# Introduction

## Purpose

The project “Billing system” is an application to automate the process of ordering and billing

parts. The application can be used easily for record keeping and reporting. The application can be installed on desktop . The system is easy to use with simple knowledge of computer.

This application also administrates its users .

## Product Scope

This application keeps the data in a **centralized way** which is available is in billing system. It is very easy to manage historical data in database.

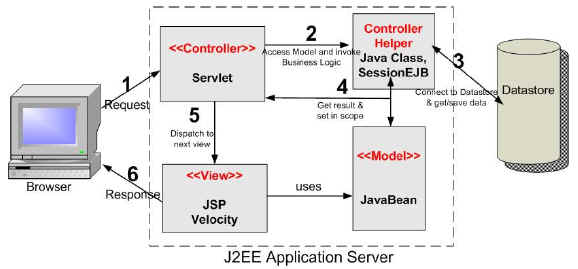
## Overview

* Now a days, for any billing system there is not more options rather than paper work(file) to save the details of parts available . It’s very time consuming process to collect the details from market, make report and analyze all reports.
* Also for employees, they have to manage all client’ reports, buy reports, sale reports etc. i.e. time consuming and lots of paper work.
* So to overcome from all above the problems and typical process we create a web application that is very helpful to employees as well as customers as well as manager.

1. **Technology and Literature Review**

**2.1 About Tools and Technology**

We use the java for this project using netbeans and for database we use SQLyog Community . In java we use the jsp and servlet. how this work we see in fig.



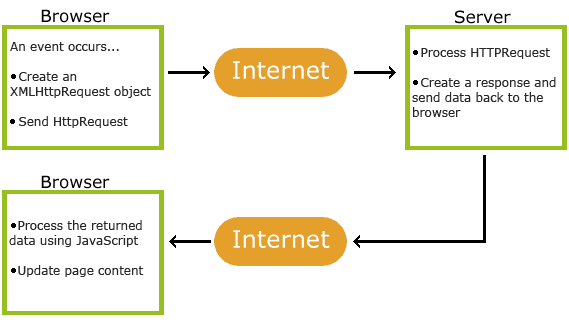
**AJAX**

AJAX = Asynchronous JavaScript and XML.

AJAX is a technique for creating fast and dynamic web pages. AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page. Classic web pages, (which do not use AJAX) must reload the entire page if the content should change

AJAX is about updating parts of a web page, without reloading the whole page.

**How ajex work?**



**2.2 Brief History of Work Done**

**2.2.1 Project Plan**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ACTIVITY** | **DAYS** | **APPROX. WEEKS** |
| 1 | Defining Project | 7 | 1 |
| 2 | Understanding User Requirements & requirement Gathering | 15 | 2 |
| 3 | System Design | 30 | 4 |
| 4 | DFD &class diagram | 10 | 2 |
| 5 | Data dictionary | 14 | 2 |
| 6 | Form design | 14 | 2 |
| 7 | Project Coding | 60 | 8 |

## 3 System Requirements Study

## 3.1 Hardware Interfaces

1. PC should be sufficiently fast with adequate memory space. At least 64 MB RAM and 2 GB hard-disk space is required to run this application.
2. Screen resolution of atleast 800\*600 is required to properly view the screens.

## 

## 3.2 Software Interfaces

1. Any windows operating system.
2. For the database handling SQLYOG must be installed. These products are open source products.

**3.3 Design and Implementation Constraints**

* **Database:**  
  The system shall use the SQLYOG Database, which is open source and free.
* **Operating System**  
  The Development environment shall be Windows based.
* **Web-Based**  
  The system shall be a Web-based application.

**3.4 Assumptions and Dependencies**

**Assumptions:**

* No one can enter the system without logging-in.
* All the rights and access is only given to the administrator.
* Customer will be able to upload complaints but only administrator can reply to it.

**Dependencies:**

* The system depends on the operation of admin/users.
* Admin will be given enough training to use the system.
* System also depends on the internet and server. In case of server failure, they should

contact developer.

1. **System Analysis**

**4.1 Problem of existing system**

**1. Inability of modification of data:** The managing of huge data effectively and efficiently for efficient results, storing the details of the consumers etc. in such a way that the database can be modified as not possible in the current system.

**2. Not user friendly:** The existing system is not user friendly because the retrieval and storing of data is slow and data is not maintained efficiently.

**3. Difficulty in reports generating:** Either no reports generating in a current system or they are generated with great difficulty reports take time to generate in the current system.

**4. Manual operator control:** Manual operator control is there and lead to a lot of chaos and errors.

**5. Lot of paperwork:** Existing system requires lot of paper work and even a small transaction require many papers fill. Moreover any unnatural cause (such as fire in the organization) can destroy all data of the organization. Loss of even a single paper led to difficult situation because all the papers are interrelated.

**6. Inability of sharing the data:** Data cannot be shared in the existing system. This means that no two persons can use the same data in existing system. Also the two departments in an organization cannot interact with each other without the actual movement of data.

* 1. **Characterstic of the proposed system**

**1. Easiness in modification of data:** The proposed system provides managing of huge data effectively and efficiently for efficient results, storing the details of the customers, employees etc. in such a way that the database can be modified.

**2. User friendly:** The proposed system is user friendly because the retrieval and storing of data is fast and data is maintained efficiently. Moreover the graphical user interface is provided in the proposed system, which provides user to deal with the system very easily.

**3. Reports are easily generated:** Reports can be easily generated in a proposed system. So any type of reports can be generated in a proposed system, which helps the managers in a decisions- making activity.

**4. Sharing the data is possible:** Data can be shared in proposed system. This means that two or more persons can use the same data in existing system provided that they have right to access that data. Also the two or more departments in an organization can easily interact with each other without the actual movement of data.

**5. No or very few paperwork:** The proposed system either does not require paper work or very few paper works is required. All the data is feted into the computer immediately and various bills and reports can be generated through computers. Since all the data is kept in a database no dataof the organization can be destroyed. Moreover work becomes very easy because there is no need to keep data on papers.

**6. Computer operator control:** Computer operator control will be there no errors. Moreover storing and retrieving of information is easy. So work can be done speedily and in time.

**4.3 Features:**

Billing System has lots of features as following:

* **Admin :**
* login
* Easily save parts details
* Easily save company details
* View sold part excel file
* **Staff:**
* Generate purchase order form
* Generate delivery challan form
* Generate invoice and other report

4.4 Class Diagram

* The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application.
* The class diagram describes the attributes and operations of a class and also the constraints imposed on the system.
* The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages.
* The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a structural diagram.

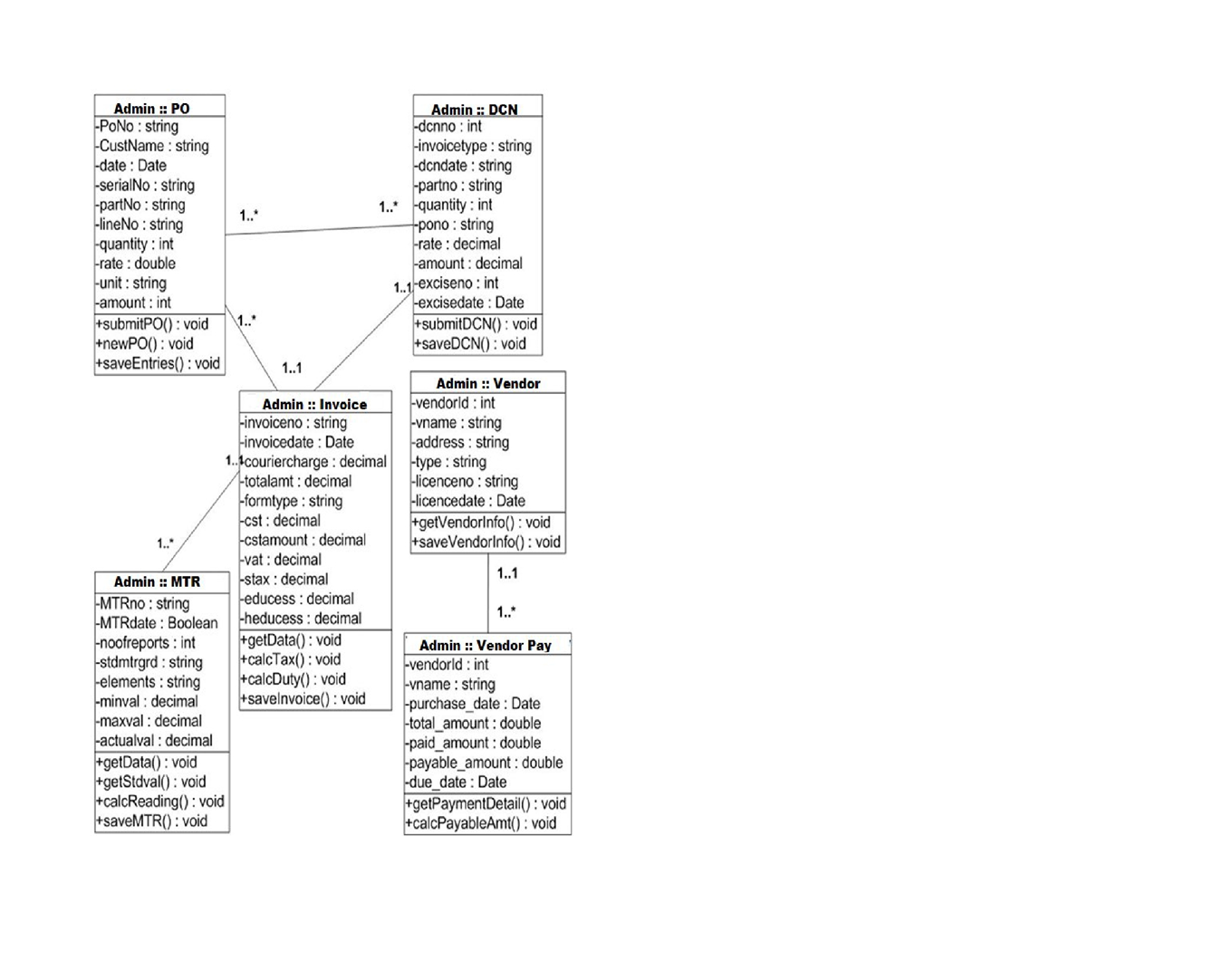


Fig-4.4(a)

4.5 Use case diagram

* To introduce use case diagrams two things to remember is:

1) Actors

2) Use cases

Dependency relationships between use cases:

* An *actor* is anyone or anything that must interact with the system.
* Actors are not part of the system
* In the UML, an actor is represented as a stickman.
* A use case must deliver something of value to an actor.
* The collection of use cases for a system constitutes all the defined ways the system may be used.
* In the UML, a use case is represented as an oval.
* A use case diagram is a diagram that shows a set of use cases and actors and their relationships.
* Actors may be connected to use cases only by association.
* An association between an actor and a use case indicates that the actor and the use case communicate with one another, each one possibly sending and receiving messages.

**Use case diagrams are used to …**

* Model the context of a system.
* Model the requirements of a system

**Dependency Relationships between Use Cases:**

* Extend: Specifies that the target use case extends the behavior of the source use case.
* Include: Specifies that the source use case explicitly incorporates the behavior of another use case at a location specifies by the source.

**Symbols for Use case diagram**

|  |  |
| --- | --- |
| **Symbol name** | **Symbol** |
| System Boundary |  |
| Use-Case |  |
| Uses |  |
| Actor |  |





4.5(b)



4.5(c)

4.5(d)



4.6 Sequence Diagram

It illustrates how objects interact with each other. It emphasizes time ordering of

messages it can model simple sequential flows, branching, iteration, recursion and

concurrency.

