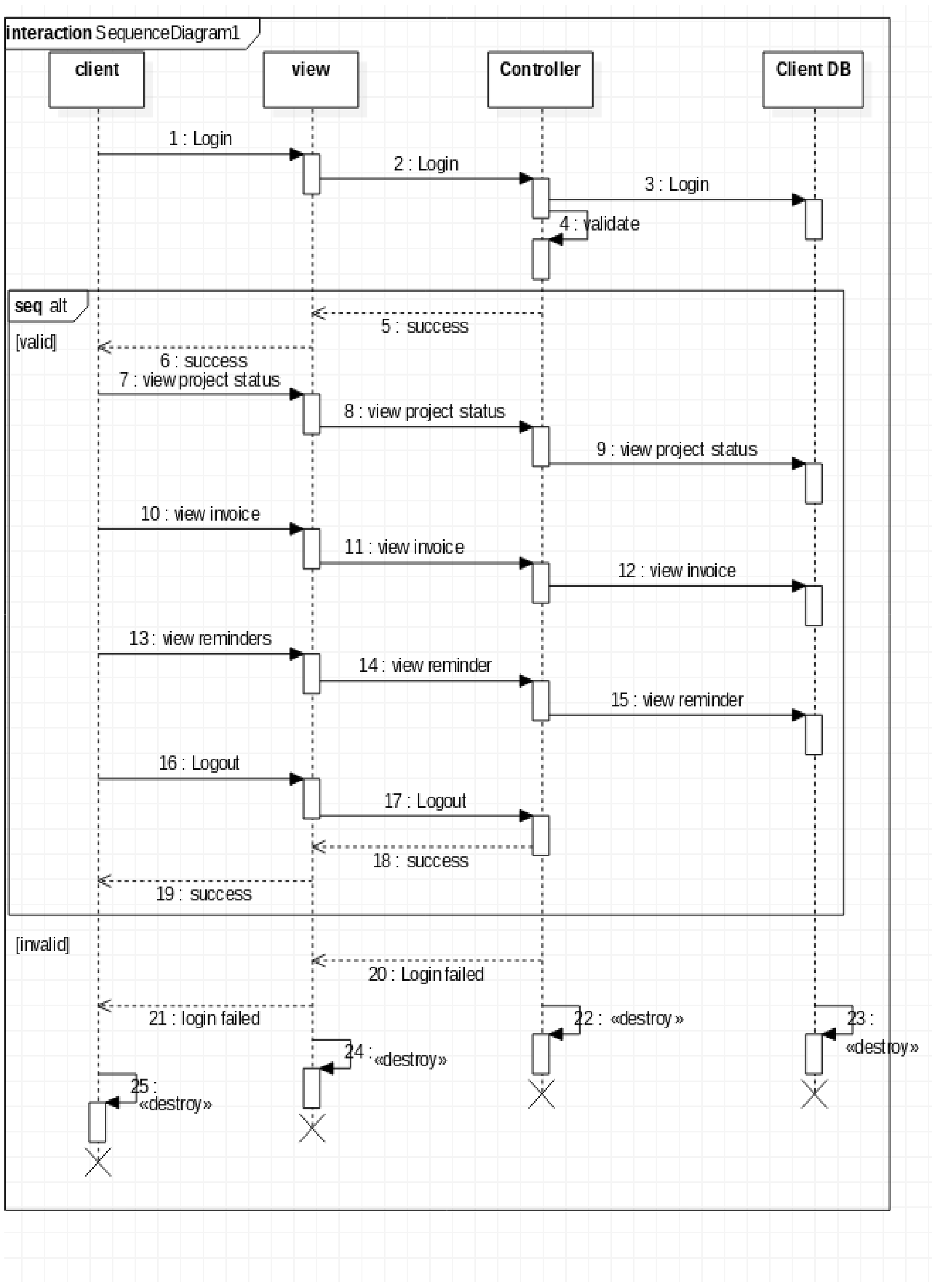
2 . Manager Sequence Diagram



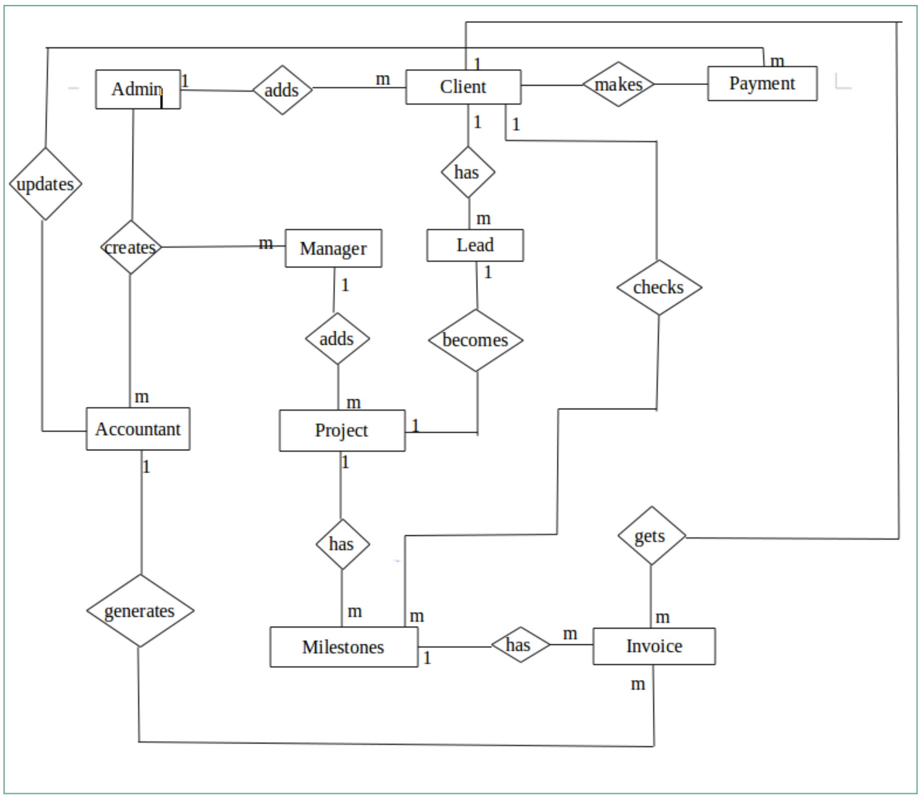
3. Accountant Sequence diagram



4. Client Sequence Diagram

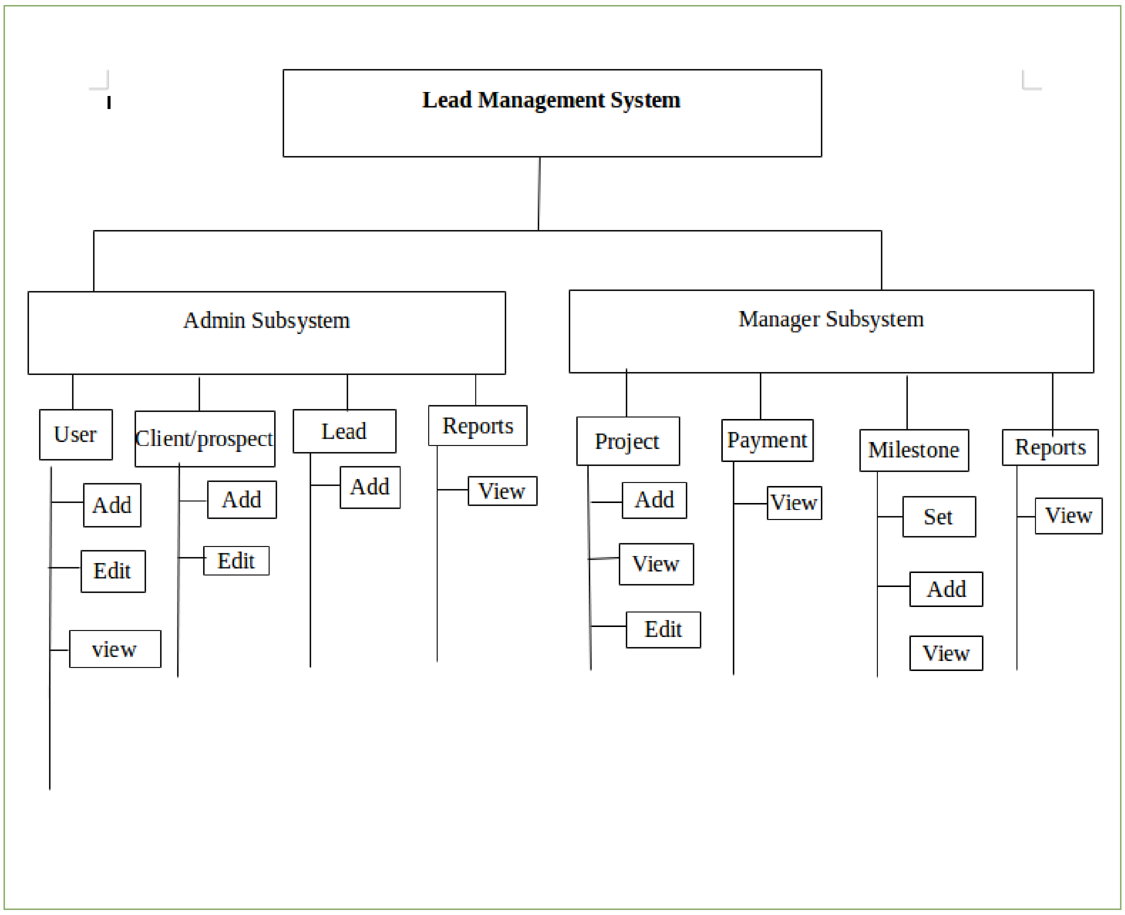


3.6 Entity relationship diagram

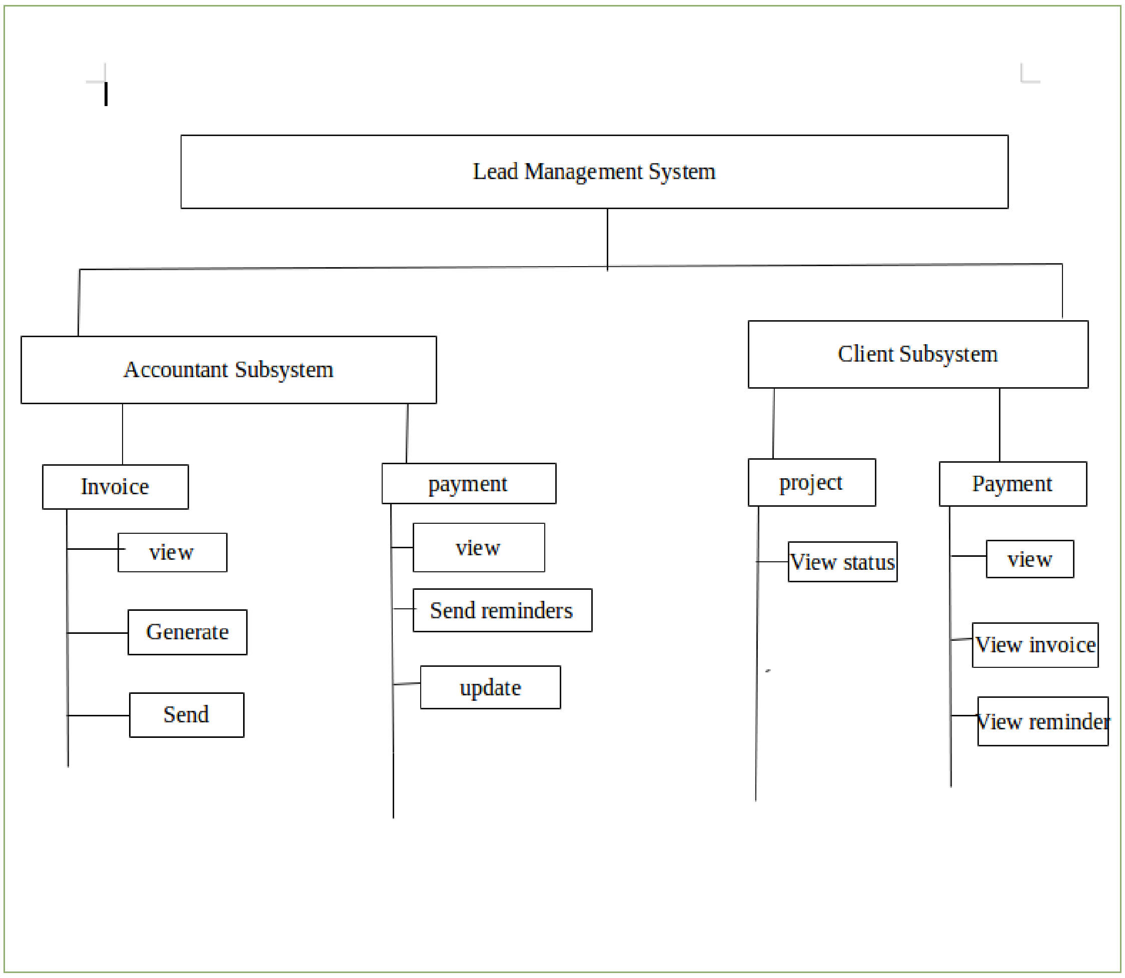


3.7 Module Heirarchy

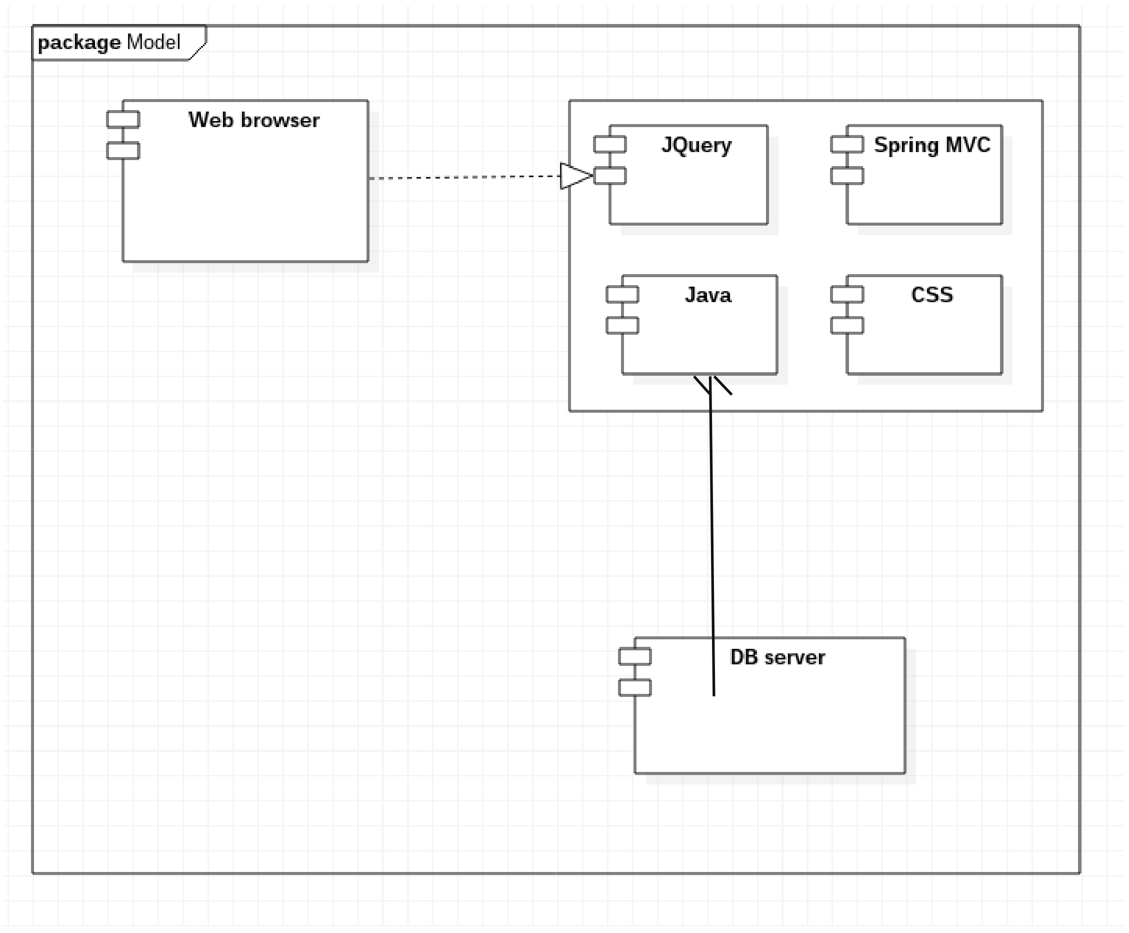
1. For Admin and Manager



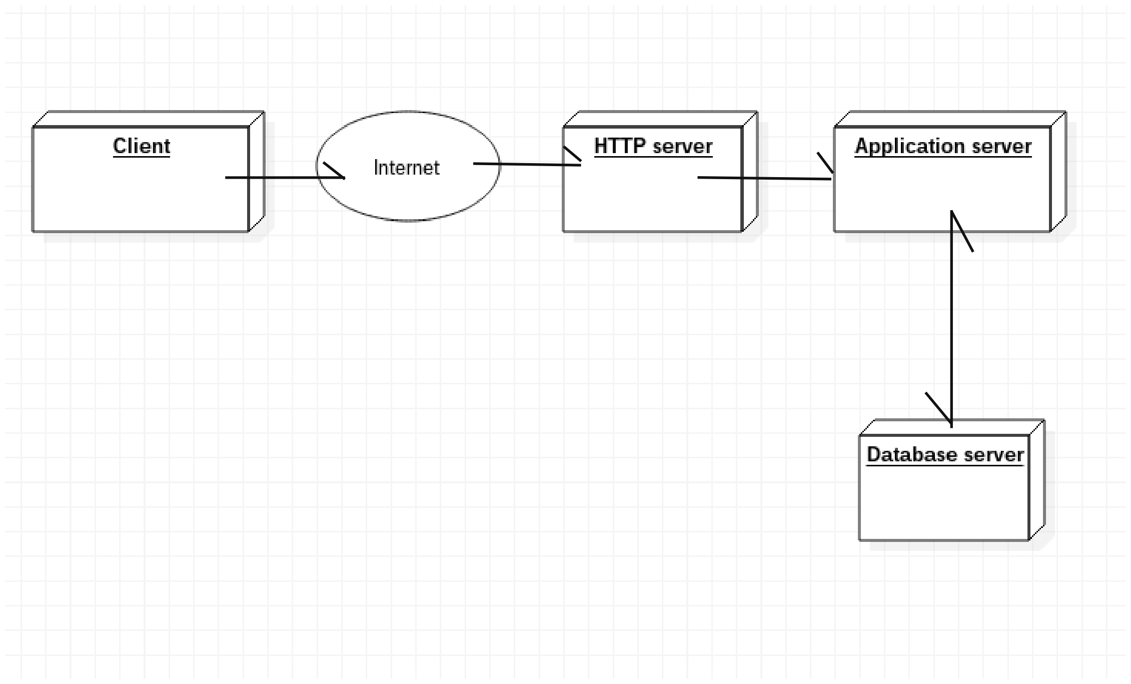
2. For Accountant and Client



3.8 Component Diagram



3.9 Deployment diagram



• 3.10 MODULE SPECIFICATION

- Client/Prospect -

-Contains operations related to clients and prospects.

-It consists of operations like adding,viewing,editing of clients and prospects.

- All the forms showing the operations come under this category.

- Lead

-Contains operations realted to leads in the system.

-It consists of operations such as add,view,edit lead.

-Whenever a lead comes in the system,it is entered into the system through a form and saved in the database.

- Project

-Contains operations related to projects in the system.

-It consists of operations such as add,view,edit projects.

-It consists of sub module-

- Milestones.

-Contains operations related to milestones of a project.

-It consists of operations such as set milestone,edit milestones and viewing of milestones.

- Payment

-Contains operations related to payment operation.

-It consists of operations such as vewing of payment,updating the status of the payment.

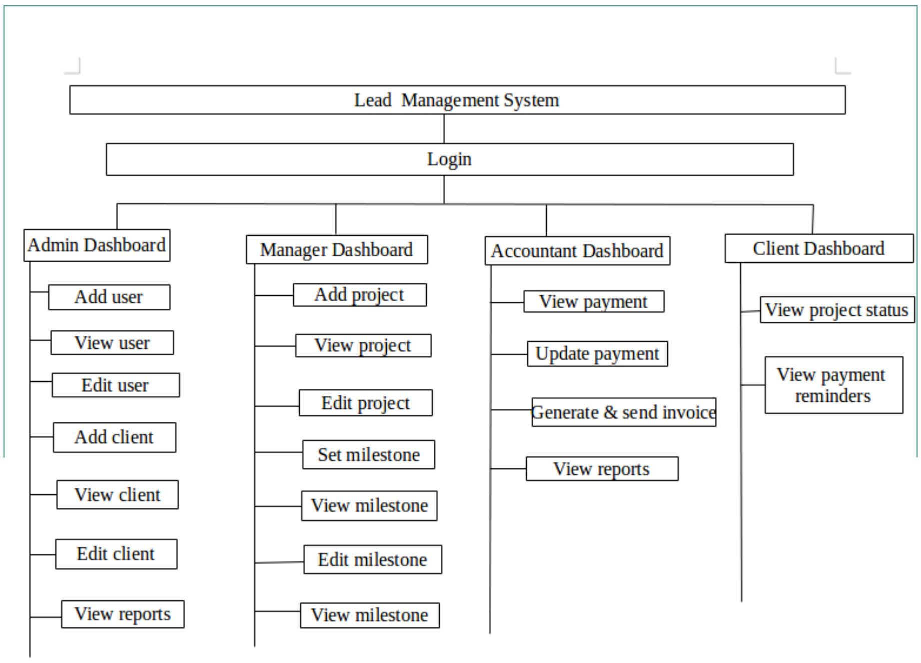
-It consists of sub module such as invoice.

- Invoice

-Contains operation related to invoice.

