**CHAPTER-1**

**INTRODUCTION**

1. **INTRODUCTION**

**1.1. PROJECT OVERVIEW**

The project named as “Hire and build” deals with the perfect utilization of equipment. This will help the ssuperwisers (who would like to involve construction activities) to manage and search the resources easily and effectively. It also provides the facility to rent equipment. The registered parties can participate in the renting process. Handing resources manually is impractical so we have to develop a system which willeasily schedule, track and maintain the equipment. And also it will eliminate the guess work of tedious scheduling and routing to maximize on-time delivery. There will bean administrator login which is accessed by the admin who has full permission.Admin can put equipment for hiring and the registered parties (contractors) can participate in the bidding process. The booking process is fully automated.

The functions are performed by customers and contractors. They can register to this website and join this company only after the approval of admin.

**MODULES:**

Admin Module.

Site Engineer Module.

* Customer Module.

**CHAPTER-2**

**SYSTEM ANALYSIS**

**2. SYSTEM ANALYSIS**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvements on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect the detailed information on the system and the feasibility study of the proposed system. This analysis focuses on the flow of the system module by module and the efficiency of each. To design the proposed system we need the exact processing logic as well as the extended features of the existing system such as reliability, consistency, storage capacity etc. This report will discuss the advantages and drawbacks/disadvantages of the existing system and the modifications and enhancements can be done. This analysis will concentrate on the information gathering for the efficient, user friendly and reliable system, which will carry forward the features of the existing system.

**2.1. REQUIREMENT ANALYSIS**

Requirements analysis results in the specification of software’s operational characteristics, indicates software’s interface with other system elements, and establishes constraints that software must meet. Requirements analysis allows you to elaborate on basic requirements established during the inception, elicitation, and negotiation tasks that are part of Requirements engineering.

**REQIUREMENT GATHERING**

The requirement gathering can be done by following ways.

* Interview.
* Website visit.

For this project is used Website visit and Interview method.

For this project I visited following resources.

* I visited few Construction Websites. These sites are given below:
* http://www.Pinterest.com
* http://www.athemes.com

These websites helps to get the information about the construction company, what are the main functionalities of the construction company and how to develop a website very usefully.

* Interview method helped to collect more information from supplier, and customer.
* I interviewed four peoples to gather information
* I used interview technique for gathering information and I asked the following questions.

To the contractor of Jeena group mr Manoj

1. How you storing the information about the customer?

Data are stored in files and records.

1. What are the difficulties faced by the site engineer?

When the data is stored in the files, it is difficult to identify the old user for any other reference.

1. Is there any equipment renting system?

No. There is no booking system.

1. What are the legal formalities for handling human resources?

Inform police station with their id cards and keep one of the copy in hand. Medical checkup proof. Personal data etc



* Next i interviewed a JCB driver Mr. Nameesh for gathering information and I considered he is a supplier and I asked following questions.

1. Is there any available booking system available now?

A: No. there is not any JCB booking system is available now.

1. How to calculate JCB rent normally?

A: Normally calculate the rent per hour and also consider the risk of the work.

1. All times do you have a work?

A : Not at all rainy season work is very less.

1. How the customer identify you .how they book your work?

A: Normally they contact by phone calls .i have a visiting card they got my advertisement. Or

the customer directly contact me.

1. Work is centralized nearest place or far place ?

A : Both nearest and far places

* I interviewed another small scale contractor Mr. Sanith for gathering information and I considered he is a user and I asked following questions

.

1. How to collect equipment for construction?

A: Normally all the equipments are in my hands if more equipments needed on the time of construction then only hire the equipments from others.

1. So you take the equipments from others .How many suppliers did you meet?

A: One or two suppliers only I meet they are nearest to me.

1. Suppose the equipment cost is very high from the current suppliers what you do?

A: We don’t have more option so we will take the equipments from that supplier.

1. If you make any damages on the equipments then do you repair or make the corresponding payments?

A: Its depend on the situation sometime maybe make the payments sometime maybe repair.

* Next I interviewed Mr. Sajeevan for gathering information and I considered he is supplier and I asked following questions.

1. How the customer identifies the equipments are in your hand?

A: The customers are nearest place so everyone knows to me.

1. So all the customers are from nearest .there is any opportunity to identifies you other customer?

A: Not any proper system is available.

1. How to measure the rent for each items?

A: Normally consider per day particular amount.

**2.2 EXISTING SYSTEM**

The existing system was handling manually. The Existing system is not an integrated system and it is not fully automated. The information’s which consists of human resources and rental of equipment, auctions etc are scattered in the entire system. All documentations are kept as MS-Word documents or excel sheets. In the current system not easy to find the resources for customer. This will make the current system more time consuming. It can't provide proper interaction between the users and customer. Customer didn’t know who is the best supplier where he can get equipment with minimum cost and minimum distance.

**From the evaluation of the existing system, we can find out the following drawbacks:**

* No proper handling of resources.
* There are no provisions for the search using the equipment names and other search terms.
* Existing system is too slow for getting recourses.
* All the records done manually more human effort is needed.
* Since it requires large number of paper work, files and books and there are chances of misplacing of important details and files containing crucial data. This will lead to loss of data.
* Time consuming.

**2.3. PROPOSED SYSTEM**

The disadvantages of the Existing System have been solved by automating the ‘***HIRE AND BUILD’*** which helps to current and accurate details. Proposed system is a step ahead in handling day to day activities in a construction parties with the help of computerized system. , since a change from manual system to a computer system saves a lot of time here provide more selecton oppertnities thus saving enormous amount of money. The system identifies which is the best option for the cusomers here applied the simplex method we reduce the cost by using this method and get the optimum solution. The system was designed in such a way that it is useful to any construction parties or a single men .

Details of the customer information, amount paid, acknowledgement details, other information can be provided to the supplier on the click of the button. It is designed keeping in mind all the drawbacks of the present system to provide a permanent solution of the existing system. The main aim of the software is to customer can get the equipments on a minimum cost with minimum distance and provide more details of a suppilers so customer gets a lots of resources .

The proposed system is also expected to reduce the amount of paper work normally these kind of datas are stored in document. The hardcopies of only the necessary document needed to be taken and the rest can be avoided.

**Merits of proposed system**

* Customer can collect the equipments minimum cost with minimum distance
* Minimum time is required.
* Lots of supplier can be available to the customer.
* Increased speed and accuracy.
* Provide long term storage and fast retrieval of data.
* Eliminate all of the paper works.
* User friendly and interactive

**2.4. FEASIBILITY STUDY**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that is spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness. Feasibility study is a test of the system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it’s approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, legal, economic and behavioral feasibilities.

The feasibility of a project can be ascertained in terms of technical, economic, behavioral and legal factors. A feasibility study is documented with a report showing all the ramifications of the project.