**Table: 4.4 Symbols for Sequence diagram**

|  |  |
| --- | --- |
| **Symbol Name** | **Symbol** |
| Object | Object |
| Life Line |  |
| Active |  |
| Message |  |

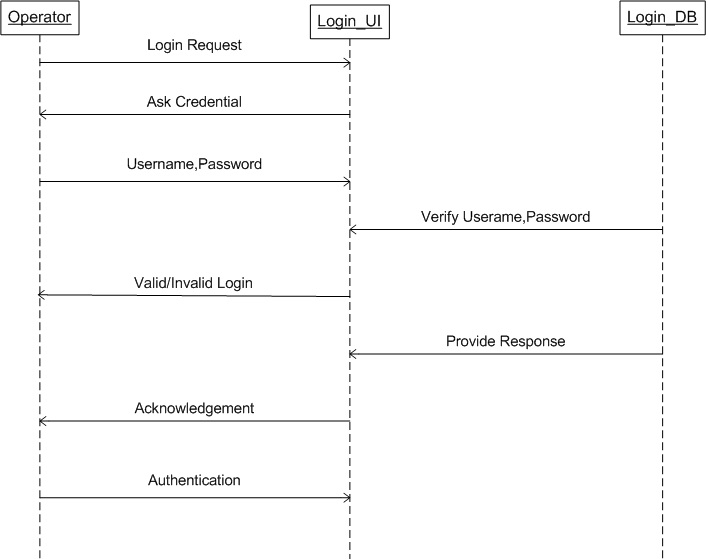


Fig: 4.6(a)

**Master**

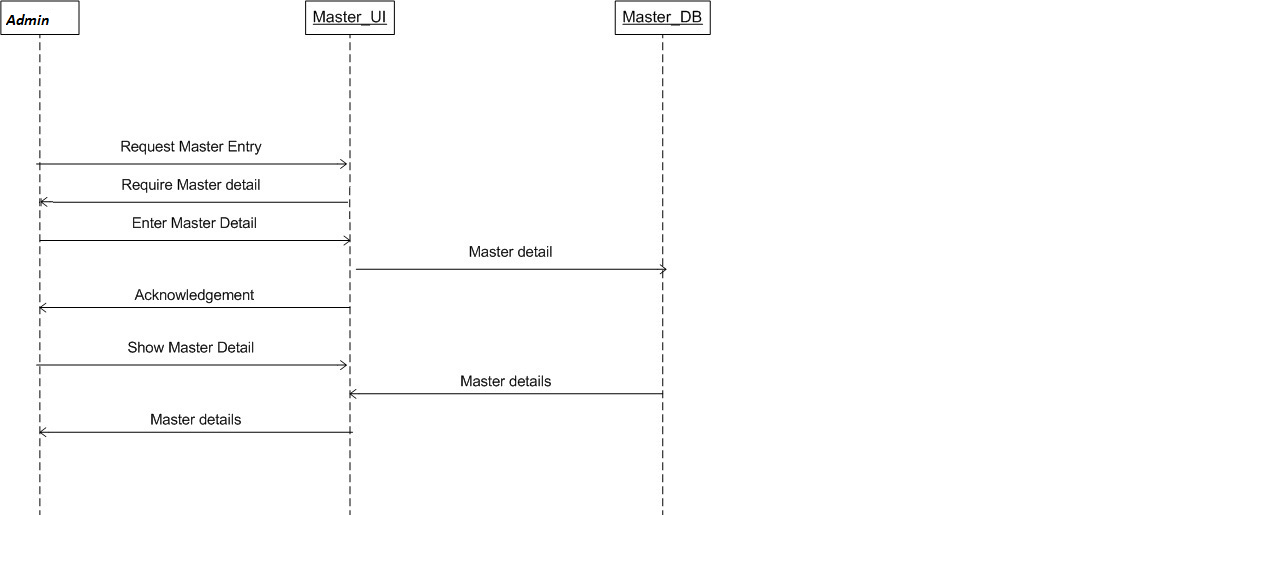


fig-4.6(b)

**PurchaseOrder**

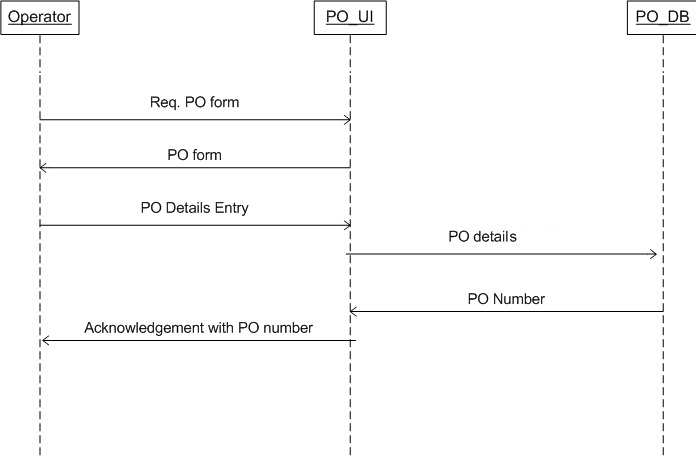


Fig-4.6(c)

**Delivery Challan**

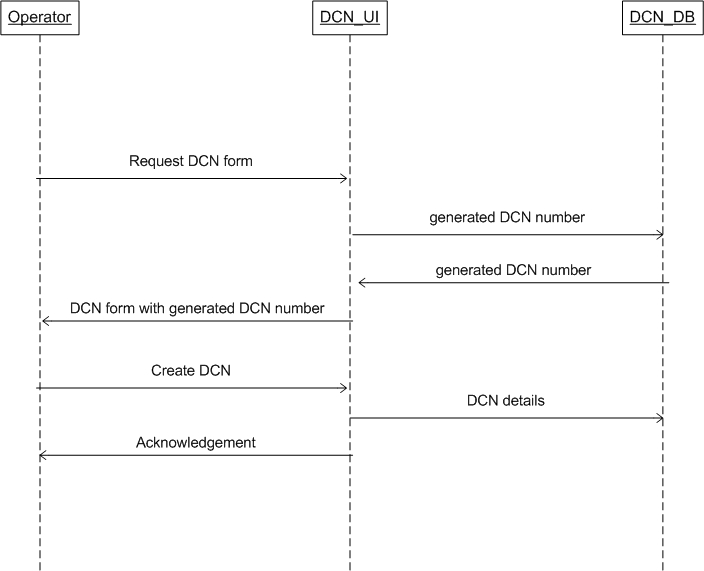


Fig-4.6(d)

**Invoices**

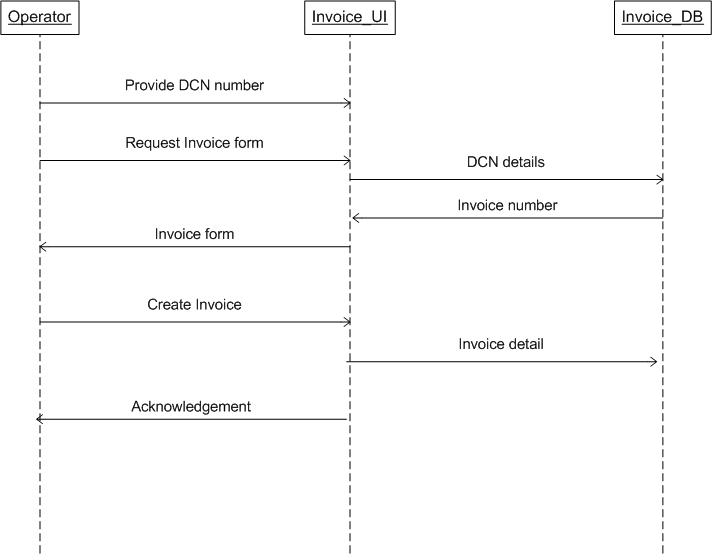
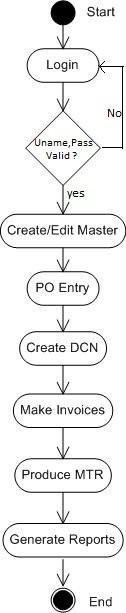


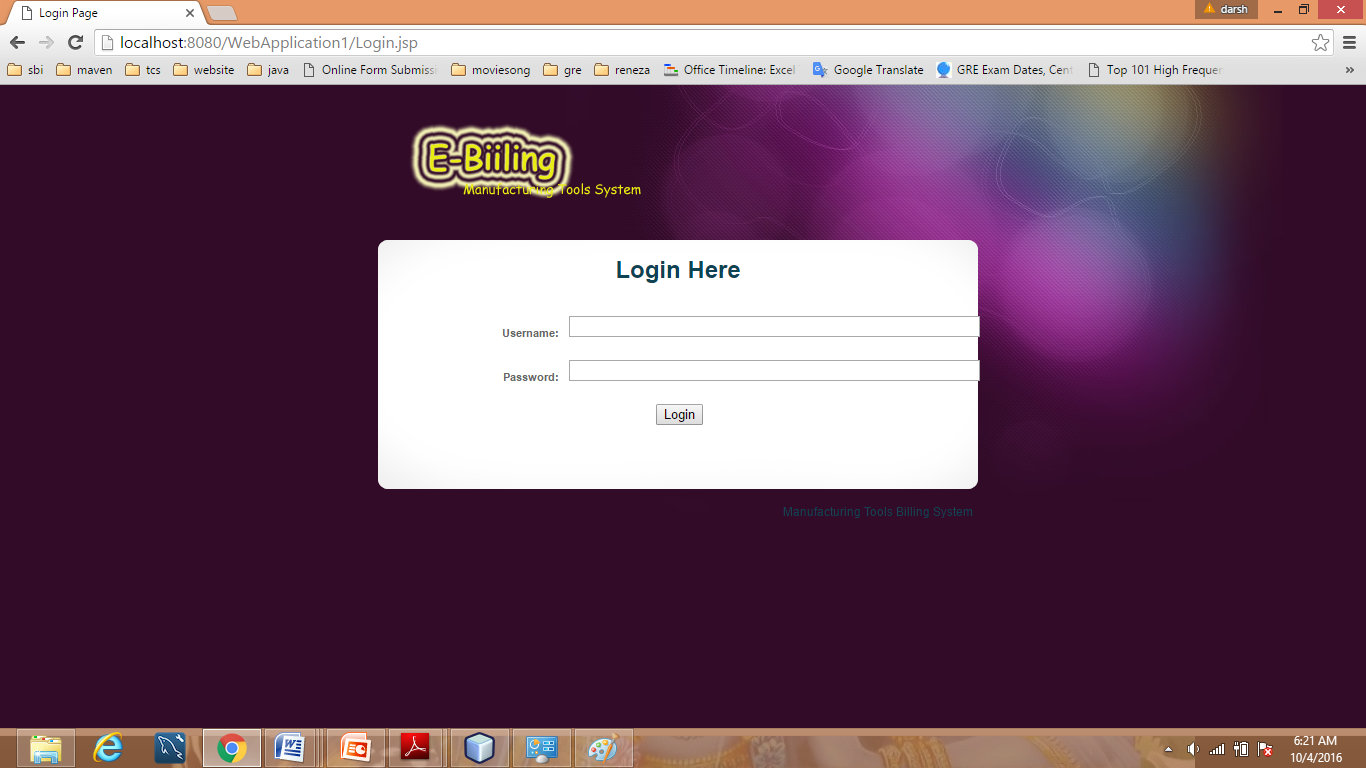
Fig-4.7(e)

**4.7.ActivityDiagram**



1. **System Design**

**5.1 System Application Design**



**5.2 Database Design**

1. **Login**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| uid | int(20) | NO | PRI | User id |
| utype | varchar(30) | YES |  | User type |
| unm | varchar(30) | YES |  | User name |
| pass | varchar(20) | YES |  | Password |

**ii.state\_entry**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| stateid | int(10) | NO | PRI | State id |
| statename | varchar(10) | YES |  | state name |

**iii. city\_entry**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| cityid | int(10) | NO | PRI | city id |
| stateid | int(10) | YES | Foreign | State id |
| cityname | varchar(10) | YES |  | city name |

**Iv Company Detail**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| cid | int(11) | NO | PRI | Company id |
| Cname | varchar(30) | YES |  | Company name |
| Address | varchar(50) | YES |  | Address |
| City id | varchar(30) | YES | Foreign key | City id |
| Stateid | varchar(30) | YES | Foreign | State id |
| Country | varchar(30s) | YES |  | Country |
| pinno | int(20) | YES |  | Pin code |
| Poneno | int(20) | YES |  | Phone number |
| Fax | int(20) | YES |  | Fax number |
| Email | varchar(15) | YES |  | e-mail address |
| Tinno | int(20) | YES |  | Company tin number |
| Vendercode | int(20) | YES |  | Company vender code |
| Stccode | int(20) | YES |  | Company stc code |
| Stcdate | Date | YES |  | stc date |

**v. Product Detail**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Pid | int(20) | NO | PRI | Product id |
| Partname | Varchar(30) | YES |  | Part name |
| Partdesc | Varchar(50) | YES |  | Part description |

**vi. Purchase Order**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data Type** | **Null** | **Key** | **Description** |
| PO\_No | int(20) | NO | PRI | Purchase Order number |
| cid | Varchar(50) | YES | Foreign | Company id |
| Orderdate | Date | YES |  | Order date |

**vii. Purchase Order Line**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **Null** | **Key** | **Description** |
| Polineno | int(10) | NO | PRI | Purchase order line no |
| PO\_No | Varchar(30) | YES | Foreign | Purchase order number |
| Serialno | Varchar(10) | YES |  | Serial number |
| Part no | Varchar(20) | YES | Foreign | Part number |
| Partdesc | Varchar(50) | YES | Foreign | Part Description |
| Lineno | int(10) | YES |  | Line number |
| Duedate | Date | YES |  | During date |
| Qut | int(10) | YES |  | Totalquantity |
| Unit | Varchar(20) | YES |  | Unit of part |
| Rate | Float(20) | YES |  | Rate of part |
| Amount | Float(20) | YES |  | Purchase order amount |

