**INTRODUCTION**

* 1. **PROJECT OVERVIEW**

The **Enterprise Resource Planning** (ERP) system, show-cases a **business process management software for an organization** to use integrated applications to manage businesses and automate many back-office functions.

System includes modules not only related to business’s ERP but also to maintain its core, such as Projects of business, Quality controls, its Teamwork, Document management, Campaigning, Helpdesk, etc., with more features and advanced building logics.

* 1. **PROJECT SUMMARY**

The Project is part of ABS, a product developed and managed by Axelor Technologies. The Axelor Business Suite allows you to follow the progress of business task by task, as well as their progress through the automatic calculation of their advance or their delay apart from that system includes GST modules for tax calculations of the purchase and sales. Provides a better view of data to measure, report and analyze, whereupon it is showed in form of cards, graphs, charts and calendar events to easily identify the data. And also, data is highlighted according to their status as planned.

* 1. **PURPOSE**

The purpose of Axelor business suite is to allow you to follow the progress of business task by task, as well as their progress through the automatic calculation of their advance or their delay apart from that system includes GST modules for tax calculations of the purchase and sales. Provides a better view of data to measure, report and analyze, whereupon it is showed in form of cards, graphs, charts and calendar events to easily identify the data. And also, data is highlighted according to their status as planned. Also managing relation with customer handling through CRM, and employee’s management through HRM module, providing marketing campaigns

* 1. **OBJECTIVE**

The main objective of any ERP is Centralized management of Data. Apart from that Objectives of ABS Falls under the same tree with different goals such as,

Better Information Management through Business Automation

Improved Workflow

Streamlining Of Processes

Modular Yet Integrated Approach

Elimination of Redundancies

Providing Standalone as well as integrated approaches.

**1.5 SCOPE**

The Axelor Business Suite modules not only related to business’s ERP but also to maintain its core, such as Projects of business, Quality controls, its Teamwork, Document management, Campaigning, Helpdesk, etc., with more features and advanced building logics and developments.

Abs is way ahead of its competitors in terms of rapid developments as well as providing modules according to client’s requirements.

**1.6 TECHNOLOGY**

**Technologies:**

* Technologies: HTML, JavaScript, JQuery, NodeJS, JAVA8,Java
* Framework: Axelor Development Kit
* Databases: PostgreSQL
* Web/Application server: TomCat 8.5

**Tools**

* IDE: Eclipse

**PROJECT MANAGEMENT**

**2.1 PROJECT PLANNING**

**2.1.1 Project Development Approach and Justification (Process Model Used)**

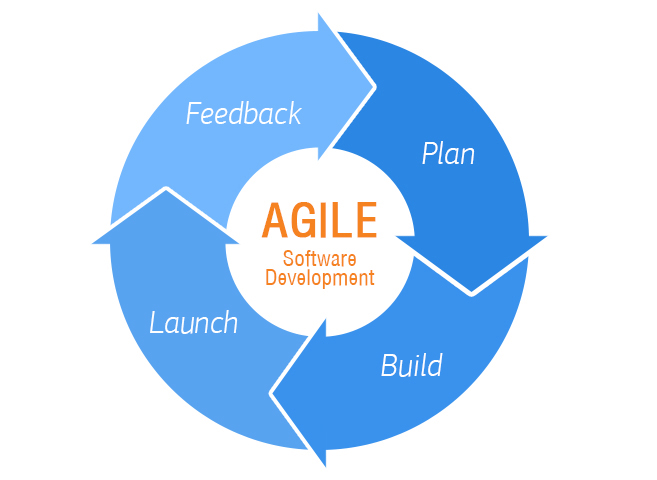
Agile software development methodology is a process for developing software which differs

Significantly from other methodologies. In English, Agile means ability to move quickly and easily and responding swiftly to change – this is a key aspect of agile software development as well.

With Agile development methodology –

* In the Agile methodology, each project is broken up into several ‘Iterations’.
* All Iterations should be of the same time duration (between 2 to 8 weeks).
* At the end of each iteration, a working product should be delivered.
* In simple terms, in the agile approach the project will be broken up into 10 releases (assuming each iteration is set to last 4 weeks).
* Rather than spending 1.5 months on requirements gathering, in agile software development,
* The team will decide the basic core features that are required in the product and decide which of these features can be developed in the first iteration.
* Any remaining features that cannot be delivered in the first iteration will be taken up in the
* Next iteration or subsequent iterations, based on priority.
* At the end of the first iterations, the team will deliver a working software with the features that were finalized for that iteration.
* There will be 10 iterations and at the end of each iteration, the customer is delivered a working software that is incrementally enhanced and updated with the features that were shortlisted for that iteration.

The iteration cycle of an agile project is shown in the image below.