The system proposed is tested whether it is feasible by conducting the following

* Technical feasibility
* Economic feasibility
* Behavioral feasibility
* Legal feasibility

**2.4.1 TECHNICAL FEASIBILITY**

The proposed system is a web based application, no extra tools are needed. Later any additional module can also add to the existing one module. The application will provide user friendly environment also.

**2.4.2. ECONOMIC FEASIBILITY**

The application is economically feasible because it provide fast and efficient way of processing .there is no complexity in the application; very less user training is required here.

This syatem is economically feasible because customer get lots of oppertunities instead of one or two suppliers so he can deciside with minimum cost cost with minimum distance

**2.4.3. BEHAVIORAL FEASIBILITY**

People are resistant to change and computers have been known to facilitate change. The Resource scheduler is designed in user friendly manner and no need to provide training for the persons using this software. The operating system used is Windows 8, which is also user friendly. It does not have any operational barriers. So need to provide any special training for using this application software and hence it is behaviorally feasible.

**2.4.4. LEGAL FEASIBILITY**

The use of Resource scheduler System will not violate rules and regulations of law. The management is also supportive. It will not make any violation in norms and rules of government. So the system is legally feasible.

**2.5 SYSTEM REQUIREMENT SPECIFICATION**

System requirements are expressed in a software requirement document. The Software requirement specification (SRS) is the official statement of what is required of the system developers. This requirement document includes the requirements definition and the requirement specification. The software requirement document is not a design document. It should set out what the system should do without specifying how it should be done. The requirement set out in this document is complete and consistent.

**2.5.1. ACTOR IDENTIFICATION**

An actor is someone or something that interacts with the system. An actor is he /she what uses the system. An actor exchanges information with the system. Asking certain questions as detailed below can identify the actors of the system.

|  |  |  |
| --- | --- | --- |
| **1.** | Who will use the main functionality of the system? | Administrator, customer |
| **2.** | Who will lead support from the system and do their daily tasks? | Administrator, supplier, customer |
| **3.** | Who will maintain and administrate the system? | Administrator |
| **4.** | With which other systems, does this system need to interact? | Database. |
| **6.** | Who was interest in the result produced by the system? | Administrator, supplier, customer |

As per the above answers we can conclude the actors. They are:

* Administrator.
* Supplier
* customer

**2.5.2 USE CASE IDENTIFICATION**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact. To find out the use cases, ask the following questions to each of the actors.

* Which functions does the actor require from the system? What does the actor need to do?
* Does the actor need to read, create, destroy, modify or store some kind of information in the system?
* Could the actor’s daily work be simplified or made more efficient by adding new functions to the system?

**2.5.2.1. USE CASES**

**Use case for the actor Administrator**

|  |  |  |
| --- | --- | --- |
| **1** | Which functions does the Administrator require from the system? What does the Admin need to do? | Administrator requires the following functionalities from the system such as Add equipment, view booking, approved Booking, ,view complaints, Add to rent, view payment pending, view equipment,check the bank details(profit).Remove the supliers from website |
| **2** | Does the Administrator need to read, create, destroy, modify or store some kind of information in the system? | Yes. Administrator need to create and view the data if require. |
| **3** |  |  |

Above questions give the following use cases for the actor Administrator.

* Login
* Add equipment
* view booking
* approve supplier registration
* approved Booking
* view sites
* view complaints
* Add to rent
* view equipment
* Communication with supplier
* Logout

**Use case for the actor supplier**

|  |  |  |
| --- | --- | --- |
| **1** | Which functions does the Supplier  require from the system? What does the Supplier need to do? | Supplier requires the following functionalities from the system such as create and view Add complaints, add his own equipmet, View equipment, view site.  Add the payment due details,check the bank details |
| **2** | Does the Supplier need to read, create, destroy, modify or store some kind of information in the system? | Yes. Supplier need to create and view the data if require. |
| **3** | Could the Site Engineer’s daily work be simplified by adding new functions to the system? | Yes, the system can reduce his/her daily work. |

Above questions give the following use cases for the actor Site supplier.

* register
* Login
* view complaints
* Add complaints
* View equipment
* View his customer
* Current availability of resource
* Add complaints
* Check bank details
* Blacklisting customers
* Communication with admin and customer
* Map for source to destination
* Logout

**Use case for the actor Customer**

|  |  |  |
| --- | --- | --- |
| **1** | Which functions does the customer require from the system? What does the customer need to do? | Customer requires the following functionalities from the system such as Add complaints, view rent, View equipment, equipment booking, book rent, and communication with supplier, payment option. |
| **2** | Does the customer need to read, create, destroy, modify or store some kind of information in the system? | Yes. Customer need to create and view the data if require. |
| **3** | Could the customer’s daily work be simplified by adding new functions to the system? | Yes, the system can reduce his/her daily work. |

Above questions give the following use cases for the actor Customer.

* Register
* Login
* Add complaints
* View details of suppliers
* Map for source to destination
* view rent
* View equipment
* equipment booking
* book rent
* rent payment
* communication with supplier
* Logout

**USE CASE DIAGRAM**



**SUPPLIER**



**ADMIN**



**CUSTOMER**

**2.5.3. ACTIVITY DIAGRAM**

The activity diagram supplements the use case by providing a graphical representation of the flow of interaction within a specific scenario. It uses rounded rectangles to imply a specific system function, arrows to represent flow through the system, decision diamonds to depict a branching decision, and solid horizontal lines to indicate that parallel activities are occurring.

The basic purposes of activity diagrams are similar to other diagrams. It captures the dynamic behavior of the system. Other diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

So the purposes can be described as:

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

Activity diagram symbols

Start/ Initial Node

ACTIVITY

Activity / Action State

|  |  |
| --- | --- |
|  |  |

Control Flow / Edge

Object Flow / Control Edge

Activity Final Node

Flow Final Node

Note/ Comment

Decision Node

**Diagram for login all users**

REGISTERED USERS

USER ENTERS LOGIN NAME AND PASSWORD

CHECK NAME PASSWORD

INCORRECT NAME OR PASSWORD

**NO**

**YES**

USER SUCCESSFULLY LOGIN

USER PROFILE PAGE IS DISPLAYED

USER LOGIN TO THE SYSYTEM

**Diagram for customer booking equipment and payment**

ADD REQUIREMENTS

ITEM SEARCH

**NOT FOUND**

REQUIREMENT NOT AVAILABLE

**FOUND**

VIEW ITEM

ADD TO THE CART

VIEW ITEMS IN CART

EDIT ITEMS

CONFIRM ORDER

PAYMENT

PAYPAL

CREDIT CARD

RECIVES ORDER

**2.5.4. SEQUENCE DIAGRAM**

Sequence diagrams are an easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment. A sequence diagram shows an interaction arranged in a time sequence. It shows the objects participating in the interaction by their life lines and the messages they exchange, arranged in a time sequence.

