* 1. **OBJECTIVE**

The aim of the project Distributor Retailer Interactive System is to develop the order processing system and inventory management system for any distributor associated with a set of retailers. Application software is needed to manage effectively requests for orders and their details and to support the business flow. The objectives of system are to provide help in reducing paper work, automation of the business process and processing of order details faster and more accurately. It also aims at improving the services and sales by using the proposed interactive system.

Distributor Retailer Interactive System is important to ensure quality control in businesses that handle transactions revolving around consumer goods. Without proper inventory control and processing of orders, a large retail store may run out of stock on an important item. A good Interactive System will alert the retailer when it is time to reorder. This system is also an important means of automatically tracking large shipments.

The main objective of Distributor Retailer Interactive System is to maintain inventory at an appropriate level by interacting constantly with the retailers and to keep a track of the retailers to avoid excessive or shortage of inventory because both the cases are undesirable for any business. Thus, management is faced with the following conflicting objectives:

1. To keep inventory at sufficiently high level to perform production and sales activities smoothly.
2. To minimize investment in inventory at minimum level to maximize profitability.
3. To keep a track of the retailers so that the distributor is aware of the purchase trends of the retailer and provide any offers or schemes for the same.
4. To minimize the cost of maintaining a manual system and to reduce human labor.
5. To ensure that the supply and distribution of products will remain continuous both at the sender end and at the retailer end so that the processing of orders is not halted and the demands of the retailer are duly met.
6. To reduce the losses of theft, obsolescence and wastage etc.
7. To make some arrangement for the sale of slow moving items.
8. To protect the system against stock-out.
9. To achieve satisfactory level of customer service while keeping the maintenance cost within reasonable bounds.
10. To provide an online transaction system for the payments involved in processing of orders.
11. To provide an easy interface to the users so that they can easily access the system.
12. To connect the whole system to e-mail so that important messages can be sent as and when required.

The proposed system is capable to fulfill these objectives and thus provide a better means to keep a check on the stock as well as interaction related to the retailers.

* 1. **PROJECT DESCRIPTION**

The Distributor Retailer Interactive System is a web based solution on PHP platform to ease the functioning between a distributor and a retailer. It is a solution for small or mid-sized distributors to keep a track of their retailers and it is also used for inventory control and placing of orders by the retailers.

The system would enable the distributors to interact with the retailers with whom they are in a business relation as well as to attract new retailers who want an automated system for placing orders and reduce the tedious manual labor. The distributor has the access privilege to add a new retailer to its system.

Both the distributor and the retailer need to login to access the system. This helps in the authentication process and also ensures that access is given only to the authenticated users. The distributor has the right to associate with a set of retailers with whom the distributor wants to associate in a business relation by adding their details and sending requests to them through e-mail.

The distributor has a different interface as compared to the retailer. At the distributor’s interface, the provision for adding and editing retailers, adding and editing products, viewing reports, generating invoices, viewing and generating estimates and sending requests for invitation or general messages is provided.

At the retailer’s interface, viewing invoices and paying them, changing their custom settings and viewing the status of their order as well as generating queries to the distributor and sending general messages are provided.

Both the distributor and the retailer need to log out once they have accessed the system so that security of the system is not breached.

**2.1. EXISTING AND PROPOSED SYSTEM**

The existing system worked as follows:

The retailers use MS Excel, and maintain their records; however it is not possible for them to share the data from multiple systems in multi user environment. There is lot of duplicate work, and chances of mistake. When the records are changed they need to update each and every excel file. There is no option to find and print previously saved records. There is no security; anybody can access any report and sensitive data, also no reports to summary report. This Distributor Retailer Interactive System is used to overcome the entire problem which they are facing currently, and making complete atomization of manual system to computerized system.

The problems associated with the existing system are:

* The existing system only provides text-based interface, which is not as user-friendly as Graphical User Interface.
* Since the system is implemented in manual, so the response is very slow.
* The transactions are executed in off-line mode, hence on-line data capture and modification is not possible.
* Off-line reports cannot be generated due to batch mode execution.

Hence, there is a need of reformation of the system with more advantages and flexibility. The Distributor Retailer Interactive System eliminates most of the limitations of the existing software. It has the following objectives:

* **Enhancement**

The main objective of Distributor Retailer Interactive System is to enhance and upgrade the existing system by increasing its efficiency and effectiveness. The software improves the working methods by replacing the existing manual system with the computer-based system.

* **Automation**

The Distributor Retailer Interactive System automates each and every activity of the manual system and increases its throughput. Thus the response time of the system is very less and it works very fast.

* **Accuracy**

The Distributor Retailer Interactive System provides the user a quick response with very accurate information regarding the users etc.

* **User-Friendly**

The Distributor Retailer Interactive System has a very user-friendly interface. Thus the users will find it very easy to work on it. The system provides accuracy along with a pleasant interface. It also makes the present manual system more interactive, speedy and user friendly.

* **Availability**

The transaction reports of the system can be retrieved as and when required. Thus, there is no delay in the availability of any information, whatever needed, can be captured very quickly and easily.

* **Maintenance Cost**

Reduce the cost of maintenance.

**2.2. FEASIBILITY STUDY**

The proposed Distributor Retailer Interactive System is much more feasible over the exiting traditional system of maintaining the inventory system and the order processing system manually or through spreadsheets by the following factors:

* **Attractive Dashboard**

The dashboard of both the retailer and the distributor are very attractive so as to make it easy for the users to use it. The features of the dashboard include:

* Yearly sales report with graph chart
* Total revenue, cost, tax details
* Latest order details
* Recently added Product
* Store summary eg: Total Product, Order, Invoice, Customer
* **Purchase Manager**

The purchasing module of the system is interactive and provides the following features:

* Add new product
* Manage all products
* Purchase product from supplier and provide to retailers
* Purchase invoice generate
* Retailer history with purchase record
* **Customer Management**

The customer management module of the system is interactive and provides the following features to give an easy and handy user interface:

* Add new retailer
* Manage your all retailer
* Customer discount setup
* Email campaign for customer
* Send invoice to customer by email
* **Product Management**

The product management module of the system provides the following features:

* Create product category/subcategory
* Add product in store
* Product general rate, special offer, tier price
* Manage product inventory
* Alert notification for product running out of stock
* Product attribute and tag
* Product barcode generate facility
* Product barcode print
* Manage damage product
* Damage product will auto deduct from inventory
* **Order Management**

The order management module of the system provides the following features:

* Add new order
* You can select retailer for new order procedure
* If the retailer has discount it will apply
* Auto tax calculation
* Retailer shipping rate
* You can make order pending with late payment
* Order and invoice number generate
* PDF, print and email invoice facility
* **Email Campaign**

Associating e-mail with the system provides the following features:

* Email campaign for your store
* Create email template
* Manage email campaign template
* Send email to all your retailers
* Email campaign report

**2.3. TOOLS AND TECHNOLOGIES USED**

The Tools and Technologies which are used in this project are:

* **PHP 5.4**

PHP is a [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting) language designed for [web development](https://en.wikipedia.org/wiki/Web_development) but also used as a [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). PHP originally stood for Personal Home Page, but it now stands for the [recursive](https://en.wikipedia.org/wiki/Recursive_acronym) acronym PHP: Hypertext Preprocessor. Fast, flexible and pragmatic, PHP powers everything from blogs to the most popular websites in the world. PHP code may be embedded into [HTML](https://en.wikipedia.org/wiki/HTML) code, or it can be used in combination with various [web template systems](https://en.wikipedia.org/wiki/Web_template_system), web content management system and [web frameworks](https://en.wikipedia.org/wiki/Web_framework). PHP code is usually processed by a PHP [interpreter](https://en.wikipedia.org/wiki/Interpreter_(computing)) implemented as a [module](https://en.wikipedia.org/wiki/Plugin_(computing)) in the web server or as a [Common Gateway Interface](https://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a [command-line interface](https://en.wikipedia.org/wiki/Command-line_interface) (CLI) and can be used to implement [standalone](https://en.wikipedia.org/wiki/Computer_software) [graphical applications](https://en.wikipedia.org/wiki/Graphical_user_interface).

* **MySQL**

MySQL is an [open-source](https://en.wikipedia.org/wiki/Open-source) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS). MySQL is a popular choice of database for use in web applications, and is a central component of the widely used [LAMP](https://en.wikipedia.org/wiki/LAMP_(software_bundle)) open-source web application software stack. LAMP is an acronym for "[Linux](https://en.wikipedia.org/wiki/Linux), [Apache](https://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, [Perl](https://en.wikipedia.org/wiki/Perl)/ [PHP](https://en.wikipedia.org/wiki/PHP)/ [Python](https://en.wikipedia.org/wiki/Python_(programming_language))". [Free-software](https://en.wikipedia.org/wiki/Free_software) open-source projects that require a full-featured database management system often use MySQL. MySQL is also used in many high-profile, large-scale [websites](https://en.wikipedia.org/wiki/Website), including [Google](https://en.wikipedia.org/wiki/Google) (though not for searches), [Facebook](https://en.wikipedia.org/wiki/Facebook), [Twitter](https://en.wikipedia.org/wiki/Twitter), [Flickr](https://en.wikipedia.org/wiki/Flickr), and [YouTube](https://en.wikipedia.org/wiki/YouTube).

* **Laravel 5.1 framework of PHP**

Laravel is a free, [open-source](https://en.wikipedia.org/wiki/Open-source) PHP [web framework](https://en.wikipedia.org/wiki/Web_framework), created by Taylor Otwell and intended for the development of web applications following the [model–view–controller](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller) (MVC) [architectural pattern](https://en.wikipedia.org/wiki/Architectural_pattern). Some of the features of Laravel are a modular [packaging system](https://en.wikipedia.org/wiki/Application-level_package_manager) with a dedicated dependency manager, different ways for accessing [relational databases](https://en.wikipedia.org/wiki/Relational_database), utilities that aid in [application deployment](https://en.wikipedia.org/wiki/Application_deployment) and maintenance, and its orientation toward [syntactic sugar](https://en.wikipedia.org/wiki/Syntactic_sugar). The [source code](https://en.wikipedia.org/wiki/Source_code) of Laravel is hosted on [GitHub](https://en.wikipedia.org/wiki/GitHub" \o "GitHub) and licensed under the terms of [MIT License](https://en.wikipedia.org/wiki/MIT_License).