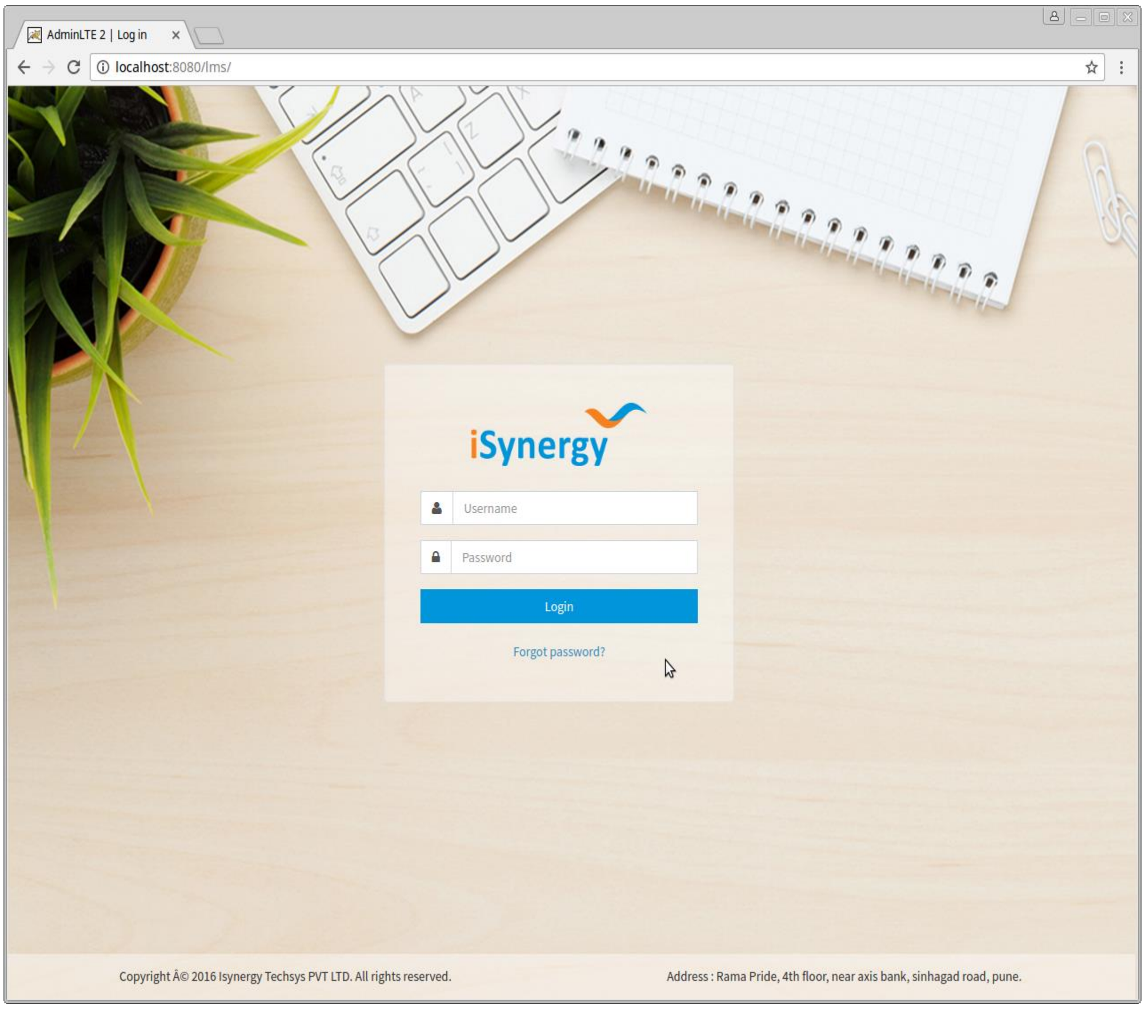
-It contains of operations such as generating and sending the invoice once the milestone status becomes completed.

- **3.11 Website Map Diagram**



• 3.12 USER INTERFACE DESIGN

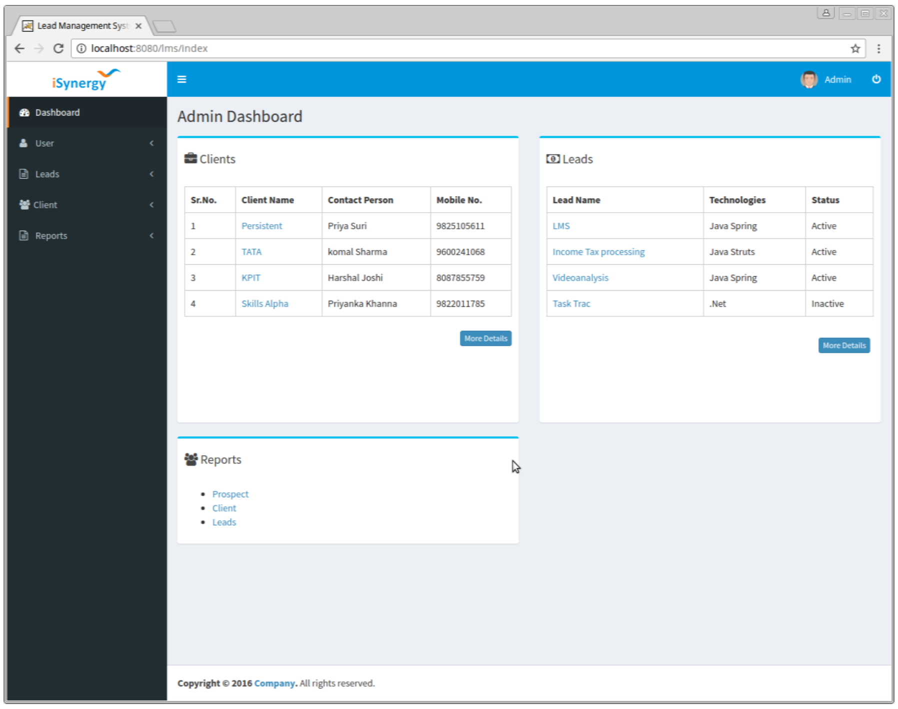
• 1-Login page



Copyright Â© 2016 Isynergy Techsys PVT LTD. All rights reserved.

Address : Rama Pride, 4th floor, near axis bank, sinhagad road, pune.

2- Admin Dashboard



Admin Dashboard

Clients

Sr.No.	Client Name	Contact Person	Mobile No.
1	Persistent	Priya Suri	9825105611
2	TATA	komal Sharma	9600241068
3	KPIT	Harshal Joshi	8087855759
4	Skills Alpha	Priyanka Khanna	9822011785

More Details

Leads

Lead Name	Technologies	Status
LMS	Java Spring	Active
Income Tax processing	Java Struts	Active
Videoanalysis	Java Spring	Active
Task Trac	.Net	Inactive

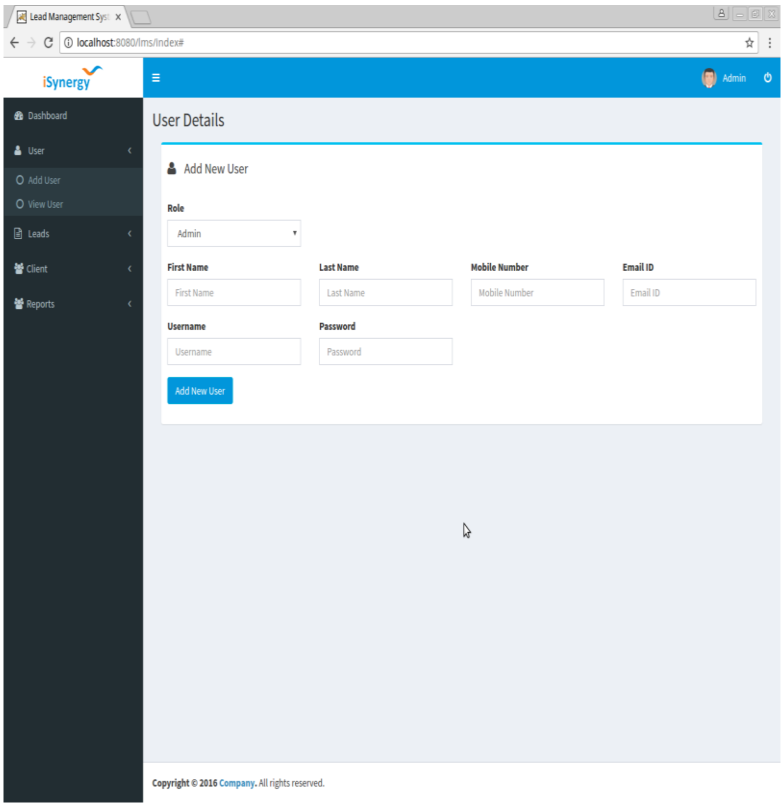
More Details

Reports

- Prospect
- Client
- Leads

Copyright © 2016 Company. All rights reserved.

- Add user



User Details

Add New User

Role

Admin

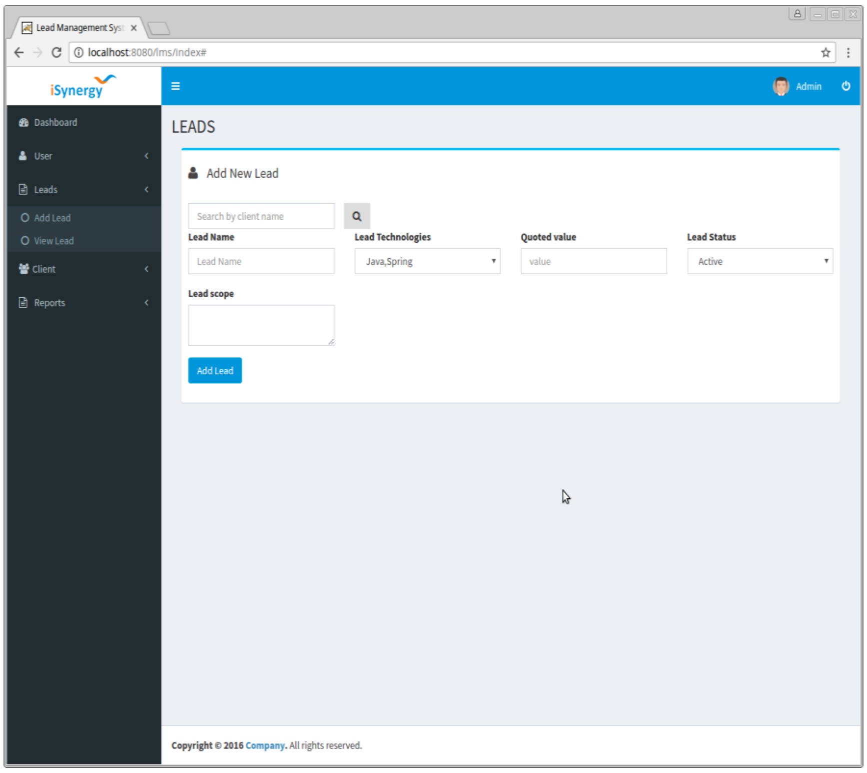
First Name	Last Name	Mobile Number	Email ID
First Name	Last Name	Mobile Number	Email ID

Username	Password
Username	Password

Add New User

Copyright © 2016 Company. All rights reserved.

Add Lead



LEADS

Add New Lead

Search by client name

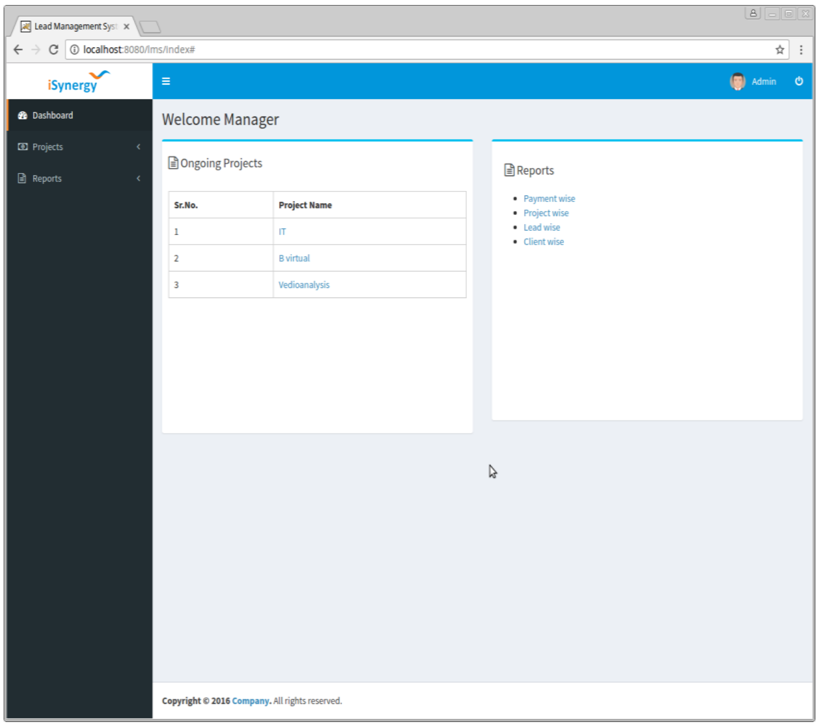
Lead Name	Lead Technologies	Quoted value	Lead Status
Lead Name	Java,Spring	value	Active

Lead scope

Add Lead

Copyright © 2016 Company. All rights reserved.

5 Manager Dashboard



Welcome Manager

Ongoing Projects

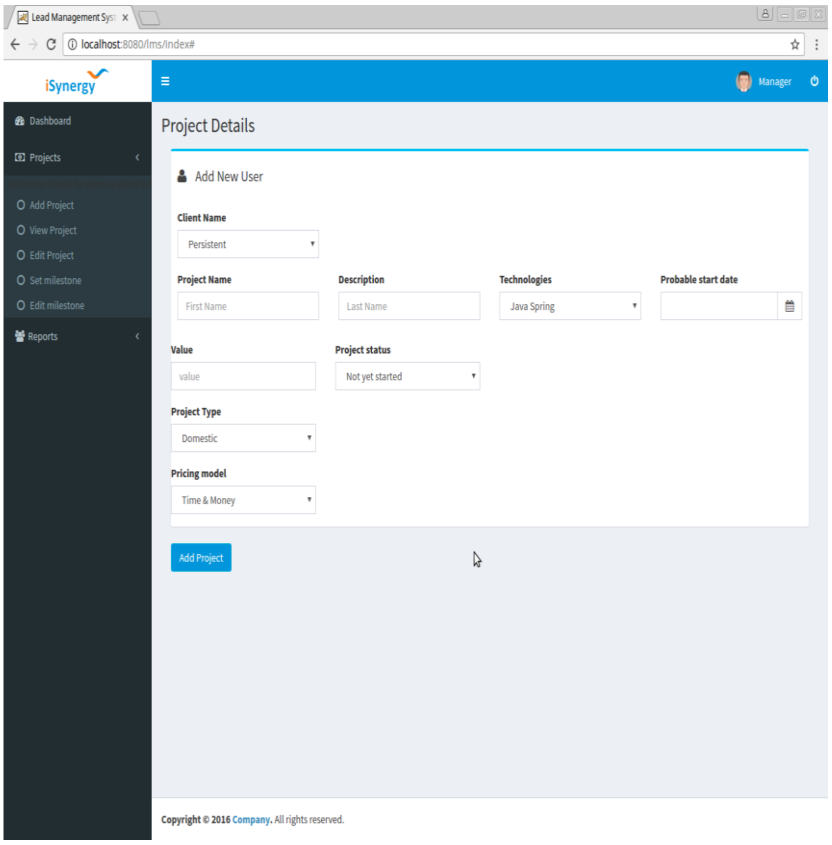
Sr.No.	Project Name
1	IT
2	B virtual
3	Vedioanalysis

Reports

- Payment wise
- Project wise
- Lead wise
- Client wise

Copyright © 2016 Company. All rights reserved.

Add projects



Project Details

Add New User

Client Name: Persistent

Project Name: First Name

Description: Last Name

Technologies: Java Spring

Probable start date:

Value: value

Project status: Not yet started

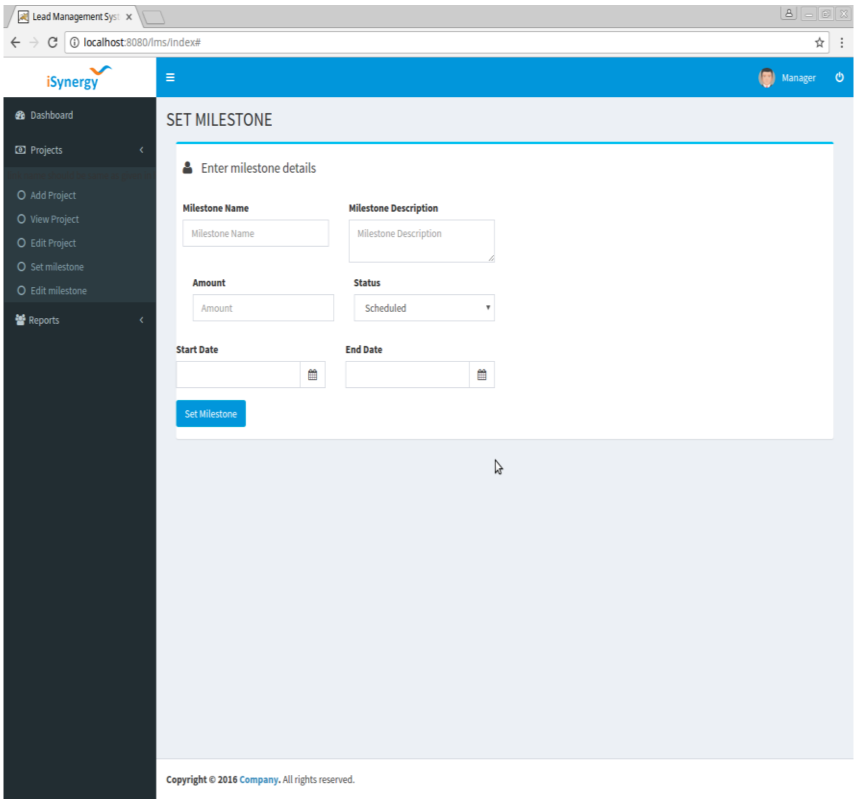
Project Type: Domestic

Pricing model: Time & Money

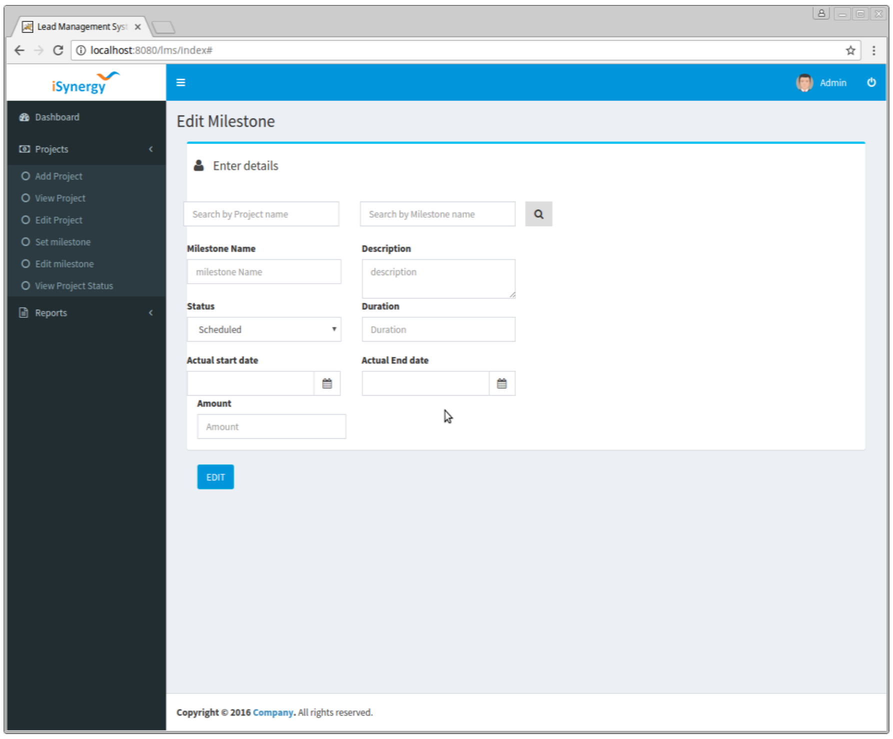
Add Project

Copyright © 2016 Company. All rights reserved.

7) set milestone



8.) Edit milestone



Edit Milestone

Enter details

Search by Project name | Search by Milestone name

Milestone Name: milestone Name

Description: description

Status: Scheduled

Duration: Duration

Actual start date

Actual End date

Amount: Amount

Copyright © 2016 Company. All rights reserved.

3.13 DATA DICTIONARY

SR NO	Field name	Data type	Description
1	actual_end_date	Long	Stores actual end date of milestone in milestone table
2	actual_start_date	Long	Stores actual start date of milestone in milestone table
3	contact_email	varchar(45)	Stores contact email in prospect table

4	Contact_name	varchar(30)	Stores contact name in prospect table
5	contact_phone	int(10)	Stores contact phone in prospect_client table
6	comment	varchar(100)	Stores comment in quotation table
7	created_date	Long	Stores created date of new user in user table
8	customer_addressline1	varchar(50)	Stores address of customer in

			prospect_client table
9	customer_addressline2	varchar(50)	Stores address of customer in prospect_client table
10	customer_name	varchar(30)	Stores customer name in prospect_client table
11	duration	varchar(10)	Stores duration of project in project table
12	email	varchar(45)	Stores email address of user in user

			table
13	end_date	long	Stores end date of milestone in milestone table
14	first_name	varchar(15)	Stores first name of user in user table
15	invoice_amount	double	Stores invoice amount in invoice table
16	Invoice_date	long	Stores invoice date in invoice table
17	Invoice_id	Int(6)	Stores unique invoice_id in invoic

			e table
18	Last_name	varchar(15)	Stores last name of user in user table
19	Last_update_date	Long	Stores last update date of user in user table
20	Lead_id	Int(6)	Stores unique lead id in lead table
21	Lead_name	varchar(30)	Stores lead name in lead table
22	Lead_scope	varchar(100)	Stores lead scope in lead table
23	Lead_status	Enum{inactive,declined,active}	Stores lead status

			in lead table
24	Lead_technologies	varchar(15)	Stores lead technologies in lead table
25	location	varchar(15)	Stores location of client in prospect_client table
26	Milestone_amount	double	Stores milestone amount in milestone table
27	Milestone_description	varchar(100)	Stores description of milestone in milestone table

28	Milestone_id	Int(6)	Stores unique milestone id in milestone table
29	Milestone_name	varchar(30)	Stores name of the milestone in milestone table
30	Mobile_no	Int(10)	Stores mobile number of user in user table
31	password	varchar(10)	Stores password for user in user table
32	Payment_currency	varchar(10)	Stores currency in payment

			table
33	Pricing_model	Enum{time & money/fixed cost}	Stores pricing model in project table
34	Probable_start_date	long	Stores probable start date of project in project table
35	Project_description	varchar(100)	Stores description of project in project table
36	Project_id	Int(6)	Stores unique project id of project in project table
3	Project_na	varchar(30)	Stores

7	me		project name in project table
38	Prospect_client_id	Int(6)	Stores unique prospect client id in prospect_client table
39	Quotation_id	Int(6)	Stores unique quotation id in quotation table
40	Quoted_value	double	Stores quoted value in the lead table
41	reference	varchar(15)	Stores reference name in prospe

			ct_client table
42	Role_id	Int(6)	Stores unique role id in role table
43	status	Enum{not yet started,ongoing,completed,delivered}	Stores status of the project in project table
44	tax	Double	Stores tax in invoice table
45	technologies	varchar(15)	Stores technologies used in project table
46	type	Enum{domestic/international}	Stores type of project in project table

47	User_id	Int(6)	Stores unique user id in user table
48	username	varchar(30)	Stores unique username in user table
49	value	double	Stores value of the project in project table

• 3.15 TABLE SPECIFICATION

1) Role

Field name	Data type	width	Constraints
Role_id	int	6	PK
Role_type	Enum{admin/manager/accountant/client}		Not null

2) User

Field name	Data type	width	Constraints
User_id	Int	6	Pk
Role_id	int	6	Fk
First_name	varchar	15	Not null
Last_name	varchar	15	Not null
username	varchar	30	Not null,unique
password	varchar	10	Not null
Mobile_no	int	10	Not null
email	varchar	45	Not null
Last_update_date	long		Not null
Created_date	long		Not null

3) Prospect_client

Field name	Data type	width	Constraints
Prospect_client_id	int	6	Pk
Customer_name	varchar	30	Not null
location	varchar	15	Not null
Customer_addressline1	varchar	50	Not null
Customer_addressline2	varchar	50	Can be null

Customer_phone	int	10	Not null
Contact_name	varchar	30	Not null
Contact_phone	int	10	Not null
Contact_email	Varchar	45	Not null
Project_description	varchar	100	Not null
reference	varchar	15	Not null
status	Enum{prospect/client}		Not null
Created_date	long		Not null