A sequence diagram has two dimensions: a vertical dimension represents time, horizontal dimension represents different objects. The vertical line is called the object’s lifeline. The lifeline represents the object’s existence during the interaction. This form was first popularized by Jacobson. An object is shown as a box at top of a dashed vertical line. A role is slot for an object within a collaboration that describes the type of object that may play the role and its relationships to other roles. However, a sequence diagram does not show the relationships among the roles or the association among the objects. An object role is shown as a vertical dashed line, the life line.

Each message is represented by an arrow between the life lines of two objects. The order in which these messages occur shown top to bottom on the page.Each message is labeled with the message name. The label also can include the argument and some control information and show self-delegation, a message that an object sends to itself, by sending the message arrow back to the same lifeline. The horizontal ordering of the lifelines is arbitrary. Often, all arrows are arranged to proceed in one direction across the page, but this is not always possible and the order conveys no information.

The sequence diagram is very simple and has immediate visual appeal- this is its greatest strength. A sequence diagram is an alternative way to understand the overall flow of the control of a program. Instead of looking at the code and trying to find out the overall sequence of behavior, we can use the sequence diagram to quickly understand that sequence.

**SEQUENCE DIAGRAM**

ADMIN SUPPLIER DATABASE CUSTOMER

VIEW EQUIPMENT

VIEW COMPLAINT

REPLY

VIEW AVAILABILTY

VIEW CUSTOMER

VIEW BLACK LIST

VIEW FEEDBACK

VIEW FEEDBACK

CHECK BANK A/C

CHECK BANK A/C

LOGOUT

LOGOUT

LOGIN LOGIN

ADD EQUIPMENT

ADD SUPPLIER

LOGIN

ADD EQUIPMENT

ADD RENT VIEW EQUIPMENT

ADD DISTANCE VIEW SUPPLIERS

ADD MAP

VIEW MAP S-D

BOOK THE EQUIPMENT

VIEW BOOKING

VIEW BOOKING

APPROVE BOOKING

COMMUNICATION COMMUNICATION

RENT PAYMENT

ADD COMPLAINT

ADD COMPLAINT

ADD FEEDBACK

BLACK LIST

LOGOUT

**2.6. SYSTEM REQUIREMENTS**

**2.6.1. HARDWARE REQUIREMENTS**

The hardware required for the application depends on the following:

* Determining size and capacity requirements
* Computer evaluation and measurements
* Financial factors
* Maintenance and support.

Processor : Core I3 higher version (32bit/64bit) Processor

RAM : 2GB and Above

Hard Disk : 50GB

Keyboard : Standard Keyboard with 104 keys

Mouse : Standard Mouse with 3 buttons

Monitor : Display Panel (1366\*768)

**2.6.2. SOFTWARE REQUIREMENTS**

The software required for the application depends on the following factors:

* The flexibility of the software
* Software contracts
* Limitation of the software

Programming language - JAVA,JAVASCRIPT

DBMS - SQL Server 2008

Development tool - Netbeanse IDE

Development platform - Windows 10

**CHAPTER-3**

**SYSTEM DESIGN**

**3. SYSTEM DESIGN**

Design is a meaningful engineering representation of something that is to be built. It is an iterative process through which requirements are translated in to a blueprint for constructing the software. The goal of the design phase is to plan a solution of the problem specified by the requirements document.

Major activities during the design phase are:

* Data Base Design
* Architectural Design
* Interface Design

**3.1. DATABASE DESIGN**

A database is collections of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy, and security are given higher priority. Database design is an integrated approach to file design. This activity deals with the design of the physical database. All entries and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to reduce the duplication of data, minimize storage space, and retrieve the data efficiently.

Guidelines for designing a database:

* Design a relational schema so that it is easy to explain its meaning. Do not combine attributed from multiple entity and relationship types into a single relation.
* Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
* As far as possible, avoid placing attributes in a base relation whose values may frequently be null.
* Design relation schemas so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

**Advantage**

* Ease of use
* Data independence
* Accuracy and integrity
* Avoiding inordinate delays
* Recovery from failure
* Privacy and security.

**3.1.1. E-R DIAGRAM**

An entity-relationship diagram is a data modeling technique that creates a graphical representation of the entities, and relationship between entities, within an information system.

**There are three basic elements in ER models:**

* **Entities** are the “things” about which we seek information
* **Attributes** are the data we collect about entities.
* **Relationships** provided the structure needed to draw information from multiple entities.

**E-R Diagram Symbols:**

Entity

Attributes

Relation

**E-R DIAGRAM**

M

1

M

M

M

1

SUPPLIER

VIEW

M

1

M

M

M

M

1

M

1

CAN LOGIN

1

1

ADD

M

RESOURCE

CUSTOMER

BLACKLIST

RESOURCE

DITANCE

VIEW

LOG IN

SUPPLIER

VIEW

BOOK

MAIN RESOURCE

ADMIN

COMMUNICATE

DAMAGE DETAILS

ADD

QUERIES

**3.1.2 TABLE DESIGN**

In the database all the information are stored in the form of tables. A table is simply a way storing data in rows and columns. In the system data is stored in many tables.

The table structures are shown below with sample data.

**Table 1: admin\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Id | Bigint | Primary key | Uniquely identify the user |
| Username | Varchar(10) | Not null | Username |
| Password | Varchar(10) | Not null | Password |

**Table2:Customer\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Id | Bigint | primary key | Uniquely identify the user |
| First name | Varchar(15) | Not null | To store first name |
| Middle name | Varchar(10) | Not null | To store middle name |
| Last name | Varchar(10) | Not null | To store last name |
| Gender | Varchar(10) | Not null | To store gender |
| DOB | Date | Not null | To store DOB |
| Place | Varchar(15) | Not null | To store place |
| Post | Varchar(15) | Not null | To store post |
| Pin code | Varchar(15) | Not null | To store pin code |
| State | Varchar(15) | Not null | To store state |
| Phone no | Varchar(15) | Not null | To store phone no |
| Username | Varchar(10) | Not null | Login |
| Password | Varchar(8) | Not null | Password |

**Table 3: addresourcemain\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Resourceid | Bigint | Primary Key | uniquely identify the resource |
| Resourcename | Bigint | Not null | Resource name |
| Description | Int | Not null | To store the description |
| Uploads | Varchar(10) | Not null | To store the photos related files |

**Table 4: add\_supplier\_resource\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Sresourceid | Bigint | Primary Key | uniquely identify the equipment |
| Quantity | Bigint | Foreign key | Quantity of resource |
| Amount | int | Not null | To store the amount |
| Travel\_cost | int | Not null | store travel cost per km |
| Resource\_id | Int | Not null | Forign key |
| Supplier\_id | int | Not null | Forign key |