Laravel 5.1, released in June 2015, is the first release of Laravel to receive [long-term support](https://en.wikipedia.org/wiki/Long-term_support) (LTS), with planned availability of [bug fixes](https://en.wikipedia.org/wiki/Bug_fix) for two years and [security patches](https://en.wikipedia.org/wiki/Security_patch) for three years.

* **HTML**

HyperText Markup Language, commonly abbreviated as HTML, is the standard [markup language](https://en.wikipedia.org/wiki/Markup_language) used to create [web pages](https://en.wikipedia.org/wiki/Web_page). Along with [CSS](https://en.wikipedia.org/wiki/Cascading_Style_Sheets), and [JavaScript](https://en.wikipedia.org/wiki/JavaScript), HTML is a cornerstone technology used to create web pages, as well as to create user interfaces for mobile and [web applications](https://en.wikipedia.org/wiki/Web_applications). [Web browsers](https://en.wikipedia.org/wiki/Web_browser) can read HTML files and render them into visible or audible web pages. HTML describes the structure of a [website](https://en.wikipedia.org/wiki/Website) [semantically](https://en.wikipedia.org/wiki/Semantic) and, before the advent of Cascading Style Sheets (CSS), included cues for the presentation or appearance of the document (web page), making it a markup language, rather than a [programming language](https://en.wikipedia.org/wiki/Programming_language). HTML can embed [scripts](https://en.wikipedia.org/wiki/Scripting_language) written in languages such as [JavaScript](https://en.wikipedia.org/wiki/JavaScript) which affect the behavior of HTML web pages. HTML markup can also refer the browser to [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS) to define the look and layout of text and other material.

* **jQuery**

jQuery is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [JavaScript library](https://en.wikipedia.org/wiki/JavaScript_library) designed to simplify the [client-side scripting](https://en.wikipedia.org/wiki/Client-side_scripting) of HTML. jQuery is the most popular [JavaScript library](https://en.wikipedia.org/wiki/JavaScript_library) in use today, with installation on 65% of the top 10 million highest-trafficked sites on the Web. jQuery is [free, open-source software](https://en.wikipedia.org/wiki/Free_and_open_source_software) licensed under the [MIT License](https://en.wikipedia.org/wiki/MIT_License). jQuery's syntax is designed to make it easier to navigate a document, select [DOM](https://en.wikipedia.org/wiki/Document_Object_Model) elements, create [animations](https://en.wikipedia.org/wiki/Animation), handle [events](https://en.wikipedia.org/wiki/Event_(computing)), and develop [Ajax](https://en.wikipedia.org/wiki/Ajax_(programming)) applications. jQuery also provides capabilities for developers to create [plug-ins](https://en.wikipedia.org/wiki/Plug-in_(computing)) on top of the JavaScript library. This enables developers to create [abstractions](https://en.wikipedia.org/wiki/Abstraction_(computer_science)) 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](https://en.wikipedia.org/wiki/Dynamic_web_page) and Web applications.

* **phpMyAdmin**

phpMyAdmin is a [free and open source](https://en.wikipedia.org/wiki/Free_and_open_source) tool written in [PHP](https://en.wikipedia.org/wiki/PHP) intended to handle the administration of [MySQL](https://en.wikipedia.org/wiki/MySQL) or [MariaDB](https://en.wikipedia.org/wiki/MariaDB" \o "MariaDB) with the use of a [web browser](https://en.wikipedia.org/wiki/Web_browser). It can perform various tasks such as creating, modifying or deleting [databases](https://en.wikipedia.org/wiki/Database), [tables](https://en.wikipedia.org/wiki/Table_(database)), [fields](https://en.wikipedia.org/wiki/Field_(computer_science)) or [rows](https://en.wikipedia.org/wiki/Row_(database)); executing [SQL](https://en.wikipedia.org/wiki/SQL) statements; or managing users and permissions.

**2.4. HARDWARE AND SOFTWARE REQUIREMENTS**

The Hardware Requirement for the project are:

* RAM - 1GB or more
* Hard Disk - 20GB or more
* SVGA Display - 1300x768
* Processor - Dual Core Processor
* Network connection (Ethernet cable or LAN connection)

The Software Requirement for the project are:

* Front End: PHP 5.4 or above
* Back End: Database in MySQL
* Operating System: Windows Operating System of 64 bit
* Framework: Laravel 5.1
* Web Technologies: HTML, jQuery
* Documentation Tool: Microsoft Word
* Server: Wamp Server 64 bit

**3.1. USERS**

A system cannot be designed in isolation without the active involvement of the user.There are 2 types of users in Distributor Retailer Interactive System: Distributor and Retailer. Both the distributor and the retailer need to login to access the system.

The distributor has a different interface as compared to the retailer. The distributor has the facility for adding and editing retailers, adding and editing products, viewing reports, generating invoices, viewing and generating estimates and sending requests for invitation or general messages is provided.

The retailer avails the facility of viewing invoices and paying them, changing their custom settings and viewing the status of their order as well as generating queries to the distributor and sending general messages are provided.

The constant interaction between the distributor and the retailer becomes the driving force for the system. The distributor has the right or access to include the retailers with whom it wants to interact in order to set up a business relation. There can be several retailers related to a single distributor. All the products available with the distributor can be ordered by the retailers and the distributor needs to keep a constant check over its inventory so that the products don’t run out of stock.

These two type of users act as the base to provide automation to the existing system involving the use of human labor or excel sheets.

**3.2. FUNCTIONAL REQUIREMENTS**

The functional requirements of the system include:

* The System aims at providing an efficient interface to the user for managing of inventory and processing orders, it shall also provide the user varied options for managing the inventory through various functions at hand.
* The product levels are continuously monitored based on their usage and are checked for the threshold levels in the inventory and accordingly the user is alerted about low levels of certain products.
* The design is such that the user does not have to manually update the system every time, whenever a new retailer or a new product is added.
* The System calculates and predicts the amount of usage for specific set days that are pre-set by the distributor, it also alerts the user of an impending action to order products before the specific day set by the user. Therefore the user never has to worry about manually calculating the estimated usage of the products as the System does it for the user.
* The simple interface of the System has functions like adding a retailer or product, removing or updating the retailer or product.

**3.3. NON-FUNCTIONAL REQUIREMENTS**

There are several non-functional requirements such as usability, reliability, performance, supportability, implementation and interfacing related to the system.

**3.3.1. Usability**

* The system must be easy to use by both distributors and retailers such that they do not need to read an extensive amount of manuals.
* The system must be quickly accessible by both distributors and retailers.
* The system must be intuitive and simple in the way it displays all relevant data and relationships.
* The modules of the system must be easily navigable by the users with buttons that are easy to understand.

**3.3.2. Reliability**

* The System must give accurate inventory status to the user continuously. Any inaccuracies are taken care by the regular confirming of the actual levels with the levels displayed in the system.
* The System must successfully add any new retailers or products given by the distributor and provide estimations and inventory status in relevance with the newly updated entities.
* The system must provide a password enabled login to the user to avoid any foreign entity changing the data in the system.
* The system should provide the user updates on completion of requested transactions and if the requested transactions fail, it should provide the user the reason for the failure.
* The system should not update the data in any database for any failed transaction.

**3.3.3. Performance**

* The system must not lag, because the users using it don’t have down-time to wait for it to complete an action.
* The system must complete updating the databases, successfully every time the user requests such an action.
* All the functions of the system must be available to the user every time the system is turned on.
* The calculations performed by the system must comply according to the norms set by the user and should not vary unless explicitly changed by the user.

**3.3.4. Supportability**

* The software is designed such that it works even on systems having the minimum configuration.
* The system is adaptable even if additional plug-ins or modules are added at a later point.

**3.3.5. Implementation**

* The System User Interface is built on Laravel framework of PHP.
* The Programming is done using PHP and jQuery.
* The Database is implemented on PHPMyAdmin.

**3.3.6. Interfacing**

* The system must offer an easy and simple way of viewing the current inventory.
* The system must be able to display the relationships between retailers, products and invoices in an intuitive manner.

**4.1. SYSTEM PERSPECTIVE**

The systems objectives outlined during the feasibility study serve as the basic from which the work of system design is initiated. Much of the activities involved at this stage is of technical nature requiring a certain degree of experience in designing systems, sound knowledge of computer related technology and through understanding of computers available in the market and the various facilities provided by the vendors.

Nevertheless, a system cannot be designed in isolation without the active involvement of the user. The user has a vital role to play at this stage too. As we know that data collected during feasibility study will be utilized systematically during the system design. It should, however be kept in mind that detailed study of the existing system is not necessarily over with the completion of the feasibility study. Depending on the plan of feasibility study, the level of detailed study will vary and the system design stage will also vary in the amount of investigation that still needs to be done. This investigation is generally an urgent activity during the system. Sometimes, but rarely, this investigation may form a separate stage between feasibility study and computer system design.

Designing a new system is a creative process, which calls for logical as well as lateral thinking. The logical approach involves systematic moves towards the end product keeping in mind the capabilities of the personnel and the equipment at each decision making step. Lateral thought implies encompassing of ideas beyond the usual functions and equipment. This is to ensure that no efforts are being made to fit previous solutions into new situations.

The system design process is not a step-by-step adherence of clear procedures and guidelines. Though, certain clear procedures and guidelines have emerged in recent days, but still much of design work depends on knowledge and experience of the designer. When designer starts working on system design, he/she will face different type of problems. Many of these will be due to constraints imposed by the user or limitations of the hardware and software available in the market. Sometimes, it is difficult to enumerate the complexity of the problems and solutions thereof since the variety of likely problems is so great and no solutions are exactly similar. However, following considerations should be kept in mind during the system designing phase:

* **Practicality**

The system must be stable and can be operated by people with average.

* **Efficiency**

This involves accuracy, timeliness and comprehensiveness to the system output.

* **Cost**

It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy all the requirements.

* **Flexibility**

The system should be modifiable depending on the changing needs of the user. Such modifications should not entail extensive reconstructing or recreation of software. It should also be portable to different computer systems.