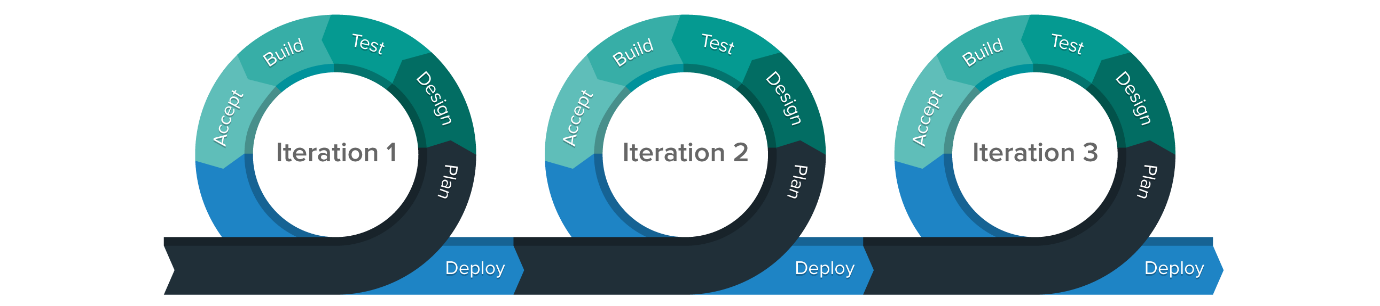


**Figure 2.1 Iteration cycle of an agile project**

This approach allows the customer to interact and work with functioning software at the end of each

Iteration and provide feedback on it. This approach allows teams to take up changes more easily and make course corrections if needed. In the Agile approach, software is developed and released incrementally in the iterations. An example of how software may evolve through

Iterations is shown in the image below.



**Figure 2.2 Evolution of a Software during Agile Process**

Agile methodology gives more importance to collaboration within the team, collaboration with the customer, responding to change and delivering working software. In the traditional approach, each job function does its job and hands over to the next job function. The previous job functions have to sign off before it is handed over the next job function authenticating that the job is full and complete in all aspects. For example, Requirement gathering is completed and handed over to the design phase and it is subsequently handed over to the development and later to testing and rework. Each job function is a phase by itself.

In an Agile way of working, each feature is completed in terms of design, development, code, testing and rework, before the feature is called done. There are no separate phases and all the work is done in single phase only.

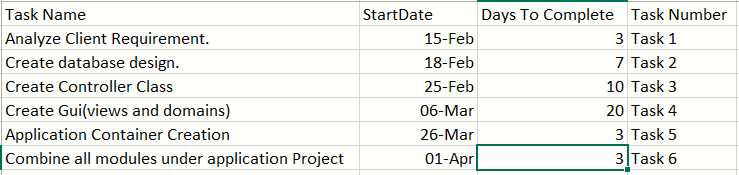
**2.1.2 Tools Used**

The Project includes various technologies in Front-end and Back-end. The Axelor Business Suite is developed Using Axelor Development Kit.

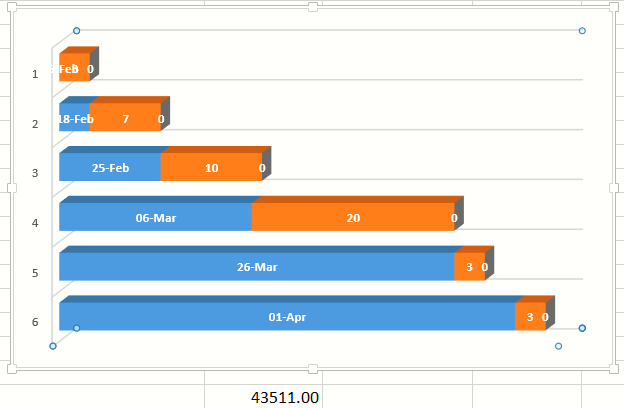
* Eclipse: Integrated Development Environment.
* PostgreSQL: PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads.
* Hibernate:Hibernate is an open source object relational mapping (ORM) tool that provides a framework to map object-oriented domain models to relational databases for web applications.