**Table 5:distance\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Distance\_id | Bigint | Primarykey | Uniquely identify |
| Source | Varchar(10) | Not null | Resource availabile place |
| destination | Varchar(15) | Not null | Destination |
| distance | Int | Not null | To calculate the distance |

**Table 6: place\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Placeid | Bigint | Primary key | Uniquely identify the site |
| placename | Varchar(20) | Notnull | To store the place name |

**Table 7: queries\_to\_admin\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Query\_id | Bigint | Primary key | Uniquely identify |
| Supplier\_id | Bigint | Foreign key | Uniquely identify |
| About | Varchar(50) | Not null | To store description |
| Details | Varchar(100) | Not null | description |
| Status | Varchar(10) | Not null | Viwed or not |

**Table 8: Addfeedback**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Feedback\_Id | Bigint | primary key | Uniquely identify |
| Costumerid | Bigint | Foreign key | Uniquely identify |
| Description | Varchar(50) | Not null | To store description |
| Date | Date | Not null | Date of complaint |

**Table 9: queries\_reply\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| reply\_id | Bigint | Primary key | Uniquely identify |
| Query\_id | Varchar(50) | Forignkey | Uniquely identify |
| Reply | Varchar(50) | Not null | response |
| Date | date | Not null | date |

**Table 10: resourceorder\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Order\_id | Bigint | Primarykey | Uniquely identify |
| Resource\_id | Bigint | Foreign key | Uniquely identify |
| Supplier\_id | Varchar(10) | Foreign key | Uniquely identify |
| Days | int | Not null | Days for resource |
| Required\_qunatity | Varchar(10) | Not null | Quantity |
| Totalcost | Varchar(10) | Not null | Cost |

**Table 11: supplier\_tb**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Constraints** | **Description** |
| Supplier\_id | Bigint | primary key | Uniquely identify the user |
| First name | Varchar(15) | Not null | To store first name |
| Middle name | Varchar(10) | Not null | To store middle name |
| Last name | Varchar(10) | Not null | To store last name |
| Gender | Varchar(10) | Not null | To store gender |
| DOB | Date | Not null | To store DOB |
| Place | Varchar(15) | Not null | To store place |
| Post | Varchar(15) | Not null | To store post |
| Pin code | Varchar(15) | Not null | To store pin code |
| State | Varchar(15) | Not null | To store state |
| Phone no | Varchar(15) | Not null | To store phone no |
| Username | Varchar(10) | Not null | Login |
| Password | Varchar(8) | Not null | Password |
| Type | Varchar(8) | Not null | usertype |
| Proof | Varchar(8) | Not null | Identity |

**3.2 ARCHITECTURAL DESIGN**

The architectural design develops a modular program structure and represents the control relationships between modules. It also defines interfaces that enable data to flow throughout the program.

**3.2.1. DATA FLOW DIAGRAM**

A data flow diagram is a graphical technique that depicts data flow and transforms that are applied as data move from input to output. The DFD is used to represent increasing information flow and functional details. A Level 0 DFD also called a fundamental system model or context model represents the entire software elements as a single bubble with input and output indicated by incoming and outgoing arrows respectively. Additional process and information flow parts are represented in next level i.e., Level 1 DFD. Each of the processes represented at level 1 are sub functions of overall system depicted in the context model.

**Data flow diagram symbol:**

Source/Destination of Data

Data flow

Storage

**Level 0: Context Level**

USERS

REQUEST

RESPONSE

REQUEST

DATABASE

RESPONSE

**Level 1Hiring process**

ADMIN

SUPPLIER

CUSTOMER

REQUEST

REQUEST

REQUEST

RESPONSE

RESPONSE

RESPONSE

DATABASE

**Level 2: Admin process**

ADMIN

ADMIN\_TB

SUPPLIER

SUPPLIER\_TB

ADDRESOURCE MAIN \_TB

PLACE\_TB

DISTANCE\_TB

QUERIES\_TB

REPLY\_TB

**Leval 2: Supplier process**

SUPPLIER

SUPPLIER\_TB

ADDRESOURCE \_TB

SUPPLERRESOURCE \_TB

BLACKLIST\_TB

DAMAGE\_TB

DISTANCE\_TB

REPLYB

CUSTOMER

RESOURCE ORDR\_TB

REPLY\_TB

QUERY ADMIN\_TB

**Leval 2: Customer process**

CUSTOMER

RESOURCE\_TB

DISTANCE\_TB

RESOURCE\_TB

CUSTOMER\_TB

SUPPLIER\_TB

BLACKLIST\_TB

**3.3. INTERFACE DESIGN**

An interface design elements for the software tell how information flows into and out of the system and how it is communicated among the components as part of the architecture.

**3.3.1 INPUT DESIGN**

Input design is the link between the information system and users and those steps that are necessary to put transaction data into a usable form for processing data entry. Instructing the computer to read data from a written printed document can active the activity of putting data into the computer for processing or it can occur by keying data directly into the system. The design of input focusing on controlling the errors, avoid delay, and keeping the process simple. System analyst decides the following input design details.

* What data to input?
* What medium to use?
* How the data is arranged and coded?

In my project named Smart scheduling, I tried to include the following design constrains provided in the software engineering.

**1: Avoid scattering of fields in the forms**

In all forms of the software the textboxes (which provided to input some data), label (which label the text boxes), combo box (list a set of values) etc all are arranged in a neat and well format. It provides a simple look to the pages. The buttons are placed at the bottom of the page and easily accessible to the user. The menus are arranged below the heading and at a minimum level of menus are arranged with pages. Menu provides the continuity to the pages.

**2: User only needs to enter a minimum amount of data**

All forms contain a minimum amount data, but most essentials. No page provides or wanted bulky of data. It provides more easiness to the user. It creates more the software to the end user. Also the operation continues by single click.

**3: Avoid confusion in the forms**

All forms have a well defined menus and each menu name indicate its purpose. So the user can easily access various forms without confusion. Each form and its sub forms are well labelled. So the user can easily identify the forms and work on that.

**The following are the input forms present in this project:**

* User login form
* Add equipment form
* Add booking form
* Approved Booking form
* Add distance form
* Add place form
* Add queries form
* Add map form
* Add damage form
* Add black list form
* Add equipment form for supplier
* Profile updating forms

**E.g. Input design for add the requirements:**



**3.3.2 OUTPUT DESIGN**

Designing computer should proceed in well thought out manner. The term output means any information produced by the information system weather printed or displayed. Output design is a process that involves designing necessary output that have to be used by various users according to requirement. The efficient intelligent output design should remove the system relationship with the users and help in decision making.