* **Security**

This is very important aspect of the design and should cover areas of hardware reliability, fall back procedures, physical security of data and provision for detection of fraud and abuse.

The major system design activities include:

* **Database design**

This activity deals with the design of the physical database. A key is to determine how the access paths are to be implemented.

* **Design Process**

The computer system design process is an exercise of specifying how, the system will work. It is an iterative process, which is based on what the system will be do as shown in the feasibility report.

* **Output Design**

The starting point of the design process is the proper knowledge of system requirements which will normally be converted in terms of output.

* **Input Design**

Once the output requirements have been finalized, the next step is to find out what data need to be made available to the system to produce the desired outputs.

* **File Design**

Once the input data is captured in the system, these may to be preserved either for a short or long period. These data will generally be stored in files in a logical manner. The designer will have to devise the techniques of storing and retrieving data from these files.

* **Procedure Design**

This step involves specifications of how processing will be performed

* **Control Design**

The control design indicates necessary procedures which will ensure correctness of processing, accuracy of data, timely output etc. this will ensure that the system is functioning as per plan.

**4.2. DATABASE DESIGN**

Database named ‘inventory’ is created using PHPMyAdmin. The different tables involved in the database are as follows:

**Table 4.1: alerts Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| products | int(1) |

**Table 4.2: client Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| name | varchar(255) |
| country | varchar(255) |
| state | varchar(255) |
| city | varchar(255) |
| zip | varchar(255) |
| address | varchar(255) |
| contact | varchar(255) |
| phone | varchar(255) |
| email | varchar(255) |
| website | varchar(255) |
| bank | varchar(255) |
| bank\_account | varchar(255) |
| description | text |
| updated\_at | date |
| created\_at | date |

**Table 4.3: currencies Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| name | varchar(100) |
| position | tinyint(1) |

**Table 4.4: estimates Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| client\_id | int(11) |
| currency\_id | int(11) |
| estimate | varchar(100) |
| reference | varchar(100) |
| amount | double(10,2) |
| discount | double(10,2) |
| type | tinyint(1) |
| start\_date | date |
| due\_date | date |
| description | text |
| terms | text |
| status | tinyint(1) |
| updated\_at | date |
| created\_at | date |

**Table 4.5: estimate\_products Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| estimate\_id | int(11) |
| product\_id | int(11) |
| quantity | int(11) |
| price | double(10,2) |
| tax | double(10,2) |
| discount | double(10,2) |
| discount\_type | tinyint(1) |
| discount\_value | double(10,2) |
| amount | double(10,2) |

**Table 4.6: format\_emails Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| client | text |

**Table 4.7: generals Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| version | varchar(10) |

**Table 4.8: images Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| name | varchar(255) |
| width | int(2) |
| height | int(2) |

**Table 4.9: invitations Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| client\_id | int(11) |
| status | tinyint(1) |

**Table 4.10: invitation\_settings Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| title | varchar(255) |
| content | text |

**Table 4.11: invoices Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| client\_id | int(11) |
| status\_id | tinyint(1) |
| currency\_id | int(11) |
| number | int(11) |
| amount | double(10,2) |
| discount | double(10,2) |
| type | tinyint(1) |
| start\_date | date |
| due\_date | date |
| description | text |
| terms | text |
| state | tinyint(1) |
| updated\_at | date |
| created\_at | date |

**Table 4.12: invoice\_payments Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| invoice\_id | int(11) |
| payment\_id | int(11) |
| payment\_date | date |
| payment\_amount | double(10,2) |

**Table 4.13: invoice\_products Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| invoice\_id | int(11) |
| product\_id | int(11) |
| quantity | int(11) |
| price | double(10,2) |
| tax | double(10,2) |
| discount | double(10,2) |
| discount\_type | tinyint(1) |
| discount\_value | double(10,2) |
| amount | double(10,2) |

**Table 4.14: invoice\_settings Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| number | int(11) |
| code | varchar(100) |
| text | text |

**Table 4.15: invoice\_statuses Tabl**e

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| name | varchar(100) |

**Table 4.16: languages Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| name | varchar(255) |
| short | varchar(100) |

**Table 4.17: language\_texts Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| name | text |
| content | text |
| type | tinyint(1) |

**Table 4.18: messages Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| from\_id | int(11) |
| parent\_id | int(11) |
| title | varchar(255) |
| content | text |
| status | tinyint(1) |
| state | tinyint(1) |
| updated\_at | date |
| created\_at | date |

**Table 4.19: payments Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| name | varchar(255) |

**Table 4.20: products Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| name | text |
| quantity | int(11) |
| code | varchar(255) |
| price | double(10,2) |
| description | Text |
| state | tinyint(1) |
| updated\_at | date |
| created\_at | date |

**Table 4.21: schedules Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| invoice\_id | int(11) |
| user\_id | int(11) |
| start\_date | int(1) |
| updated\_at | date |
| created\_at | date |

**Table 4.22: settings Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| currency\_id | int(11) |
| name | varchar(255) |
| country | varchar(255) |
| state | varchar(255) |
| city | varchar(255) |
| zip | varchar(255) |
| address | varchar(255) |
| contact | varchar(255) |
| phone | varchar(255) |
| email | varchar(255) |
| website | varchar(255) |
| bank | varchar(255) |
| bank\_account | varchar(255) |
| description | text |
| receive\_emails | varchar(255) |

**Table 4.23: taxes Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| user\_id | int(11) |
| value | double(10,2) |

**Table 4.24: users Table**

|  |  |
| --- | --- |
| **Name** | **Data Type** |
| id | int(11) |
| role\_id | int(11) |
| parent\_id | int(11) |
| language\_id | int(11) |
| name | varchar(200) |
| email | varchar(200) |
| password | varchar(64) |
| status | tinyint(1) |
| remember\_token | varchar(100) |
| updated\_at | date |
| created\_at | date |

The ER Diagram related to the above database is as follows:

id

title

content

Messages

send/replyy

send/replyy

id

id

id

name

name

email

email

name

quantity

id

amount

start\_date

view

Retailer

Invoice

Products

generates

order

adds

Distributor

**Fig 4.1: ER Diagram**

**4.3. CONTEXT DIAGRAM (DFD)**

The context diagram used in the system are divided in 2 levels of hierarchy:

Distributor

Retailer

Retailer

Retailer

Retailer

Distributor Retailer Interactive System

**Fig 4.2: Level 0 DFD**

Distributor

Retailer

Distributor Retailer Interactive System

Data Store

Login

Login

Process Order

View Status of Order

Inventory Update

Generate Invoice

**Fig 4.3: Level 1 DFD**

**4.4. USE CASE DIAGRAM**

The use case diagram for Distributor Retailer Interactive system is:



**Fig 4.4: Use Case Diagram**

**4.5. SEQUENCE DIAGRAMS**

There are a few sequences which determine the flow of process:

Start

Stop

Log out

Schedules

Languages

Settings

Messages

Reports

Dashboard

Estimates

Products

Clients

Invoices

Login

**Fig 4.5: Sequence diagram for distributor**

Start

Login

Dashboard

Estimates

Invoices

Settings

Messages

Stop

Log out

**Fig 4.6: Sequence diagram for retailer**

Start

Stop

Create new client

Update an existing client

Input data as required

Delete an existing client

**Fig 4.7: Sequence diagram for client information on distributor interface**

Start

Stop

Create new product

Update an existing product

Input data as required

Delete an existing product

**Fig 4.8: Sequence diagram for product information on distributor interface**

Start

Stop

Create new estimate

Update an existing estimate

Input data as required

Delete an existing estimate

**Fig 4.9: Sequence diagram for estimate information on distributor interface**

Start

Stop

Create new invoice

Update an existing invoice

Input data as required

Delete an existing invoice

**Fig 4.10: Sequence diagram for invoice information on distributor interface**

Start

Stop

Create new schedule

Update an existing schedule

Input data as required

Delete an existing invoice

**Fig 4.11: Sequence diagram for schedule information on distributor interface**

Start

Stop

Create new message

Delete an existing message

Input data as required

Update an existing message

**Fig 4.12: Sequence diagram for messages information on distributor interface**

**5.1. SCREENSHOTS**

Some important code snippets are as follows:

**Fig 5.1: Dashboard of Distributor**

**Fig 5.2: Dashboard of Client**

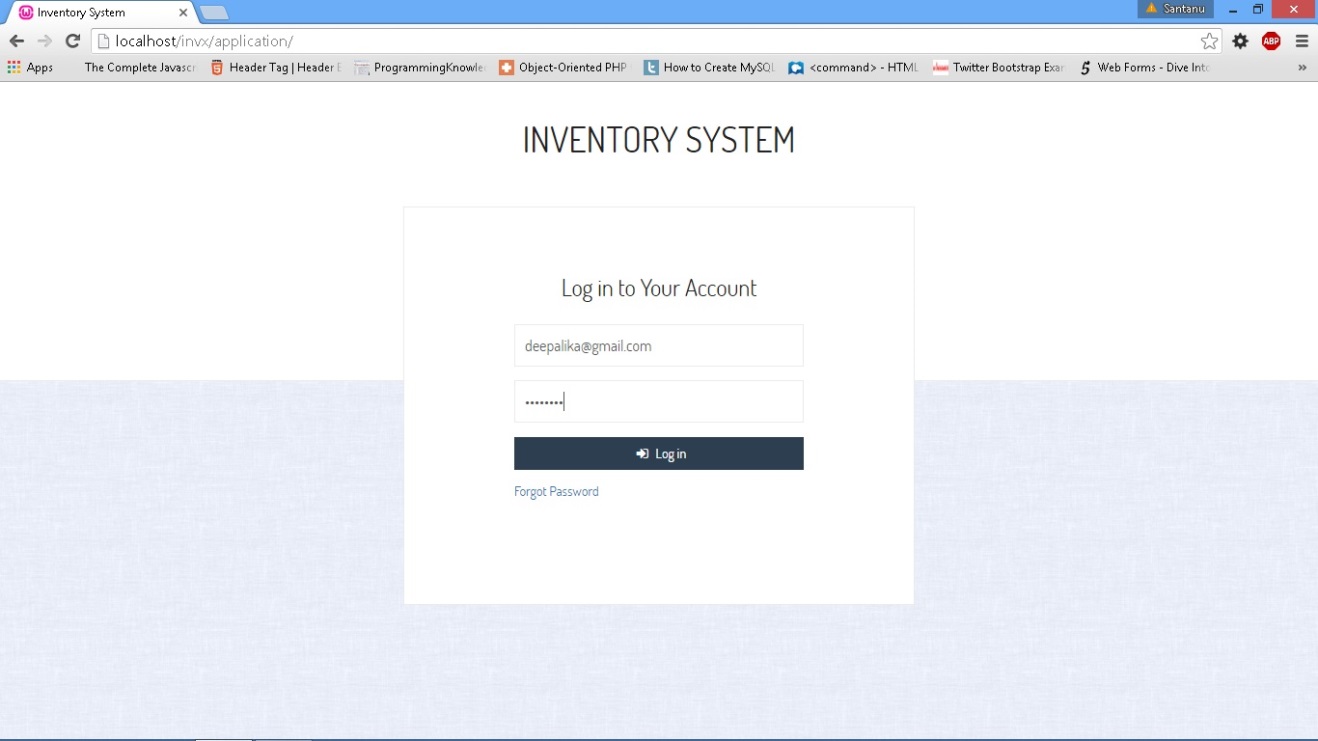
****

**Fig 5.3: Dashboard of Retailer information**

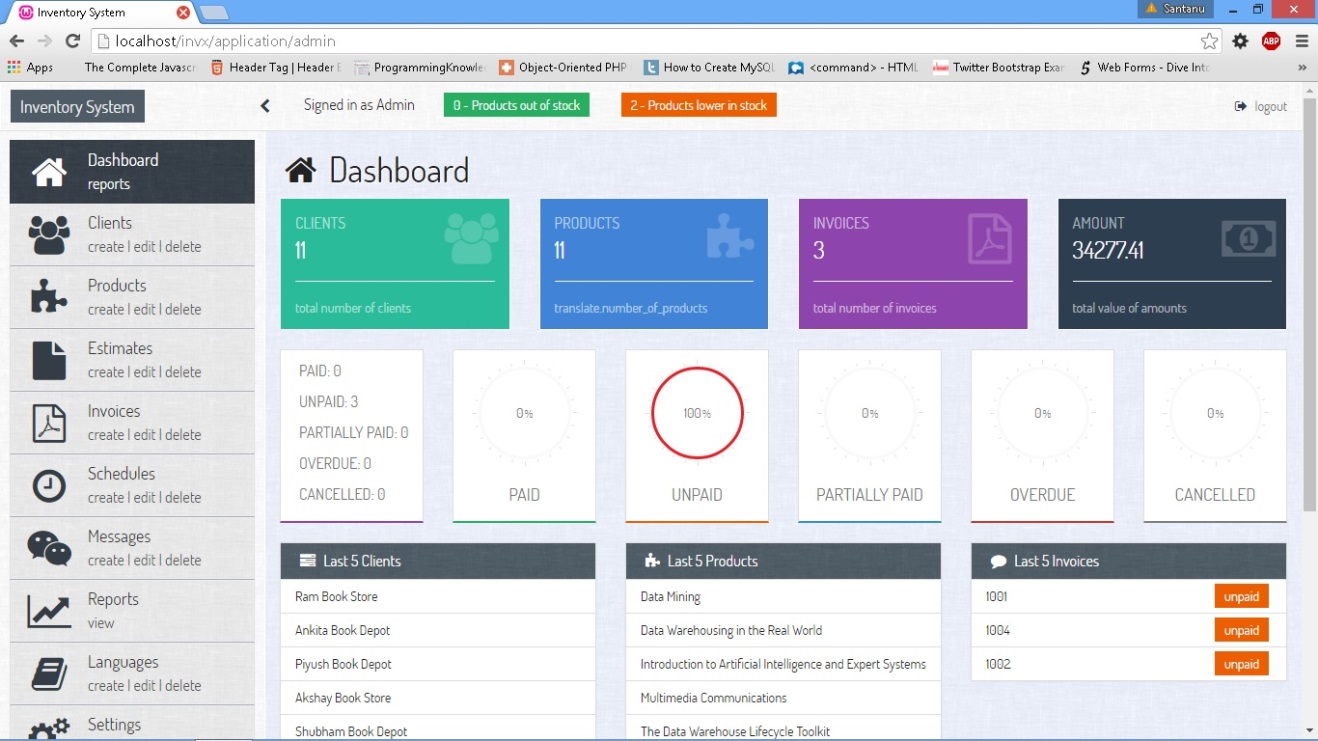
****

**Fig 5.4: Dashboard of Product information**

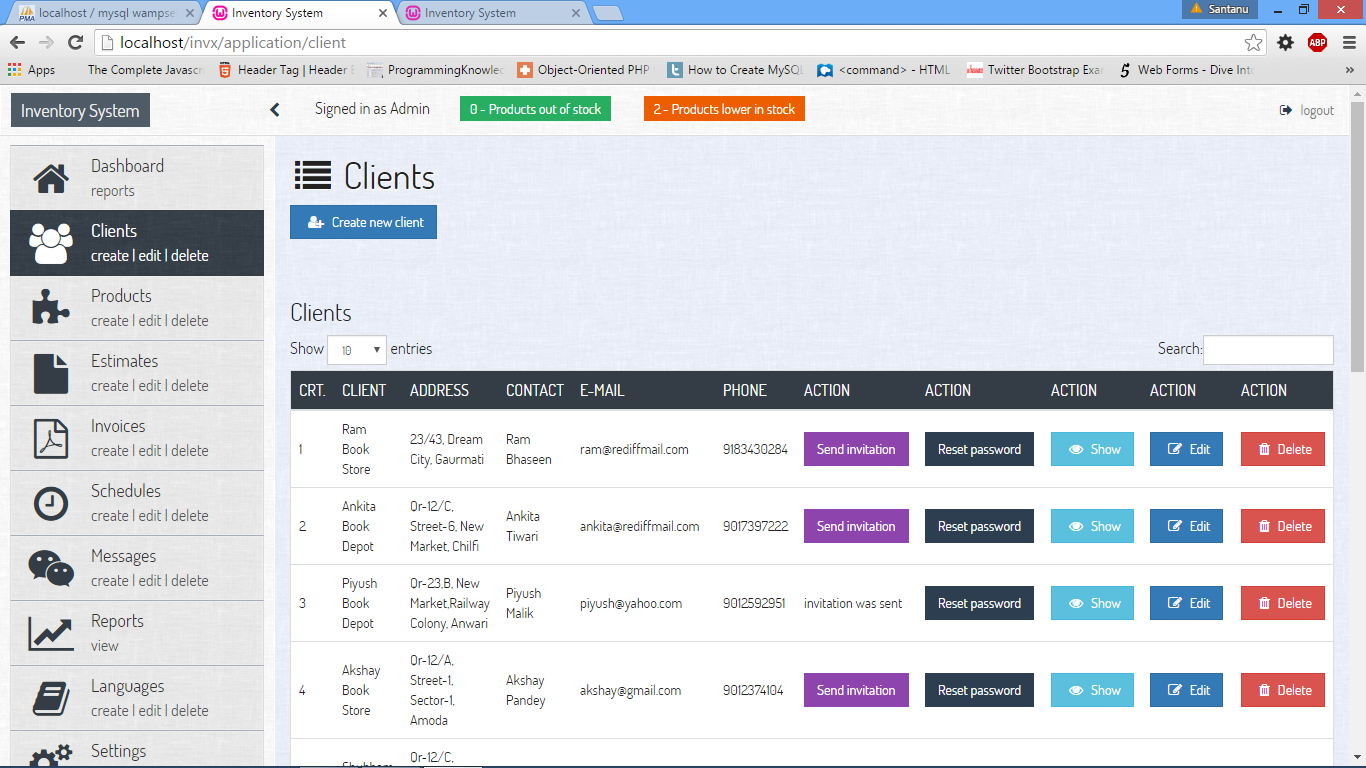
The resultant interfaces of the project are as follows:

****

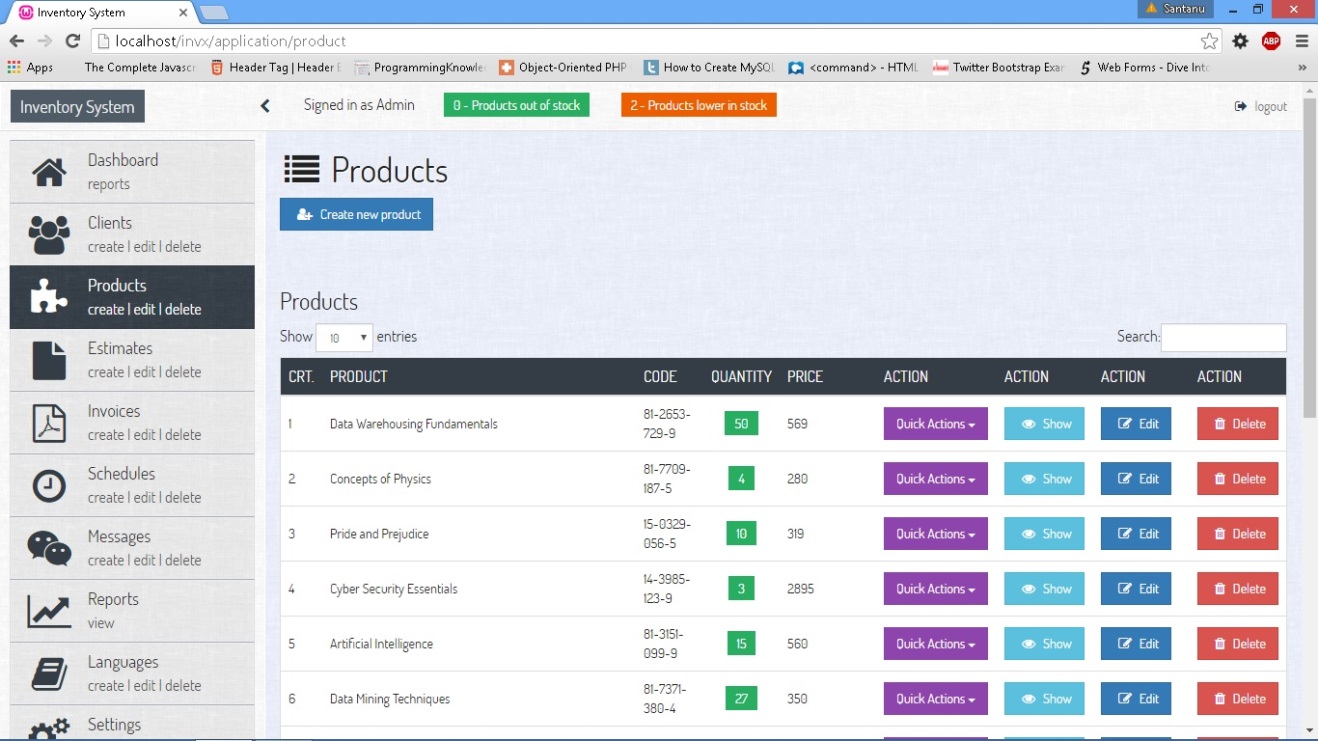
**Fig 5.5: Distributor Retailer Login Interface**