**2.2 PROJECT WORK SCHEDULING**

**2.2.1 Gantt Chart**

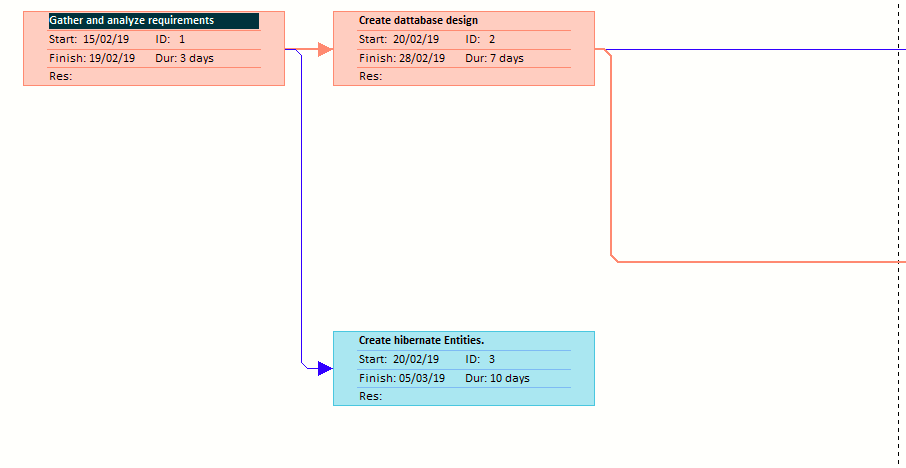


**Fig 2.3 Gantt chart**

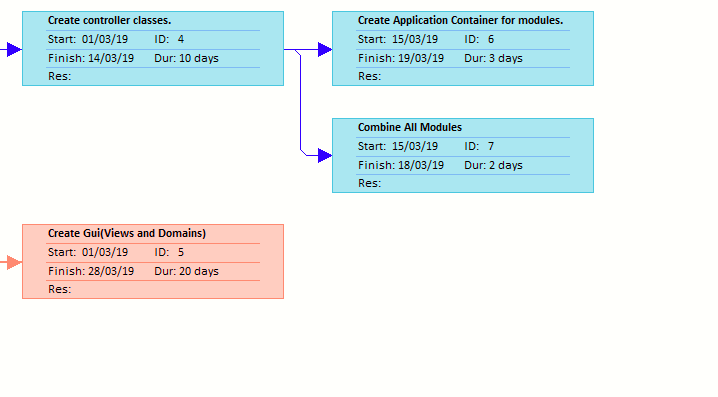


**Fig 2.4 Gantt chart**

**2.2.2 PERT Chart**



**Fig 2.5 PERT Chart- I**

**Fig 2.5 PERT Chart- II**

**2.3 FEASIBILITY STUDY**

Once the scope has been identified, it is reasonable to ask whether we can build software that meets this scope. Is this project feasible?

The feasibility of software can be tested in four dimensions:

**2.3.1 Technical Feasibility**

What we have planned to implement is technically feasible. Do we have a sufficient amount of knowledge or technology to make it a reality? And the answer is fairly easy because we have found out that development is done through the ADK(Axelor Development Kit) and its quite easy to manage the Controller and model with the hibernate entities.

**2.3.2 Time Schedule Feasibility**

We checked whether our system can be ready in time without any error. We have planned all its phase keeping the aspect in our mind, that if we find any bug or error after testing phase then we can move our deadline to 2-4 days, as we set our deadline before the actual submission date to the client

**2.3.3 Operational Feasibility**

How the project will work and who will use it, all such concerns arise in this phase. We have to study the requirements of the client carefully as its just raw data with no functional direction.

Our Client ranges in various fields form manufacturing of goods to IT sector fields.

**2.3.4 Implementation Feasibility**

The Implementation of project is checked in here.

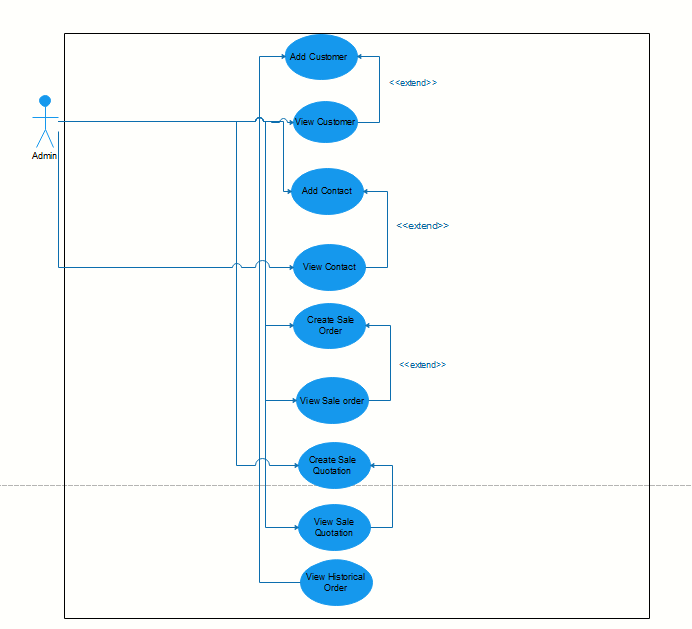
We have found out that with MVC structure and rapid development provided by Hibernate entities it’s quite easy to implement the requirements of clients in the real-world applications.

**System Requirements Study**

**3.1 USER CHARACTERISTICS**

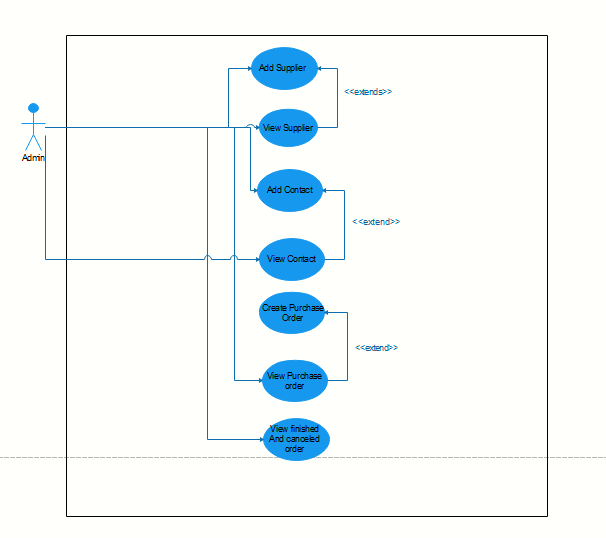
This application mostly intended to be used by the administrator or User with Access Rights of using it.