**viii.Challan Invoice Item**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Dcno | int(5) | No | PRI | Delivery challan number |
| Invoicetype | Varchar(25) | YES |  | Challan type |
| cid | Varchar(25) | YES |  | Company id |
| Dcn\_date | Date | YES |  | Delivery Challan Date |
| Poid | int(20) | YES | Foreign | Purchase order number |
| Part\_ no | Varchar(30) | YES | Foreign | Part number |
| Part \_desc | Varchar(50) | YES | Foreign | Part description |
| Qut | int(10) | YES |  | Total quantity |
| Rate | Float(20) | YES |  | Rate of part |
| Amount | Float(20) | YES |  | Amount of part |
| Exicechano | int(10) | YES |  | Excise challan number |
| Exicedate | Date | YES |  | Excise challan date |

**ix. Retail Invoice**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Retailno | varchar(10) | No | PRI | Retail invoice number |
| Dcno | int(5) | YES | Foreign | Delivery Challan Number |
| Compnm | Varchar(50) | YES |  | Company name |
| Invodate | Date | YES |  | Invoice date |
| Be\_no | int(10) | YES |  | Bill number |
| Cour\_char | int(10) | YES |  | Courier Charge |
| Be\_date | Date | YES |  | Bill date |
| Tot amo | Float(20) | YES |  | Total amount |
| Tr\_ch\_no | int(10) | YES |  | Tr challan number |
| Formtype | Varchar(10) | YES |  | Form type |
| Tr\_ch \_date | Date | YES |  | Tr challan date |
| Cst1/vat1 | int(10) | YES |  | Cst1/vat1 |
| Cst2/vat2 | Float(10) | YES |  | Cst1/vat2 |
| Add \_duty | int(10) | YES |  | Add duty |
| Affamount | Float(10) | YES |  | Affected amount |
| Rdoff | Float(10) | YES |  | Round off |
| Net amount | int(10) | YES |  | Net amount |
| Netamword | Varchar(10) | YES |  | Amount in word |

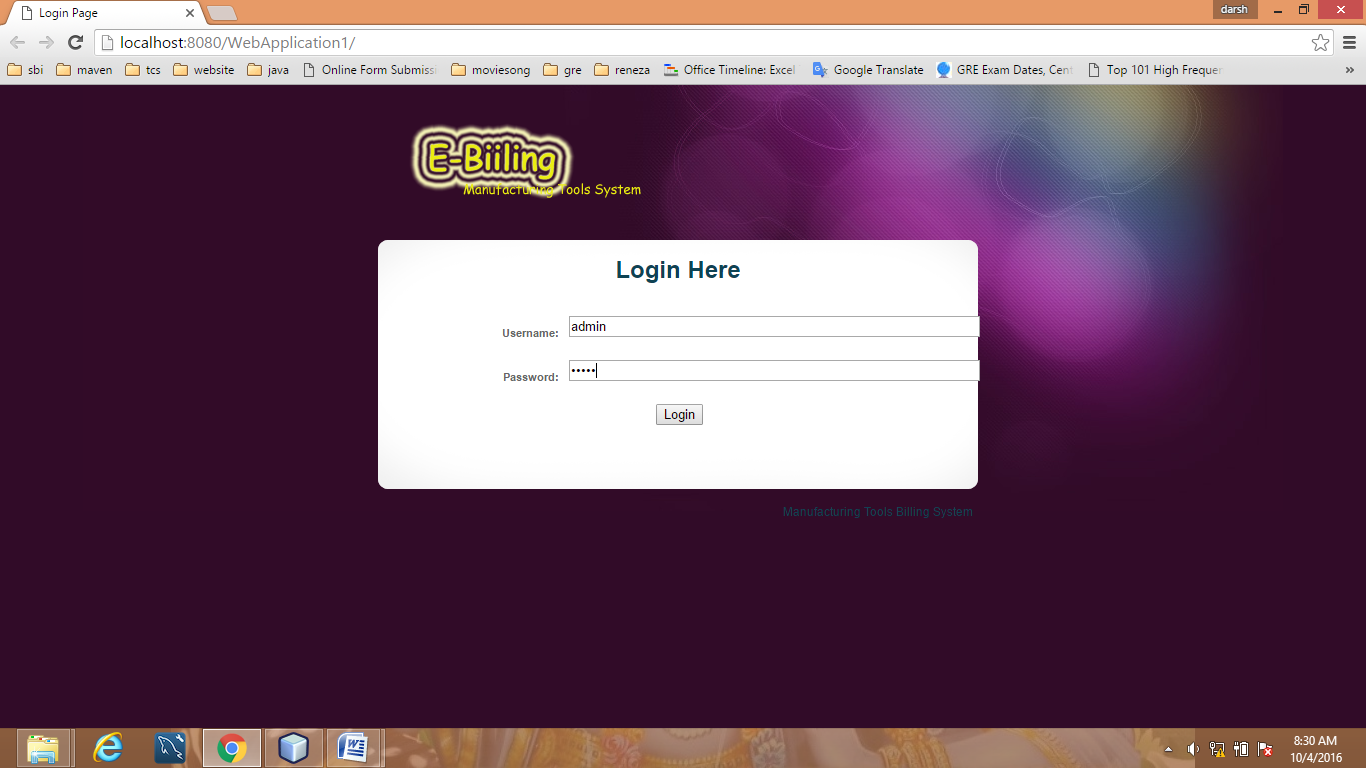
**x. Tax Invoice**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Tax no | Varchar(10) | NO | PRI | Retail invoice number |
| Dcno | int(5) | YES | Foreign | Delivery Challan Number |
| Comp\_ nm | Varchar(50) | YES |  | Company name |
| Invo\_ date | Date | YES |  | Invoice date |
| Be \_no | int(10) | YES |  | Bill number |
| Cour\_ char | int(10) | YES |  | Courier Charge |
| Bedate | Date | YES |  | Bill date |
| Totamo | Float(20) | YES |  | Total amount |
| Tr\_ ch\_ no | int(10) | YES |  | Tr challan number |
| Vat1 | int(10) | YES |  | Vat1 |
| Vat2 | Float(10) | YES |  | Vat2 |
| Tr \_ch \_date | Date | YES |  | Tr challan date |
| Add \_vat1 | int(10) | YES |  | Cst1/vat1 |
| Add \_vat2 | Float(10) | YES |  | Cst1/vat2 |
| Add \_duty | int(10) | YES |  | Add duty |
| Affamount | Float(10) | YES |  | Affected amount |
| Rdoff | Float(10) | YES |  | Round off |
| Net amount | int(10) | YES |  | Net amount |
| Netamword | Varchar(50) | YES |  | Amount in word |

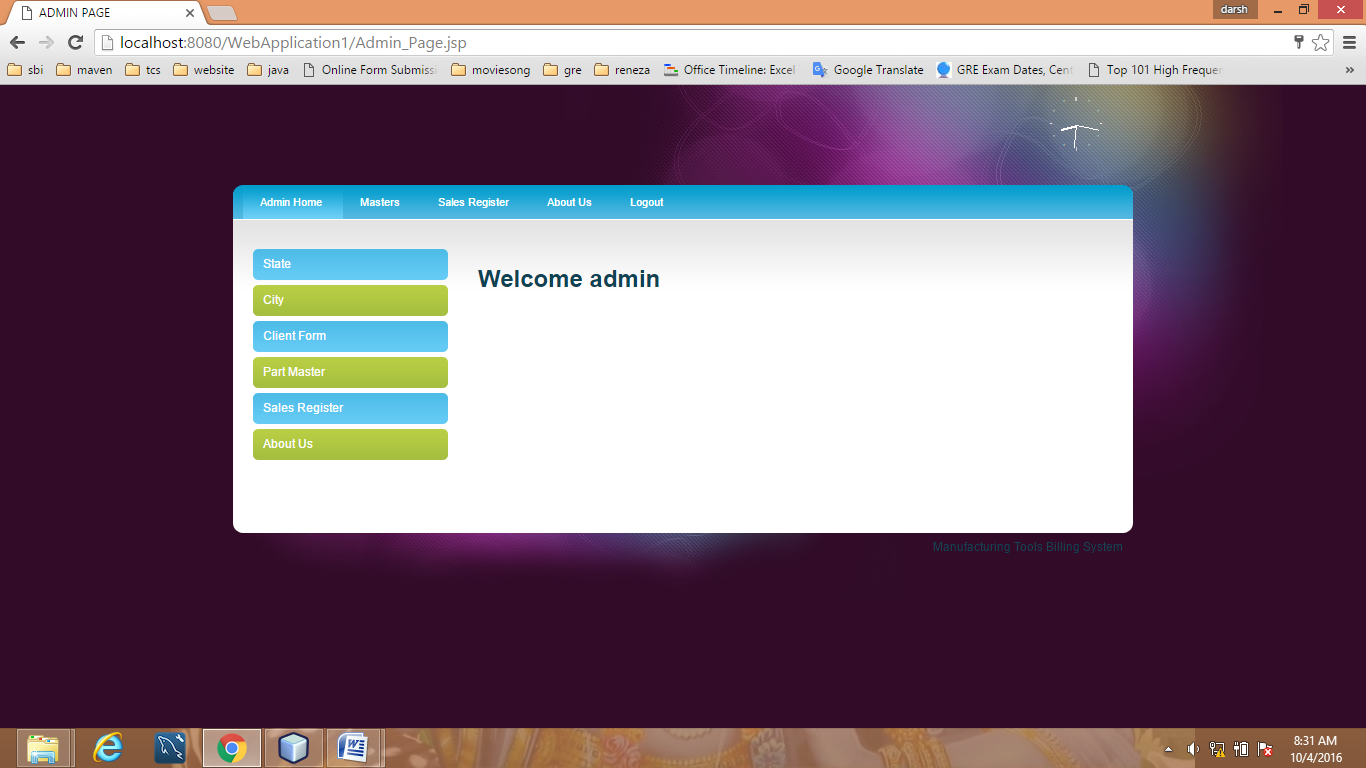
**xi. Job Work**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| job no | varchar(10) | NO | PRI | Retail invoice number |
| Dcno | int(5) | YES | Foreign | Delivery Challan Number |
| Comp \_nm | Varchar(50) | YES |  | Company name |
| Invo \_date | Date | YES |  | Invoice date |
| Be \_no | int(10) | YES |  | Bill number |
| Cour\_ char | int(10) | YES |  | Courier Charge |
| Be \_date | Date | YES |  | Bill date |
| Tot \_amo | Float(20) | YES |  | Total amount |
| Tr\_ ch\_ no | int(10) | YES |  | Tr challan number |
| Ser\_ tax1 | int(10) | YES |  | Service tax1 |
| Ser\_ tax2 | Float(10) | YES |  | Service tax2 |
| Tr\_ch\_date | Date | YES |  | Tr challan date |
| Edu\_cess1 | int(10) | YES |  | Education center for economic social study1 |
| Edu\_cess2 | Float(10) | YES |  | Education center for  economic social study2 |
| Addduty | int(10) | YES |  | Add duty |
| Hi\_edu\_cess1 | int(10) | YES |  | Hi education center for  economic social study1 |
| Hi \_edu\_cess2 | Float(10) | YES |  | Hi Education center for  economic social study2 |
| Affamount | Float(10) | YES |  | Affected amount |
| Rdoff | Float(10) | YES |  | Round off |
| Net amount | int(10) | YES |  | Netamount |
| Netamword | Varchar(50) | YES |  | Amount in word |