****

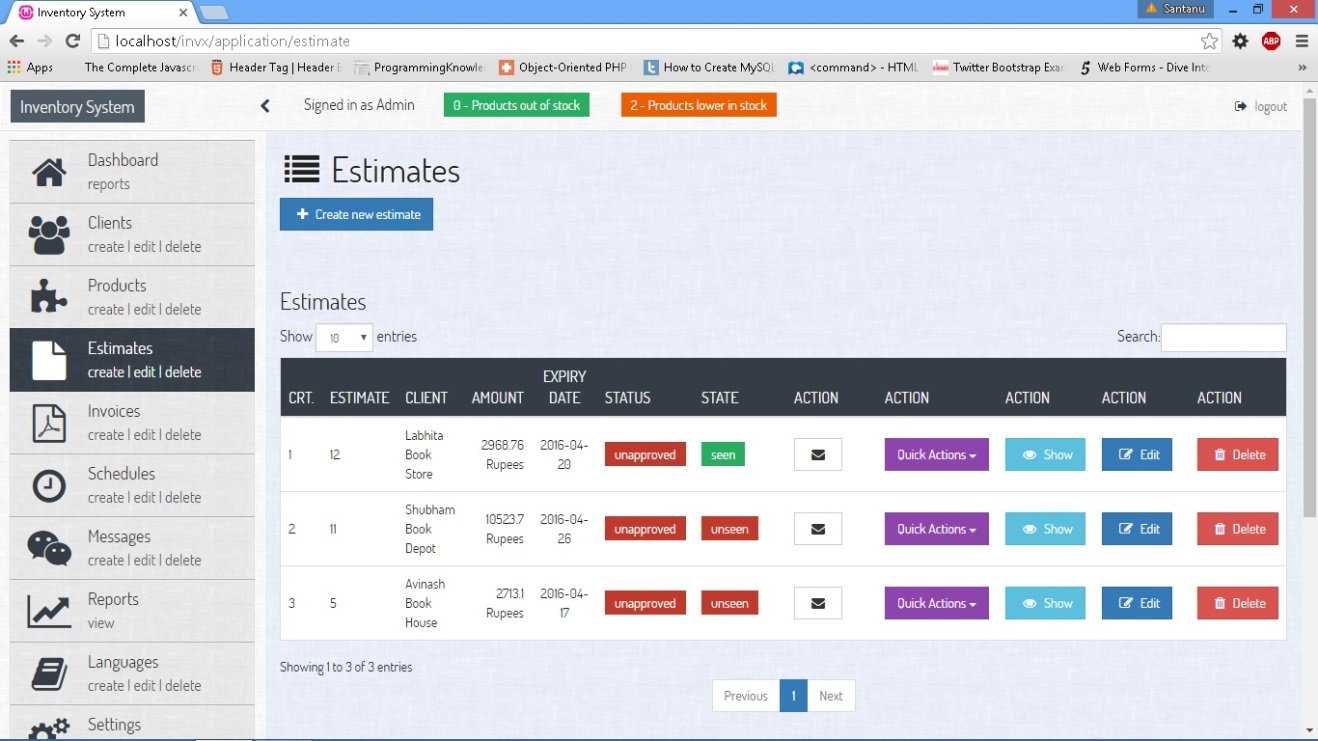
**Fig 5.6: Distributor Dashboard**

****

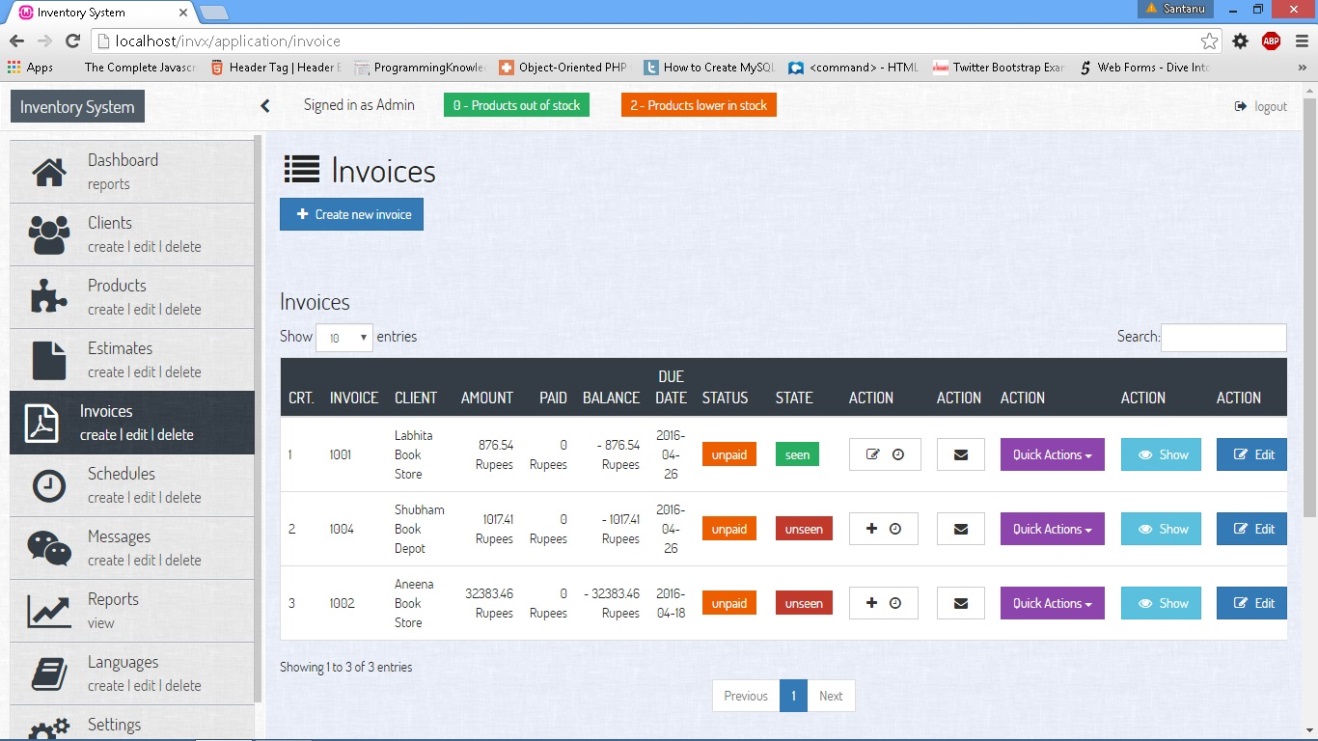
**Fig 5.7: Client Details on Distributor**

****

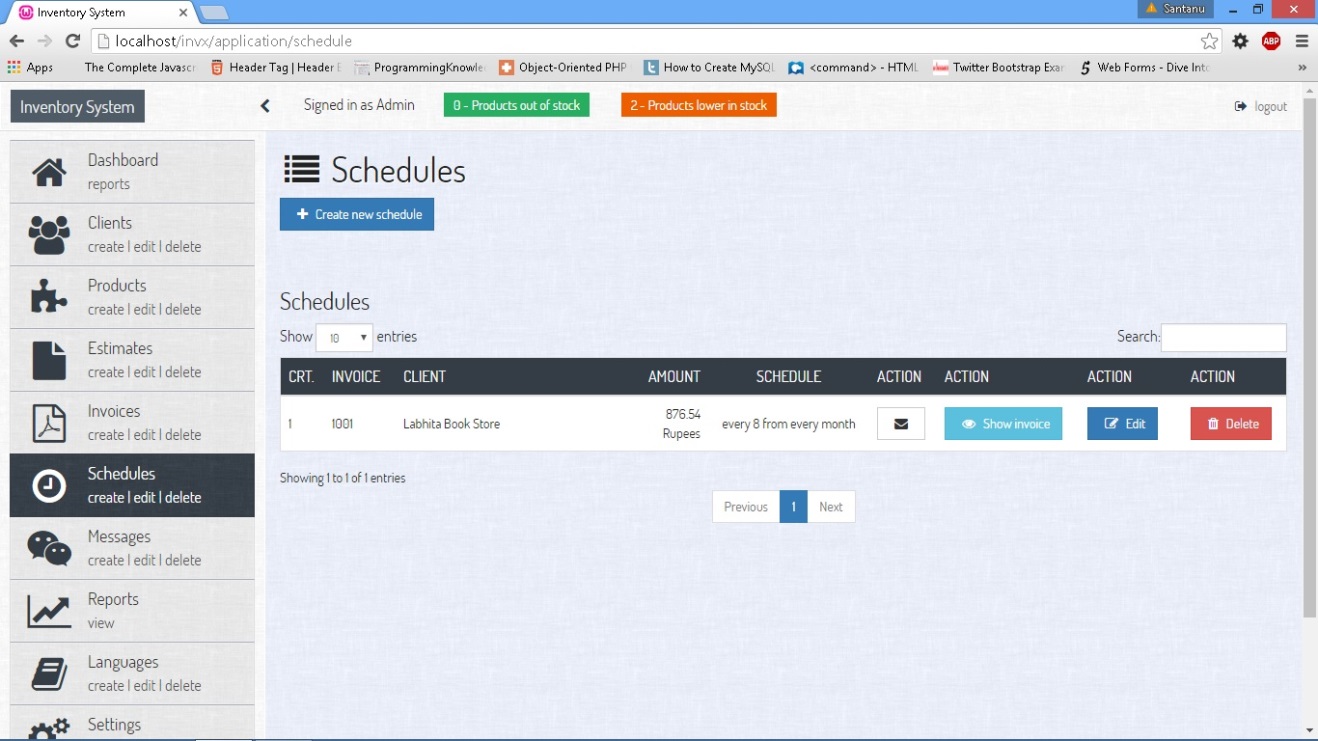
**Fig 5.8: Product Details on Distributor**