**Sale Module**



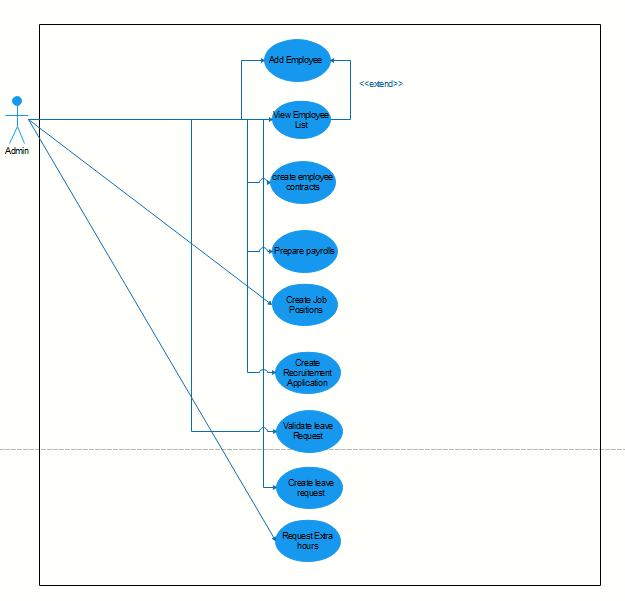
**Fig 3.1 Use-Case Diagram**

**Purchase Module**



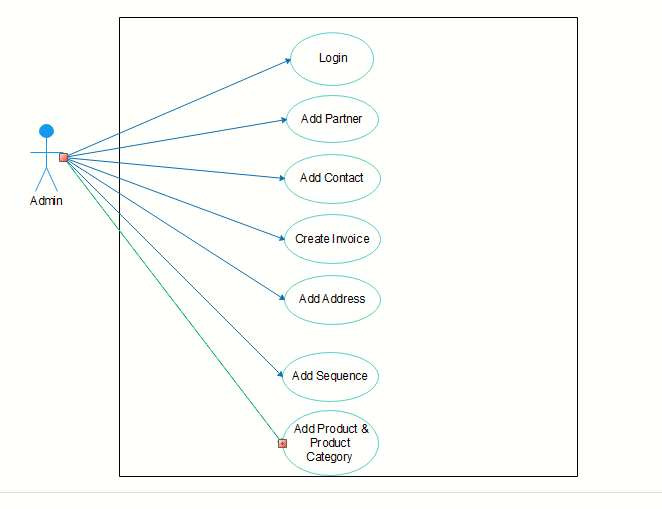
**Fig 3.2 Use-Case Diagram**

**HR Module**



**Fig 3.3 Use-Case Diagram**

**GST Module**



**Fig 3.4 Use-Case Diagram**

**3.2 Hardware and Software Requirements**

Hardware configuration of machine used

|  |  |  |
| --- | --- | --- |
| *  | * Processor Speed | : Intel Core I5 with 2.40 GHz |
| *  | * RAM | : 4.00 GB or Higher |
| *  | * Hard Disk | : 80 GB or Higher |

Software used along with the version

* Operating System: Linux
* Language: Java
* Technologies: HTML, JavaScript, JQuery, NodeJS, JAVA8
* Framework: Axelor Development Kit
* Databases: PostgreSQL
* Web/Application server: TomCat 8.5

**3.3 ASSUMPTIONS AND DEPENDENCIES**

Following are the assumptions that have been taken while documenting the report:

* The code is error-free and bug-free.
* The instance has enough storage to store the data.
* The user is well-versed in the English language.
* All the software and hardware requirements are fulfilled.

Following are the dependencies that should be taken care of before using this software:

* This application is based on Java, Gradle, Hibernate, and PostgreSQL.
* The Application Should be hosted on a Server.

**SYSTEM ANALYSIS**

**4.1 STUDY OF THE CURRENT SYSTEM**

There are several existing ERP’s available in the market. Some of the famous ERP are the SAP. One common thing between all the ERP is the centralised management of data. After reading about existing ERP, we found that there is no feature provided to customer about changing the fields of the form of various module. Here’s where ABS differs where change in the views or domain of the system can be done by the user.

**4.2 PROBLEMS AND WEAKNESSES OF THE CURRENT SYSTEM**

The existing system does provide the rapid development of views, domain and controller but with ADK we can achieve this. Also a client with bit of technical knowledge can make changes in the views and domains, so for changes client doesn’t have to reach developer each and every time.

Usually the ERP software’s are large and very complex to understand but in ABS complexity is reduced by introducing Hibernate entities, what it does is it act as an layer between the underlying Database and the controller.

**4.3 REQUIREMENTS OF NEW SYSTEM**

**4.3.1 Functional Requirements**

Manage Sales: This application manages the sales of the company. It allows to create quotations, to print them or to generate them in PDF format and to send them directly from the application. Once the estimate has been validated and finalized, it automatically changes into a sales order.

Manage Purchase: It will include the basic functionalities related to purchasing of any kind of raw materials or things required in organizations. It also specifies functionalities to add suppliers manage contracts, purchase orders viewing historical data related to purchases made.Manage Customer: CRM module helps to manage & track detailed information of the customer like communication history ,calls, meetings, details of purchases made by customer, contract duration etc. CRM module can be integrated with Sales module to enhance sales opportunities.Manage Modules: App builder will allows the clients to make changes in their system without consulting any help from the owner the ERP, it reduces the time and cost of maintenance for the both parties.