When designing the output, system analyst must accomplish the following:

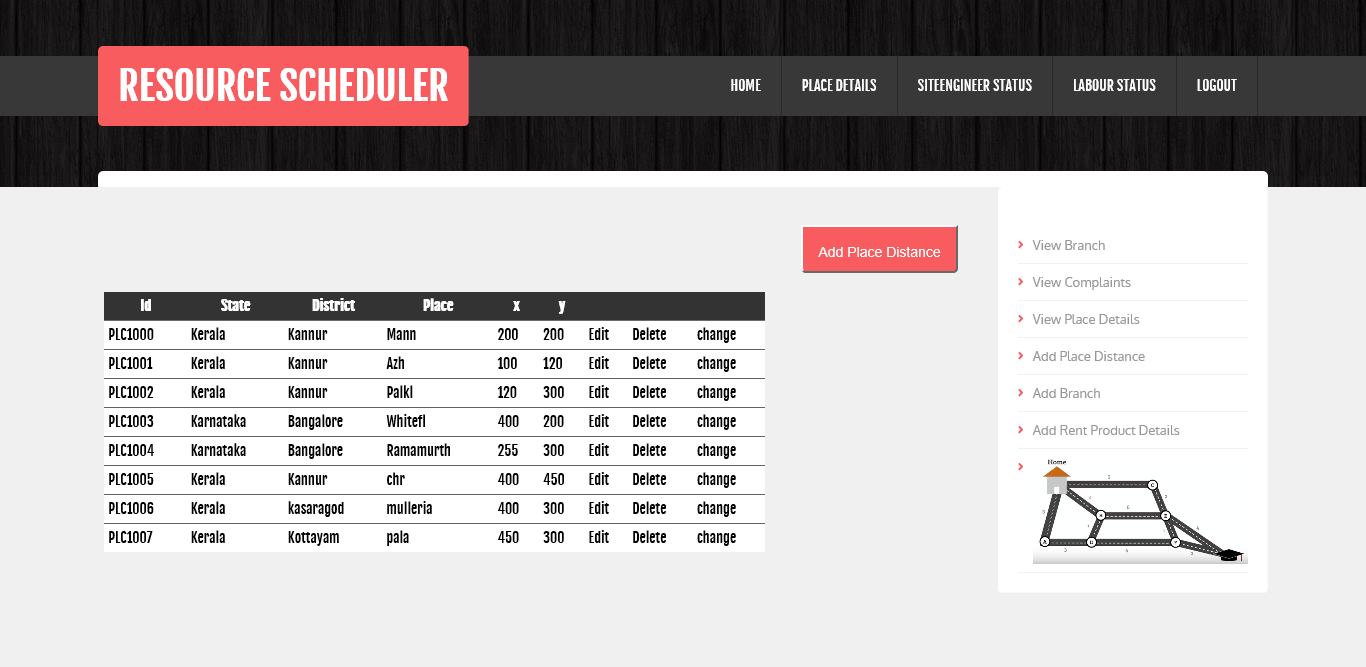
* Determine the information present
* Decide whether to print, display the information and select output medium
* Arrange information in acceptable format.

In my project, the outputs are in the form of reports. They are well format and it provides the output in a correct and neat format.

**The following are the output forms present in this project:**

* Form for view equipment details
* Form for view booking details
* Form for view order details
* Form for view pending details
* Form for view map
* Form for view supplier details
* Form for view my order details
* Form for view place details
* Form for view distance details
* Form for view black listing customer details
* Form for view customer details
* Form for view payment details
* Form for view commission details
* Form for view premium for customer details
* Form for view bill creation details

**E.g. Output design for viewing branch details**



**CHAPTER-4**

**CODING**

**4. CODING**

**4.1 ABOUT THE SOFTWARE TOOLS USED**

**4.1.1 .NET**

.NET is a set of technologies designed to allow applications to work together whether they reside on the user’s hard drive, the local network, a remote computer, or the Internet. It is the common code framework used by nearly all Microsoft applications, such as Windows, Internet Explores and IIS Web server, and it is also the framework for third – party application development based on Microsoft technology. Microsoft .NET relies on standards such as XML, SOAP and UDDI to make it easy for networked objects to discover other networked objects and communicate with them. Because .NET applies to almost all Microsoft products, the company divides .NET into several areas including .NET servers and the .NET Framework.

**4.1.2 . NET FRAMEWORK**

The .NET Framework includes the runtime and compile time services required to run a .NET application. Compile time is when the developer is compiling the source. Runtime is when the compiled code is executing in memory. At the center of the runtime execution of .NET code is the Common Language Runtime (CLR). The CLR is a virtual machine that runs as a process on the computer on which it is installed.

**4.1.3 COMMON LANGUAGERUNTIME**

The Common Language Runtime (CLR) serves as the execution environment for the .NET Framework. The CLR is responsible for managing the compiled code of .NET applications, which can be written in different languages including VB, C#, Java, and Perl. The cross-language integration is achieved through the two major components of CLR: Intermediate Language and Metadata.

Intermediate Language (IL) is an assembly language that runs on almost any type of CPU. IL achieves this versatility by using stacks to handle functions that normally occur in registers. As managed code, IL is just-in-time (JIT) compiled when .NET applications are executed. JIT compilers convert IL into machine language that is specific to the host CPU. [During runtime, JIT compilers have the luxury of choosing stacks, registers or other stores to implement IL stacks.] Various JIT compilers are provided by the CLR, making it possible for different computer architectures to execute IL. Unlike other assembly languages, IL integrates high-level concepts which make CLR code more robust. As a high level language, IL is strongly typed and uses the ideas behind structured-exception handling, deployment support, and component interaction. Thus a range of software can run on the .NET Framework as long as the compiler can produce IL.   
Metadata, the second component of the CLR, is a description of the implemented code. The Metadata is responsible for ensuring that the CLR executes the code securely. To prevent modules of software from breaking type definitions, Metadata stores information regarding classes, methods, and types. Registers are no longer required to keep track of information because relevant data is stored with the compiled code or IL. By housing information on classes and registrations, Metadata allows the CLR to function more efficiently since programs are less likely to get hung up on version and inheritance dependencies.

**4.1.4 Microsoft SQL Server 2008**

Microsoft SQL Server 2008 provides the Microsoft Windows Server System integrated server software with a database platform for the next generation of connected, scalable, and reliable enterprise applications. The breadth and depth of innovation in this version is in response to t he needs of customers. This white paper is targeted to database administrators, to give you an understanding of the new features in and capabilities of SQL Server 2005. From many enhancement of existing features, to an entirely new security model, database administrative is now more productive and in tune with needs of the administrator.

At the core of SQL Server 2008 are new infrastructure application capabilities. SQL Service Broker is a distributed application framework that provides a new form of scalability and reliability for asynchronous message delivery. Though not new, Microsoft SQL server Notification Services, Reporting Services, and SQL Server Mobile Edition (formerly called SQL Server CE) are all greatly enhanced in SQL Server 2005.

SQL (Structured Query Language) is a database computer language designed for the retrieval and management of data in relational database scheme creation and modification, and database object access control management.