****

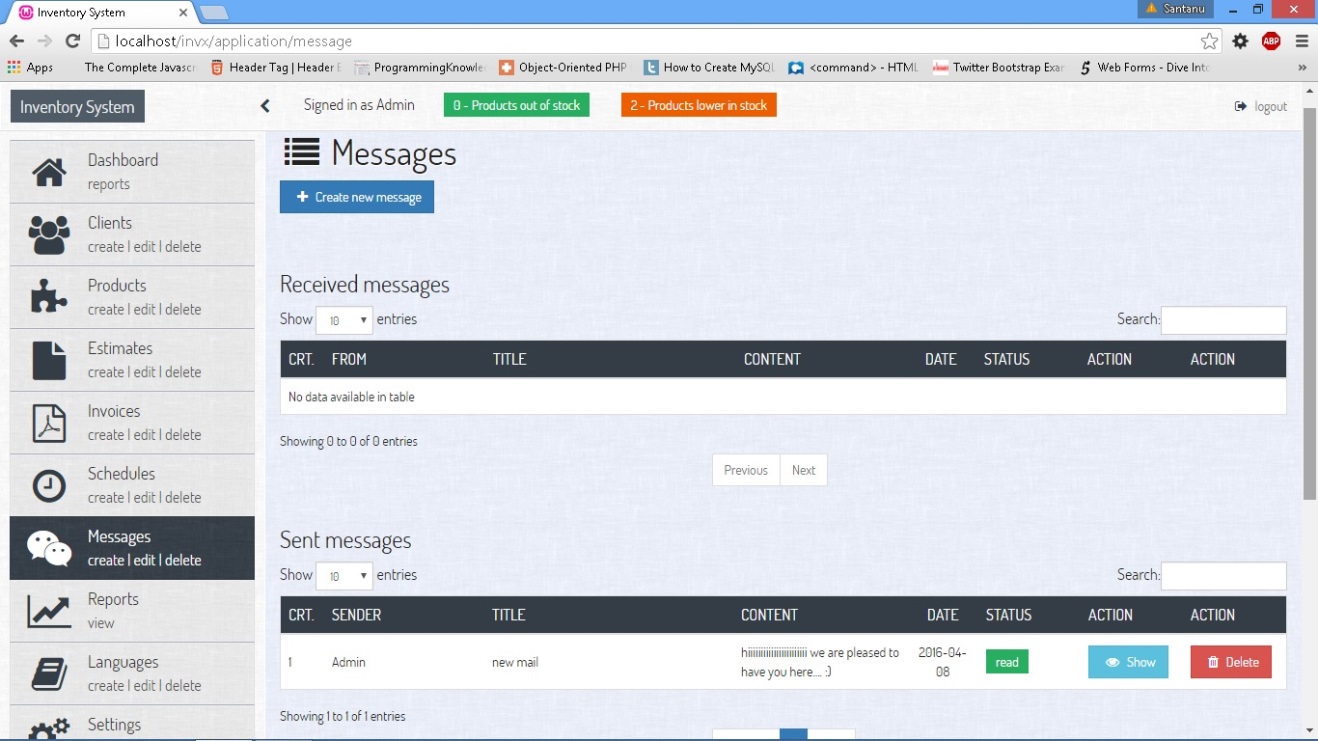
**Fig 5.9: Estimates Details on Distributor**

****

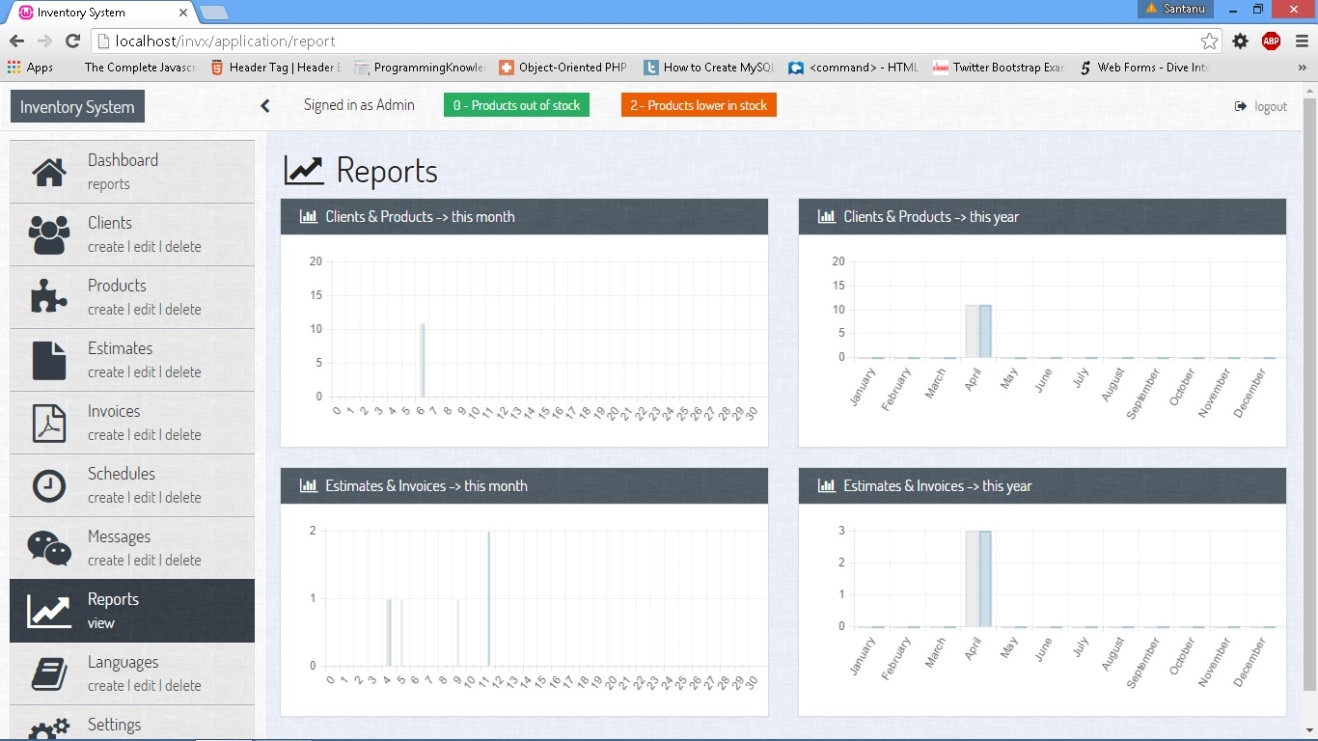
**Fig 5.10: Invoices Details on Distributor**

****

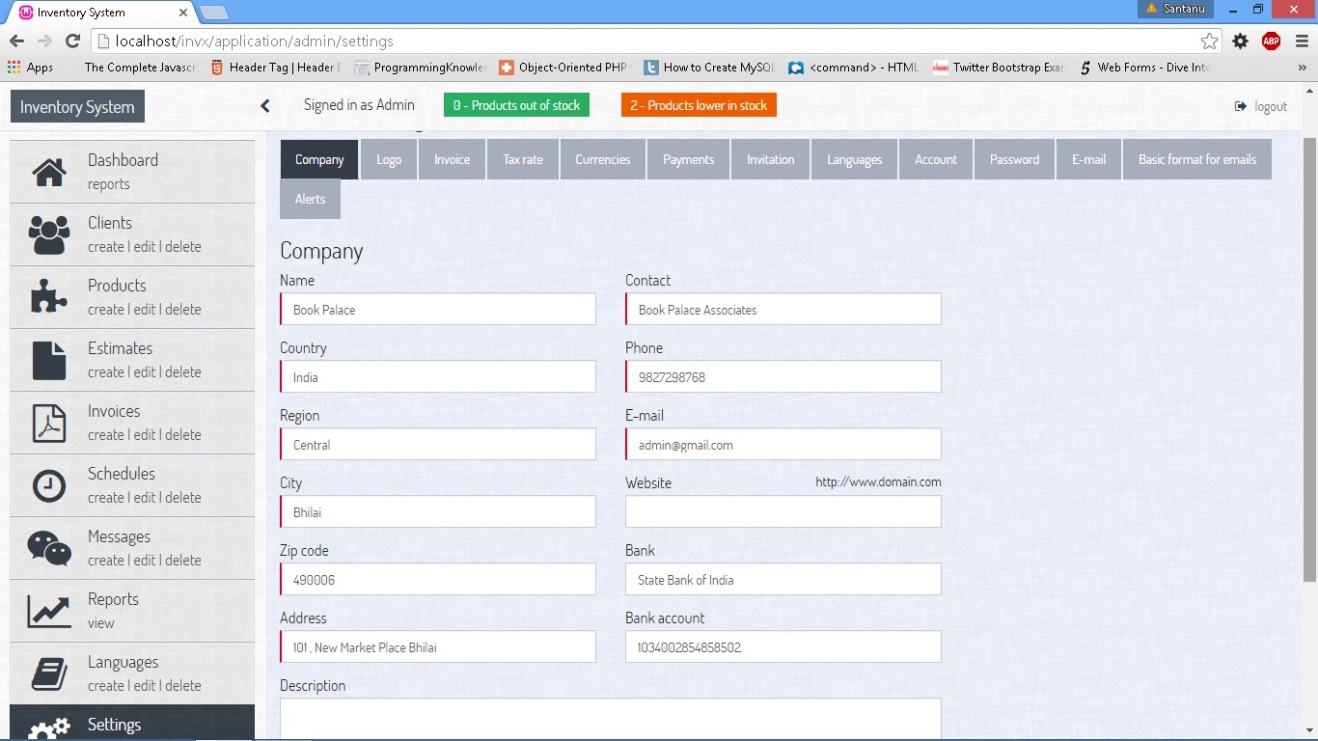
**Fig 5.11: Schedules Details on Distributor**