**4.3.2 Non Functional Requirements**

Authentication: Proper Authentication has been maintained, along with flexibility to modify and view data.Minimized Code: We have minimized the lines of code as well as kept reusability of code as top priority. Less processing and loading time has been kept under consideration.Performance Improvements: The application shall accommodate high number of Users without any fault.Customizable Modules: Many modules related to ERP has been included, out of which, according to the user’s requirement modules can be included, excluded and modifiable.Error Handling: System shall handle expected and non-expected errors in ways that prevent loss in information and long downtime period.

**4.4 FEASIBILITY STUDY**

Once the scope has been identified, it is reasonable to ask whether we can build software that meet this scope. Is this project feasible?

**4.4.1 Does The System Contribute to The Overall Objectives of The Organization?**

How the project will work and who will use it, all such concerns arise in this phase. We have to study what the existing system’s problem is, and is it worth solving or not. And yes System does contribute to the overall objective of the organization, as discussed early in the document the System has its unique features as being an ERP Product like rapid development, managing of modules that gives the product an edge in the market and in turn it does contribute the overall objective of he organization.

**4.4.2 Can the System be Implemented Using the Current Technology and within the Given Cost and Schedule Constraints?**

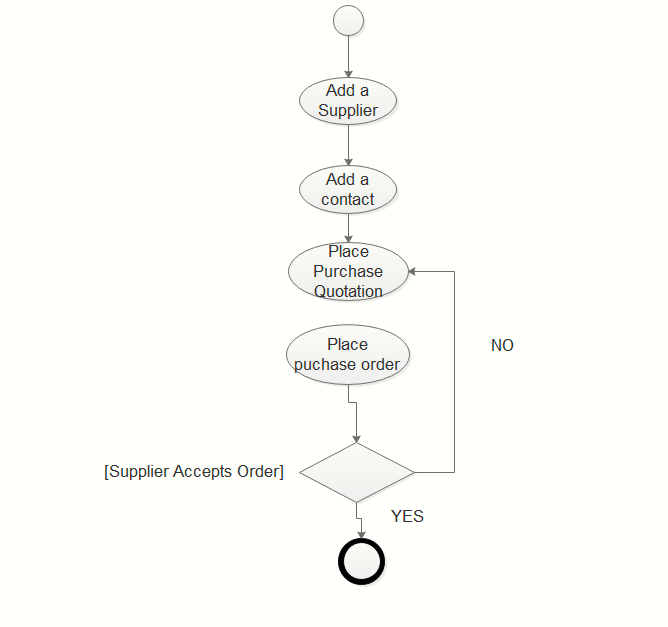
What we have planned to implement is technically feasible. Do we have a sufficient amount of knowledge or technology to make it a reality? And the answer is fairly easy because we have found that development done using the POJO classes so its very easy and efficient to develop modules and manage them.

**4.4.1 Can The System Be Integrated With Other Systems Which Are Already in Place?**

Our system cannot be integrated with other systems. It’s an ERP system developed and managed by the Axelor Technologies.

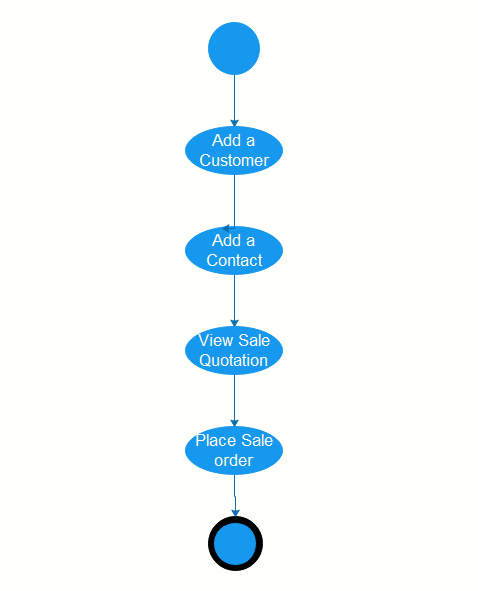
**4.5 Activity /Process in New System (Use event table/activity diagram)**

**Purchase Module**



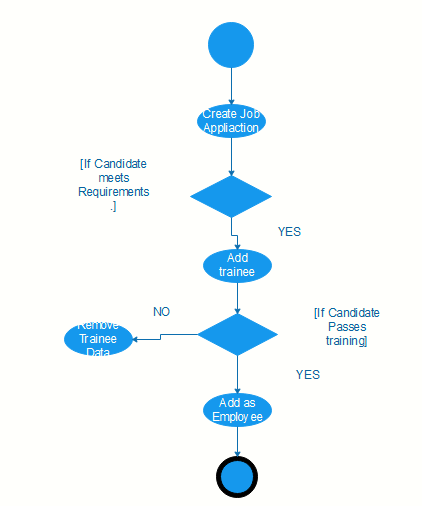
**Figure 4.1** Activity Diagram

**Sales Module**



**Figure 4.2** Activity Diagram

**HR Module**



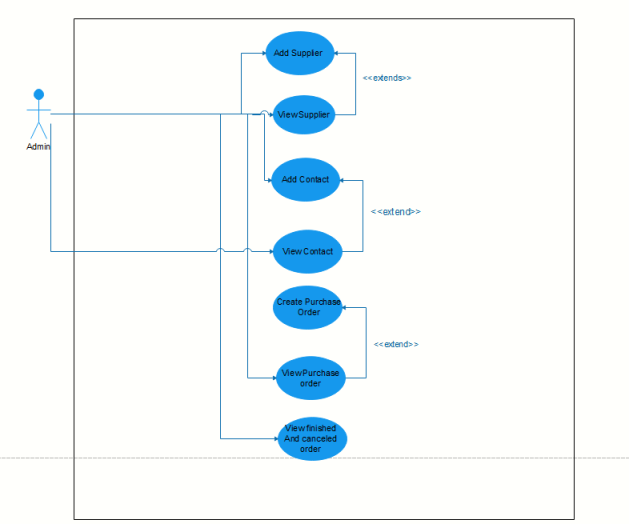
**Figure 4.3** Activity Diagram

**4.6 FEATURES OF NEW SYSTEM**

As discussed early during the document new features in the system can be considered as the rapid development as well as the handling of database through the POJO classes.