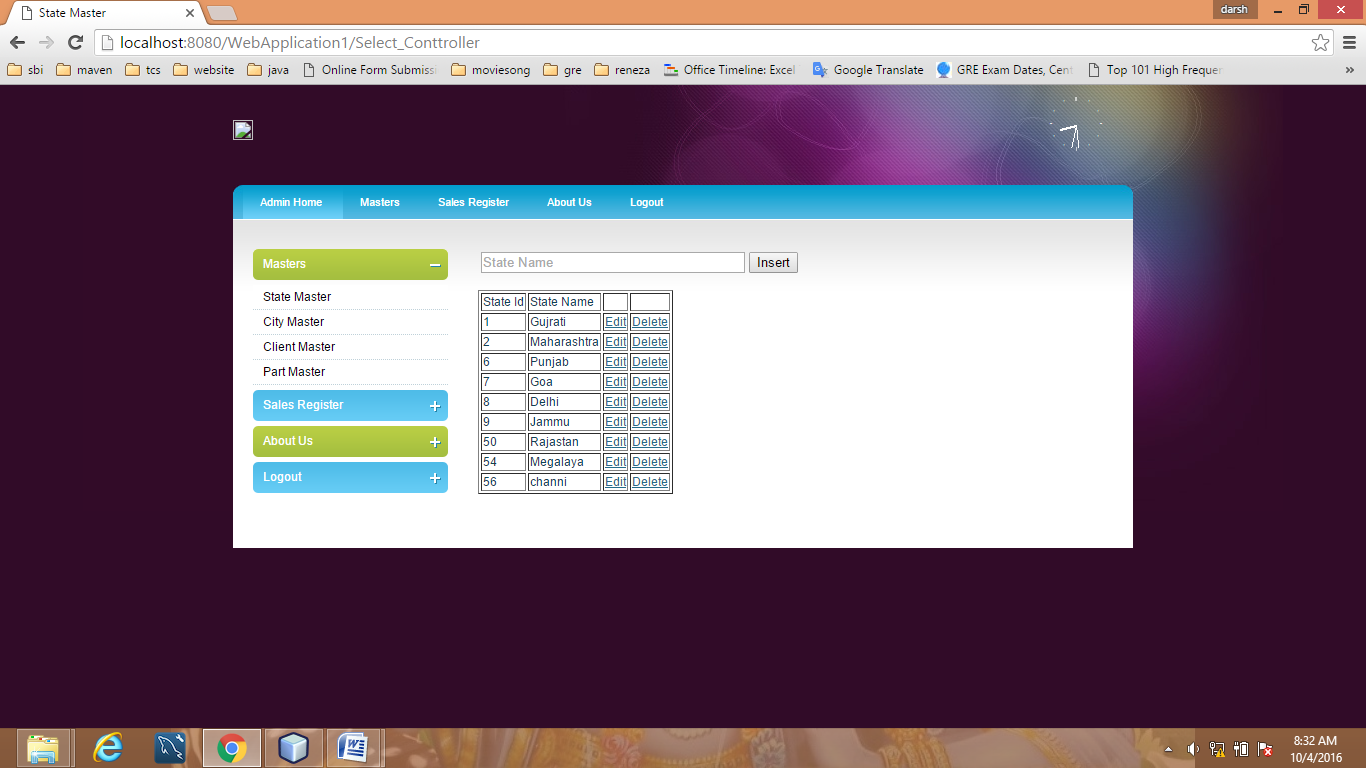
* 1. **Samples Of Forms, Reports and Interface**

****

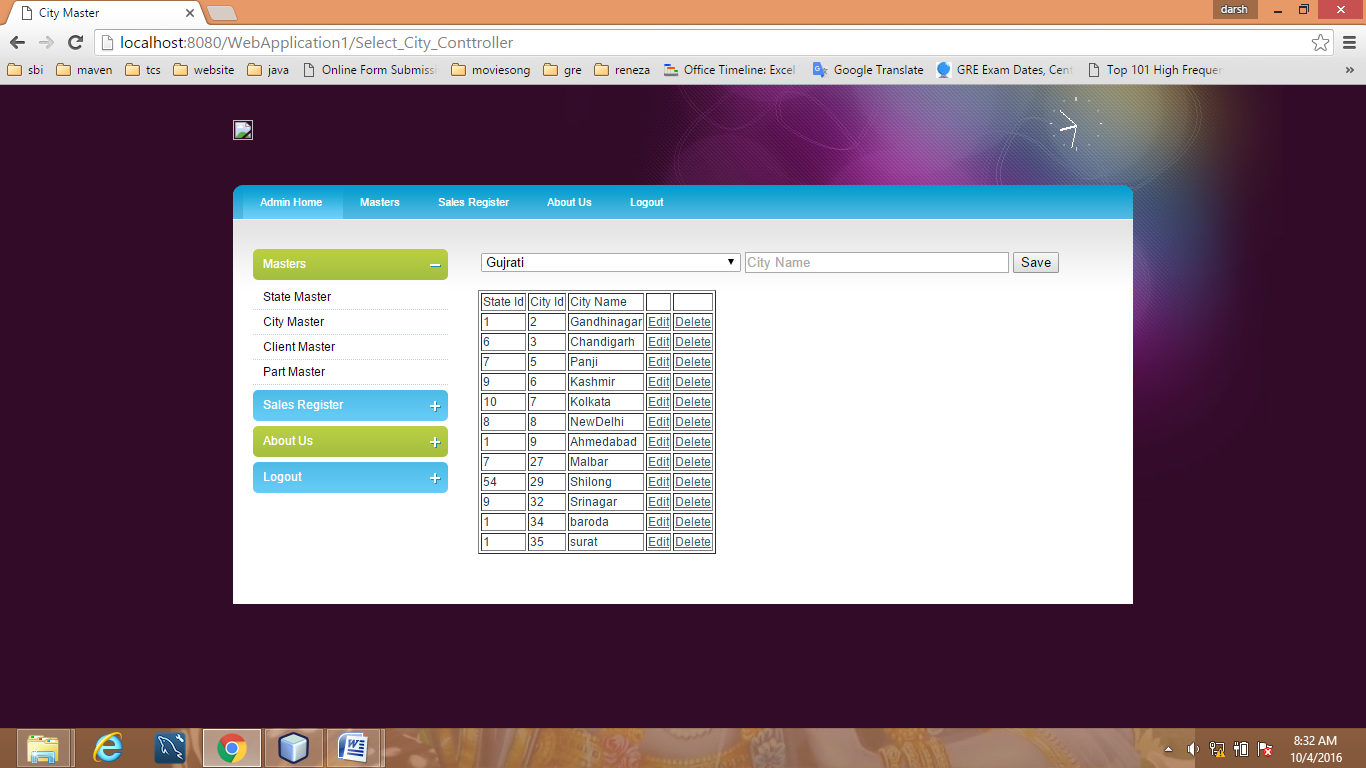
When admin enter correct username and password see below page