****

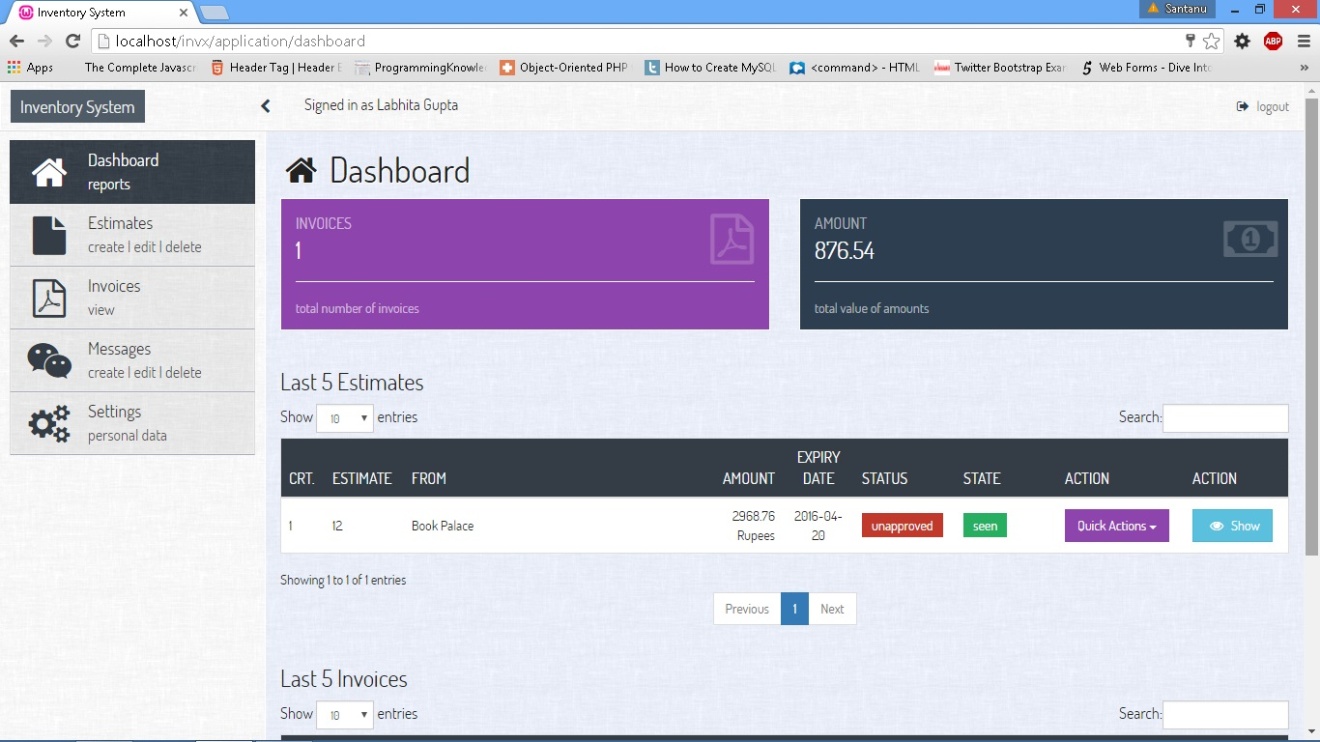
**Fig 5.12: Messages Details on Distributor**

****

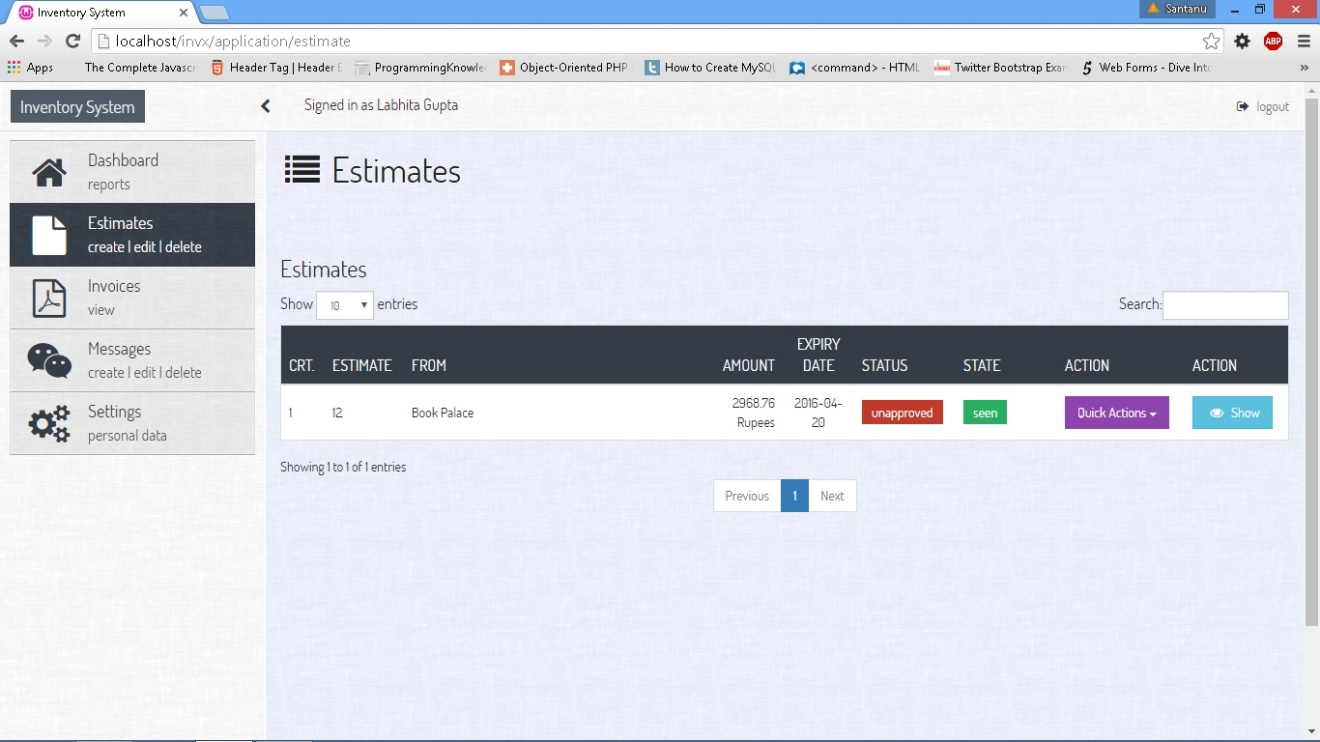
**Fig 5.13: Reports Details on Distributor**

****

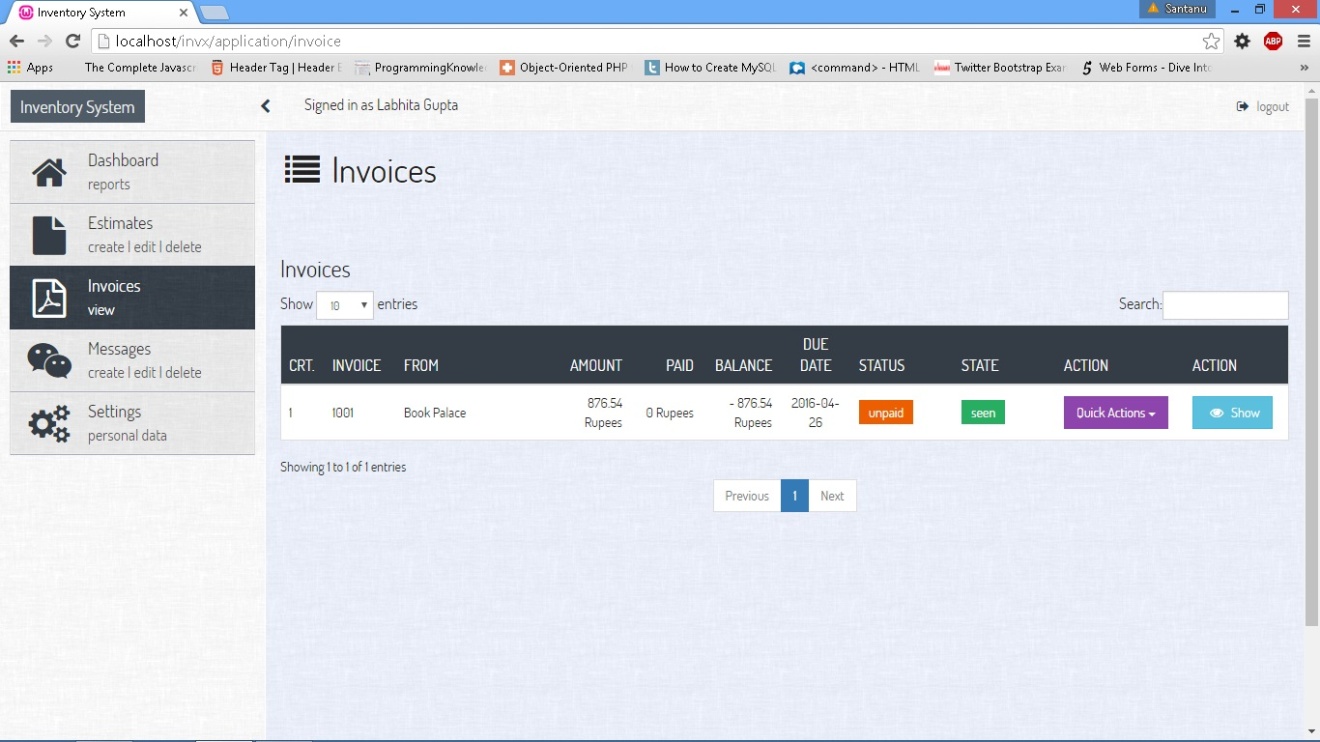
**Fig 5.14: Setting Details on Distributor**