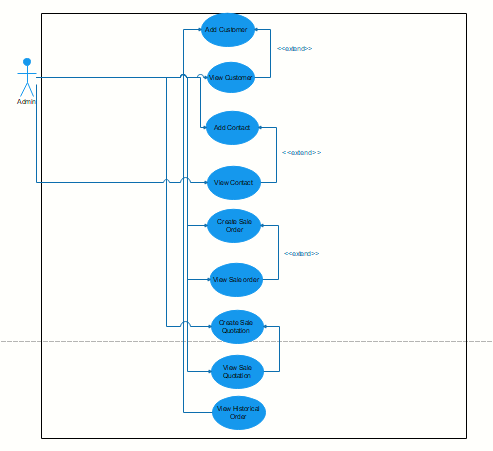
**4.7 USE CASE DIAGRAM**

**Purchase Module**



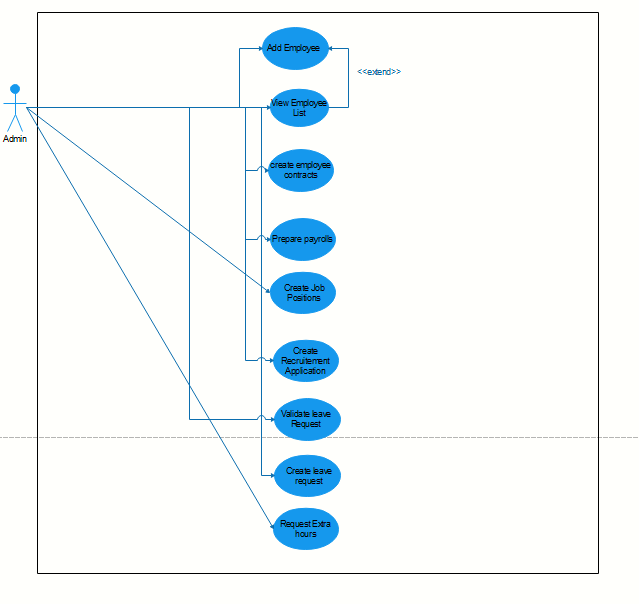
**Figure 4.4** Use Case Diagram

**Sale Module**



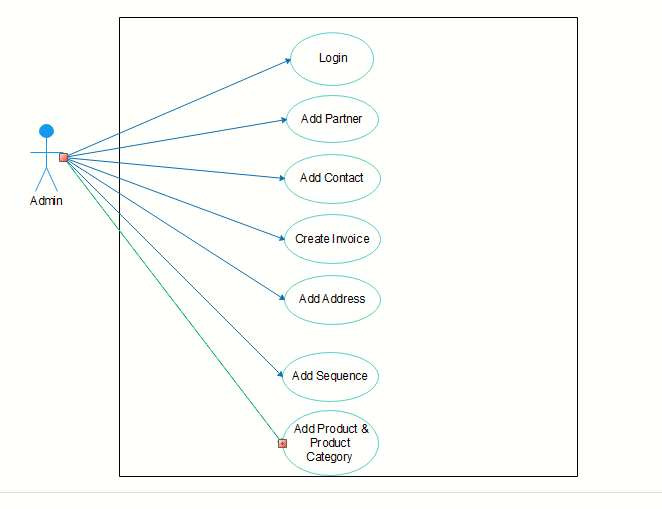
**Figure 4.5** Use Case Diagram

**HR Module**



**Figure 4.6** Use Case Diagram

**GST Module**



**Figure 4.7** Use Case Diagram

**4.8 List Main Modules of New System**

**Sales**

This application manages the sales of the company. It allows to create quotations, to print them or to generate them in PDF format and to send them directly from the application. Once the estimate has been validated and finalized, it automatically changes into a sales order.

**Purchase**

This application allows to manage the purchases of the company. It allows you to create quotations that, once validated and finalized, automatically transform into purchase orders. You can generate supplier arrivals from a purchase order to schedule the associated stock movements, and generate the control invoice.

The application also makes it possible to send quotation requests to several suppliers, thus making it possible to select the most competitive offer.

The sub-application "Purchase requests" generates pricing requests on specific products to your suppliers.

**CRM**

The CRM application allows you to manage company interactions with prospects and customers. First of all, it allows to manage the prospects.

The first step is to create leads to track sales and marketing of these new leads. A lead is usually created at the beginning of the business relationship.

It is a person/company interested in your products and services, with whom you are in the early stage of the business relationship.