4) Project

Field name	Data type	width	Constraints
project_id	int	6	Pk
Prospect_client_id	int	6	Fk
Project_name	varchar	30	Not null
Project_description	varchar	100	Not null
technologies	varchar	15	not null
Probable_start_date	long		Not null
duration	varchar	10	Not null
value	double		Not null
status	Enum{not yet started/ongoing/completed/delivered}		Not null
type	Enum{domestic/international}		Not null
Pricing_model	Enum{fixed cost/time & money}	15	Not null
Created_date	Long		Not null
Last_update_date	long		Not null

5) Lead

Field name	Data type	width	Constraint
Lead_id	int	6	Pk
Prospect_client_id	int	6	Fk
Lead_scope	varchar	100	Not null
Lead_technologies	varchar	15	Not null
Lead_status	Enum{declined/inaction/active}		Not null
Quoted_value	double		Not null
Lead_name	varchar	30	Not null
Created_date	long		Not null
Last_update_date	long		Not null

6) Milestone

Field name	Data type	width	Constraints
Milestone_id	int	6	Pk
Project_id	int	6	Fk
Start_date	long		Not null
End_date	long		Not null
Milestone_name	varchar	30	Not null
Milestone_description	varchar	100	Not null
Milestone_amount	double		Not null
Milestone_status	Enum{completed/partial}		Not null
Actual_start_date	long		Not null
Actual_end_date	long		Not null

7) Invoice

Field name	Data type	width	Constraints
Invoice_id	int	6	Pk
Milestone_id	int	6	Fk
Invoice_date	long		Not null
Invoice_amount	Float		Not null
Tax	Float		Not null

8) Payment

Field name	Data type	width	Constraints
Payment_id	int	6	Pk
Project_id	int	6	Fk
Payment_amount	Float		Not null
Payment_date	long		Not null
Payment_currency	varchar	10	Not null

9) Invoice_payment

Field name	Data type	width	Constraints
Payment_id	int	6	fk
invoice_id	int	6	Fk

10) Quotation_history

Field name	Data type	width	Constraints
quotation_id	int	6	Pk
lead_id	int	6	Fk
Quoted_value	Float		Not null
comment	varchar	100	Not null
Create_date	long		Not null
Last_update_date	long		Not null

• 3.16 TEST PROCEDURES AND IMPLEMENTATION

Test case ID	1
Test case name	Validate login

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
To verify that username and password on login page should not be	Do not enter username and password on login page and submit form	An error message "Please enter username and password" should display	An error message "Please enter username and password" is displayed	Pass
	Do not enter	An error message	An error message	Pass

empty	username on login page and submit form	"Please enter username" should display	"Please enter username" is displayed	
	Do not enter password on login page and submit form	An error message "Please enter password" should display	An error message "Please enter password" is displayed	Pass
1. To verify that username and password on login page	Enter username and password where username and/or password not exists in	An error message "Incorrect username and/or password" should display	An error message "Incorrect username and/or password" is displayed	Pass

exists in databa se	database			
	Enter username and password which exists in database	Should redirect to Home page	Redirected to Home page	Pass

Test case ID	2
Test case name	Add User

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field displayed	Pass

2. UserName existence validation	Enter UserName which exists in database	An error message "UserName already exists" should display	An error message "Username already exists" is displayed	Pass
	Enter UserName which does not exists in database	Registration should be successful	Registration is successful	Pass

Test case ID	3

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field should display	Pass

Test case ID	4

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field displayed	Pass

Test case ID	5
Test case name	Add Project

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration	An error message for empty mandatory field should display	An error message for empty mandatory field displayed	Pass

	form			

Test case ID	6
Test case name	Edit project

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration	An error message for empty mandatory field should display	An error message for empty mandatory field displayed	Pass

	form			
2. Project Name existence validation	Enter Project name to search	An error message “Project does not exist” should display	An error message “Project does not exist” is displayed	Pass
	Enter Project Name	Details displayed successfully	Details displayed successfully	Pass

Test case ID	7

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field should display	Pass

2. Date validation	Enter invalid start date and end date	An error message “Invalid date” should display	An error message “Invalid date ” is displayed	Pass
	Enter valid dates	Date accepted	Date accepted	Pass

Test case ID	8
Test case name	Edit milestone-1

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field should display	Pass

2. Enter by Project Name	Enter Project name to search	An error message “Project does not exist” should display	An error message “Project does not exist” is displayed	Pass
	Enter Project Name	Details of project should be displayed successfully	Details displayed successfully	Pass

Test case ID	9
Test case name	Edit milestone-2

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and submit registration form	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field should display	Pass

2. Enter by milestone Name	Enter milestone name to search	An error message “milestone name does not exist” should display	An error message “milestone name does not exist” is displayed	Pass
	Enter milestone Name	Details of project should be displayed successfully	Details displayed successfully	Pass

Test case ID	10
Test case name	Update payment

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1. Enter by client Name	Enter client name to search	An error message "client does not exist" should display	An error message "client does not exist" is displayed	Pass
2. Enter by amount	Enter client name for search	"projects of the following client displayed"	"projects of the following client displayed" "Amount	Pass pass

	Enter Amount paid and update status	“Amount updated”	updated”	

Test case ID	11
Test case name	Show Reports-1

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1.Mandatory fields validation	Do not enter any data and “Show report”	An error message "Please enter details" should display	An error message "Please enter details" is displayed	Pass
	Leave mandatory fields as blank one at a time and submit registration form	An error message for empty mandatory field should display	An error message for empty mandatory field displaye	Pass

2. Enter Date	Enter invalid date	An error message “Invalid date” is displayed	An error message “Invalid date” is displayed	Pass
3. Enter Category	Select Category	“Report generated”	Report generated”	Pass

Test case ID	12
Test case name	Generate invoice

Test case Description	**Test steps**			**Status**
	Input	**Expected output**	**Actual output**	
1. Mandatory feilds	Do not enter any data and click on "generate invoice"	Error message "Select project name" will be displayed	Error message as "Select project name is displayed"	Pass
	Select project name and click on generate button	Message displayed as "invoice generated for the milestone"	Message displayed as "invoice generated for the milestone"	pass

CHAPTER 4

USER MANUAL

• 4.1 USER MANUAL

Admin -

1) Add user

Steps are as follows -

1. Adding a new user to the system is done by admin.
2. Select the user widget on the dashboard after loggin in.
3. Select the add user option on the loaded page.
4. Fill out the form and click “ADD USER” button.
5. A message will appear on the window “User added succesfully”.

2) Edit user

Steps are as follows -

1. Select the user widget on the dashboard after loggin in.
2. Select the edit user option.
3. Form with the list of users will appear on the page.

4. Click on the user you want to edit.A pop up with all the editable feilds will appear.
5. Edit the values and click on “Save” button.
6. New Values will be saved with a message displayed as “Edited successfully”.

3)View User

Steps are as follows-

1. Select the user widget on the dashboard after loggin in.
2. Select the view user option.
3. A list of system users will be displayed.
4. Click on any value to view detailed information.

4) Delete User

Steps are as follows -

1. Select the user widget on the dashboard after loggin in.
2. Select delete option.
3. A list of system user will be displayed with a delete icon next to each value.
4. Select the delete icon and the respective value will be deleted.
5. A message will be displayed on the page “Deleted succesffully”

Client/Prospect

1)Add client/prospect

1. Select the client/prospect widget on the dashboard after loggin in.
2. Select add new client/prospect option.
3. Fill out the form and click on “ADD “ button.
4. A message will be displayed “Added successfully!”

2) Edit client/prospect

1. Select the client/prospect widget on the dashboard after loggin in.
2. Select Edit option .
3. A list of existing clients/prospect will be displayed .
4. Click on the value in the list.
5. A pop up will appear with all editable values.
6. Edit the values and click on “Save” button.
7. New values will be saved with a message displayed as “Edited successfully”.

3) View client/prospect

1. Select the client/prospect widget on the dashboard after loggin in.
2. Select the view client/prospect option.
3. List of existing client/prospects will be displayed.
4. Click on any value to view detailed information.

<u>Lead</u>

1) Add Lead

1. Click on the lead widget on the dashboard after loggin in.
2. Select add lead option.
3. Fill out the form.
4. Click “Add Lead” option.
5. A message will be displayed “Added successfully”.

2) View Lead

1. Click on the lead widget on the dashboard after loggin in.
2. Select view lead option.
3. List of leads will be displayed.
4. Click on any value to get detailed information about the lead

Reports

1)View Report

1. Select the report widget on the dashboard after loggin in.
2. Select the type of report you want.
3. Report will be displayed on the screen.

Manager

Project

1)Add Project

1. Click on the project widget on the dashboard after loggin in.
2. Select add project option.
3. Fill out the form and click "add button".
4. Message will appear as "Project added successfully".

2) View Project

1. Click on the project widget on the dashboard after loggin in.
2. Select view project option.
3. Project list will be displayed.
4. Select any value.
5. A pop up window will be displayed showing detailed information about the projects.
6. Select any value to get detailed information .

3) Edit Project

1. Click on the project widget on the dashboard after loggin in.
2. Select edit project option.
3. Project list will be displayed.
4. Select any value.
5. Pop up message will appear with editatble values.
6. Edit and click the save button.
7. Message will appear “Edited successfully”.

Client

1) View client

1. Click on the client widget on the dashboard after loggin in.
2. Select view client option.
3. List of client will be displayed.

Payment

1) View payment

1. Click on the payment widget on the dashboard.
2. Select view payment
3. List of projectwise payment will be displayed.

Milestones

1) Set milestone

1. Click on the milestone widget after loggin in.
2. Select set milestone option.
3. Fill in the form and click on the button “set milestone”
4. Message will be displayed “milestone set”

2) View milestone

1. Click on the milestone widget after loggin in
2. Select view milestone option.
3. List of project wise milestone will be displayed.

Reports

1) View Reports

1. Select Report widget on the dashboard after loggin in.
2. Select view reports
3. Select the criteria for reports.
4. Click on view reports.

Accountant

Payment

1)Add Payment

1. Select Payment widget on the dashboard after loggin in.
2. Select add payment option.
3. Fill in the form and click “Save” button.
4. Message will be displayed “Payment added succesfully.

2) View Payment

1. Select Payment widget on the dashboard after loggin in.
2. Select view payment option.
3. List of projectwise payments will be displayed.
4. Click on any value to see detailed information.

3) Edit Payment

1. Select Payment widget on the dashboard after loggin in.
2. Select edit payment option
3. List of payment will be displayed.
4. Click on any value.
5. Pop up window will be displayed with editable values.
6. Edit and click on save

4) Update payment status

1. Select Payment widget on the dashboard after loggin in.
2. Select update payment status.
3. Enter the status,details and click on update.
4. Message will be displayed “Updated successfully”.

5) Send payment reminders

1. Select Payment widget on the dashboard after loggin in.
2. Select send payment reminder option.

3. Check status of project.
4. Select client.
5. Fill in the payment details.
6. Click on send reminder button.
7. Message will be displayed “Reminder sent”.

Invoice

1) Generate invoice

1. Select invoice widget on the dashboard after loggin in.
2. Select generate invoice option.
3. Select client.
4. Click on generate invoice option.
5. Invoice will be generated in pdf format.

2) View invoice

1. Select invoice widget on the dashboard after loggin in.
2. Select view invoice option.
3. Select category.
4. Invoice for the selected category will be displayed.

3) Send invoice

1. Select invoice widget on the dashboard after loggin in.
2. Select send invoice option.
3. Select the client name.
4. Click on send invoice option.

Client

<u>Project</u>

1) View project

1. Select project widget on the dashboard after loggin in.
2. Select option view project.
3. List of project given by the client will be displayed.
4. Click on the any value ,pop up window will be displayed.
5. Details of the project will be displayed.

2) View payment reminder

1. Select payment widget on the dashboard after loggin in.
2. Select view payment reminder
3. List of payment reminders(if any) will be displayed for the corresponding project.
4.

- **4.2 MENU EXPLANATION**

I) Admin

Admin Dashboard-

1)User

1. Add User

Admin can “ADD USER” to the system by clicking on the ADD USER button. A form appears having various fields for data. Filling the form and submitting it results in successful addition of user to the system.

2. View User

Admin can “View User” of the system by selecting this option.

A list of the system users will be displayed on the screen.

3. Edit User

Admin can select on the “view user” option.A link appears on the names of the users.Admin can select any user to edit the user.Editable form appears on the screen.

2) Lead

1. Add Lead

Admin can “Add Lead” to the system by clicking on this option.

A form appears on the screens having various feilds.Filling the form and submitting it results in successful addition of lead to the system.

2. View Lead

Admin can “View Leads” in the system by selecting this option.

A list appears on the screens showing the leads in the system.

3. Edit Lead

Admin can “Edit Lead” in the system by selecting this option.

List of leads will be displayed.Select the lead to be edited.A form with editable feilds will appear on the screen.Fill it and click the edit button.This will result in successfull editing of lead.

II) Manager

Manager dashboard

1)Add project

Manager can “ADD Project” to the system by clicking on the ADD Project button. A form appears having various fields for data. Filling the form and submitting it results in successful addition of project to the system.

2) View Projects

Manager can “View Project” added to the system by clicking this option.

A List of existing projects will be displayed on the screen.

3) Edit Projects

Manager can “Edit project” by clicking this option.Enter the project name to be edited. Fill out the values that need to be chanaged.Click the edit button to save the changes.

4) Set milestones

Manager can set milestones for a project by selecting this option.

Manager can fill out a form and then click the “set milestone” button.

5) Edit milestone

Manager can “edit milestone” by selecting this option.Enter the project name and milestone name and the details will appear in the editable format.Enter the details to be changed and click the “edit” button.

III) Accountant

1) View Payment

Accountant can view payment details by selecting this option.

List of projects and payment details will be displayed on the scren.

2) Update payment status

Accountant can update the payment status by selecting this option.Payment is done manually and the status is updated in the system.

3) Generate and send invoice

Accountant can generate and send invoice by selecting this option.

A list of project whose milestone status is “completed” will be selected.

Select it and click the generate and send the invoice.

IV) Client

1) View project status

Client can view the status of the project by selecting this option.

Select the current project from the list and project details will be displayed.

2) View Payment reminder

Client can view payment reminders of the project by selecting this option.

Project details(payment related) will be displayed on the screen.

• 4.3 PROGRAM SPECIFICATION

Following are the program specification used in the development process explained with the help of flow chart.

User Authentication (Login)

Module Name	Authentication and Authorization	
Program Name	Login to the system	
Purpose	Check authentication of user	
Event	Click on “Login” Button.	
Input	**Constraints**	**Description**
Login Detail	The required field should not be null	Login details gets checked against database
Output	Login details get checked against database to check the authentication of the user and user get direction to their respective page.	

Add User

Module Name	Add user.	
Program Name	Add user details.	
Purpose	Add user details.	
Event	Click on “Save” Button.	
Input	**Constraints**	**Description**
User Detail	The required field should not be null	User Details are stored
Output	User details are stored in the User table and Admin gets acknowledgement message of record added.	

Add Project

Module Name	Add project	
Program Name	Add project details.	
Purpose	Add project details.	
Event	Click on “Add Project” Button.	
Input	**Constraints**	**Description**
User Detail	The required field should not be null	ProjectDetails are stored
Output	Project details are stored in the project table and Manager gets acknowledgement message of record added.	

Add Lead

Module Name	Add lead	
Program Name	Add lead details.	
Purpose	Add lead details.	
Event	Click on “Add Lead” Button.	
Input	**Constraints**	**Description**
User Detail	The required field should not be null	Lead details are stored
Output	Leads details are stored in the Lead table and Admin gets acknowledgement message of record added.	

DRAWBACKS AND LIMITATIONS

-Currently the system handles the payment manually.

Payment gateway is not included in the system.Payment is accepted manually and only the status is maintained in the system.

-Client after loggin in has only 2 options available-check project status and check payment reminders.Entire payment summary is not available to the client currently.

PROPOSED ENHANCEMENT

-Including payment gateway for online payment.

-Including “service provider” gateway to send payment reminders via messages on cell phone.

-Including the facility to show the payment summary to the client.

CONCLUSION

- System automated Lead management system.
- System proves useful for company as it manages all the tasks related to management of leads till invoice generation.
- Information provided is reliable and analytical.
- System provides management of all leads from any place via internet.
- System provides statistics about all leads and projects using attractive tables.
- System reduces cost and efforts of maintaining records.

BIBLIOGRAPHY

Bibliography:

Mastering Spring MVC 4

By: Geoffroy Warin

Websites:

http://docs.spring.io/

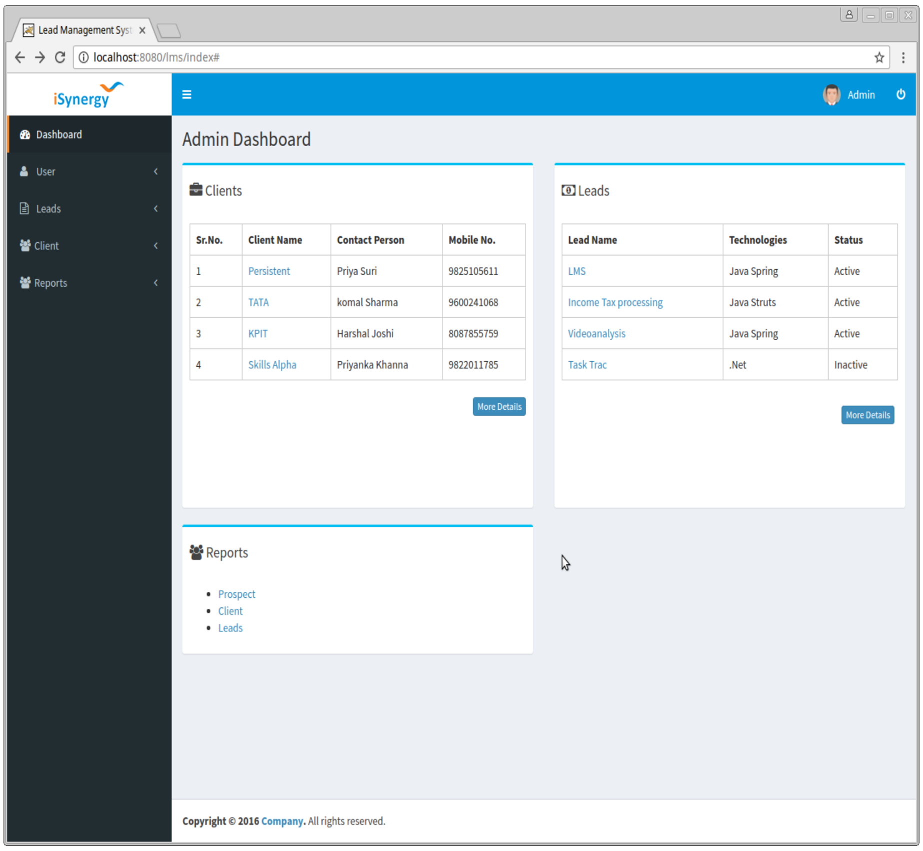
1. **http://projects.spring.io/spring-framework/**
2. **http://www.w3schools.com/**
3. **http://www.restapitutorial.com/lessons/**
4. **http://www.javatpoint.com/spring-JdbcTemplate-tutorial**

ANNEXURE 1:

USER INTERFACE

Admin Screens

1. Admin dashboard



Admin Dashboard

Clients

Sr.No.	Client Name	Contact Person	Mobile No.
1	Persistent	Priya Suri	9825105611
2	TATA	komal Sharma	9600241068
3	KPIT	Harshal Joshi	8087855759
4	Skills Alpha	Priyanka Khanna	9822011785

More Details

Leads

Lead Name	Technologies	Status
LMS	Java Spring	Active
Income Tax processing	Java Struts	Active
Videoanalysis	Java Spring	Active
Task Trac	.Net	Inactive

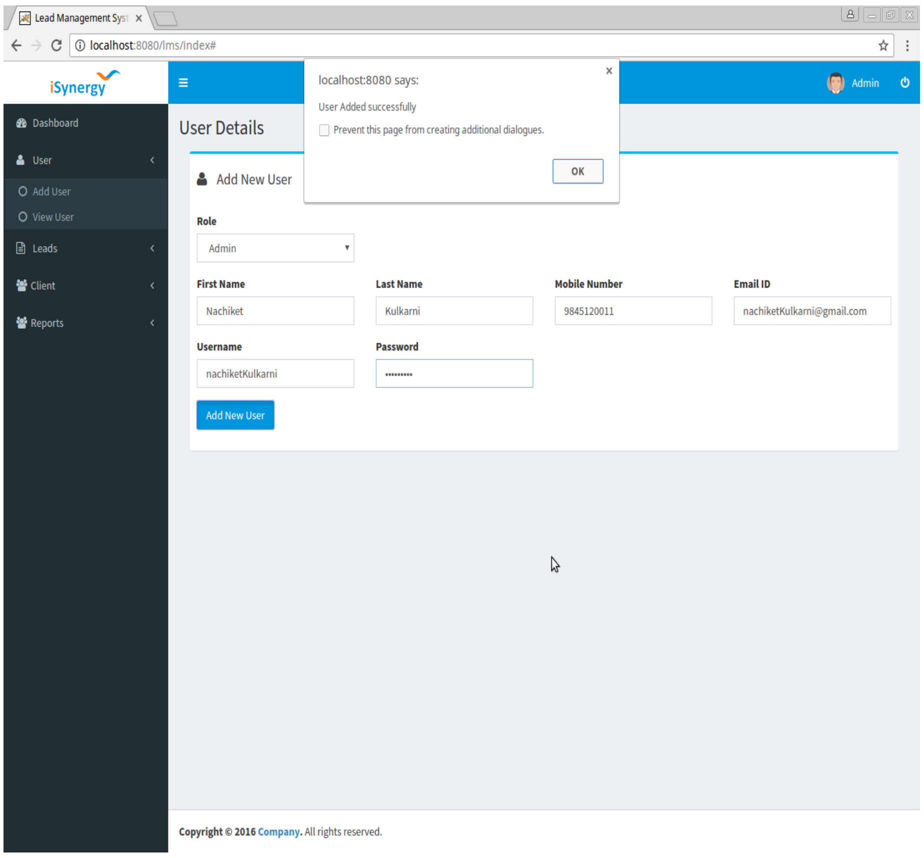
More Details

Reports

- Prospect
- Client
- Leads

Copyright © 2016 Company. All rights reserved.

2 Add user


Lead Management Sys

localhost:8080/lms/index#

localhost:8080 says:

User Added successfully

Prevent this page from creating additional dialogues.