**4.2 CODING PRINCIPLE**

The input to the coding phase is the design document. During coding phase, modules identified in the design document are coded according to the module specification. Objectives of coding phase are, to transform design into code and unit test the code.

**4.2.1 Coding Guidelines**

* Code should be easy to understand.
* Don’t take pride in cryptic code.
* Code should be well documented.
* Comments should be present.
* Functions should be small.
* Do not use Go-to statement.
* Do not use the same variable for multiple purposes.
  1. **SAMPLE CODE**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Web;

usingSystem.Web.UI;

usingSystem.Web.UI.WebControls;

usingSystem.Data;

publicpartialclassADMIN\_AddState\_Place : System.Web.UI.Page

{

datamanipulationdm = newdatamanipulation();

protectedvoidPage\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

MultiView1.SetActiveView(View1);

}

}

protectedvoid Button1\_Click(object sender, EventArgs e)

{

String id = dm.GenId("select max(State\_id) from State\_tb", "STA");

Stringsql = "insert into State\_tbvalues('" + id + "','" + txtstate.Text + "')";

int r= dm.ForExecute(sql);

if (r > 0)

{

Response.Write("<script language='javascript'>alert('State has been added... ');</script> ");

txtstate.Text = "";

}

}

protectedvoid Button2\_Click(object sender, EventArgs e)

{

//Add District

String id = dm.GenId("select max(District\_Id) from District\_tb", "DIS");

Stringsql = "insert into District\_tbvalues('"+id+"','"+ddlState.SelectedItem.Text+"','"+txtdistrict.Text+"')";

int r = dm.ForExecute(sql);

if (r > 0)

{

Response.Write("<script language='javascript'>alert('District has been added... ');</script> ");

txtdistrict.Text = "";

}

}

protectedvoid Button3\_Click(object sender, EventArgs e)

{

MultiView1.SetActiveView(View2);

String s = "select \* from State\_tb";

DataSet ds=dm.Bind(s);

if (ds.Tables[0].Rows.Count> 0)

{

ddlState.DataSource = ds;

ddlState.DataTextField = "State\_name";

ddlState.DataValueField = "State\_id";

ddlState.DataBind();

ddlState.Items.Insert(0, "Select State");

}

}

protectedvoid ddlState0\_SelectedIndexChanged(object sender, EventArgs e)

{

String s = "select \* from District\_tb where State\_name='"+ddlState0.SelectedItem.Text+"' ";

DataSet ds = dm.Bind(s);

if (ds.Tables[0].Rows.Count> 0)

{

ddldistrict.DataSource = ds;

ddldistrict.DataTextField = "District";

ddldistrict.DataValueField = "District\_Id";

ddldistrict.DataBind();

ddldistrict.Items.Insert(0, "Select District");

}

protectedvoid Button5\_Click(object sender, EventArgs e)

{

MultiView1.SetActiveView(View3);

Strings\_state = "select \* from State\_tb";

DataSetds\_state = dm.Bind(s\_state);

if (ds\_state.Tables[0].Rows.Count> 0)

{

ddlState0.DataSource = ds\_state;

ddlState0.DataTextField = "State\_name";

ddlState0.DataValueField = "State\_id";

ddlState0.DataBind();

ddlState0.Items.Insert(0, "Select State");

}

}

protectedvoidBtnAddPlace\_Click(object sender, EventArgs e)

{

String id = dm.GenId("select max(Id) from Place\_tb", "PLC");

Stringsql = "insert into Place\_tbvalues('" + id + "','" + ddlState0.SelectedItem.Text + "','" +ddldistrict.SelectedItem.Text + "','" + txtplace.Text + "','"+txt\_x.Text+"','"+txt\_y.Text+"')";

int r = dm.ForExecute(sql);

if (r > 0)

{

Response.Write("<script language='javascript'>alert('Place and its Coordinates has been added... ');</script> ");

}

txt\_x.Text = txt\_y.Text=txtplace.Text = string.Empty;

}

protectedvoid Button6\_Click(object sender, EventArgs e)

{

Response.Redirect("~/ADMIN/ViewPlace\_Coordinates.aspx");

}

}

**CHAPTER-5**

**TESTING**

**5. SYSTEM TESTING**

For software that is newly developed, primary importance is given to testing the system. It is the last opportunity for the developer to detect the possible errors in the software before handing over it to the customer. Testing is the processes by which the developer will generate a set of data, which gives the maximum probability of finding all types of errors that can occur in the software.

The various steps in testing the system can be listed as below:

1. Running the program to identify any errors that might have occurred while feeding the program into the system.
2. Applying the screen formats to regulate users to extend, so that the screens are comprehensible to the user.
3. Presenting the formats to the administration for the purpose of obtaining approval and checking if any modification has to be done. Obtaining feedbacks from users and analyzing the scope for improvement.
4. Checking the data accessibility from the data server and whether any improvement is needed or not.

# The following are the types of Testing:

# Unit Testing

# Integration Testing

* System Testing
* Validation Testing

**5.1. UNIT TESTING**

Unit testing is carried out to screen wise, each screen being identified as an object. Attention is diverted to individual modules, independently to one another to locate in coding and logic.

In unit testing,

* Module interface is tested to ensure that information properly flows into and out of the program under test.
* Local data structures are examined to ensure that data stored temporarily maintains its integrity during all steps in algorithm execution.
* Boundary condition is tested to ensure that the module operates properly at boundaries established to limit or restrict processing.
* All independent paths through the control structures are executed to ensure that all statements in the module have been executed at least once.
* Error handling paths are also tested.

**TEST CASES**

**Login Form**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Test Scenario** | **Expected Result** | **Observed Result** | **Result** |
| 1. | Enter wrong user name and pass word. | Display login form again with a warning message. | Message displayed. | Pass |
| 2. | Enter correct user name and wrong password. | Display login form again with a warning message. | Message displayed. | Pass |
| 3. | Enter correct user name and password. | Administrator or other users can login into the system. | Appropriate home page is displayed. | Pass |
| 4. | Press login button without filling the user name and password. | Display a warning message to fill the fields. | Warning message is displayed. | Pass |

**Customer registration form**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Test Scenario** | **Expected Result** | **Observed Result** | **Result** |
| 1. | Form displayed. | Display the registration form. | Form loaded | Pass |
| 2. | Enter the name in integers. | Display an invalid message. | Invalid message displayed. | Pass |
| 3. | Enter the mobile number in characters. | Display an invalid message. | Invalid message displayed. | Pass |
| 4. | Enter the mobile number more than and less than 10 integers. | Display an invalid message. | Invalid message displayed. | Pass |
| 5. | Click the save button without filling the details. | Display a warning message to fill the details. | Warning message displayed. | Pass |
| 6. | Click on save button with filled fields. | Accept the details. | Registration successfully done. | Pass |
| 7. | Click cancel button | Clear all fields to blank. | All fields cleared. | Pass |