****

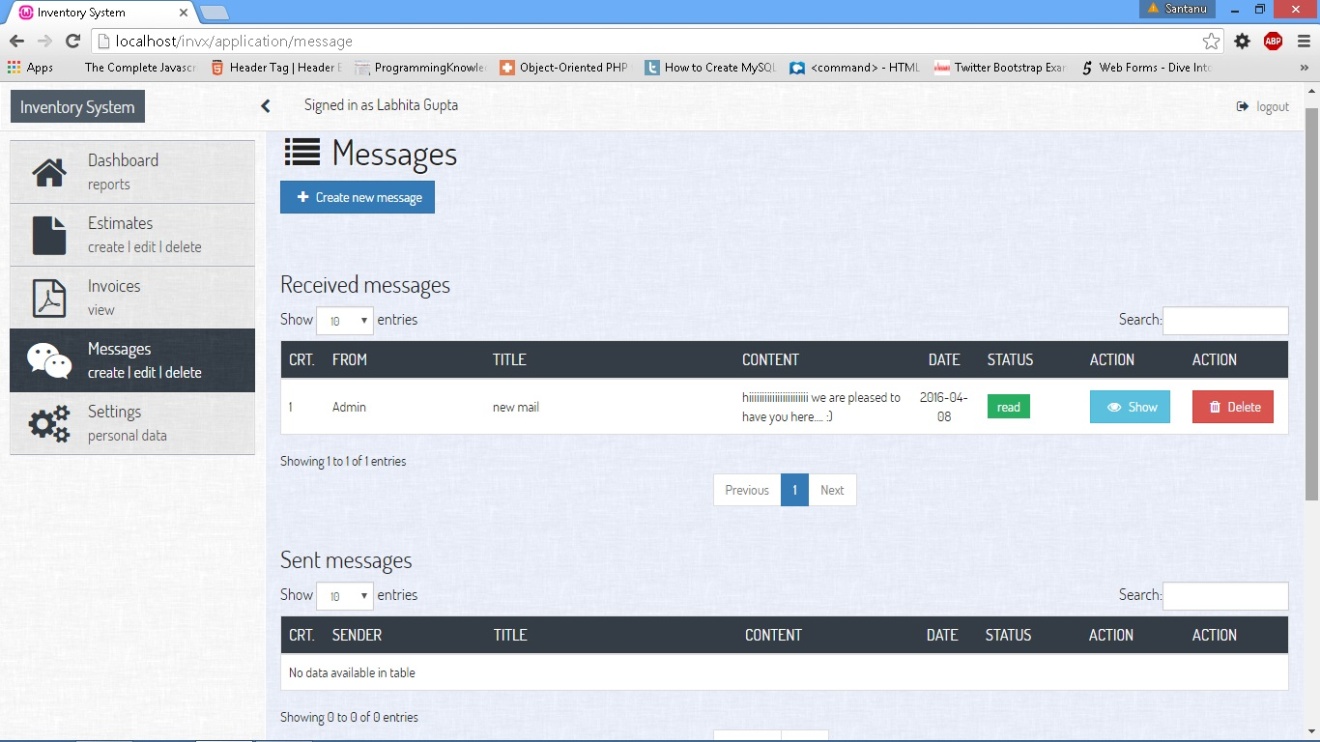
**Fig 5.15: Retailer Dashboard**

****

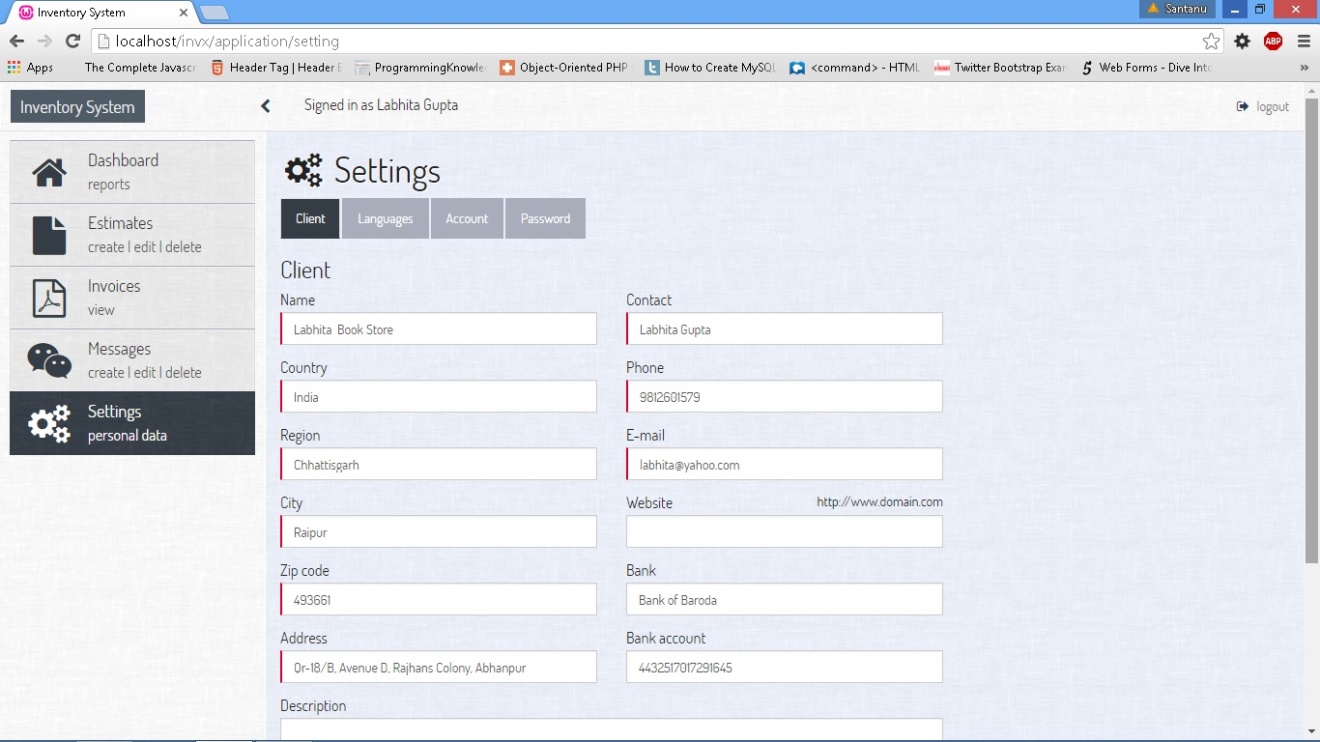
**Fig 5.16: Estimates Details on Retailer**

****

**Fig 5.17: Invoices Details on Retailer**

****

**Fig 5.18: Messages Details on Retailer**

****

**Fig 5.19: Setting Details on Retailer**

**6. SOFTWARE TESTING**

Software testing is a critical element of software quality assurance and represent the ultimate review of specification, design, coding. The purpose of product testing is to verify and validate the various work products viz. units, integrated unit, final product to ensure that they meet their requirements.

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the [software](https://en.wikipedia.org/wiki/Software) to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding [software bugs](https://en.wikipedia.org/wiki/Software_bug) (errors or other defects).

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

* meets the requirements that guided its design and development,
* responds correctly to all kinds of inputs,
* performs its functions within an acceptable time,
* is sufficiently usable,
* can be installed and run in its intended [environments](https://en.wikipedia.org/wiki/Operating_environment) and
* achieves the general result its stakeholders desire.

In Distributor Retailer Interactive System Black Box Testing has been used.

**7. CONCLUSION**

The conclusions drawn from the project Distributor Retailer Interactive System are:

* It is a web based solution on PHP platform to ease the functioning between a distributor and a retailer.
* It has removed the tedious process of holding data in an excel sheet, which provided fewer manipulation features.
* Distributor and retailer system provides various features like attractive dashboard, which represents all the basic details precisely.
* The system helps in generating reports for comparative analysis amongst the retailers and also among their product selection.
* It helps maintaining communication among the distributor and retailer by using the message feature.
* This system also provides multiple language feature which enable it to be used at different regions.
* The system helps distributor keep the proper stock details, to avoid loosing clients due to lower stock storage which might happen if it is held using a traditional system for the same.
* The system can handle a large number of retailers and maintain their records effectively and help in faster processing of orders.
* The system also aims at eliminating paperwork, and reduces time and the labor cost. When the system is used then human tensions and risks can be overcome.
* User friendly forms are designed for data entry at the retailer level. The data entered by the retailers can further be used during promotion of other goods.
* The invoice of the orders can be generated and dispatched to the retailers through e-mail.

**8. FUTURE ENHANCEMENTS**

Everything that is made has some or the other things to be added to make it better than revolutions. The project “Distributor Retailer Interactive System” has been tried to develop a robust and fault free system, still enough flexibility has been provided for further enhancements and modifications. As mentioned earlier that the designed forms are typically reflections of the developer, so it is strongly believed that the enhancement to be done with the project are acceptable for its betterment. But at the same time it should be mentioned that since one cannot claim himself/herself as a master of technology there is always some scope of technical modifications in the project that may lead to find code redundancy & storage space minimization.

The future enhancements for this project are:

* Transaction gateway can be implemented to enable the users for online transactions using credit cards instead of the regular cash or cheque transactions.
* The retailers can be provided access to order the products online through the system instead of manual registration of order.
* The system can be deployed further to run on smart phones as an Android application so that it becomes user friendly as well as can be accessed from anywhere and at anytime.

**BIBLIOGRAPHY**

**Books:**

1. Kevin Yank, “PHP & MySQL: Novice To Ninja”, SitePoint Private Limited,3th Edition, Published in: 2009
2. Larry Ullman, “PHP & MySQL for Dynamic Web Sites: Visual Quick Pro Guide”, Peachpit Press , 4nd Edition, Published in: 2011
3. Vikram Vasvani, “The Complete Reference MySQL”, McGraw-Hill Companies, 4th Edition, Published in: 2004
4. David Flanagan, “jQuery Pocket Reference”, O'Reilly Media, 1st Edition, Published in: 2010

**Websites:**

1. [php.net](http://www.java2s.com/Code/Java/CatalogJava.htm)
2. <http://stackoverflow.com/>
3. <https://en.wikipedia.org/>
4. <http://programmers.stackexchange.com/>
5. <https://laravel.com/>
6. <https://jquery.com/>