OK

User Details

Add New User

Role: Admin

First Name: Nachiket

Last Name: Kulkarni

Mobile Number: 9845120011

Email ID: nachiketKulkarni@gmail.com

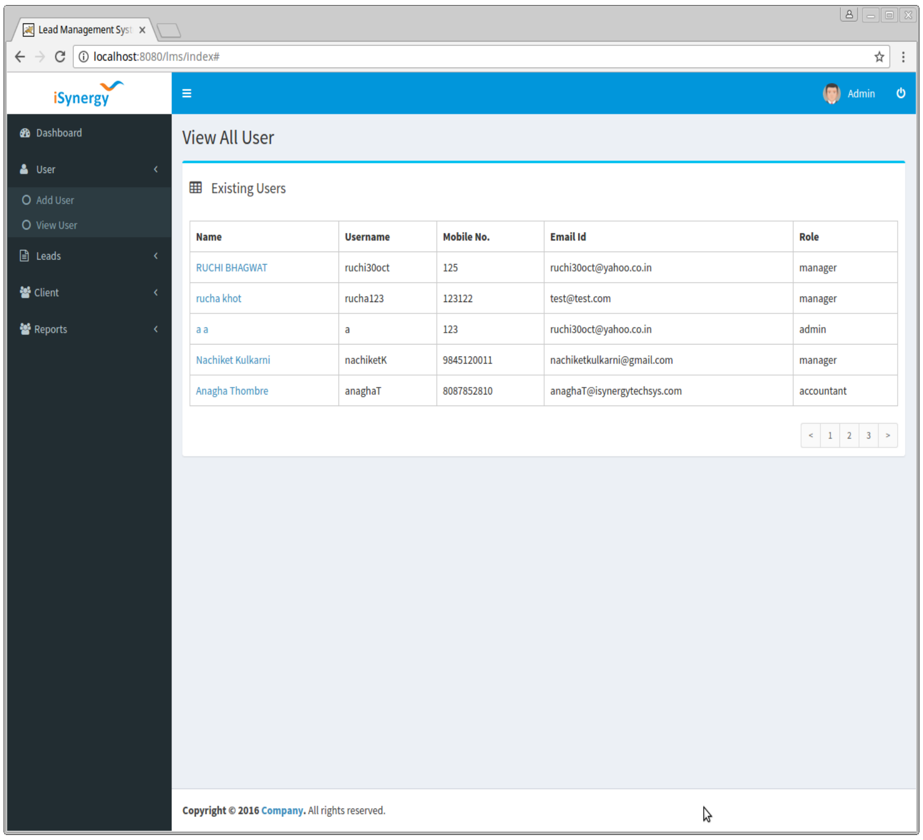
Username: nachiketKulkarni

Password: •••••••••

Add New User

Copyright © 2016 Company. All rights reserved.

View user



View All User

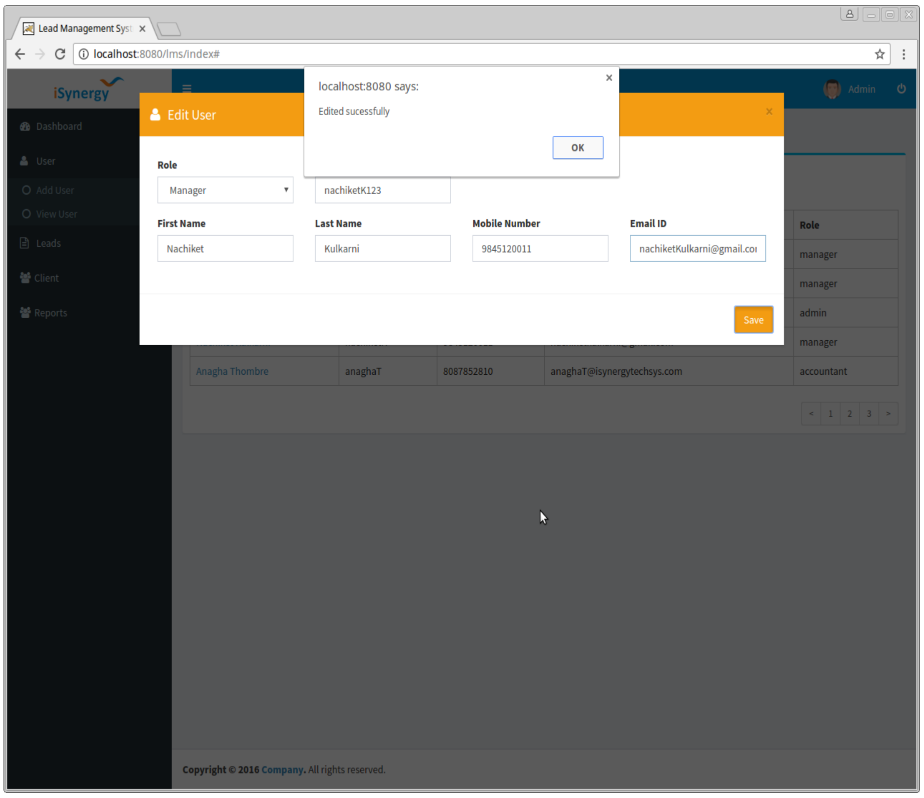
Existing Users

Name	Username	Mobile No.	Email Id	Role
RUCHI BHAGWAT	ruchi30oct	125	ruchi30oct@yahoo.co.in	manager
rucha khot	rucha123	123122	test@test.com	manager
a a	a	123	ruchi30oct@yahoo.co.in	admin
Nachiket Kulkarni	nachiketK	9845120011	nachiketkulkarni@gmail.com	manager
Anagha Thombre	anaghaT	8087852810	anaghaT@isynergytechsys.com	accountant

< 1 2 3 >

Copyright © 2016 Company. All rights reserved.

Edit user


localhost:8080 says:

Edited sucessfully

OK

Edit User

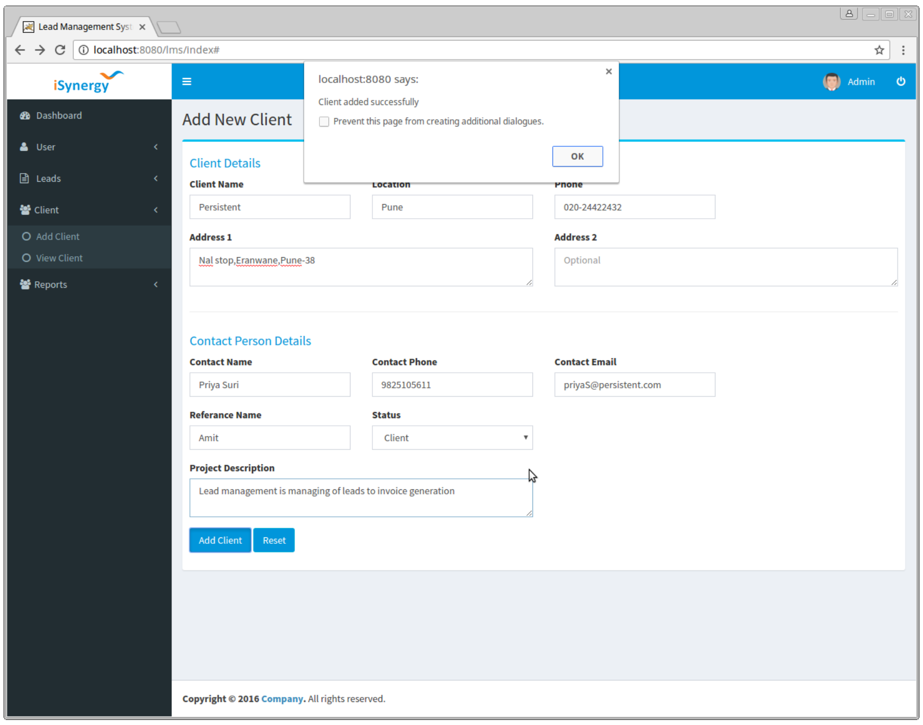
Role			
Manager	nachiketK123		
First Name	Last Name	Mobile Number	Email ID
Nachiket	Kulkarni	9845120011	nachiketKulkarni@gmail.coı

Save

				Role
				manager
				manager
				admin
				manager
Anagha Thombre	anaghaT	8087852810	anaghaT@isynergytechsys.com	accountant

Copyright © 2016 Company. All rights reserved.

5 . Add client



localhost:8080 says:

Client added successfully

☐ Prevent this page from creating additional dialogues.

OK

Add New Client

Client Details

Client Name	Location	Phone
Persistent	Pune	020-24422432
Address 1		Address 2
Nal stop,Eranwane,Pune-38		Optional

Contact Person Details

Contact Name	Contact Phone	Contact Email
Priya Suri	9825105611	priyaS@persistent.com
Referance Name	Status	
Amit	Client	

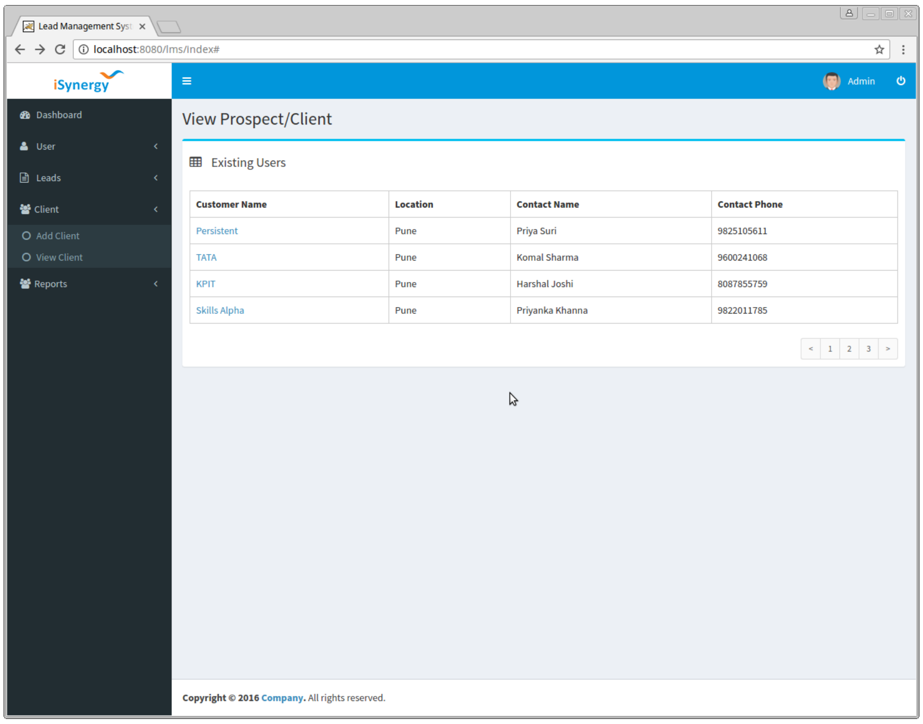
Project Description

Lead management is managing of leads to invoice generation

Add Client Reset

Copyright © 2016 Company. All rights reserved.

5. View client



View Prospect/Client

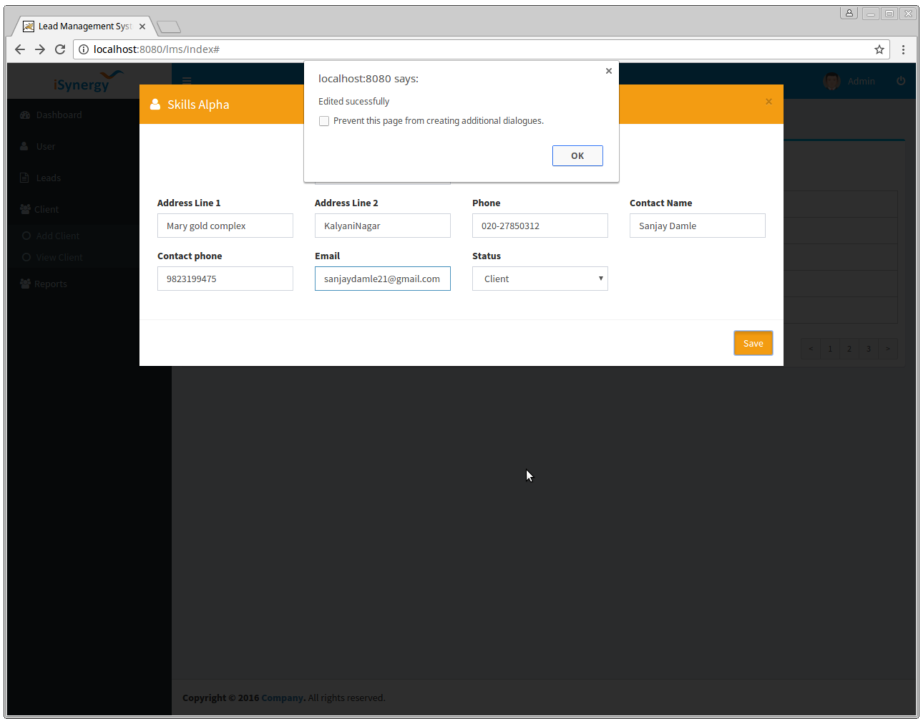
Existing Users

Customer Name	Location	Contact Name	Contact Phone
Persistent	Pune	Priya Suri	9825105611
TATA	Pune	Komal Sharma	9600241068
KPIT	Pune	Harshal Joshi	8087855759
Skills Alpha	Pune	Priyanka Khanna	9822011785

< 1 2 3 >

Copyright © 2016 Company. All rights reserved.

6. Edit client



localhost:8080 says:

Edited sucessfully

Prevent this page from creating additional dialogues.

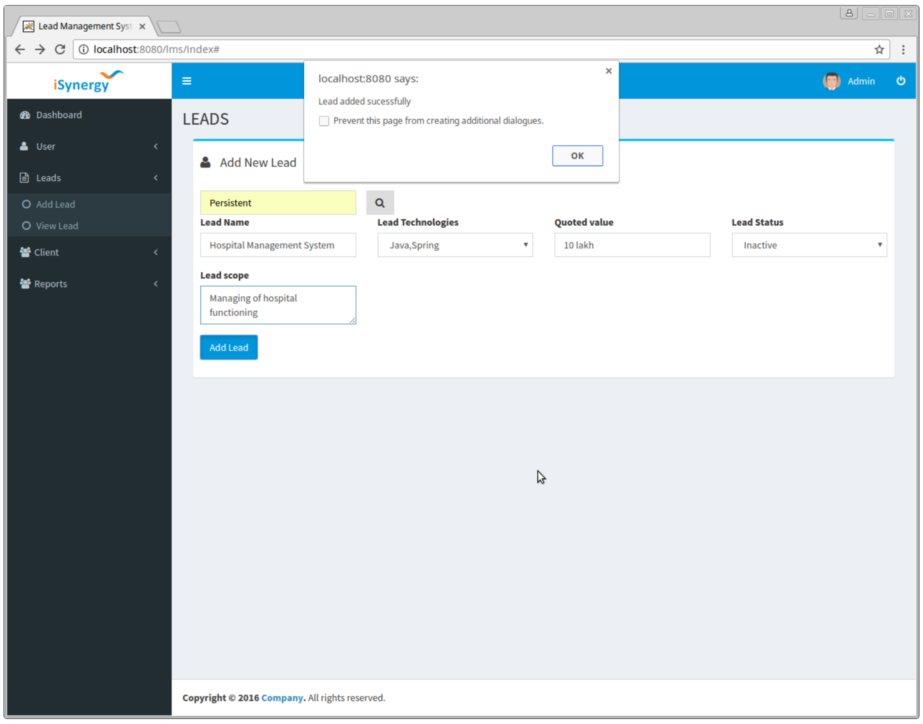
OK

Skills Alpha

Address Line 1	Address Line 2	Phone	Contact Name
Mary gold complex	KalyaniNagar	020-27850312	Sanjay Damle
Contact phone	**Email**	**Status**	
9823199475	sanjaydamle21@gmail.com	Client	

Save

7. Add lead



localhost:8080 says:

Lead added sucessfully

Prevent this page from creating additional dialogues.

LEADS

Add New Lead

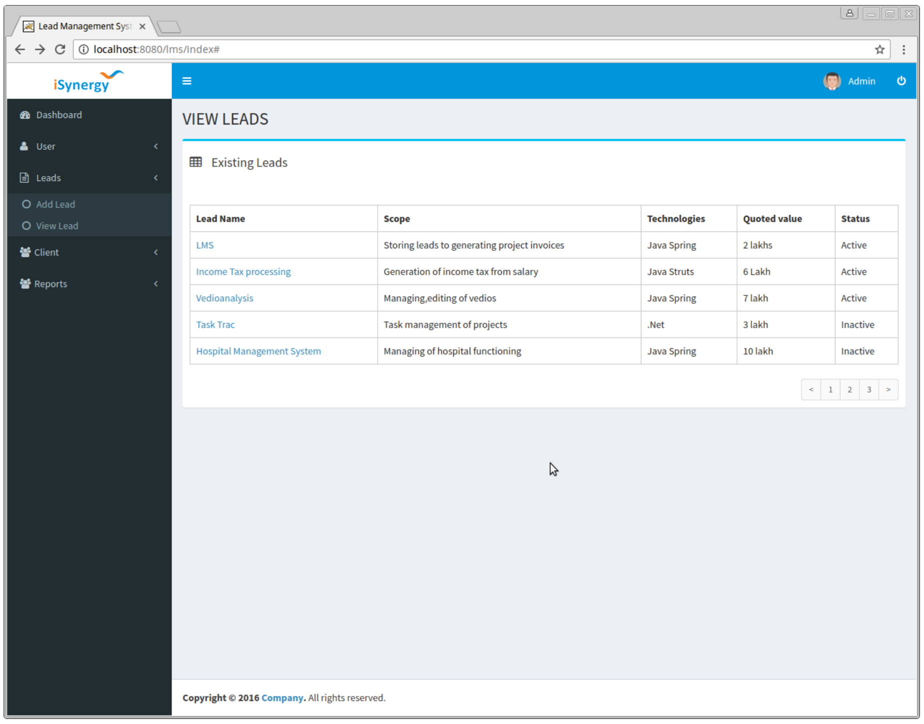
Persistent

Lead Name	Lead Technologies	Quoted value	Lead Status
Hospital Management System	Java,Spring	10 lakh	Inactive

Lead scope: Managing of hospital functioning

Copyright © 2016 Company. All rights reserved.

8. View lead



VIEW LEADS

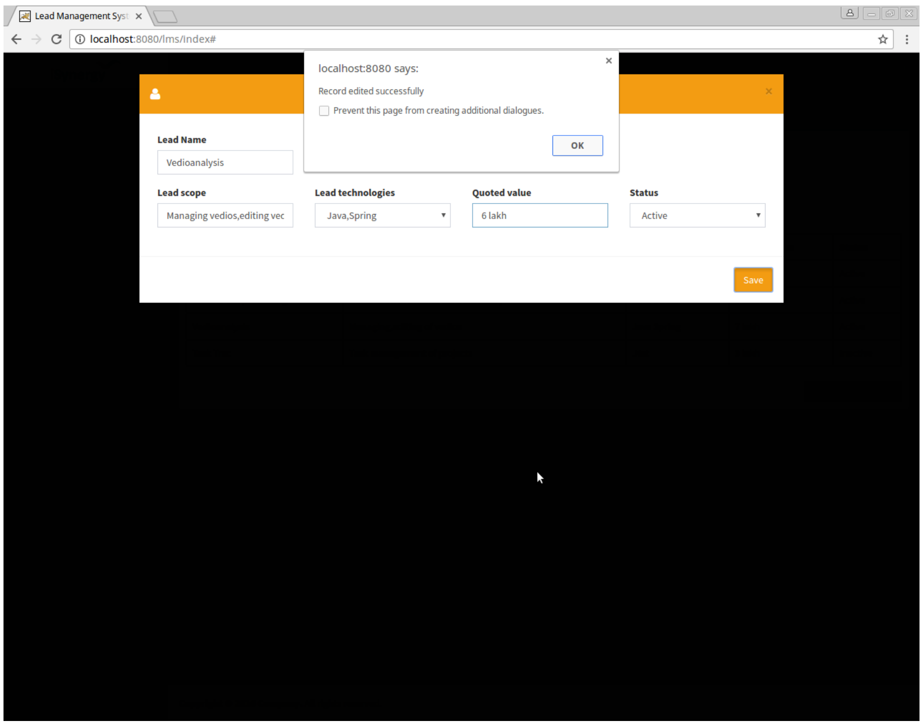
Existing Leads

Lead Name	Scope	Technologies	Quoted value	Status
LMS	Storing leads to generating project invoices	Java Spring	2 lakhs	Active
Income Tax processing	Generation of income tax from salary	Java Struts	6 Lakh	Active
Vedioanalysis	Managing,editing of vedios	Java Spring	7 lakh	Active
Task Trac	Task management of projects	.Net	3 lakh	Inactive
Hospital Management System	Managing of hospital functioning	Java Spring	10 lakh	Inactive

< 1 2 3 >

Copyright © 2016 Company. All rights reserved.

9. Edit lead



localhost:8080 says:

Record edited successfully

Prevent this page from creating additional dialogues.

Lead Name: Vedioanalysis

Lead scope: Managing vedios,editing vec

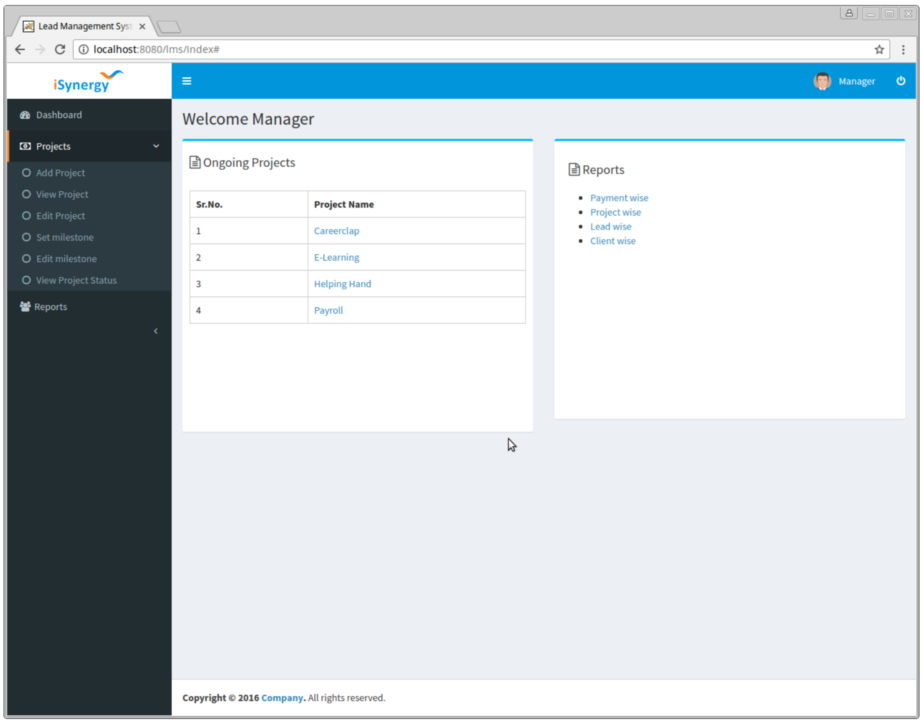
Lead technologies: Java,Spring

Quoted value: 6 lakh

Status: Active

Manager screens

1. Manager Dashboard



Welcome Manager

Ongoing Projects

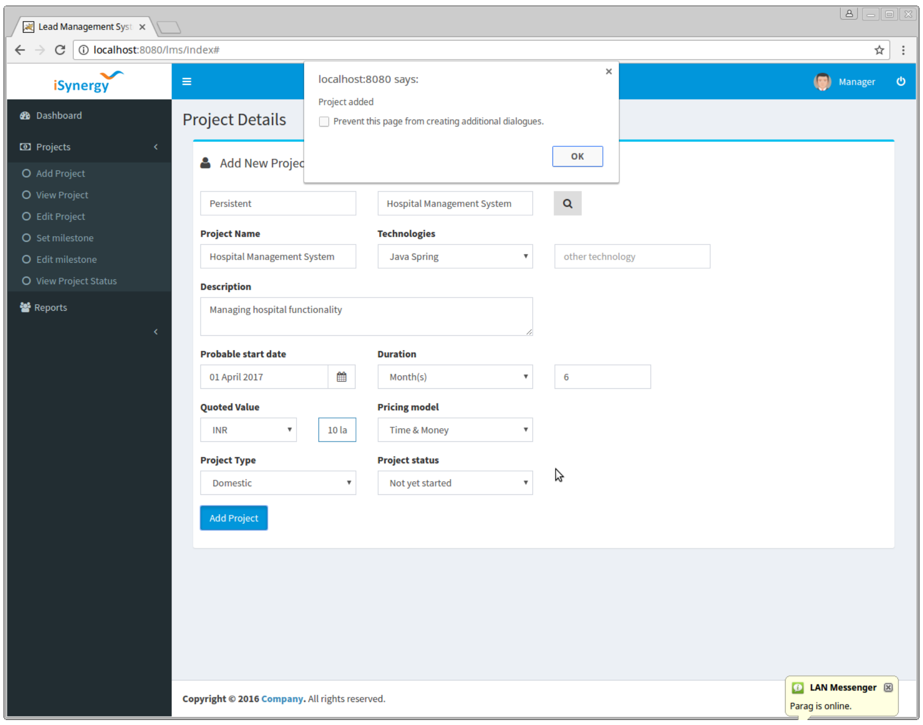
Sr.No.	Project Name
1	Careerclap
2	E-Learning
3	Helping Hand
4	Payroll

Reports

- Payment wise
- Project wise
- Lead wise
- Client wise

Copyright © 2016 Company. All rights reserved.