**HR**

This application allows you to manage the company’s human resources. It consists of different sub applications covering a broad spectrum of human resources. It allows to

manage: - Management of employees including employee contracts, payroll preparation, restaurant vouchers and bonuses. - Leave requests. - Expenses. - Timesheets. - Extra hours. - Recruitment. - Training courses. - Appraisals.

**4.10 Selection of Hardware and Software Justification**

**4.10.1 Hardware Justification**

Ingesting voluminous data requires hardware of higher configuration that stores this data.

**4.10.2 Software Justification**

The Axelor Business suite is mainly developed on JAVA, and for that Eclipse is used ass IDE.

In ABS back-end the DB used is PostgreSQL along with that hibernate is used for quick development of the domains and getter setter methods.

**System Design**

**5.1 SYSTEM APPLICATION DESIGN**

**5.1.1 Method Pseudo code**

**Purchase Module**

Purchases/: Purchase management application.

Purchases/Suppliers/: This menu gives access to the list of suppliers and allows you to create new suppliers records.

Purchases/Contacts/: This menu gives access to the list of contacts and allows you to create new contacts records.

Purchases/Purchase requests/: Purchase request management application.

Purchases/Purchase requests/All purchase requests/: Displays purchase requests to suppliers and creates new ones.

Purchases/Purchase requests/All requests sent/: Displays purchase requests sent to suppliers and creates new ones.

Purchases/Purchase requests/All requests accepted/: Displays purchase requests that have been accepted.

**Sales Module**

Sales/: Sales management application.

Sales/Customers/: This menu gives access to the list of customers and allows you to create new customers records.

Sales/Contacts/: This menu gives access to the list of contacts and allows you to create new contacts records.

Sales/Products & services/: This menu gives access to the list of products/services and allows to create new product/ service forms.

Sales/Sale orders/ : Allows to create new sale orders and displays the current and completed orders.

**HR Module**

Human resources/ Human resources/Employee Management/: Application of human resources management.

Human resources/Employee Management/List of employees/: Displays the list of employees and allows you to create employee records.

Human resources/Employee Management/Employment Contracts/: Allows to create employment contracts for employees.

Human resources/Employee Management/Payroll Preparation/: Menu for managing employee payroll preparation.

Human resources/Employee Management/Lunch voucher management/: Allows to manage lunch voucher.

**GST Module**

GST/Party: Add, remove, and edit party

GST/Contact: Add, remove, and edit contact

GST/Invoice: Create sale invoice of the customer (party)

GST/Dashboard: view the paid, unpaid invoice per customer, customer per state, invoice per status and unpaid invoice of last month.

GST/Address: Add address to be used in party and contact creation and associate with it