**View Site engineer Details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test scenario** | **Expected result** | **Observed result** | **Result** |
| 1. | Form displayed. | Display form with all controls. | Form loaded with all controls. | Pass |
| 2. | Click add button without data. | Display warning message to fill the details. | Warning message displayed. | Pass |
| 3. | Click save button with data. | Inserted successfully. | Added successfully. | Pass |

# View labourdetails

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Test scenario | Expected result | Observed result | Result |
| 1. | Form displayed. | Display form with all controls. | Form loaded with all controls. | Pass |
| 2. | Click on the hyperlink. | View form displayed with valuable details | Result displayed correctly. | Pass |

# 5.2. INTEGRATION TESTING

Integration testing is a symmetric technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. Unit tested module were taken and a single program structure was built that has been dictated by and tested in small segments, where errors were easy to locate and rectify. Each database or table manipulation operation was written as single program was tested again with numerous test data to check for its functionality.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Input/procedure** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1. | Check the value pass between different forms are appropriate format | Appropriate operations of different forms. | Same as expected. | Pass |

# 5.3. SYSTEM TESTING

System testing is used test the entire system (Integration of the all modules). It also tests to find the discrepancies between the system and the original objective, current specification and system documentation. The entire system is checked to correct deviation to achieve correctness.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Input/procedure** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1. | Check whether indented output is obtained. | All operations are carried out properly. | Same as expected. | Pass |

**5.4. VALIDATION TESTING**

At the conclusion of integration testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of software tests begins validation test has been conducted one of the two possible conditions exists. One is the function or performance characteristics confirm to specification and are accepted and the other is deviation from specification is uncovered and a deficiency list is created.

**CHAPTER-6**

**IMPLEMENTATION**

**6.1 IMPLEMENTATION**

System implementation is the stage where the theoretical design is turned into a working system. The system can be implemented only after through testing is done and if it if found to work according to specifications. The following methods were undergone.

* Testing developed programs with updating.
* Correction of errors identified.
* Creating the tables of the system with actual data.
* Making necessary changes with actual data.
* Doing a parallel run of the system to find out any errors identified and to correct them.
* Training of user personnel’s.

The implementation method used to implement Smart Scheduling is Parallel Run. That is, the new system will work parallel to the existing system. The new system will replace the existing system completely.

Smart Scheduling is developed as a web application, as usual some web development technologies are used in the implementation of the project. The language selected to program this software ASP.net with C#. The reason for selecting C#; is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

Programming language - C#

Database - SQL Server 2008

Development tool - Visual Studio 2010

Development platform - Windows 8

**CHAPTER-7**

**COnclusion**

**7. Conclusion**

To conclude the description about the project the project developed using ASP.net with C# and SQL SERVER 2008 is based on the requirement specification of the user and the analysis of the existing system, with flexibility for future enhancement. This will help the site managers to manage and search the resources easily and effectively. It also provides the facility to rent equipments. The registered companies can participate in the auction process. Handing resources manually is impractical so we have to develop a system which will easily schedule, track and maintain the equipments. And also it will eliminate the guesswork of tedious scheduling and routing to maximize on-time delivery. There will be an administrator login which is accessed by the admin who has full permission. Admin can put equipments for auction and the registered companies can participate in the bidding process. The booking process is fully automated.

**CHAPTER-8**

**REFERENCES**

**8. REFERENCES**

**8.1. WEBSITE**

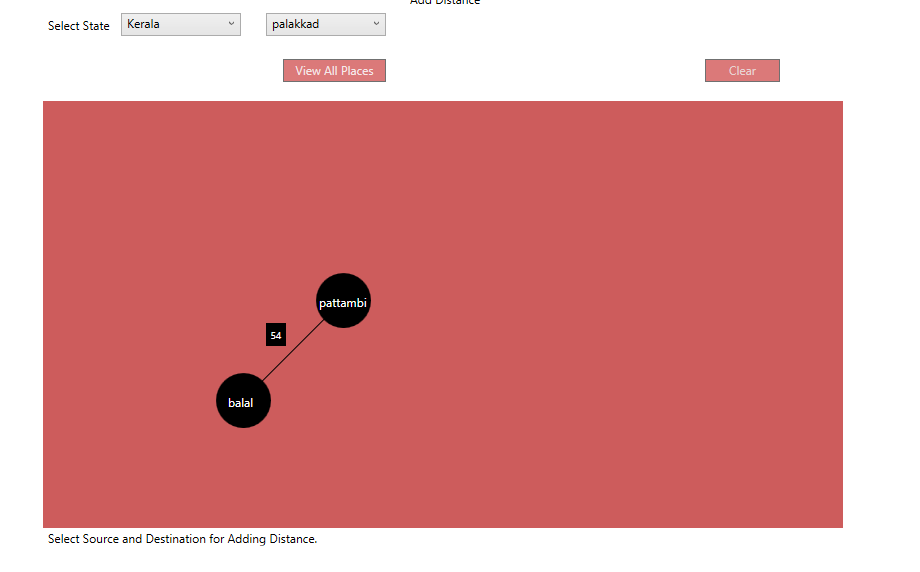
* http://www.Pinterest.com
* http://www.athemes.com

**8.2. REFERENCE BOOKS**

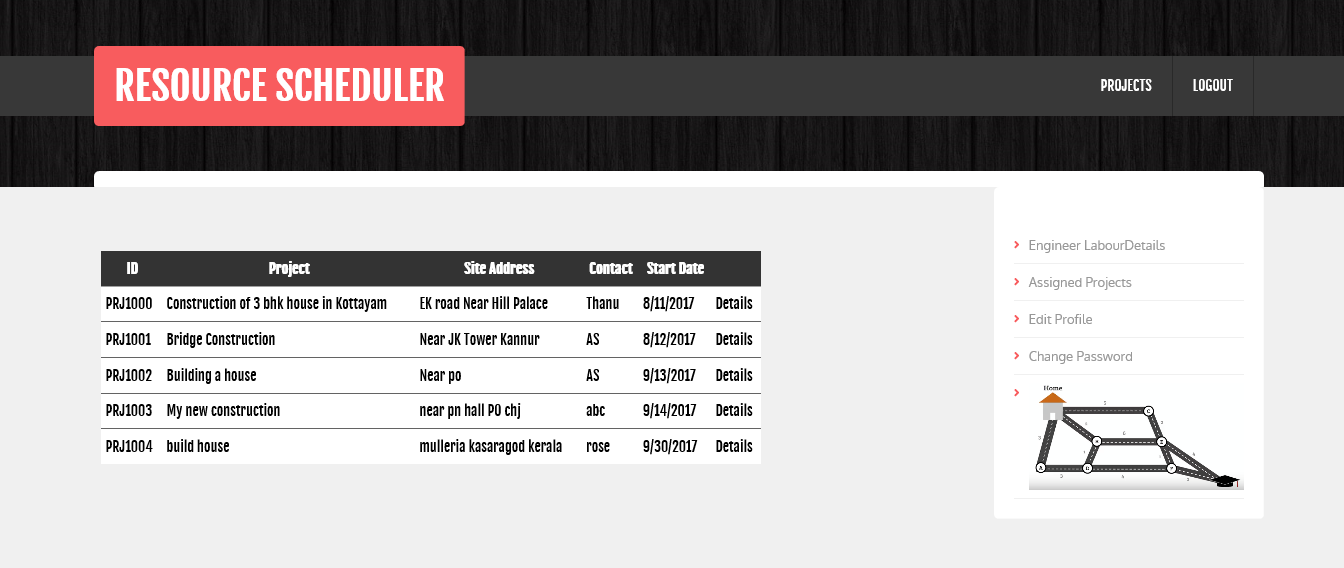
* Roger. S. Pressman, ‘Software Engineering’, Tata McGraw Hill, Fifth Edition, 2004.
* Kathleen Kalata, ‘Introduction to ASP.NET’,2nd Edition