2. Add Project



localhost:8080 says:

Project added

☐ Prevent this page from creating additional dialogues.

OK

Project Details

Add New Projec

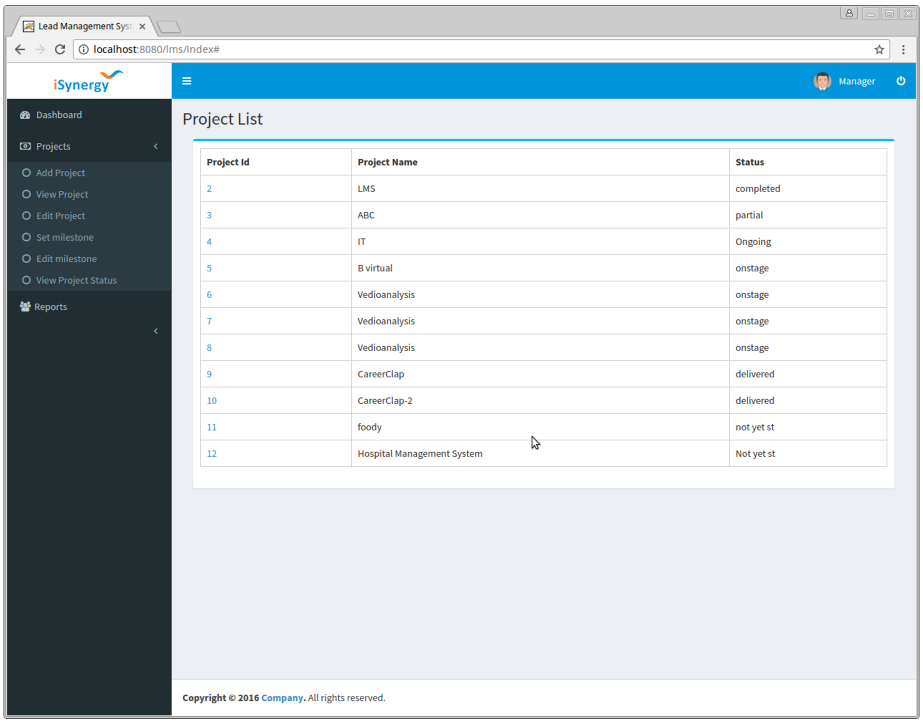
Persistent	Hospital Management System	
Project Name	**Technologies**	
Hospital Management System	Java Spring	other technology
Description		
Managing hospital functionality		
Probable start date	**Duration**	
01 April 2017	Month(s)	6
Quoted Value	**Pricing model**	
INR 10 la	Time & Money	
Project Type	**Project status**	
Domestic	Not yet started	

Add Project

Copyright © 2016 Company. All rights reserved.

Parag is online.

View Project

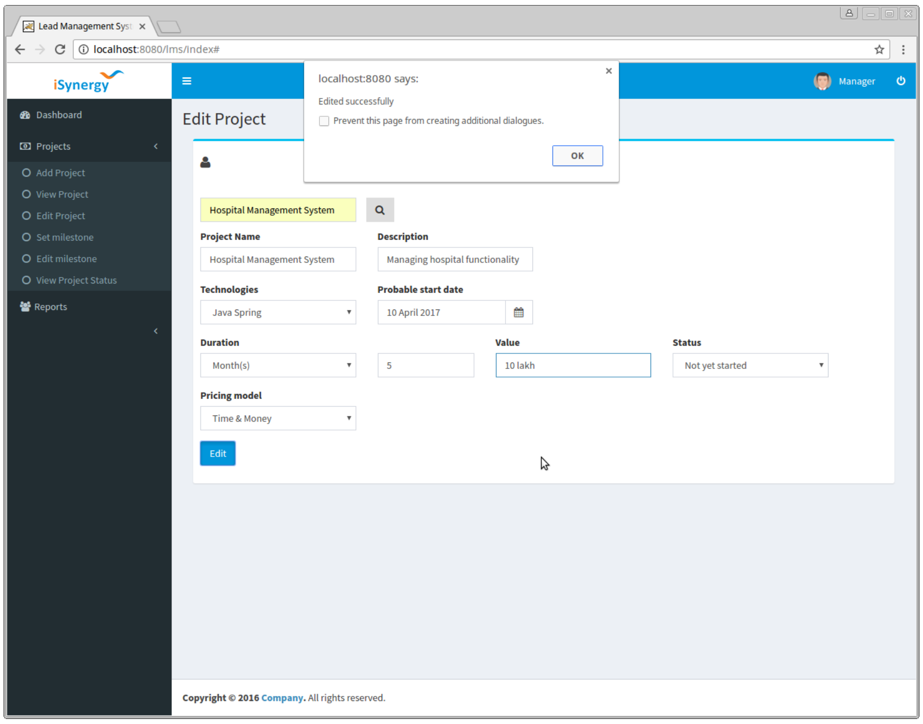


Project List

Project Id	Project Name	Status
2	LMS	completed
3	ABC	partial
4	IT	Ongoing
5	B virtual	onstage
6	Vedioanalysis	onstage
7	Vedioanalysis	onstage
8	Vedioanalysis	onstage
9	CareerClap	delivered
10	CareerClap-2	delivered
11	foody	not yet st
12	Hospital Management System	Not yet st

Copyright © 2016 Company. All rights reserved.

Edit Project



localhost:8080 says:

Edited successfully

Prevent this page from creating additional dialogues.

OK

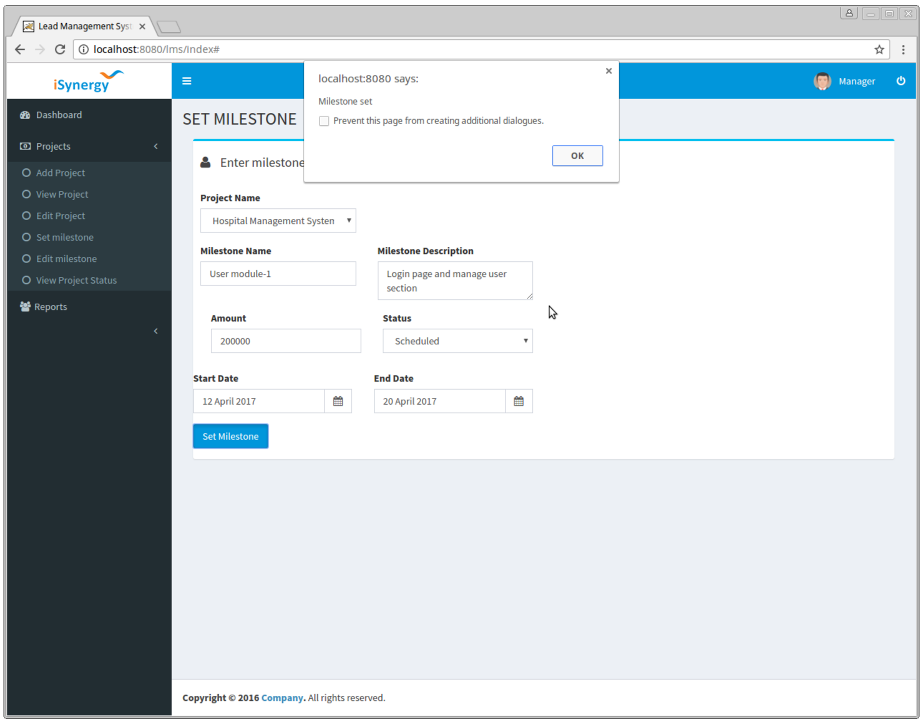
Hospital Management System

Project Name	Description		
Hospital Management System	Managing hospital functionality		
Technologies	Probable start date		
Java Spring	10 April 2017		
Duration		Value	Status
Month(s)	5	10 lakh	Not yet started
Pricing model			
Time & Money			

Edit

Copyright © 2016 Company. All rights reserved.

Set milestone



localhost:8080 says:

Milestone set

☐ Prevent this page from creating additional dialogues.

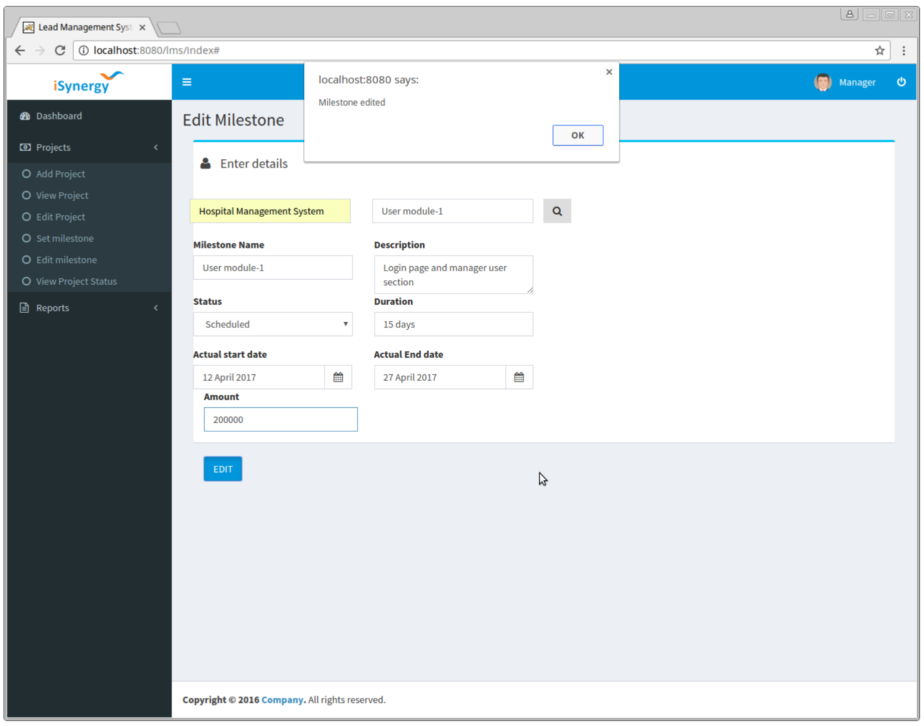
OK

Project Name	Hospital Management Systen
Milestone Name	User module-1
Milestone Description	Login page and manage user section
Amount	200000
Status	Scheduled
Start Date	12 April 2017
End Date	20 April 2017

Set Milestone

Copyright © 2016 Company. All rights reserved.

7 . Edit milestone



localhost:8080 says:

Milestone edited

OK

Edit Milestone

Enter details

Hospital Management System | User module-1

Milestone Name: User module-1

Description: Login page and manager user section

Status: Scheduled

Duration: 15 days

Actual start date: 12 April 2017

Actual End date: 27 April 2017

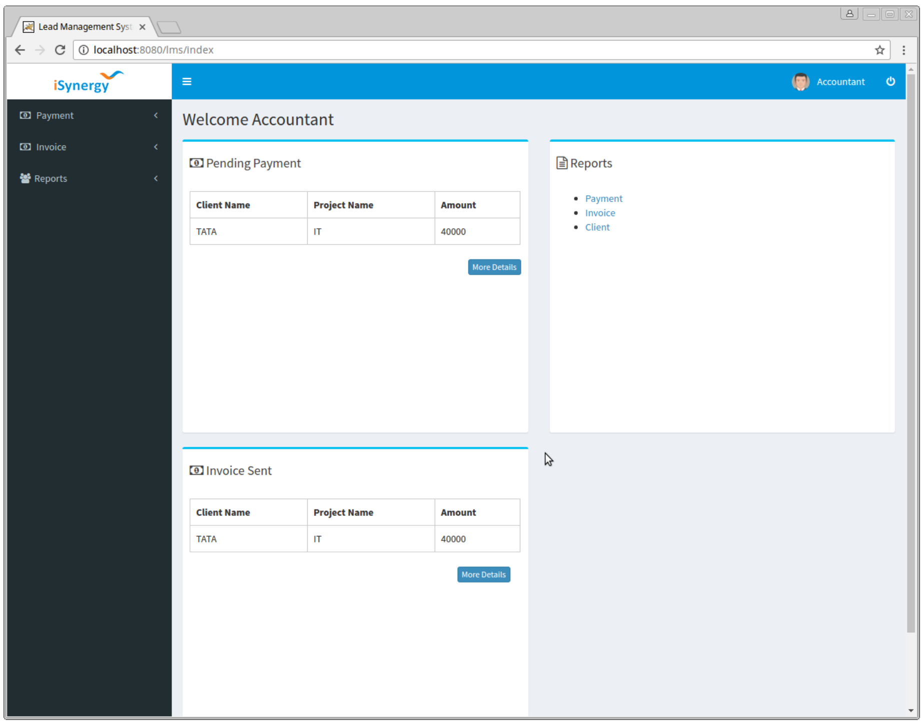
Amount: 200000

EDIT

Copyright © 2016 Company. All rights reserved.

Accountant screens

1. Accountant dashboard



Welcome Accountant

Pending Payment

Client Name	Project Name	Amount
TATA	IT	40000

More Details

Reports

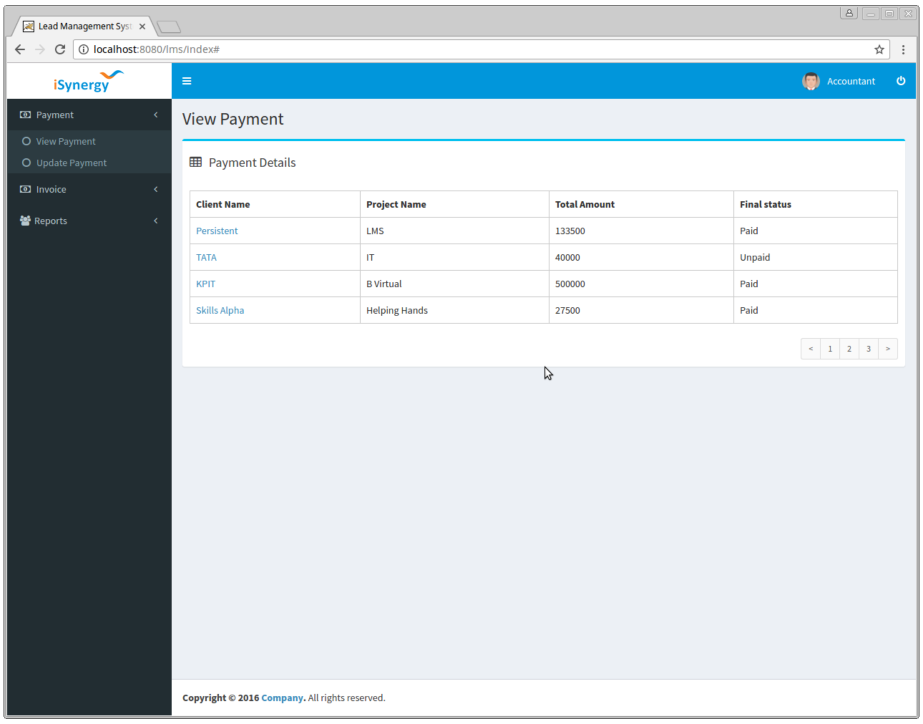
- Payment
- Invoice
- Client

Invoice Sent

Client Name	Project Name	Amount
TATA	IT	40000

More Details

2. View payment



Lead Management Sys

localhost:8080/lms/index#

View Payment

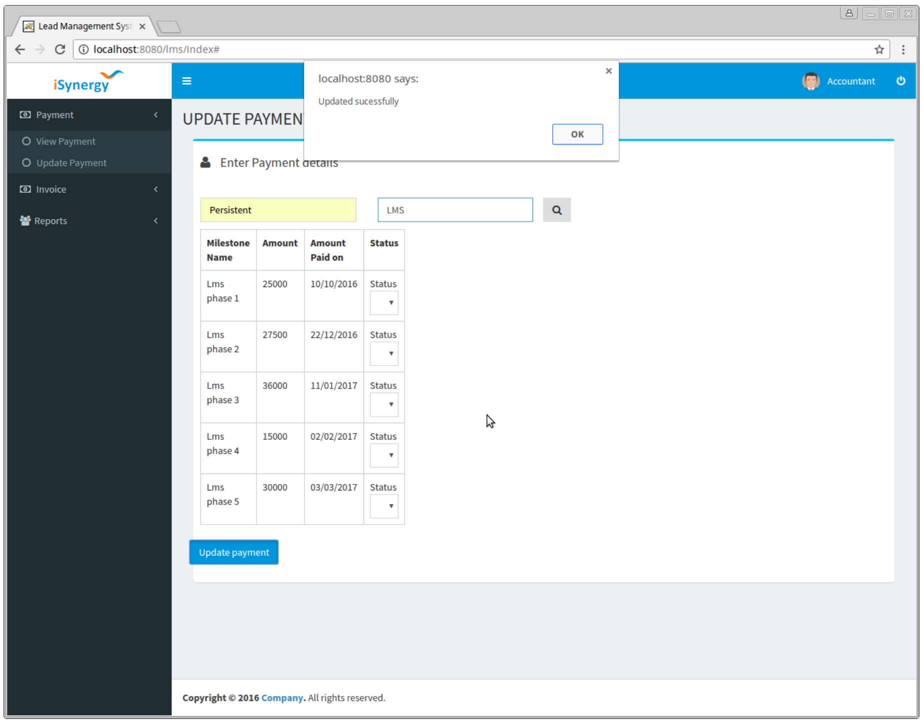
Payment Details

Client Name	Project Name	Total Amount	Final status
Persistent	LMS	133500	Paid
TATA	IT	40000	Unpaid
KPIT	B Virtual	500000	Paid
Skills Alpha	Helping Hands	27500	Paid

< 1 2 3 >

Copyright © 2016 Company. All rights reserved.

3. Update payment



Lead Management Syst

localhost:8080/lms/index#

localhost:8080 says:

Updated sucessfully

OK

UPDATE PAYMEN

Enter Payment details

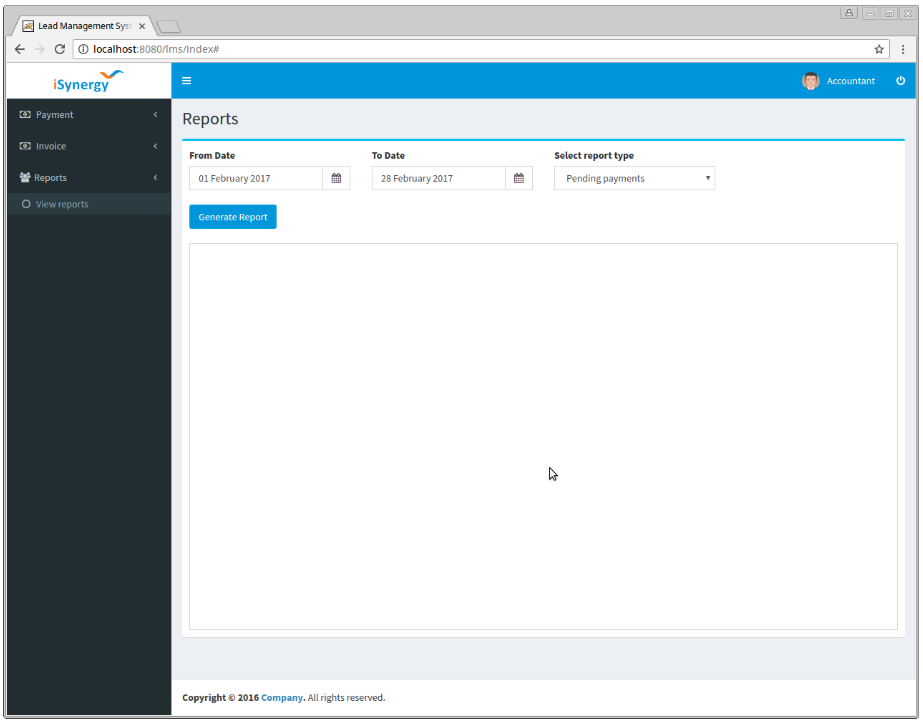
Persistent

LMS

Milestone Name	Amount	Amount Paid on	Status
Lms phase 1	25000	10/10/2016	Status
Lms phase 2	27500	22/12/2016	Status
Lms phase 3	36000	11/01/2017	Status
Lms phase 4	15000	02/02/2017	Status
Lms phase 5	30000	03/03/2017	Status

Copyright © 2016 Company. All rights reserved.

4. view report



Reports

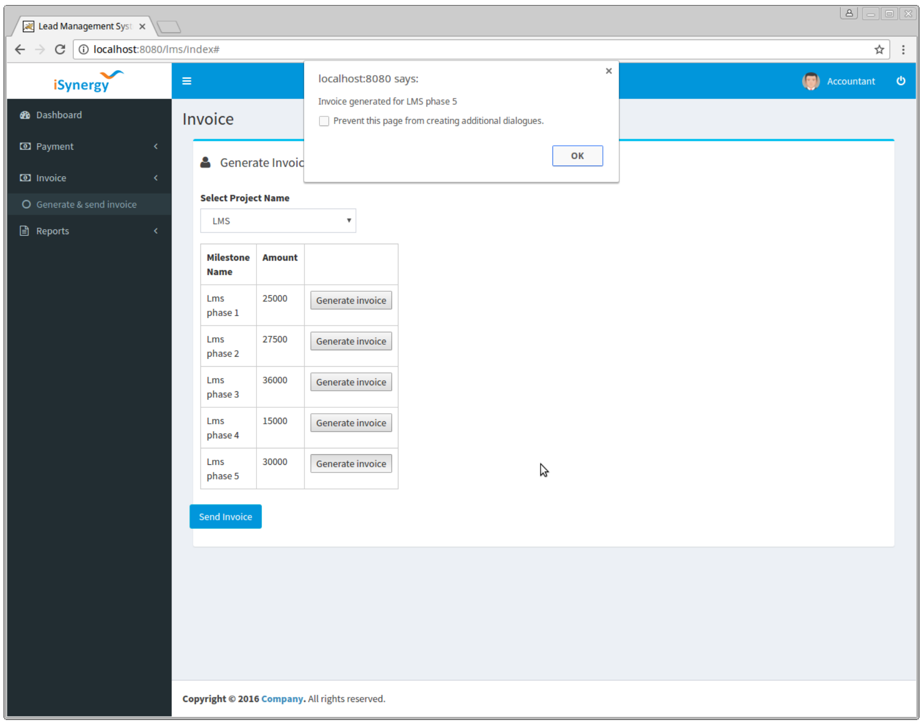
From Date: 01 February 2017

To Date: 28 February 2017

Select report type: Pending payments

Copyright © 2016 Company. All rights reserved.

5. Generate invoice



localhost:8080 says:

Invoice generated for LMS phase 5

☐ Prevent this page from creating additional dialogues.