****

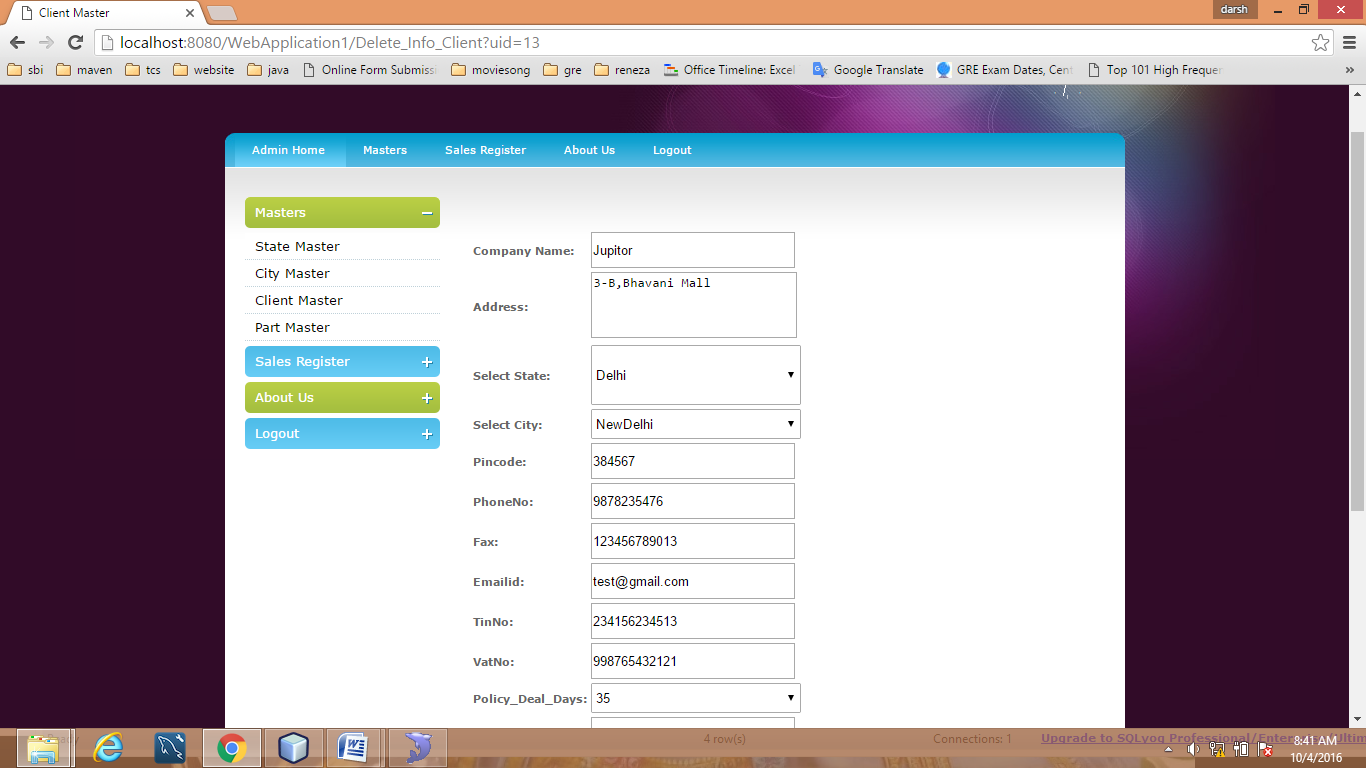
Now admin register the state

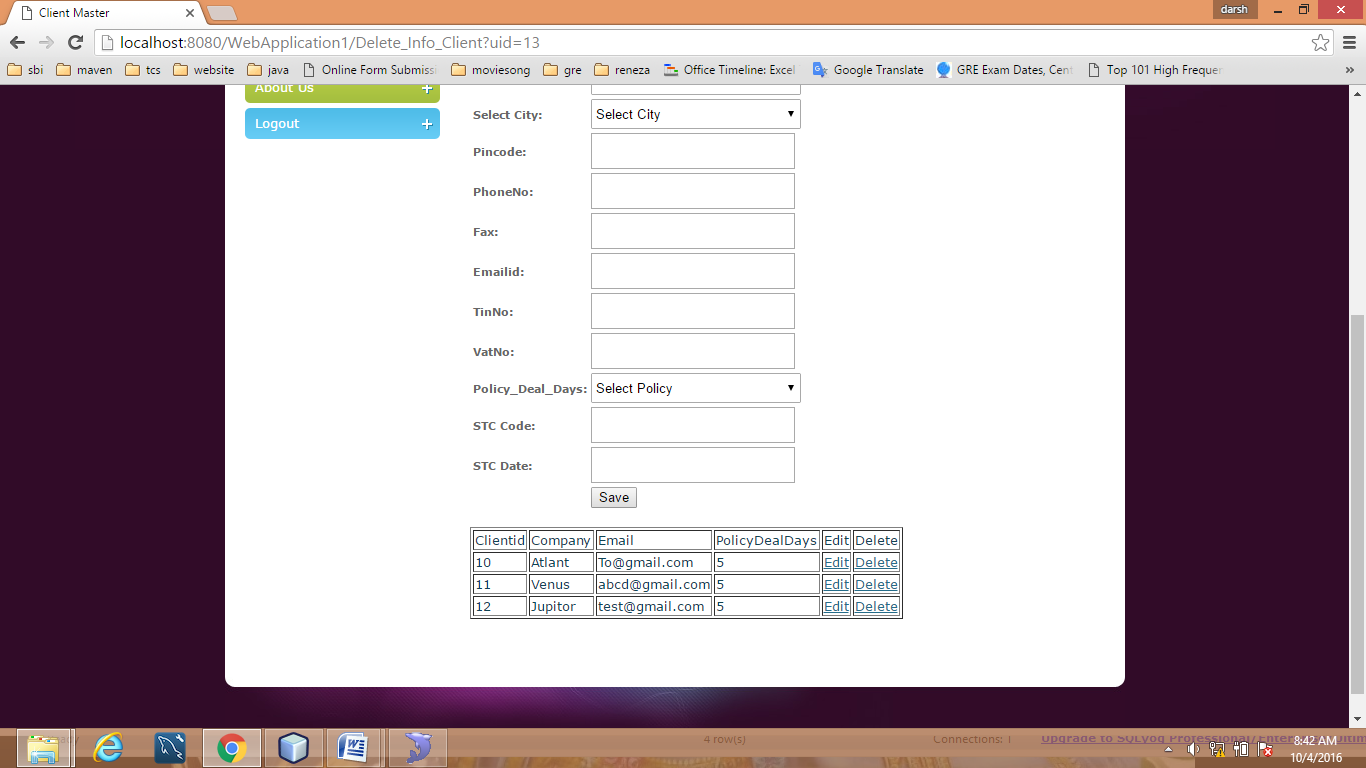
****

Now admin register city

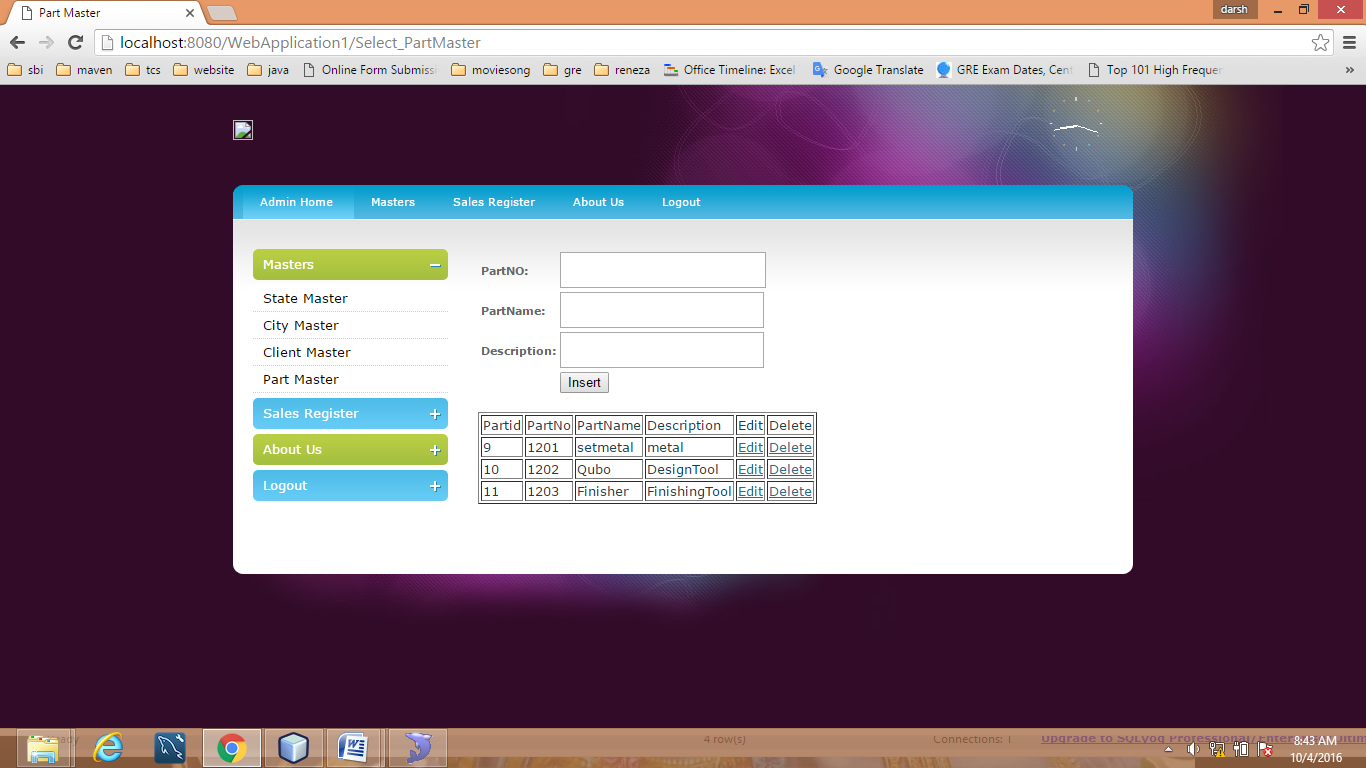
****

Now admin register seller company

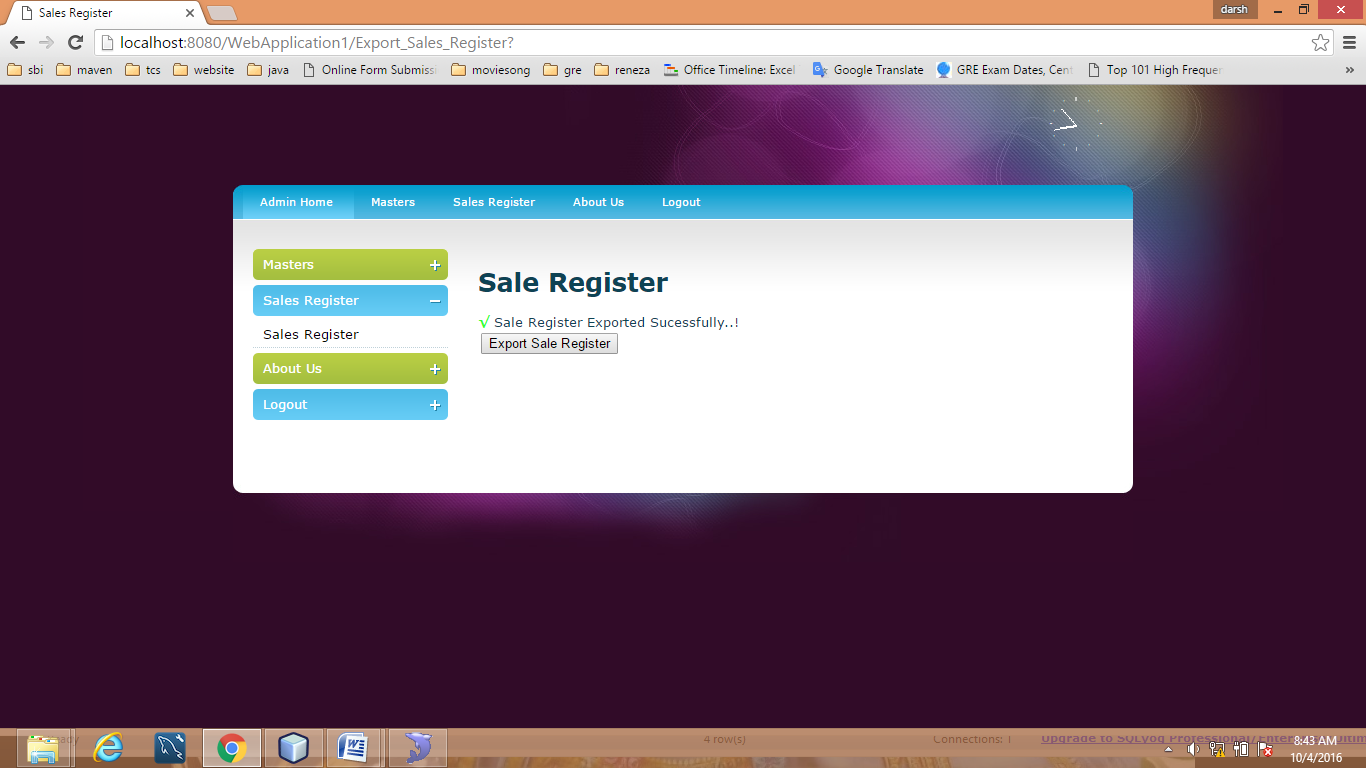
****

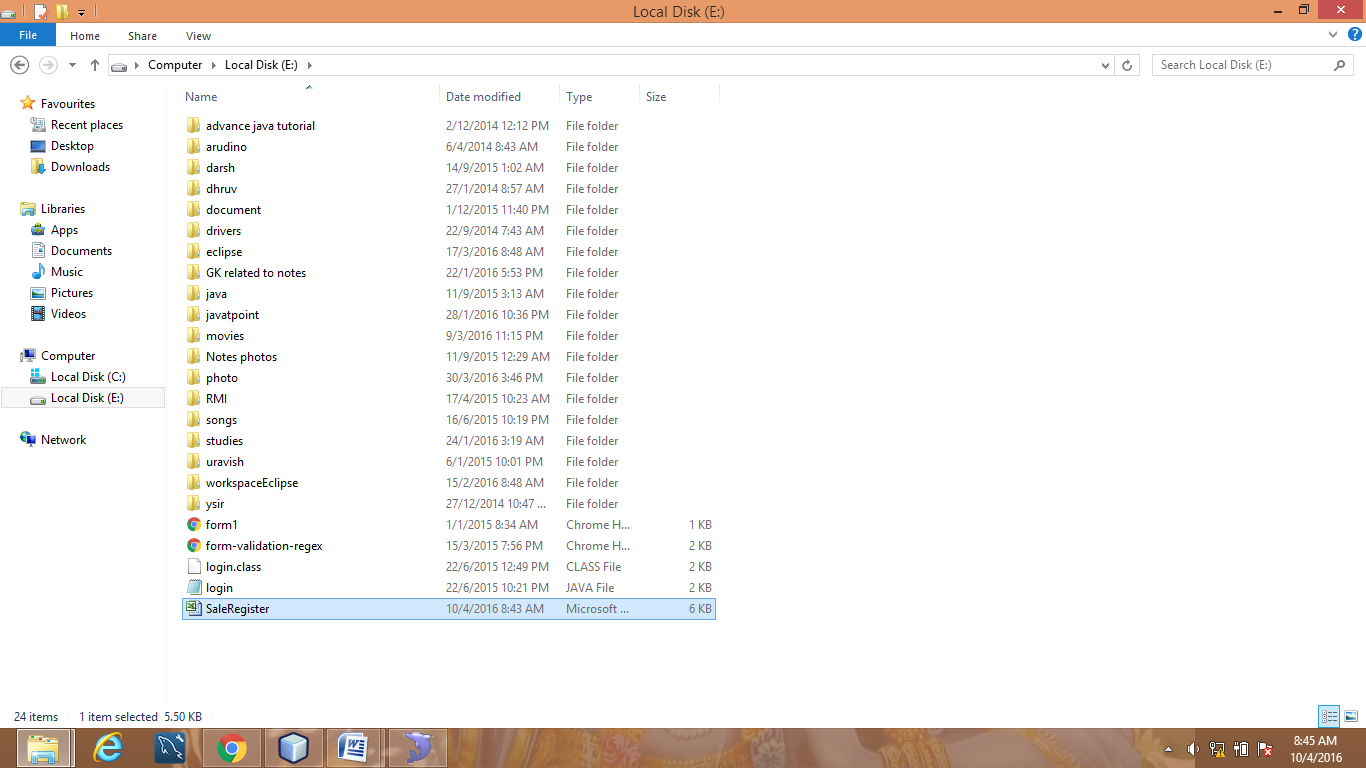
Now result can modify delete also directly ****