**APPENDIX**

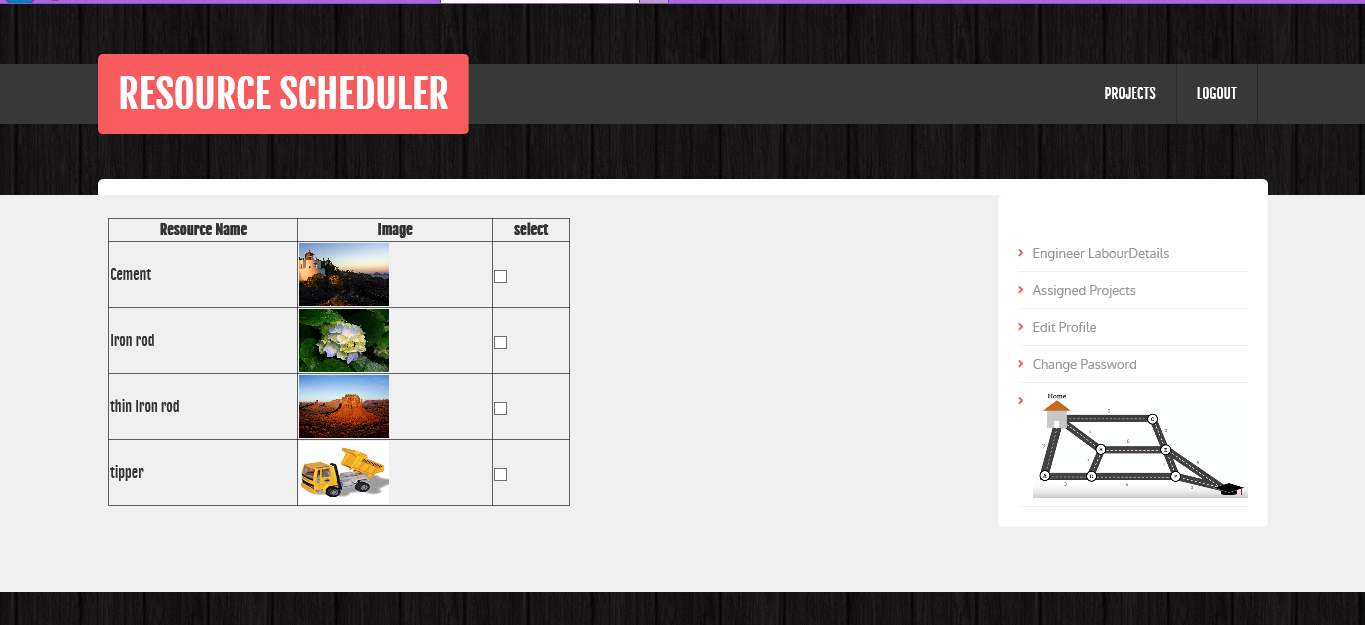
1. Form for view the distance between source and destination.



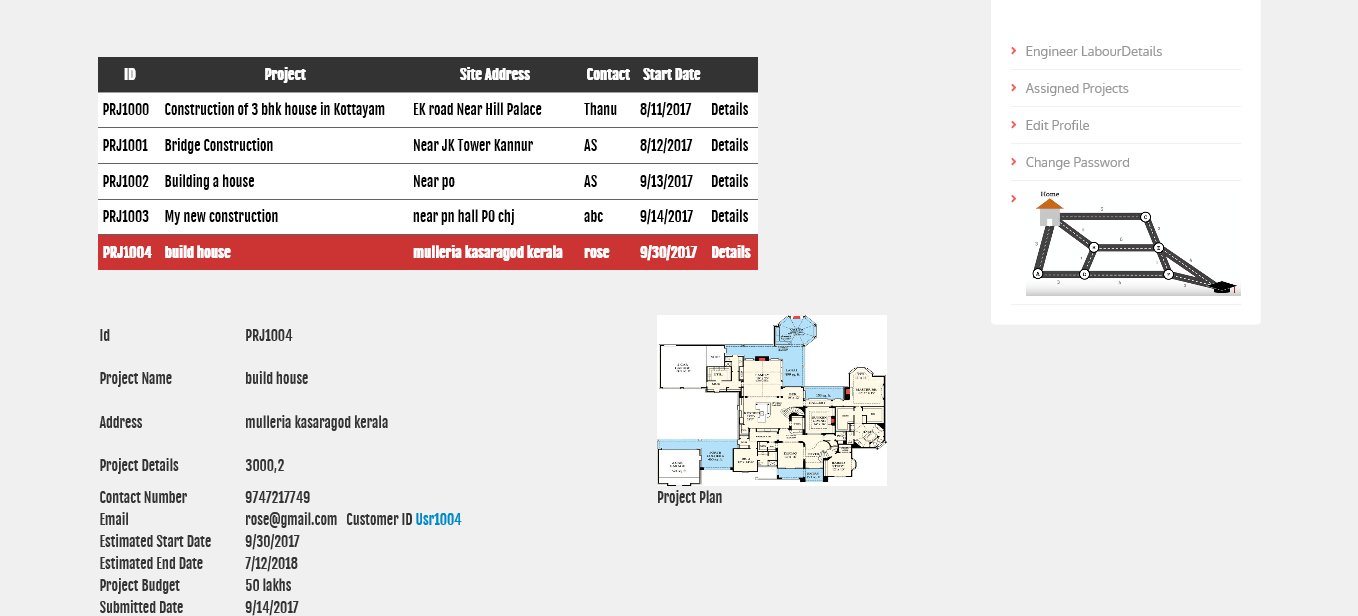
1. Site engineer view projects



1. Form for site engineer add resources for the project



1. form for site engineer view detailes of project.



1. form for site engineer view detailes of site.