OK

Invoice

Generate Invoic

Select Project Name

LMS

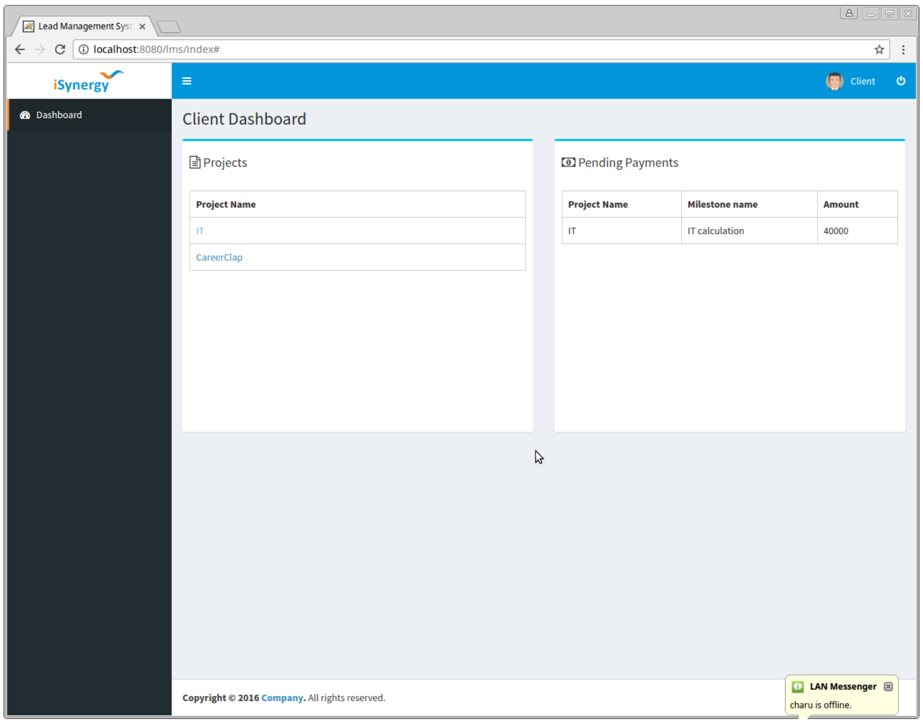
Milestone Name	Amount	
Lms phase 1	25000	Generate invoice
Lms phase 2	27500	Generate invoice
Lms phase 3	36000	Generate invoice
Lms phase 4	15000	Generate invoice
Lms phase 5	30000	Generate invoice

Send Invoice

Copyright © 2016 Company. All rights reserved.

Client screens

1. client dashboard



Lead Management Syst

localhost:8080/lms/index#

iSynergy

Dashboard

Client

Client Dashboard

Projects

Project Name
IT
CareerClap

Pending Payments

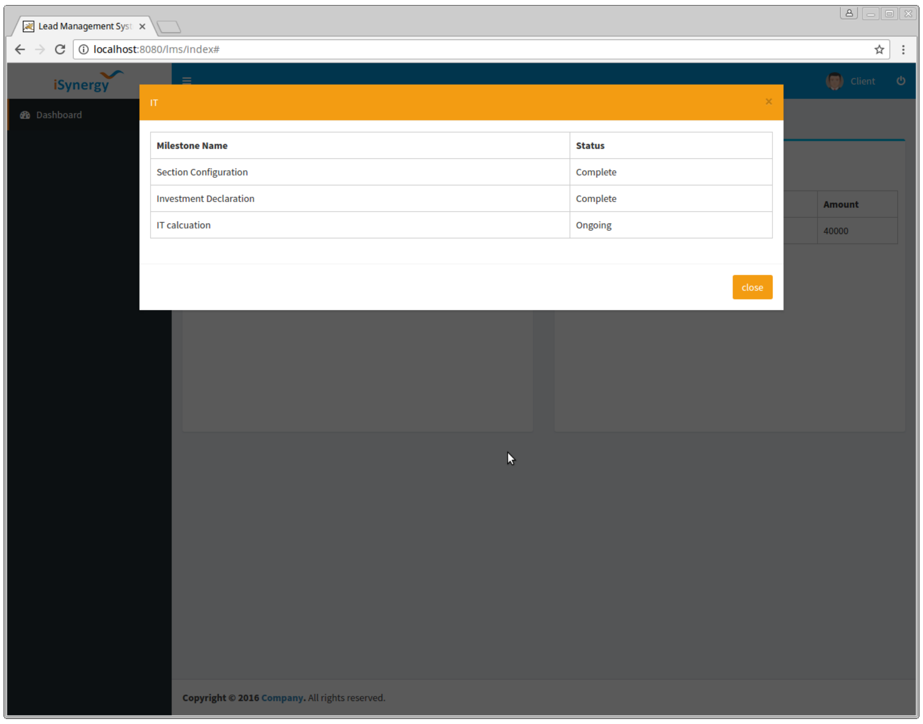
Project Name	Milestone name	Amount
IT	IT calculation	40000

Copyright © 2016 Company. All rights reserved.

LAN Messenger

charu is offline.

2. Client project status



Lead Management Syst

localhost:8080/lms/index#

iSynergy

Dashboard

Client

IT

Milestone Name	Status
Section Configuration	Complete
Investment Declaration	Complete
IT calcuation	Ongoing

close

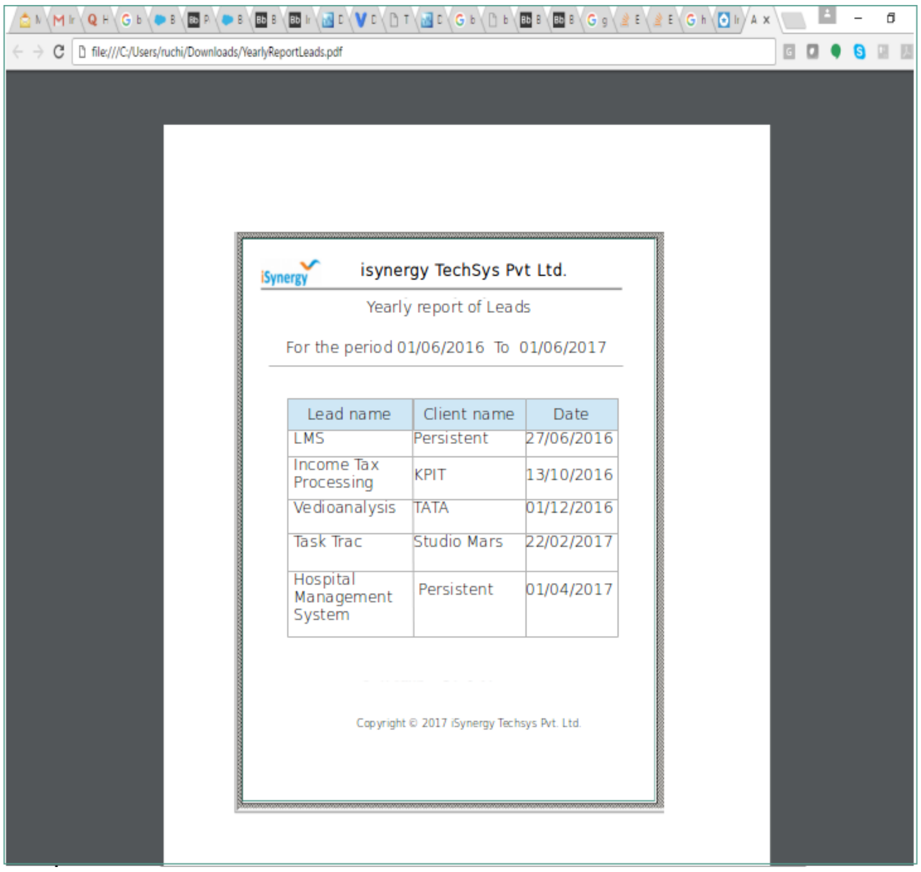
Amount
40000

Copyright © 2016 Company. All rights reserved.

ANNEXURE 2:

REPORTS

1. Yearly report of Leads



file:///C:/Users/ruchi/Downloads/YearlyReportLeads.pdf

isynergy TechSys Pvt Ltd.

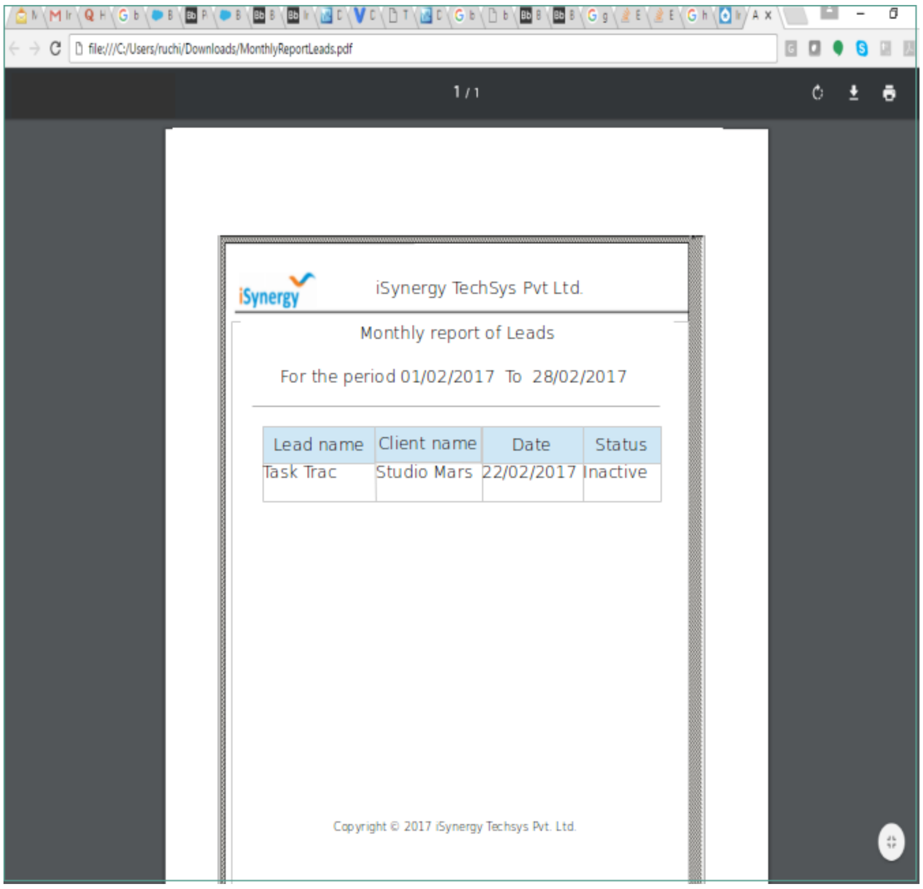
Yearly report of Leads

For the period 01/06/2016 To 01/06/2017

Lead name	Client name	Date
LMS	Persistent	27/06/2016
Income Tax Processing	KPIT	13/10/2016
Vedioanalysis	TATA	01/12/2016
Task Trac	Studio Mars	22/02/2017
Hospital Management System	Persistent	01/04/2017

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

2. Monthly report of leads



file:///C:/Users/ruchi/Downloads/MonthlyReportLeads.pdf

1 / 1

iSynergy TechSys Pvt Ltd.

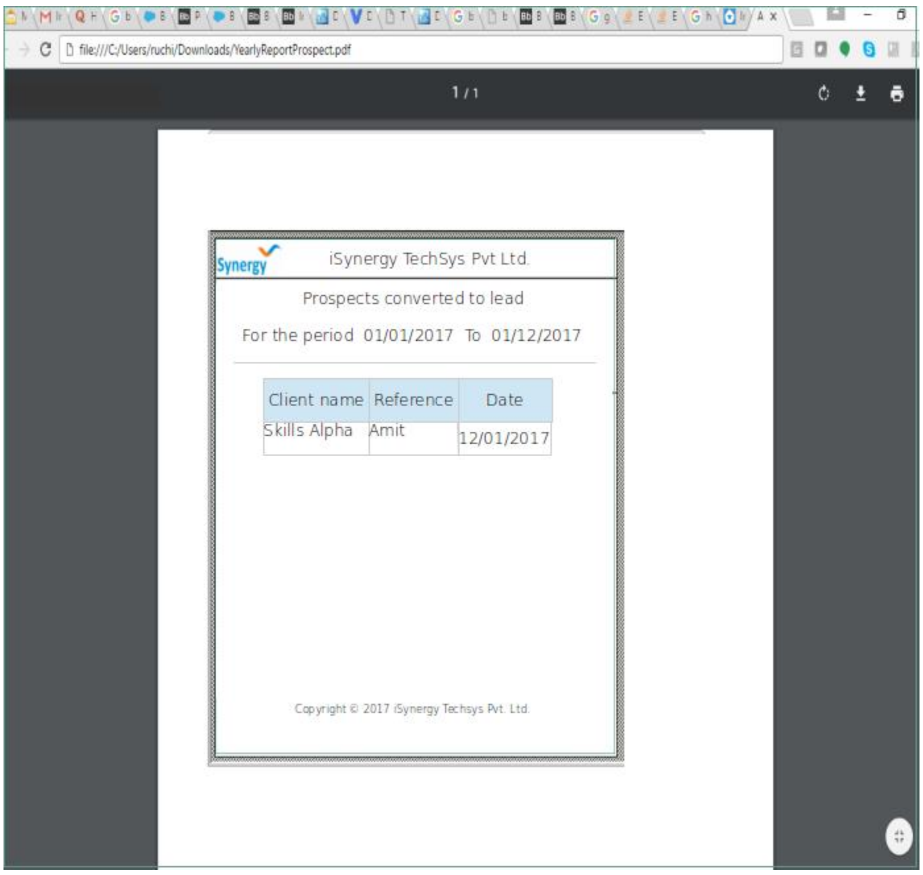
Monthly report of Leads

For the period 01/02/2017 To 28/02/2017

Lead name	Client name	Date	Status
Task Trac	Studio Mars	22/02/2017	Inactive

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

3. Yearly client to prospect



file:///C:/Users/ruchi/Downloads/YearlyReportProspect.pdf

iSynergy TechSys Pvt Ltd.

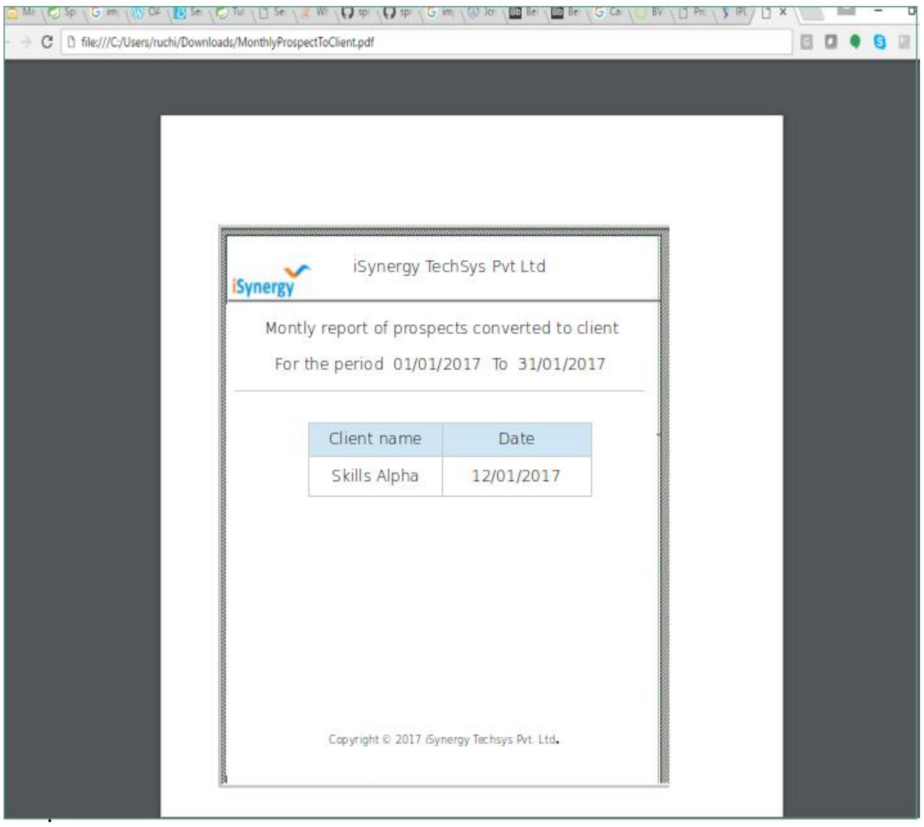
Prospects converted to lead

For the period 01/01/2017 To 01/12/2017

Client name	Reference	Date
Skills Alpha	Amit	12/01/2017

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

4. Monthly prospect to client



file:///C:/Users/ruchi/Downloads/MonthlyProspectToClient.pdf

iSynergy TechSys Pvt Ltd

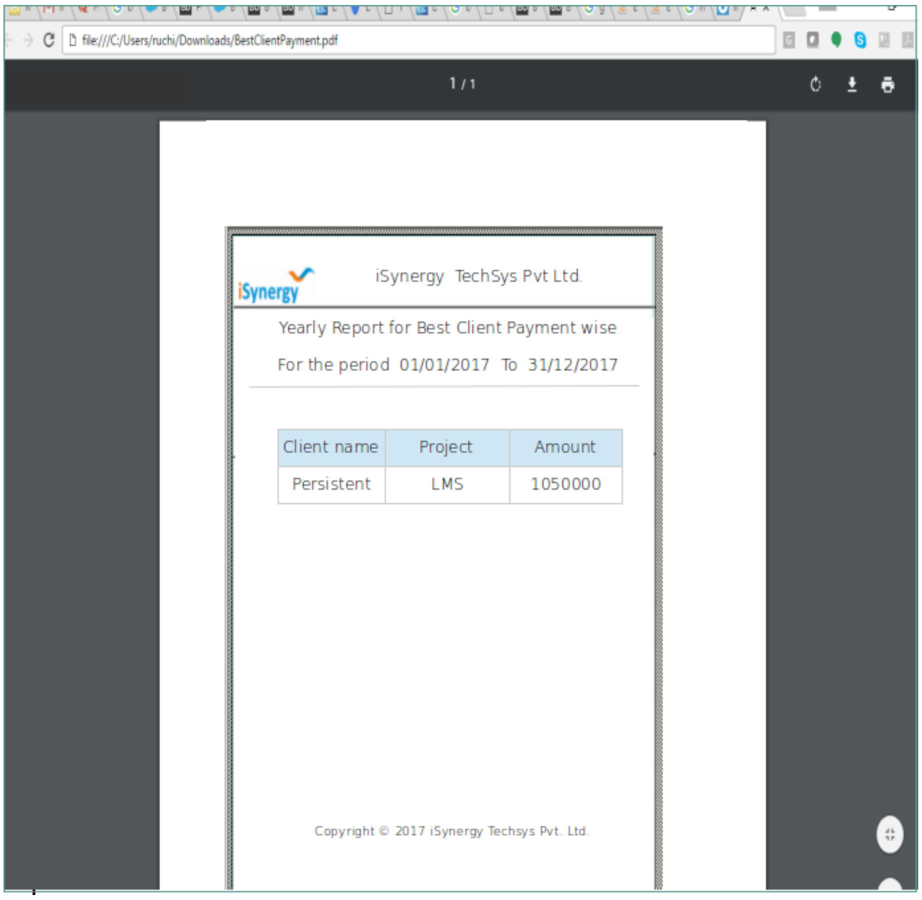
Montly report of prospects converted to client

For the period 01/01/2017 To 31/01/2017

Client name	Date
Skills Alpha	12/01/2017

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

5. Best client yearly-payment wise



file:///C:/Users/ruchi/Downloads/BestClientPayment.pdf

1 / 1

iSynergy TechSys Pvt Ltd.

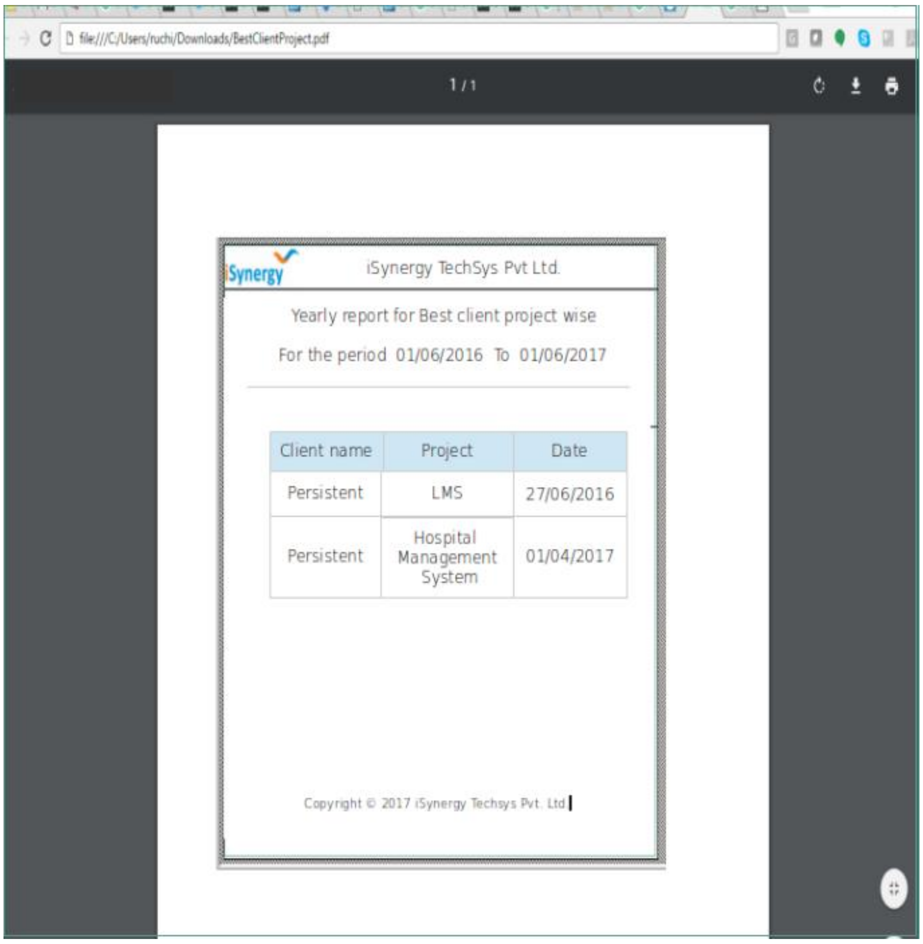
Yearly Report for Best Client Payment wise

For the period 01/01/2017 To 31/12/2017

Client name	Project	Amount
Persistent	LMS	1050000

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

6. Best client project wise



file:///C:/Users/ruchi/Downloads/BestClientProject.pdf

1 / 1

iSynergy TechSys Pvt Ltd.