Now admin enter the part detail

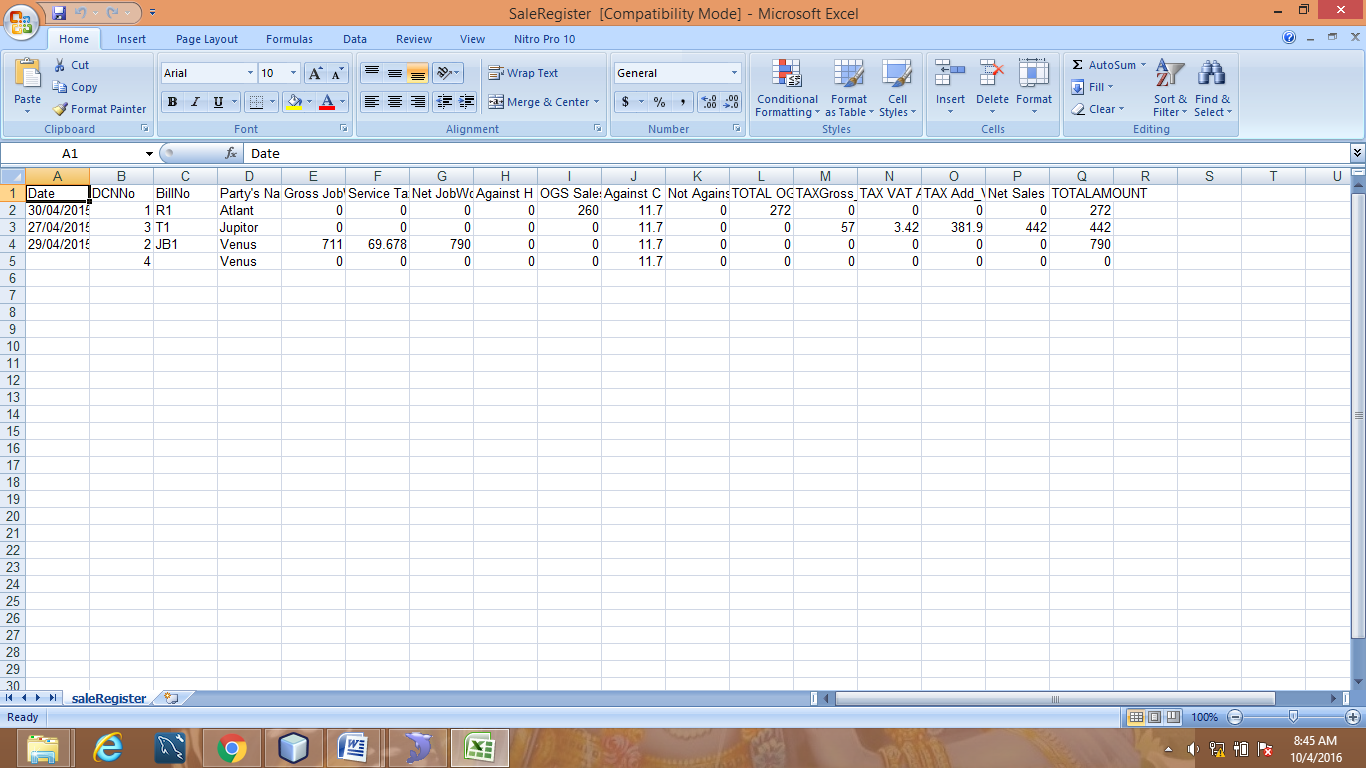
****

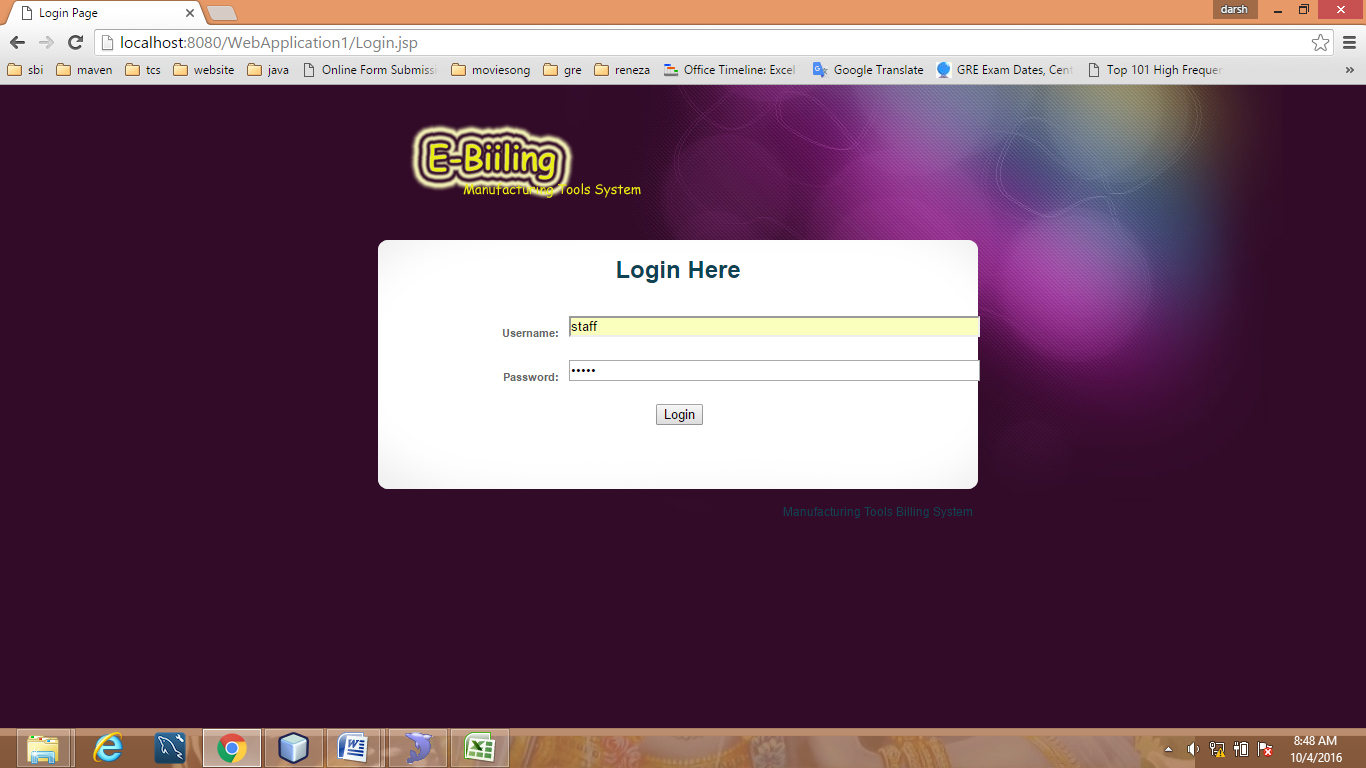
Now admin store all the data in excel file from database

****

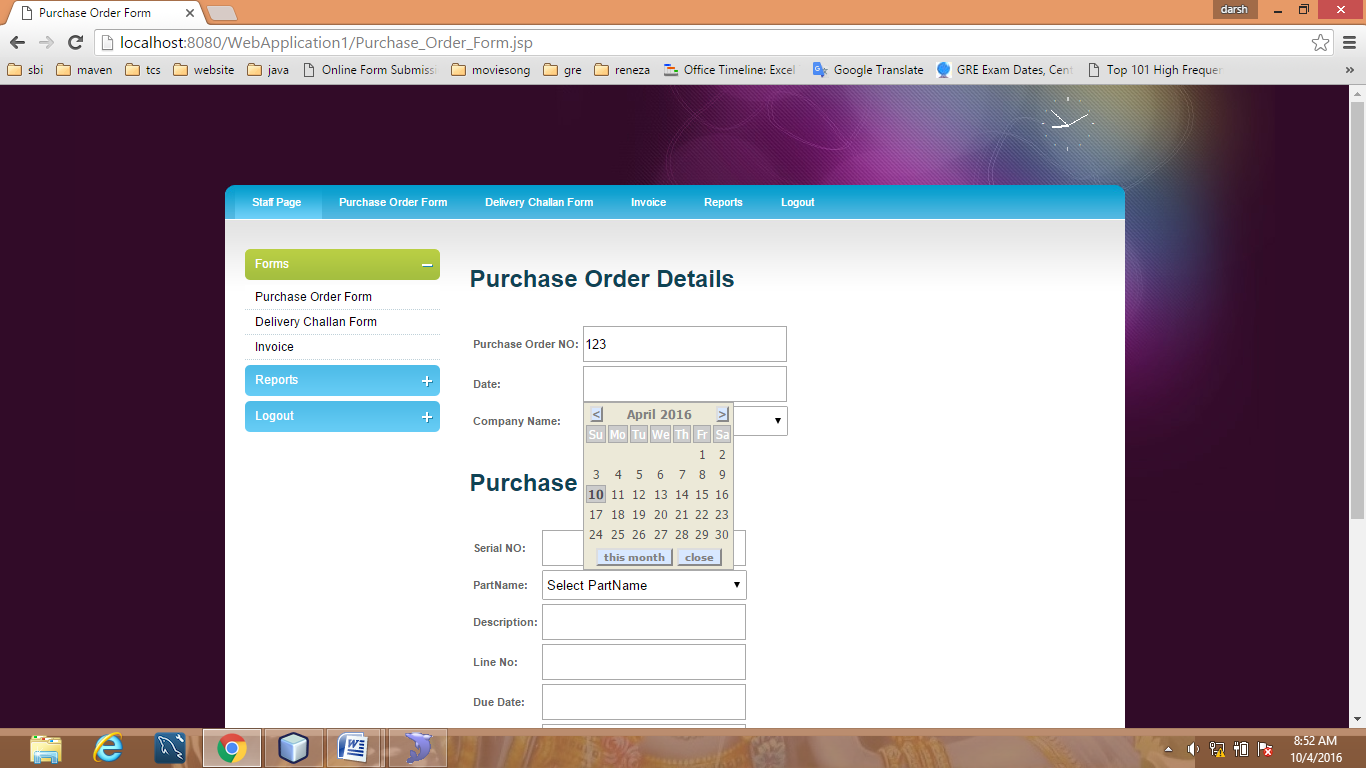
****

You show excel file

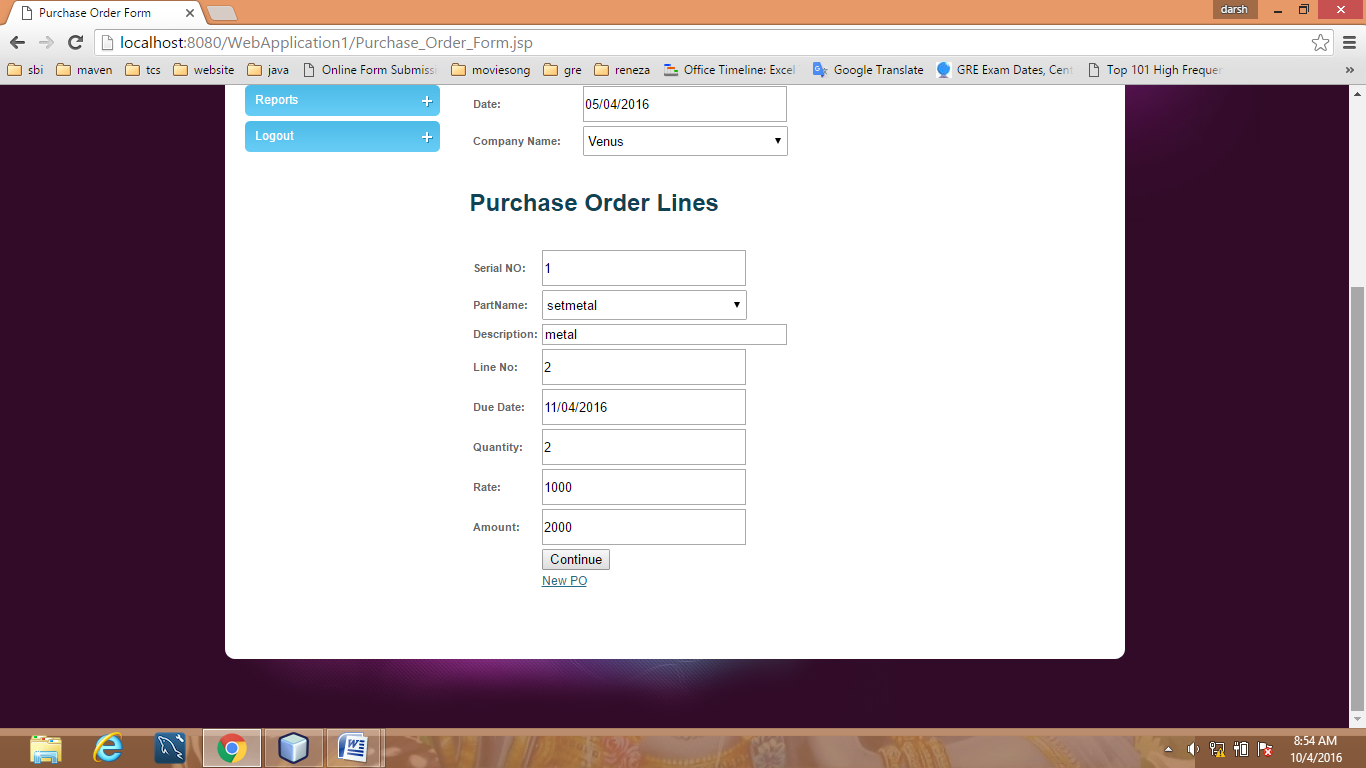
****

****

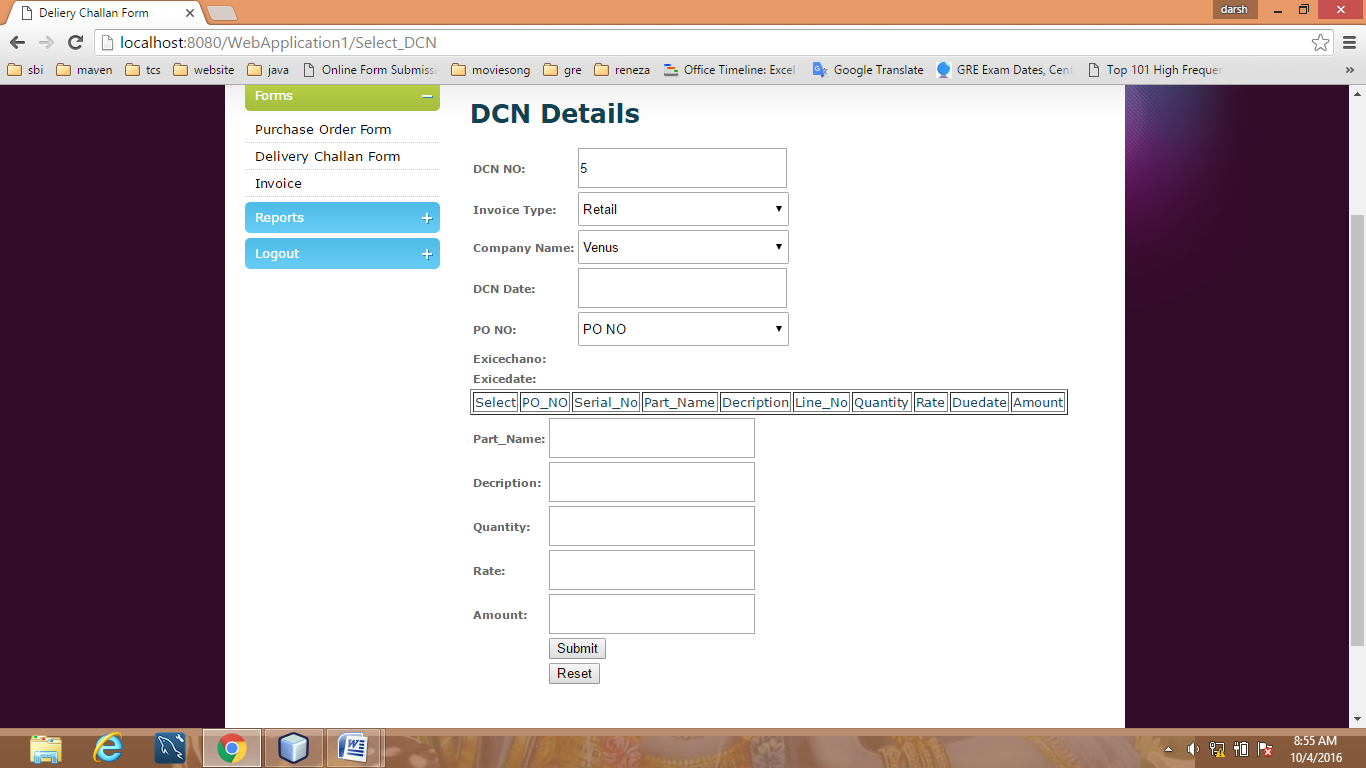
Staff enter correct password and username