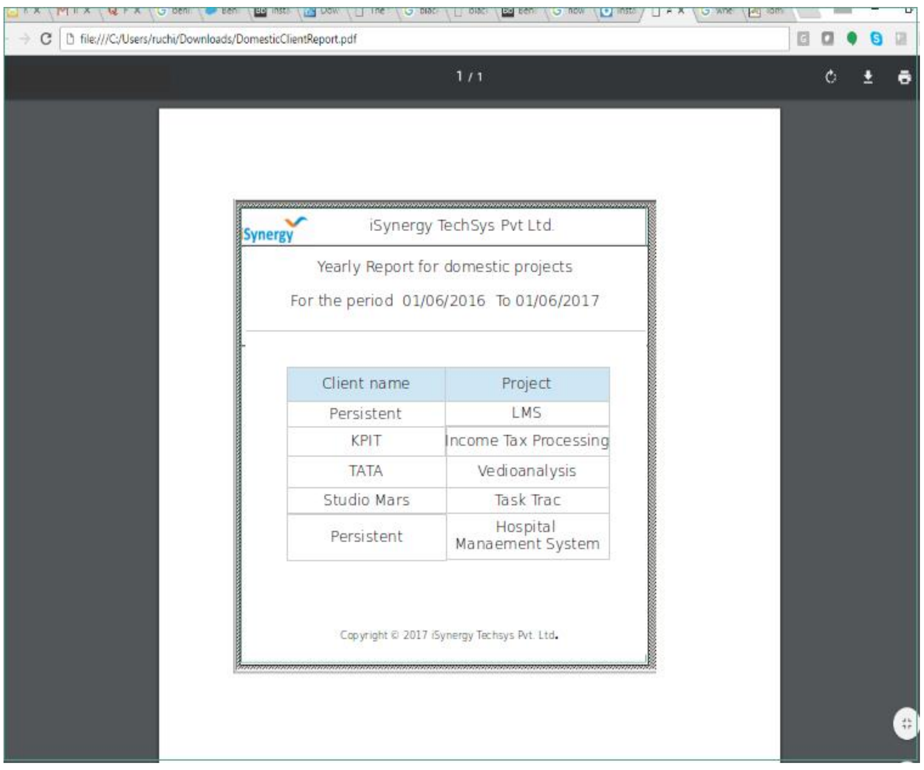
Yearly report for Best client project wise

For the period 01/06/2016 To 01/06/2017

Client name	Project	Date
Persistent	LMS	27/06/2016
Persistent	Hospital Management System	01/04/2017

Copyright © 2017 iSynergy Techsys Pvt. Ltd

7. Domestic project



file:///C:/Users/ruchi/Downloads/DomesticClientReport.pdf

1 / 1

iSynergy TechSys Pvt Ltd.

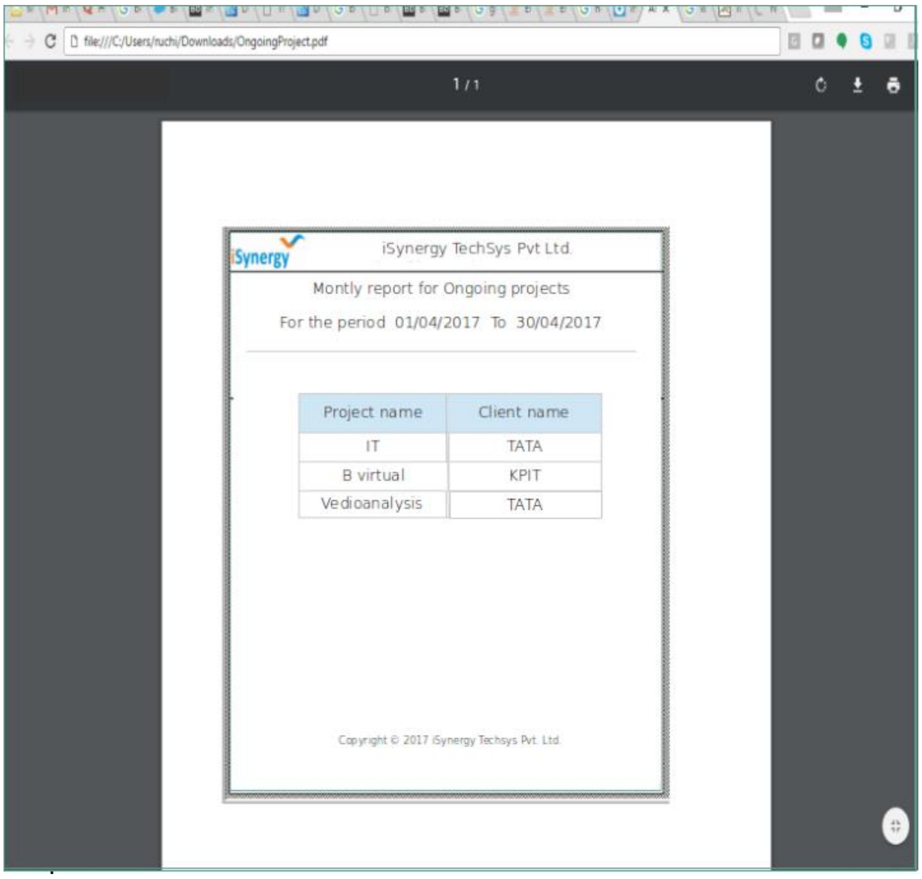
Yearly Report for domestic projects

For the period 01/06/2016 To 01/06/2017

Client name	Project
Persistent	LMS
KPIT	Income Tax Processing
TATA	Vedioanalysis
Studio Mars	Task Trac
Persistent	Hospital Manaement System

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

8.Ongoing projects



iSynergy TechSys Pvt Ltd.

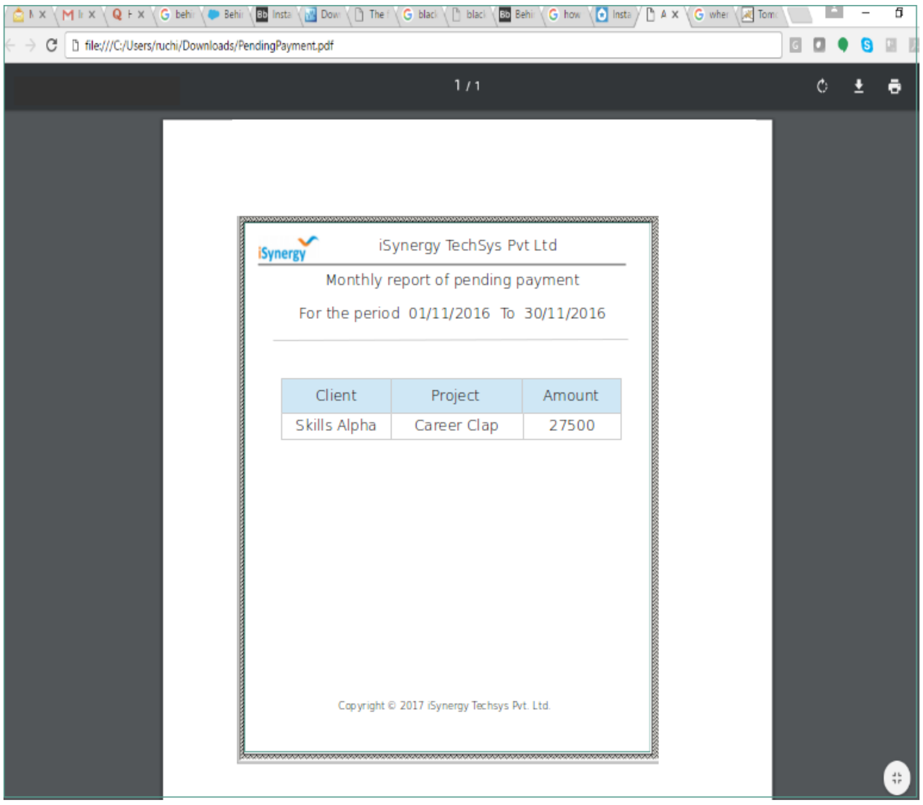
Montly report for Ongoing projects

For the period 01/04/2017 To 30/04/2017

Project name	Client name
IT	TATA
B virtual	KPIT
Vedioanalysis	TATA

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

9.Pending payment



file:///C:/Users/ruchi/Downloads/PendingPayment.pdf

iSynergy TechSys Pvt Ltd

Monthly report of pending payment

For the period 01/11/2016 To 30/11/2016

Client	Project	Amount
Skills Alpha	Career Clap	27500

Copyright © 2017 iSynergy Techsys Pvt. Ltd.

userController.java

```
package com.isynergy.lms.controller;


import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;

import com.isynergy.lms.manager.UserManager;
/*import com.isynergy.lms.vo.LeadVO;
import com.isynergy.lms.vo.Prospect_ClientVO;*/
import com.isynergy.lms.vo.ResponseMessageVO;
import com.isynergy.lms.vo.UserVO;



//@Response body is used when u want to return the response back to UI
@Controller
@RequestMapping(value="/user")
publicclass UserController
{
	@Autowired
	UserManager usermanager;

	//-------ADD NEW USER-----------------
	@RequestMapping(value="/addNewUser",method =
RequestMethod.POST,consumes = "application/json")
	public@ResponseBody ResponseMessageVO addNewUser(@RequestBody
UserVO uservo) throws Exception
	{
		try
		{
			ResponseMessageVO result =
usermanager.addNewUser(uservo);
			System.out.println("result in controller:"+result);

			returnresult;
		}

		catch(Exception e)
		{
			throwe;
		}

	}
	//-----------VIEW ALL USERS IN THE SYSTEM -------------
	@RequestMapping(value="/viewAllUsers",method = RequestMethod.GET)
	public@ResponseBody  List<UserVO> viewAllUsers() throws Exception
	{
		try
		{
			List<UserVO>userlist = new ArrayList<UserVO>();
			returnuserlist = usermanager.viewAllUsers();
		}
		catch(Exception e)
		{
			throwe;
		}


	}

	//---------GET USER BY ID--------------------
	@RequestMapping(value="/getUserById/userID/{userID}",method =
RequestMethod.GET)
	public@ResponseBody UserVO getUserById(@PathVariableintuserID)
throws Exception
	{
		try
		{
//			List<UserVO> userList = new ArrayList<UserVO>();
			UserVO userList = usermanager.getUserById(userID);
			System.out.println("user......"+userList.getUserName());
			returnuserList;
		}
		catch(Exception e)
		{
```

```
                        throwe;

                }

        }

        //---------EDIT USER-------------------

        @RequestMapping(value="/editUser/userID/{userID}",method
=RequestMethod.POST,consumes = "application/json")

        public@ResponseBody ResponseMessageVO
editUser(@PathVariableintuserID) throws Exception

        {

                try

                {

                        ResponseMessageVO result = (ResponseMessageVO)
usermanager.editUser(userID);

                        returnresult;

                }

                catch(Exception e)

                {

                        throwe;

                }

        }

}
```

userManager.java

```
package com.isynergy.lms.manager;


import java.util.ArrayList;

import java.util.List;


import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;


import com.isynergy.lms.dao.UserDAO;

/*import com.isynergy.lms.vo.LeadVO;

import com.isynergy.lms.vo.Prospect_ClientVO;*/

import com.isynergy.lms.vo.ResponseMessageVO;

import com.isynergy.lms.vo.UserVO;


@Component

publicclass UserManager

{

	@Autowired

	UserDAO userdao;


	//----------------ADD NEW USER------------------------------

	public ResponseMessageVO addNewUser(UserVO uservo) throws
Exception

	{

		try

		{
```

```
                    ResponseMessageVO result =
userdao.addNewUser(uservo);
                //System.out.println("result in projectManager is:"+result);
                    returnresult;
                }
                catch(Exception e)
                {
                    throwe;
                }
        }
        //-----------------VIEW ALL USERS IN THE SYSTEM---------------------
        public List<UserVO> viewAllUsers() throws Exception
        {
                try
                {
                List<UserVO>userlist = new ArrayList<UserVO>();
                returnuserlist = userdao.viewAllUsers();
                }
                catch(Exception e)
                {
                    throwe;
                }
        }

        //-------------EDIT USER-------------
        public ResponseMessageVO editUser(intuserID) throws Exception
        {
                try
```

```
            {
                ResponseMessageVO result = userdao.editUser(userID);
                returnresult;

            }
            catch(Exception e)
            {
                throwe;
            }
        }
        public UserVO getUserById(intuserID) throws Exception
        {
            try
            {
//              List<UserVO> userList = new ArrayList<UserVO>();
                UserVO userList = userdao.getUserById(userID);
                returnuserList;
            }
            catch(Exception e)
            {
                throwe;
            }
        }



    }
```

ANNEXURE 3:

SAMPLE CODE

userDao.java

```
package com.isynergy.lms.dao;



import java.sql.ResultSet;

import java.sql.SQLException;

importjava.util.ArrayList;

import java.util.List;


import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.core.env.Environment;

import org.springframework.jdbc.core.RowMapper;

import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;

import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;

import org.springframework.stereotype.Component;


/*import com.isynergy.lms.vo.LeadVO;

import com.isynergy.lms.vo.MilestoneVO;

import com.isynergy.lms.vo.ProjectVO;

import com.isynergy.lms.vo.Prospect_ClientVO;*/
import com.isynergy.lms.vo.ResponseMessageVO;

import com.isynergy.lms.vo.RoleVO;

import com.isynergy.lms.vo.UserVO;




@Component

publicclass UserDAO {


	public Logger logger = LoggerFactory.getLogger(UserDAO.class);


	@Autowired

	NamedParameterJdbcTemplate jdbcTemplate;


	@Autowired

	Environment env;





	// ---------QUERIES----------------------

	privatefinalstatic String INSERT_USER = " INSERT INTO `user`
(role_id,first_name,last_name,username,password,mobile,email,last_update_date,cre
ate_date)"
				+
"VALUES(:roleid,:firstname,:lastname,:username,:password,:mobileno,:email,:last_u
pdate_date,:create_date)";
```

```
        privatefinalstatic String VIEW_USER = "SELECT 
user_id,first_name,last_name,username,mobile,email,role_type FROM user,role "+

                " WHERE user.role_id = role.role_id ORDER BY user_id";

        privatefinalstatic String UPDATE_USER = " UPDATE USER SET 
first_name =: first_name ,last_name= :last_name,"+

                " username = :username,mobile = :mobile,email = :email"+

                " WHERE user_id = :user_id " ;

        privatefinalstatic String GET_USER_BY_ID = "SELECT 
first_name,last_name,username,mobile,email ,role_type FROM user,role"+

                        " WHERE user.role_id = role.role_id AND user_id= 
:user_id";


        public ResponseMessageVO addNewUser(UserVO uservo) {

                ResponseMessageVO messageVO = newResponseMessageVO();

                try {
                        MapSqlParameterSource parameterSource = 
newMapSqlParameterSource();
```

```
                    parameterSource.addValue("roleid",
userVO.getRoleID());// values

                                                    // same as

                                                    // query
                    parameterSource.addValue("firstname",
uservo.getFirstName());
                    parameterSource.addValue("lastname",
uservo.getLastName());
                    parameterSource.addValue("username",
uservo.getUserName());
                    parameterSource.addValue("password",
uservo.getPassword());
                    parameterSource.addValue("mobileno",
uservo.getMobileNo());
                    parameterSource.addValue("email", uservo.getEmail());

          parameterSource.addValue("last_update_date",System.currentTimeMillis());
                    parameterSource.addValue("create_date",
System.currentTimeMillis());

                    if (jdbcTemplate.update(INSERT_USER,
parameterSource) > 0)
                    {

          messageVO.setStatusCode(env.getProperty("RES_SUCCESS_CODE"));
          messageVO.setStatusMessage(env.getProperty("RES_MESSAGE_ADD_U
SER"));

                    }

                    else
                    {

          messageVO.setStatusCode(env.getProperty("ERR_ERROR_CODE"));

          messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_ADD_U
SER"));
                    }

               }
               catch (Exception e)
               {

          messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_ADD_U
SER"));
               }
               returnmessageVO;

          }

          public List<UserVO> viewAllUsers()
          {
```

```
                    MapSqlParameterSource parameterSource =
newMapSqlParameterSource();

                    List<UserVO>userlist = jdbcTemplate.query(VIEW_USER, new
MapSqlParameterSource(),
                                        new RowMapper<UserVO>() {

                                             @Override
                                             public UserVO mapRow(ResultSet rs,
introwNum)
                                                                throws
SQLException {

                                                       UserVO uservo =
newUserVO();
                                                       RoleVO rolevo =
newRoleVO();


          uservo.setUserID(rs.getInt("user_id"));

          uservo.setFirstName(rs.getString("first_name"));
uservo.setLastName(rs.getString("last_name"));

uservo.setUserName(rs.getString("username"));

//uservo.setPassword(rs.getString("password"));

uservo.setMobileNo(rs.getString("mobile"));

uservo.setEmail(rs.getString("email"));

rolevo.setRoleType(rs.getString("role_type"));

/*obj.setLastUpdateDate(rs.getLong("last_update_date"));

obj.setCreateDate(rs.getLong("create_date"));*/

                                        uservo.setRolevo(rolevo);;

                                        returnuservo;
                                }
                        });

        returnuserlist;

}

public ResponseMessageVO editUser(intuserID) throws Exception
```

```
            {
                        UserVO uservo = newUserVO();
                        ResponseMessageVO messageVO = newResponseMessageVO();
                        try
                        {
                                    MapSqlParameterSource parameterSource =
newMapSqlParameterSource();
                                    parameterSource.addValue("userID",userID);

            parameterSource.addValue("first_name",uservo.getFirstName());

            parameterSource.addValue("last_name",uservo.getLastName());

            parameterSource.addValue("username",uservo.getUserName());

            parameterSource.addValue("mobile",uservo.getMobileNo());
                                    parameterSource.addValue("email",uservo.getEmail());

                                    if(jdbcTemplate.update(UPDATE_USER,
parameterSource)>0)
                                    {

            messageVO.setStatusCode(env.getProperty("RES_SUCCESS_CODE"));
                                                System.out.println(messageVO.getStatusCode());

            messageVO.setStatusMessage(env.getProperty("RES_MESSAGE_EDIT_U
SER"));
```

```
            System.out.println(messageVO.getStatusMessage());
                    }
                    else
                    {

            messageVO.setStatusCode(env.getProperty("ERR_ERROR_CODE"));
                                System.out.println(messageVO.getStatusCode());

            messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_EDIT_U
SER"));

            System.out.println(messageVO.getStatusMessage());

                    }

                }
                catch(Exception e)
                {

            messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_EDIT_U
SER"));
                }
                returnmessageVO;
            }

            //---------GET USER BY ID------------------
            public UserVO getUserById(intuserID) throws Exception
```

```
        {
                try
                {
                        MapSqlParameterSource parameterSource = 
newMapSqlParameterSource();
                        parameterSource.addValue("user_id", userID);

                        System.out.println("Id is :"+userID);

                        List<UserVO>userList = jdbcTemplate.query(
                                GET_USER_BY_ID, parameterSource,
                                new RowMapper<UserVO>() {

                                        @Override
                                        public UserVO 
mapRow(ResultSet rs, introwNum)
                                                        throws 
SQLException {



                                                UserVO uservo = 
newUserVO();
                                                RoleVO rolevo = 
newRoleVO();


        uservo.setFirstName(rs.getString("first_name"));
System.out.println("first name :"+uservo.getFirstName());



uservo.setLastName(rs.getString("last_name"));

uservo.setUserName(rs.getString("username"));

uservo.setMobileNo(rs.getString("mobile"));

uservo.setEmail(rs.getString("email"));



rolevo.setRoleType(rs.getString("role_type"));

uservo.setRolevo(rolevo);

                                                        returnuservo;
                                              }
                                     });

                    returnuserList.get(0);

          }
          catch(Exception e)
          {
                    throwe;
          }
```

```
            }
        
}
```

projectController.java

```
package com.isynergy.lms.controller;


import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;

import com.isynergy.lms.manager.ProjectManager;
import com.isynergy.lms.vo.MilestoneVO;
import com.isynergy.lms.vo.ProjectVO;
import com.isynergy.lms.vo.Prospect_ClientVO;

import com.isynergy.lms.vo.ResponseMessageVO;



@Controller

@RequestMapping(value="/project")

publicclass ProjectController

{

	@Autowired

	ProjectManager projectManager;


	// use logger - org.slf4j.logger


	//----------ADD NEW PROJECT

	@RequestMapping(value="/addProject",method =
RequestMethod.POST,consumes = "application/json")

	public@ResponseBody ResponseMessageVO  addProject(@RequestBody
ProjectVO projectvo) throws Exception

	{

		try

		{

			ResponseMessageVO result =
projectManager.addProject(projectvo);

			returnresult;

		}

		catch(Exception e)

		{

			throwe;
```

```
                }


        }


        //-----------VIEW ALL PROJECTS IN THE SYSTEM

        @RequestMapping(value = "/viewProject",method = RequestMethod.GET)

        public@ResponseBody List<ProjectVO> viewProject() throws Exception

        {

                try

                {

                        List<ProjectVO>projectlist = new
ArrayList<ProjectVO>();

                        returnprojectlist = projectManager.viewProject();

                }

                catch(Exception e)

                {

                        throwe;

                }


        }


        //--------UPDATE PROJECT STATUS -PROJECT NAME

        @RequestMapping(value =
"/updateProjectStatus/projectStatus/{projectStatus}/project_name/{project_name}",
method = RequestMethod.PUT)

        public@ResponseBody ResponseMessageVO
updateProjectStatus(@PathVariable String projectStatus,@PathVariable String
project_name) throws Exception
```

```
		{
			try
			{
				ResponseMessageVO messagevo =
projectManager.updateProjectStatus(projectStatus,project_name);
				returnmessagevo;
			}
			catch(Exception e)
			{
				throwe;
			}

		}

		//------------GET MILESTONE DETAILS PROJECT WISE
			@RequestMapping(value = "/getProjectMilestoneList",method =
RequestMethod.GET)
			public@ResponseBody List<ProjectVO>
getProjectMilestoneList() throws Exception
			{
				try
				{
					List<ProjectVO>projectList = new
ArrayList<ProjectVO>();
					projectList =
projectManager.getProjectMilestoneList();
					returnprojectList;
				}
				catch(Exception e)
				{
					throwe;
				}



			}

			//------------GET MILESTONES BY PROJECT_ID-------------



		@RequestMapping(value="/getMilestoneByProjectID/project_id/{project_i
d}",method = RequestMethod.GET)
			public@ResponseBody List<MilestoneVO>
getMilestoneByProjectID(@PathVariableintproject_id) throws Exception
			{
				try
				{
					List<MilestoneVO>milestoneList = new
ArrayList<MilestoneVO>();
					milestoneList =
projectManager.getMilestoneByProjectID(project_id);
					returnmilestoneList;
				}
				catch(Exception e)
				{
					throwe;
				}
		}

		//-------------GET PROJECT DETAILS CLIENT WISE
		@RequestMapping(value = "/getProjectListClientWise",method = 
RequestMethod.GET)
		public@ResponseBody List<Prospect_ClientVO> 
getProjectListClientWise() throws Exception
		{
			try
			{
				List<Prospect_ClientVO>projectList = new 
ArrayList<Prospect_ClientVO>();
				projectList = 
projectManager.getProjectListClientWise();
				returnprojectList;
			}
			catch(Exception e)
			{
				throwe;
			}

		}

		//------------GET PROJECT COUNT CLIENT WISE
```

```
        @RequestMapping(value="/getProjectCountClientWise/customer_name/{c
ustomer_name}",method =RequestMethod.GET)
                public@ResponseBody List<ProjectVO>
getProjectCountClientWise(@PathVariable String customer_name) throws
Exception
                {
                        try
                        {
                                List<ProjectVO>projectCountList = new
ArrayList<ProjectVO>();
                                projectCountList =
projectManager.getProjectCountClientWise(customer_name);
                                returnprojectCountList;
                        }
                        catch(Exception e)
                        {
                                throwe;
                        }



                }



                //-----------------GET PROJECT LIST STATUS WISE


        @RequestMapping(value="/getProjectListStatusWise/project_status/{projec
t_status}",method = RequestMethod.GET)
```

```
                    public@ResponseBody List<ProjectVO>
getProjectListStatusWise(@PathVariable String project_status) throws Exception
                    {
                              try
                              {
                                        List<ProjectVO>projectList = new
ArrayList<ProjectVO>();
                                        projectList =
projectManager.getProjectListStatusWise(project_status);
                                        returnprojectList;
                              }
                              catch(Exception e)
                              {
                                        throwe;



                              }


                    }
                    //edit project



                    //-----------ADD NEW MILESTONES
                    @RequestMapping(value = "/addNewMilestone",method =
RequestMethod.POST,consumes = "application/json")
                    public@ResponseBody ResponseMessageVO
addNewMilestone(@RequestBody MilestoneVO milestonevo) throws Exception
                    {
```

```
                try
                {
                        ResponseMessageVO messagevo =
projectManager.addNewMilestone(milestonevo);
                        returnmessagevo;
                }
                catch(Exception e)
                {
                        throwe;
                }
        }


        //-----------SET MILESTONE STATUS-------------
        @RequestMapping(value =
"/setMilestoneStatus/milestoneStatus/{milestoneStatus}/milestoneID/{milestoneID}"
,method = RequestMethod.PUT)
        public@ResponseBody ResponseMessageVO
setMilestoneStatus(@PathVariable String
milestoneStatus,@PathVariableintmilestoneID) throws Exception
        {
                try
                {
                        ResponseMessageVO messagevo =
projectManager.setMilestoneStatus(milestoneStatus,milestoneID);
                        returnmessagevo;
                }
                catch(Exception e)
```

```
                    {
                            throwe;
                    }


            }
            //------------------------REPORTS------------------



        @RequestMapping(value="/getProjectListTypeWise/project_type/{project_
type}",method =RequestMethod.GET)
                            public@ResponseBody List<ProjectVO>
getProjectListTypeWise(@PathVariable String project_type) throws Exception
                            {
                                    try
                                    {

        List<ProjectVO>projectTypeList = new ArrayList<ProjectVO>();
                                            projectTypeList =
projectManager.getProjectListTypeWise(project_type);
                                            returnprojectTypeList;
                                    }
                                    catch(Exception e)
                                    {
                                            throwe;
                                    }

                            }
```

```
}
```

projectManager.java

```
package com.isynergy.lms.manager;


import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

/*import com.isynergy.lms.dao.UserDAO;*/
import com.isynergy.lms.dao.ProjectDAO;
import com.isynergy.lms.vo.MilestoneVO;
import com.isynergy.lms.vo.ProjectVO;
import com.isynergy.lms.vo.Prospect_ClientVO;
import com.isynergy.lms.vo.ResponseMessageVO;

@Component
publicclass ProjectManager
{
        @Autowired
        ProjectDAO projectdao;

        public ResponseMessageVO addProject(ProjectVO projectvo) throws
Exception
        {
                try
                {
                        ResponseMessageVO result =
projectdao.addProject(projectvo);
                        System.out.println("Result in projectManager:"+result);
                        returnresult;
                }
                catch(Exception e)
                {
                        throwe;
                }
        }

        public List<ProjectVO> viewProject() throws Exception
        {
                try
                {
                        List<ProjectVO>projectlist = new
ArrayList<ProjectVO>();
                        returnprojectlist = projectdao.viewProject();
                }
```

```
				catch(Exception e)
				{
						throwe;
				}


		}


		public ResponseMessageVO updateProjectStatus(String
projectStatus,String project_name) throws Exception
		{
				try
				{
						ResponseMessageVO messagevo =
projectdao.updateProjectStatus(projectStatus,project_name);
						returnmessagevo;
				}
				catch(Exception e)
				{
						throwe;
				}
		}


		public List<ProjectVO> getProjectMilestoneList() throws Exception
		{
				try
				{
						List<ProjectVO>projectList = new
ArrayList<ProjectVO>();
```

```
                        projectList = projectdao.getProjectMilestoneList();
                        returnprojectList;
                }
                catch(Exception e)
                {
                        throwe;
                }
        }

        public List<Prospect_ClientVO> getProjectListClientWise() throws
Exception
        {
                try
                {
                        List<Prospect_ClientVO>projectList
=projectdao.getProjectListClientWise();
                        returnprojectList;
                }
                catch(Exception e)
                {
                        throwe;
                }
        }

        public List<ProjectVO> getProjectListTypeWise(String project_type)
throws Exception
        {
                try
                {
                        List<ProjectVO>projectTypeList = new
ArrayList<ProjectVO>();
                        projectTypeList =
projectdao.getProjectListTypeWise(project_type);
                        returnprojectTypeList;
                }
                catch(Exception e)
                {
                        throwe;
                }
        }

        public List<ProjectVO> getProjectCountClientWise(String
customer_name) throws Exception
        {
                try
                {
                        List<ProjectVO>projectCountList = new
ArrayList<ProjectVO>();
                        projectCountList =
projectdao.getProjectCountClientWise(customer_name);
                        returnprojectCountList;
                }
                catch(Exception e)
                {
                        throwe;
                }
```

```
		}

		public List<ProjectVO> getProjectListStatusWise(String project_status)
throws Exception
		{
			try
			{
				List<ProjectVO>projectList = new
ArrayList<ProjectVO>();
				projectList =
projectdao.getProjectListStatusWise(project_status);
				returnprojectList;
			}
			catch(Exception e)
			{
				throwe;
			}
		}

		//---------------- MILESTONE---------------
		public ResponseMessageVO addNewMilestone(MilestoneVO milestonevo)
throws Exception
		{
			try
			{
				ResponseMessageVO messagevo =
projectdao.addNewMilestone(milestonevo);
				returnmessagevo;
				}
				catch(Exception e)
				{
					throwe;
				}

			}

			public ResponseMessageVO setMilestoneStatus(String
milestoneStatus,intmilestoneID) throws Exception
			{
				try
				{
				ResponseMessageVO messagevo =
projectdao.setMilestoneStatus(milestoneStatus,milestoneID);
				returnmessagevo;
				}
				catch(Exception e)
				{
					throwe;
				}

			}

			//----------GET MILESTONE LIST BY PROJECT ID---------------
			public List<MilestoneVO> getMilestoneByProjectID(intproject_id) throws
Exception
			{
```

```
                    try
                    {
                              List<MilestoneVO>milestoneList = new
ArrayList<MilestoneVO>();
                              milestoneList =
projectdao.getMilestoneByProjectID(project_id);
                              returnmilestoneList;

                    }
                    catch(Exception e)
                    {
                              throwe;
                    }
          }



}
```

projectDAO.java

```
package com.isynergy.lms.dao;


import java.sql.ResultSet;
import java.sql.SQLException;
importjava.util.ArrayList;
```

```
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.core.env.Environment;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.stereotype.Component;


import com.isynergy.lms.vo.MilestoneVO;
import com.isynergy.lms.vo.ProjectVO;
import com.isynergy.lms.vo.Prospect_ClientVO;
import com.isynergy.lms.vo.ResponseMessageVO;


@Component
publicclass ProjectDAO
{

	@Autowired
	NamedParameterJdbcTemplate jdbcTemplate;

	@Autowired
	Environment env;
	//------------QUERIES

	privatefinalstatic String INSERT_PROJECT ="INSERT INTO 
project(prospect_client_id,project_name,project_description,project_technologies,pro
bable_start_date,"+

	"duration,VALUE,project_status,project_type,created_date,last_update_date
)VALUES"+


	"(:prospectclientid_fk,:projectname,:projectdescription,:technologies,:proba
blestartdate,:duration,:value,:project_status,:project_type,:createdate,:lastupdatedate)
";

	privatefinalstatic String VIEW_PROJECT ="SELECT 
project_name,project_description,"+
	"project_technologies,probable_start_date,duration,VALUE FROM 
project";

	privatefinalstatic String UPDATE_PROJECT_STATUS = "UPDATE 
project SET project_status =:project_status"+

						" WHERE project_name = 
:project_name";
```

```
            privatefinalstatic String GET_PROJECT_LIST = "SELECT
project_id,project_name,project_status FROM project";


            privatefinalstatic String GET_MILESTONE_DETAILS = "SELECT
milestone_name,milestone_amount, milestone_status"
                                    + " FROM milestone WHERE milestone.project_id =
:project_id";

            privatefinalstatic String GET_CLIENT_LIST ="SELECT
prospect_client_id,customer_name FROM prospect_client";

            privatefinalstatic String GET_PROJECT_DETAILS ="SELECT
project_name,project_status,project_type,project_technologies from project" +

                                                  " WHERE project.prospect_client_id =
:prospect_client_id";




            privatefinalstatic String GET_PROJECT_TYPE_LIST ="SELECT
project_name ,project_description,project_technologies,project_status,"+

                                                  "pricing_model FROM project WHERE
project_type=:project_type";
```

```
        privatefinalstatic String PROJECT_COUNT_CLIENTWISE ="SELECT
customer_name, COUNT(project.prospect_client_id ) "+

                                        "FROM prospect_client  ,project
WHERE "+

                                                "
prospect_client.prospect_client_id = project.prospect_client_id "+

                                                " GROUP BY customer_name
";

        privatefinalstatic String PROJECT_LIST_STATUSWISE ="SELECT
project_name ,project_description,project_type"+

                                        " FROM project WHERE
project_status=:project_status";


        privatefinalstatic String ADD_NEW_MILESTONE ="INSERT INTO
milestone(project_id,start_date,end_date,milestone_name,"+


        "milestone_description,milestone_amount,milestone_status,actual_start_dat
e,"+

                                "actual_end_date)"+
        "VALUES(:projectID,:startDate,:endDate,:milestoneName,:milestoneDescri
ption,"+

        ":milestoneAmount,:milestoneStatus,:actual_start_date,:actual_end_date)";

        privatefinalstatic String UPDATE_MILESTONE_STATUS =  "UPDATE
milestone SET milestone_status= :milestone_status "+

                                        "WHERE milestone_id =
:milestone_id";


        privatefinalstatic String GET_MILESTONE_BY_PROJECTID = "
SELECT milestone_name,"+

                                        "
milestone_amount,milestone_status FROM milestone"+

                                        " WHERE project_id =
:project_id";
```

```
          public ResponseMessageVO addProject(ProjectVO projectvo)
          {
                    ResponseMessageVO messageVO = newResponseMessageVO();
                    try
                    {

                              MapSqlParameterSource parameterSource =
newMapSqlParameterSource();



          parameterSource.addValue("prospectclientid_fk",projectvo.getProspectClie
ntID());//values same as query

          parameterSource.addValue("projectname",projectvo.getProjectName());

          parameterSource.addValue("projectdescription",projectvo.getProjectDescrip
tion());

          parameterSource.addValue("technologies",projectvo.getTechnologies());

          parameterSource.addValue("probablestartdate",projectvo.getProbableStartD
ate());

          parameterSource.addValue("duration",projectvo.getDuration());
                              parameterSource.addValue("value",projectvo.getValue());

          parameterSource.addValue("project_status",projectvo.getProjectStatus());
```

```
        parameterSource.addValue("project_type",projectvo.getProjectType());

        parameterSource.addValue("createdate",System.currentTimeMillis());

        parameterSource.addValue("lastupdatedate",System.currentTimeMillis());

                        if(jdbcTemplate.update(INSERT_PROJECT,
parameterSource)>0)
                        {

        messageVO.setStatusCode(env.getProperty("RES_SUCCESS_CODE"));

        messageVO.setStatusMessage(env.getProperty("RES_MESSAGE_ADD_P
ROJECT"));
                        }

                        else
                        {

        messageVO.setStatusCode(env.getProperty("ERR_ERROR_CODE"));

        messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_ADD_P
ROJECT"));
                        }
```

```
		}
		catch(Exception e)
			{
				e.printStackTrace();
			}
			returnmessageVO;



		}

		public List<ProjectVO> viewProject()
		{
			MapSqlParameterSource parameterSource =
newMapSqlParameterSource();


			List<ProjectVO>projectlist =
jdbcTemplate.query(VIEW_PROJECT, new MapSqlParameterSource(),
					new RowMapper<ProjectVO>() {

						@Override
						public ProjectVO mapRow(ResultSet
rs, introwNum)
								throws
SQLException {

							ProjectVO obj =
newProjectVO();
//obj.setProjectID(rs.getInt("project_id")); // values from db

                                                  // tables

//obj.setProspectClientID(rs.getInt("prospect_client_id"));

obj.setProjectName(rs.getString("project_name"));

obj.setProjectDescription(rs.getString("project_description"));

obj.setTechnologies(rs.getString("project_technologies"));

obj.setProbableStartDate(rs.getLong("probable_start_date"));

obj.setDuration(rs.getString("duration"));

obj.setValue(rs.getFloat("VALUE"));

/*obj.setCreateDate(rs.getLong("created_date"));

obj.setLastUpdateDate(rs.getLong("last_update_date"));*/

                                       returnobj;
                               }
                       });
```

```
		returnprojectlist;



	}

	public ResponseMessageVO updateProjectStatus(String 
projectStatus,String project_name)
	{
		ResponseMessageVO messageVO = newResponseMessageVO();
		MapSqlParameterSource source = newMapSqlParameterSource();

		//values that are passed in the parameters are stored in the variables
		//so pass them directly
		source.addValue("project_status", projectStatus);
		source.addValue("project_name",project_name);


		if(jdbcTemplate.update(UPDATE_PROJECT_STATUS, 
source)>0)
		{

	messageVO.setStatusCode(env.getProperty("RES_SUCCESS_CODE"));

	messageVO.setStatusMessage(env.getProperty("RES_MESSAGE_PROJEC
TSTATUS_UPDATE"));
		}
```

```
			else
			{

		messageVO.setStatusCode(env.getProperty("ERR_ERROR_CODE"));

		messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_PROJE
CTSTATUS_UPDATE"));
			}
			returnmessageVO;



		}



		public List<ProjectVO> getProjectMilestoneList() //viewProject.jsp
		{

			//private final static String GET_PROJECT_LIST = "SELECT
project_id,project_name,project_status FROM project";

			//private final static String GET_MILESTONE_DETAILS =
"SELECT milestone_name,milestone_amount, milestone_status,actual_start_date,"
			//			+ "actual_end_date FROM milestone WHERE
milestone.project_id = :project_id";
```

```
                    List<ProjectVO>projectList =
jdbcTemplate.query(GET_PROJECT_LIST, new MapSqlParameterSource(),
                                        new RowMapper<ProjectVO>() {
                                             @Override
                                             public ProjectVO mapRow(ResultSet
rs, introwNum)
                                                       throws
SQLException {
                                                  ProjectVO projectvo =
newProjectVO();

     projectvo.setProjectID(rs.getInt("project_id"));

     projectvo.setProjectName(rs.getString("project_name")); // values from db

     projectvo.setProjectStatus(rs.getString("project_status"));

                                                  MapSqlParameterSource
parameterSource = newMapSqlParameterSource();

     parameterSource.addValue("project_id", projectvo.getProjectID());


     List<MilestoneVO>milestonesList =
jdbcTemplate.query(GET_MILESTONE_DETAILS, parameterSource,
                                                            new
RowMapper<MilestoneVO>() {
```

```
@Override

public MilestoneVO mapRow(ResultSet rs, introwNum)

        throws SQLException {




MilestoneVO milestonevo = newMilestoneVO();

milestonevo.setMilestoneName(rs.getString("milestone_name"));

milestonevo.setMilestoneAmount(rs.getFloat("milestone_amount"));

milestonevo.setMilestoneStatus(rs.getString("milestone_status"));

/*milestonevo.setActual_start_date(rs.getLong("actual_start_date"));

milestonevo.setActual_end_date(rs.getLong("actual_end_date"));*/



returnmilestonevo;
                                                              }
```

```
                                                            });



          projectvo.setMilestoneList(milestonesList);
                                                  returnprojectvo;
                                             }
                                        });
               returnprojectList;
          }




          public List<Prospect_ClientVO> getProjectListClientWise()
          {

               List<Prospect_ClientVO>clientList =
jdbcTemplate.query(GET_CLIENT_LIST, new MapSqlParameterSource(),
                                        new RowMapper<Prospect_ClientVO>() {
                                             @Override
                                             public Prospect_ClientVO
mapRow(ResultSet rs, introwNum)
                                                       throws
SQLException {
                                                  Prospect_ClientVO clientvo =
new Prospect_ClientVO();

          //clientvo.setProspectClientID(rs.getInt("prospect_client_id"));
```

```
            clientvo.setCustomerName(rs.getString("customer_name")); // values from
db

                                                            MapSqlParameterSource
parameterSource = newMapSqlParameterSource();

            parameterSource.addValue("prospect_client_id",
clientvo.getProspectClientID());

                                                            List<ProjectVO>projectList =
jdbcTemplate.query(GET_PROJECT_DETAILS, parameterSource,
                                                                                    new
RowMapper<ProjectVO>() {

            @Override

            public ProjectVO mapRow(ResultSet rs, introwNum)

                        throws SQLException {
```

```
ProjectVO projectVO = newProjectVO();

projectVO.setProjectName(rs.getString("project_name"));

projectVO.setProjectStatus(rs.getString("project_status"));

projectVO.setProjectType(rs.getString("project_type"));

projectVO.setTechnologies(rs.getString("project_technologies"));



returnprojectVO;
                                                                  }
                                                          });



clientvo.setProjectList(projectList);
                                        returnclientvo;
                                }
                        });
        returnclientList;
}

//----------------PROJECT TYPE PROJECT LIST----------------
```

```
        public List<ProjectVO> getProjectListTypeWise(String project_type)
throws Exception
        {
                try
                {
                        MapSqlParameterSource parameterSource =
newMapSqlParameterSource();
                        parameterSource.addValue("project_type", project_type);

                        List<ProjectVO>projectTypeList = jdbcTemplate.query(
                                        GET_PROJECT_TYPE_LIST,
parameterSource,
                                        new RowMapper<ProjectVO>() {

                                                @Override
                                                public ProjectVO
mapRow(ResultSet rs, introwNum)
                                                                throws
SQLException {




        //lead_name,lead_scope,lead_technologies,quoted_value
                                                        ProjectVO obj =
newProjectVO();


        obj.setProjectName(rs.getString("project_name"));
        obj.setProjectDescription(rs.getString("project_description"));

        obj.setTechnologies(rs.getString("project_technologies"));

        obj.setProjectStatus(rs.getString("project_status"));

        obj.setPricingModel(rs.getString("pricing_model"));



                                    returnobj;
                                }
                            });

                returnprojectTypeList;
            }
            catch(Exception e)
            {
                throwe;
            }



        }

        //--------------GET PROJECT COUNT CLIENT WISE
        public List<ProjectVO> getProjectCountClientWise(String
customer_name) throws Exception
```

```
          {
                    try
                    {
                              MapSqlParameterSource parameterSource = 
newMapSqlParameterSource();
                              parameterSource.addValue("customer_name", 
customer_name);

                              List<ProjectVO>ProjectCountList = jdbcTemplate.query(
                                        PROJECT_COUNT_CLIENTWISE,
parameterSource,
                                        new RowMapper<ProjectVO>() {

                                                  @Override
                                                  public ProjectVO
mapRow(ResultSet rs, introwNum)
                                                            throws
SQLException {

          returnnull;



                                                  }
                                        });

                              returnProjectCountList;
                    }
                    catch(Exception e)
```

```
                {
                        throwe;
                }


        }


        public List<ProjectVO> getProjectListStatusWise(String project_status)
throws Exception
        {
                try
                {
                        MapSqlParameterSource parameterSource =
newMapSqlParameterSource();
                        parameterSource.addValue("project_status",
project_status);

                        List<ProjectVO>projectList = jdbcTemplate.query(
                                        PROJECT_LIST_STATUSWISE,
parameterSource,
                                        new RowMapper<ProjectVO>() {

                                                @Override
                                                public ProjectVO
mapRow(ResultSet rs, introwNum)
                                                                throws
SQLException {
```

```
                                                            ProjectVO obj =
newProjectVO();

        obj.setProjectName(rs.getString("project_name"));

        obj.setProjectDescription(rs.getString("project_description"));

        obj.setProjectType(rs.getString("project_type"));



        returnobj;


                                               }
                                       });

                        returnprojectList;
                }
                catch(Exception e)
                {
                        throwe;
                }
         }



        //---------------milestones-----------------
```

```
		public ResponseMessageVO addNewMilestone(MilestoneVO milestonevo)
		{
			ResponseMessageVO messageVO = newResponseMessageVO();
			try {
				MapSqlParameterSource parameterSource = 
newMapSqlParameterSource();

				parameterSource.addValue("projectID", 
milestonevo.getProjectID());// values

											// same as

											// query
				parameterSource.addValue("startDate", 
milestonevo.getStartDate());
				parameterSource.addValue("endDate", 
milestonevo.getEndDate());
				parameterSource.addValue("milestoneName", 
milestonevo.getMilestoneName());
				parameterSource.addValue("milestoneDescription", 
milestonevo.getMilestoneDescription());
				parameterSource.addValue("milestoneAmount", 
milestonevo.getMilestoneAmount());
				parameterSource.addValue("milestoneStatus", 
milestonevo.getMilestoneStatus());
```

```
        parameterSource.addValue("actual_start_date",milestonevo.getActual_start
_date());

        parameterSource.addValue("actual_end_date",milestonevo.getActual_end_d
ate());

                        if (jdbcTemplate.update(ADD_NEW_MILESTONE,
parameterSource) > 0)
                        {

        messageVO.setStatusCode(env.getProperty("RES_SUCCESS_CODE"));

        messageVO.setStatusMessage(env.getProperty("RES_MESSAGE_ADD_M
ILESTONE"));

                        }

                        else
                        {

        messageVO.setStatusCode(env.getProperty("ERR_ERROR_CODE"));

        messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_ADD_
MILESTONE"));
                        }

                }
```

```
                        catch (Exception e)
                        {

            messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_ADD_
MILESTONE"));
                        }
                        returnmessageVO;

            }





            public ResponseMessageVO setMilestoneStatus(String
milestoneStatus,intmilestoneID)
            {
                        ResponseMessageVO messageVO = newResponseMessageVO();
                        MapSqlParameterSource source = newMapSqlParameterSource();

                        //values that are passed in the parameters are stored in the variables
                        //so pass them directly
                        source.addValue("milestone_status", milestoneStatus);
                        source.addValue("milestone_id",milestoneID);



                        if(jdbcTemplate.update(UPDATE_MILESTONE_STATUS,
source)>0)
                        {
```

```
		messageVO.setStatusCode(env.getProperty("RES_SUCCESS_CODE"));

		messageVO.setStatusMessage(env.getProperty("RES_MESSAGE_UPDAT
E_MILESTONE_STATUS"));
			}

			else
			{

		messageVO.setStatusCode(env.getProperty("ERR_ERROR_CODE"));

		messageVO.setStatusMessage(env.getProperty("ERR_MESSAGE_MILES
TONE_STATUS_UPDATE"));
			}
			returnmessageVO;



		}

		public List<MilestoneVO> getMilestoneByProjectID(intproject_id) throws
Exception
		{
			try
			{
				MapSqlParameterSource parameterSource =
newMapSqlParameterSource();
				parameterSource.addValue("project_id", project_id);
```

```
                    List<MilestoneVO>milestoneList = jdbcTemplate.query(

          GET_MILESTONE_BY_PROJECTID, parameterSource,
                                        new RowMapper<MilestoneVO>() {

                                             @Override
                                             public MilestoneVO
mapRow(ResultSet rs, introwNum)
                                                       throws
SQLException {

                                                  MilestoneVO obj =
newMilestoneVO();

          obj.setMilestoneName(rs.getString("milestone_name"));

          /*obj.setActual_start_date(rs.getLong("actual_start_date"));

          obj.setActual_end_date(rs.getLong("actual_end_date"));*/


          obj.setMilestoneAmount(rs.getFloat("milestone_amount"));

          obj.setMilestoneStatus(rs.getString("milestone_status"));

                                                  returnobj;
                                        }
                                    });

                        returnmilestoneList;

                    }
                    catch(Exception e)
                    {
                        throwe;
                    }


                }

            }
```