****

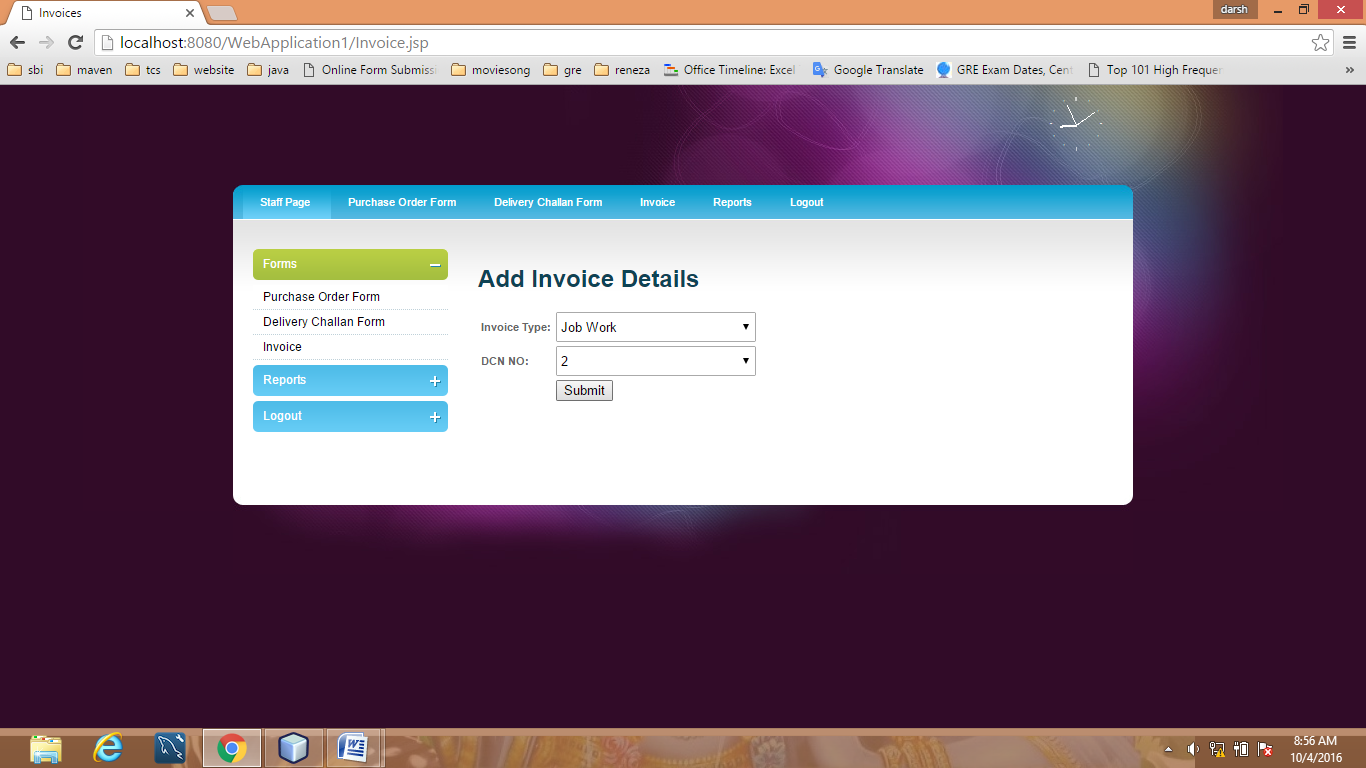
Fill purchase order form

****

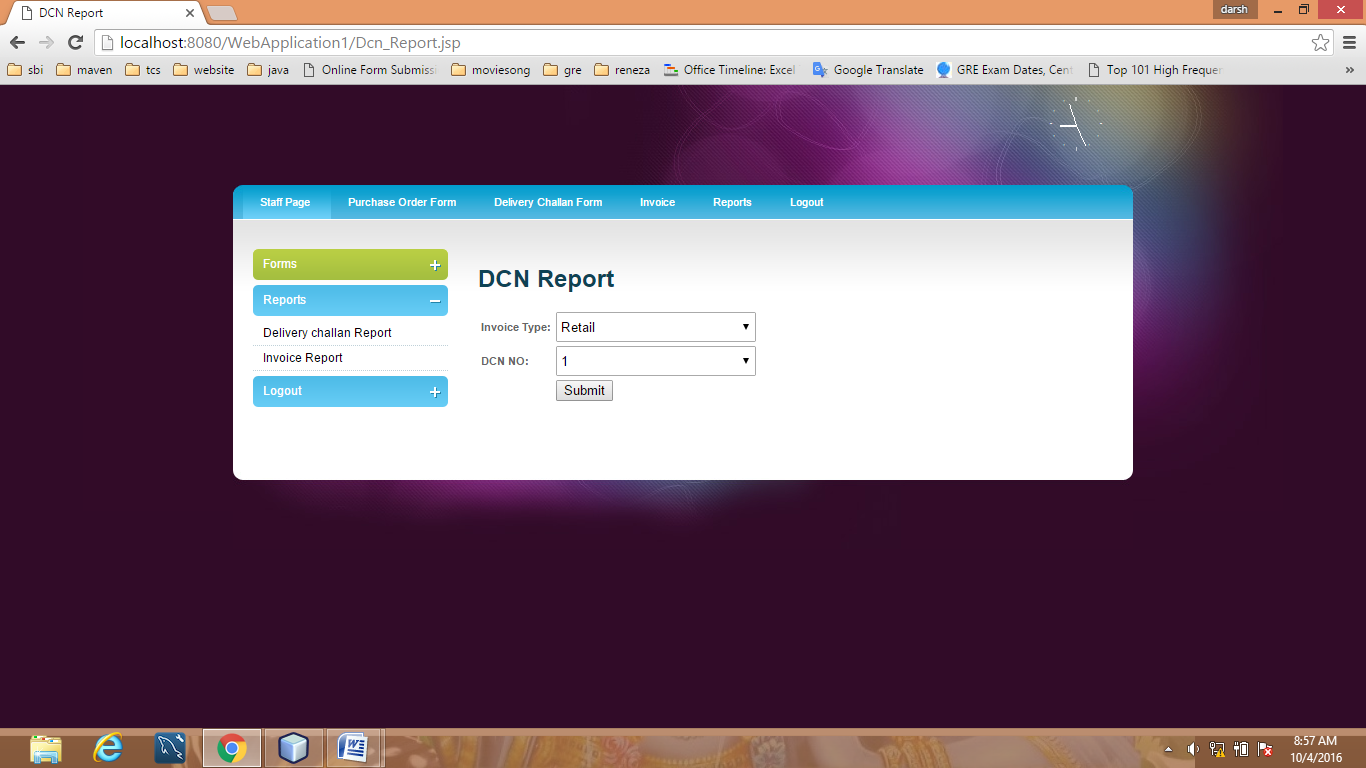
**Fill the delivery challan form**

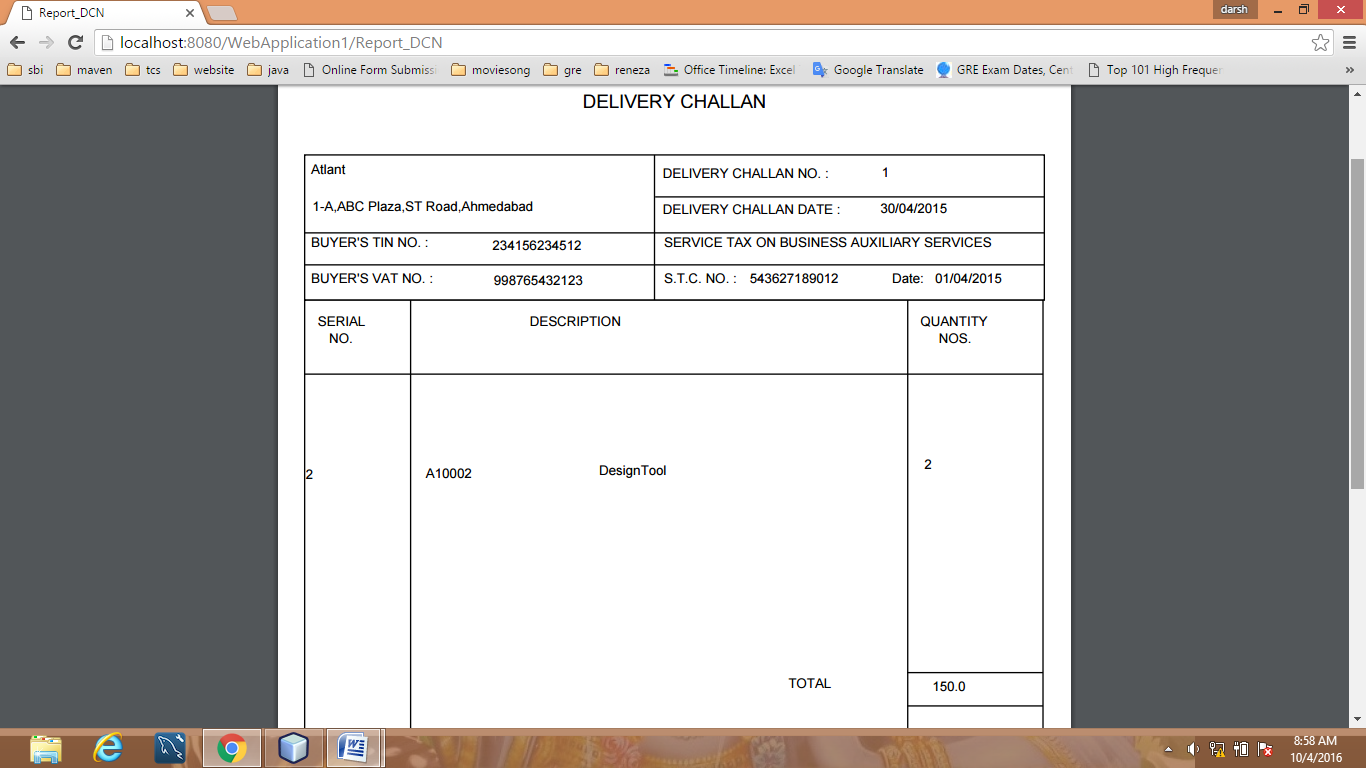
****

Select invoice

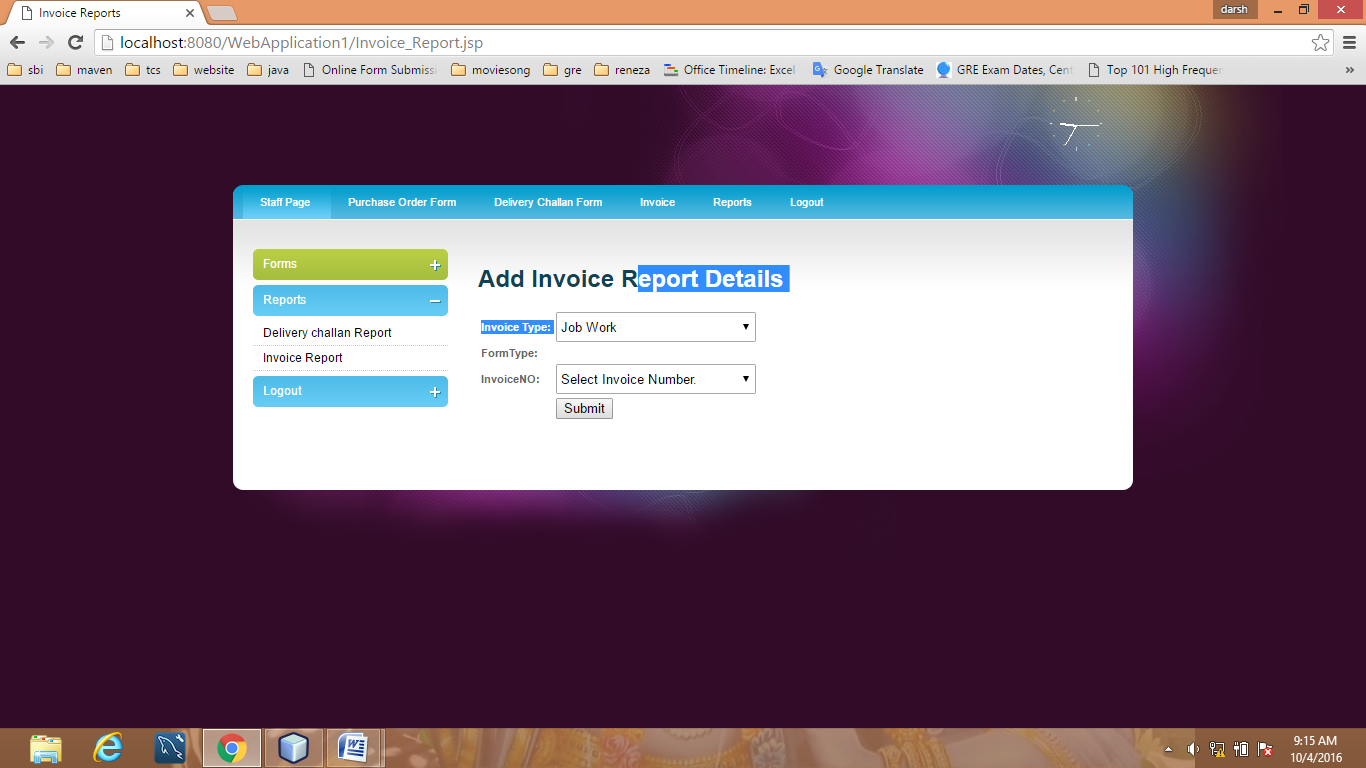
****

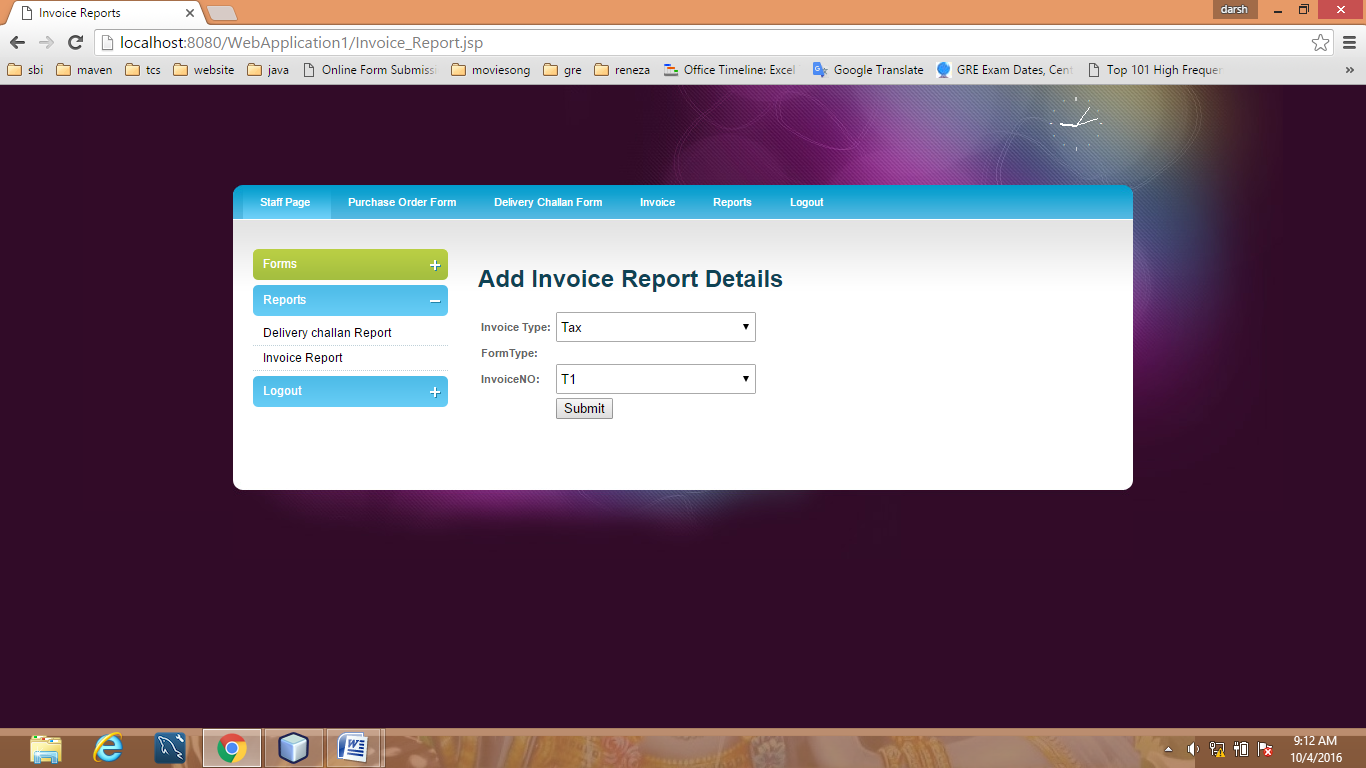
**Generate DCN report**

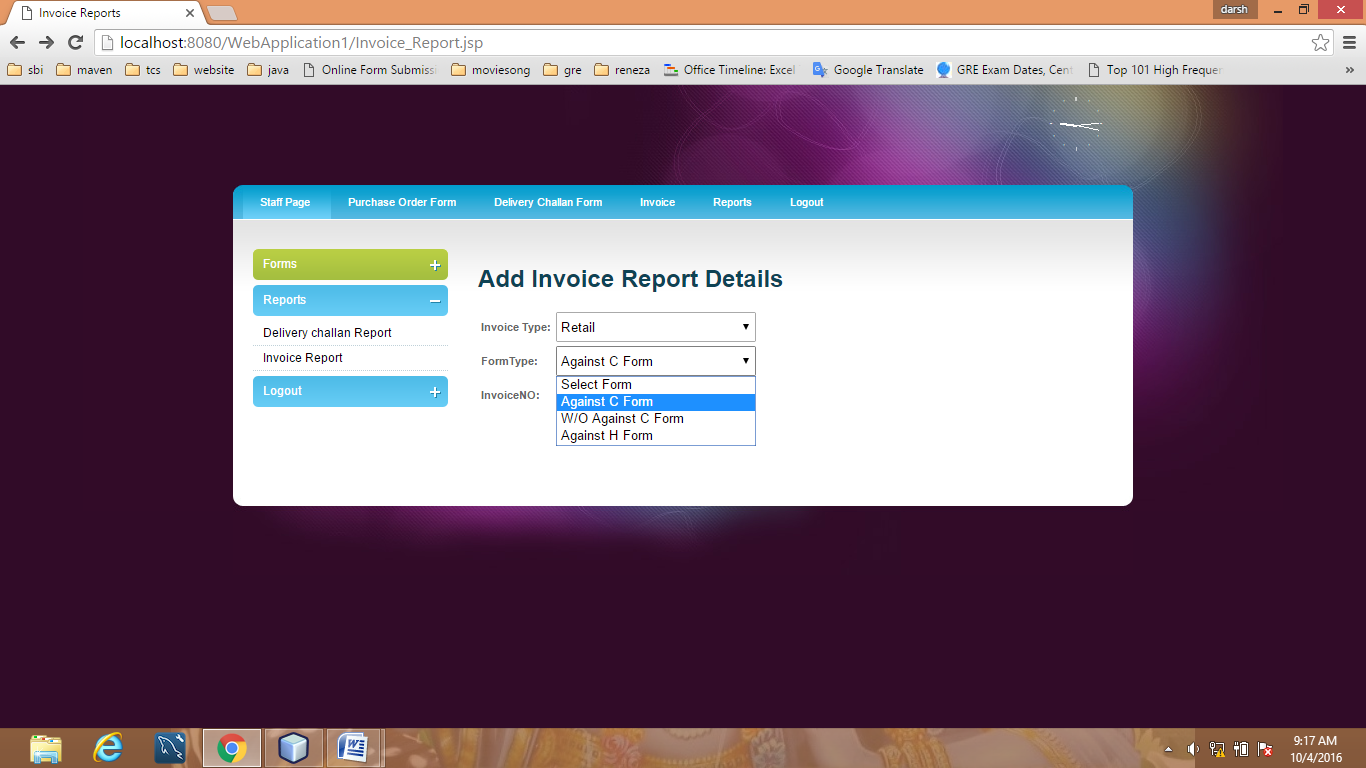
****

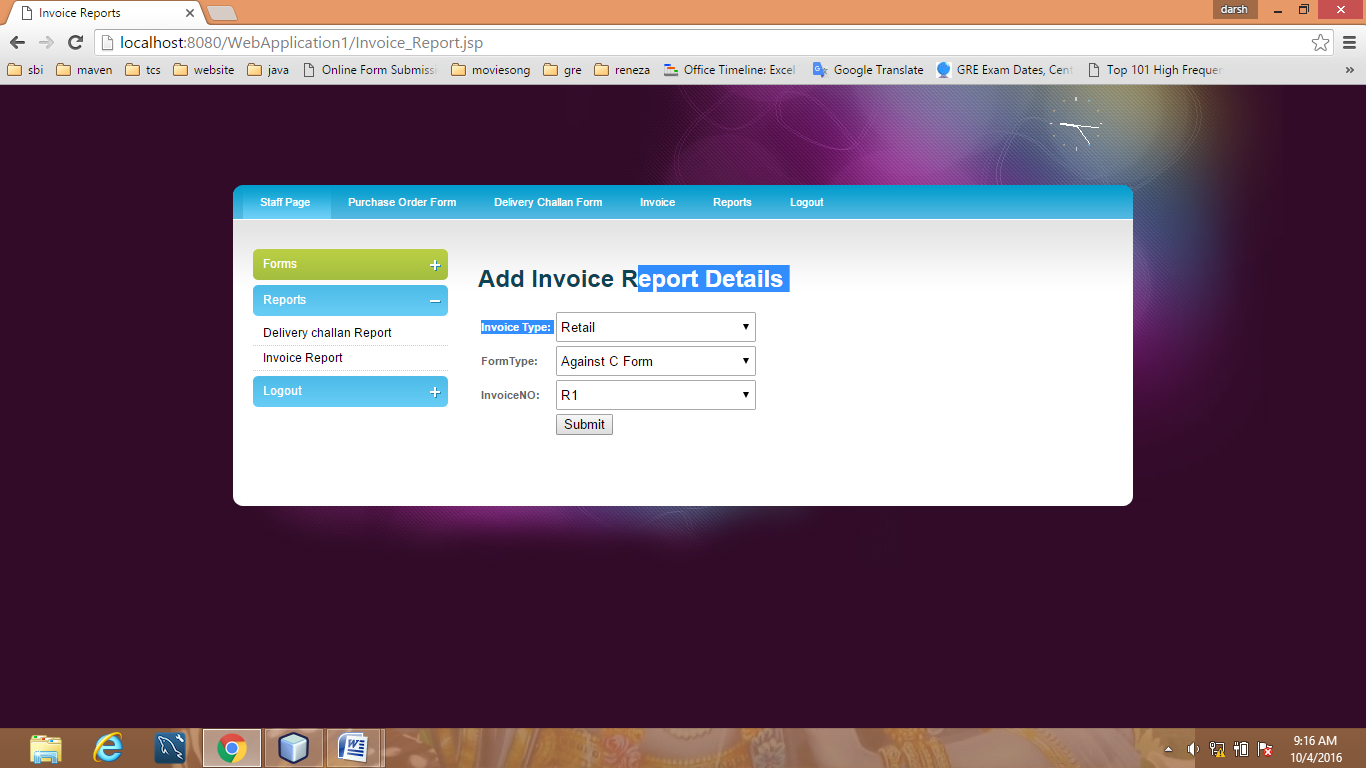
****

**Generate invoce report**

****

****

****

****

****

**6 System Testing**

## 6.1 Testing Method

### Black-Box Testing

The technique of testing without having any knowledge of the interior workings of the application is called black-box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black-box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

### White-Box Testing

White-box testing is the detailed investigation of internal logic and structure of the code. White-box testing is also called **glass testing** or **open-box testing**. In order to perform**white-box** testing on an application, a tester needs to know the internal workings of the code.The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

### Grey-Box Testing

Grey-box testing is a technique to test the application with having a limited knowledge of the internal workings of an application. In software testing, the phrase the more you know, the better carries a lot of weight while testing an application.Mastering the domain of a system always gives the tester an edge over someone with limited domain knowledge. Unlike black-box testing, where the tester only tests the application's user interface; in grey-box testing, the tester has access to design documents and the database. Having this knowledge, a tester can prepare better test data and test scenarios while making a test plan.

## 6.3 Testing Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Test Cases name | Description | Input | Expected outcome |
| 1)Admin Login | Verify that admin login with valid username and password | Username=xyz and password=123 | On clicking of the login button admin home page appears |
| 2)company  Registration | Verify that admin will add all the details in registration page | Username=xyz, [email=xyz@gmail.com](mailto:email=xyz@gmail.com),  Ect. | On clicking of the submit button home page appears. |
| 3)staff Login | Verify that doctor login with valid username and password | Username=xyz, password=123 | On clicking of the login button staff home page appears. |

**7 conclusion**

This system is vrery useful to staff and admin of company.

8 Bibliography

http://www.javatpoint.com/ajax-tutorial

<http://www.javatpoint.com/javascript-tutorial>

<http://www.javaworld.com/article/2074594/java-security/reports-made-easy-with->

jasperreports.html