**5.2 Database Design/Data Structure Design**

**GST Module**

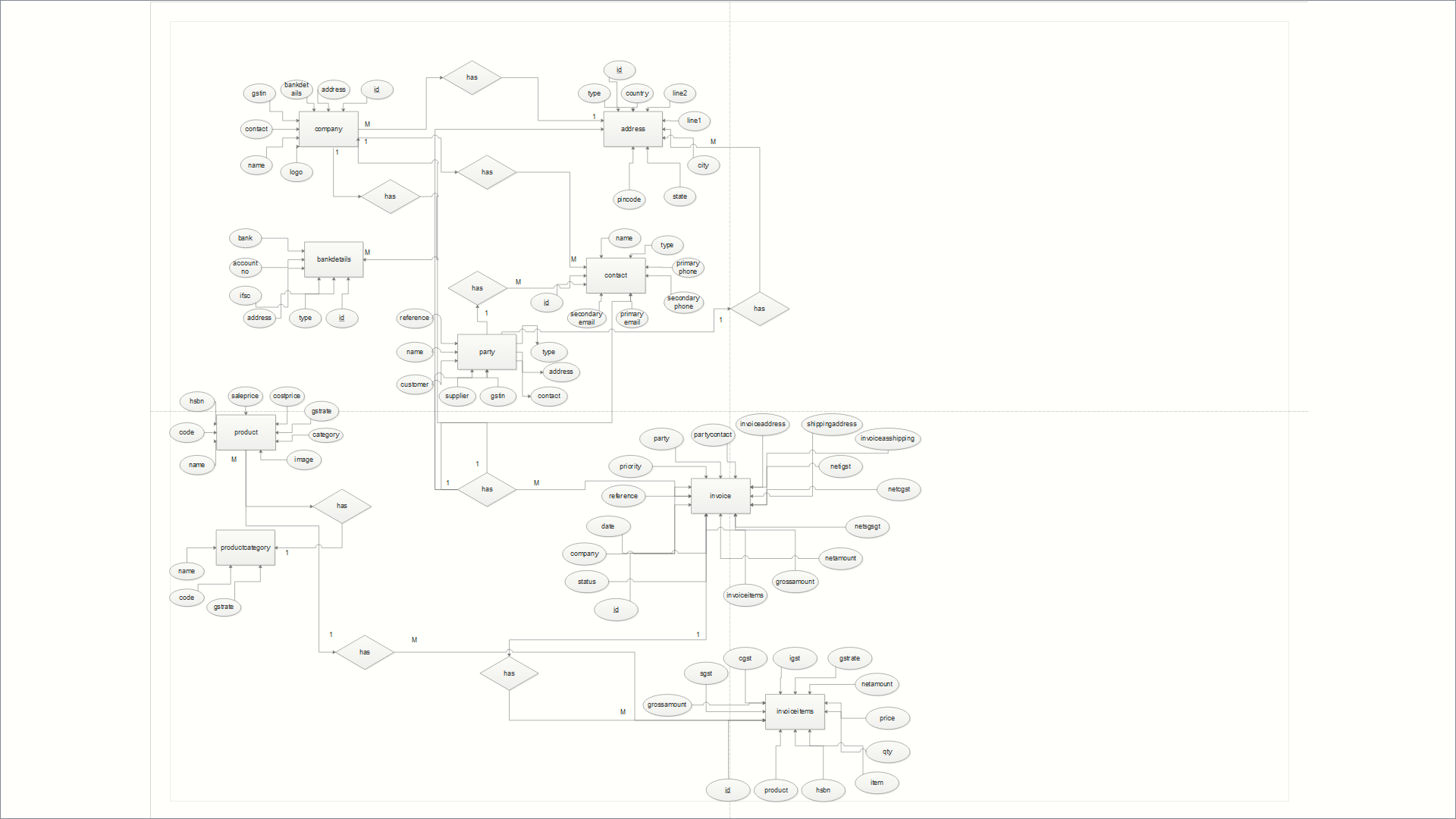
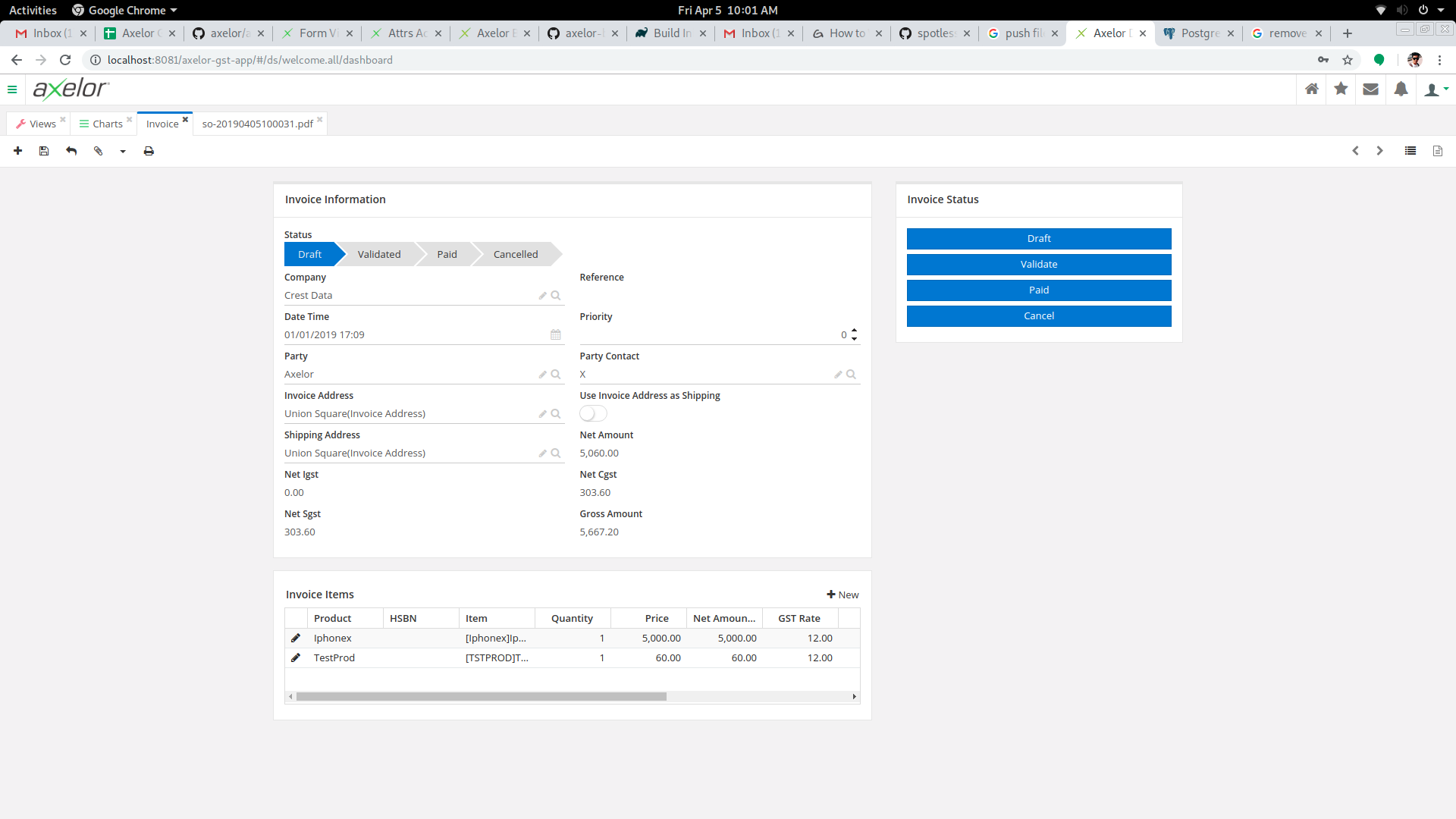


Fig 5.2.1 ER Diagram

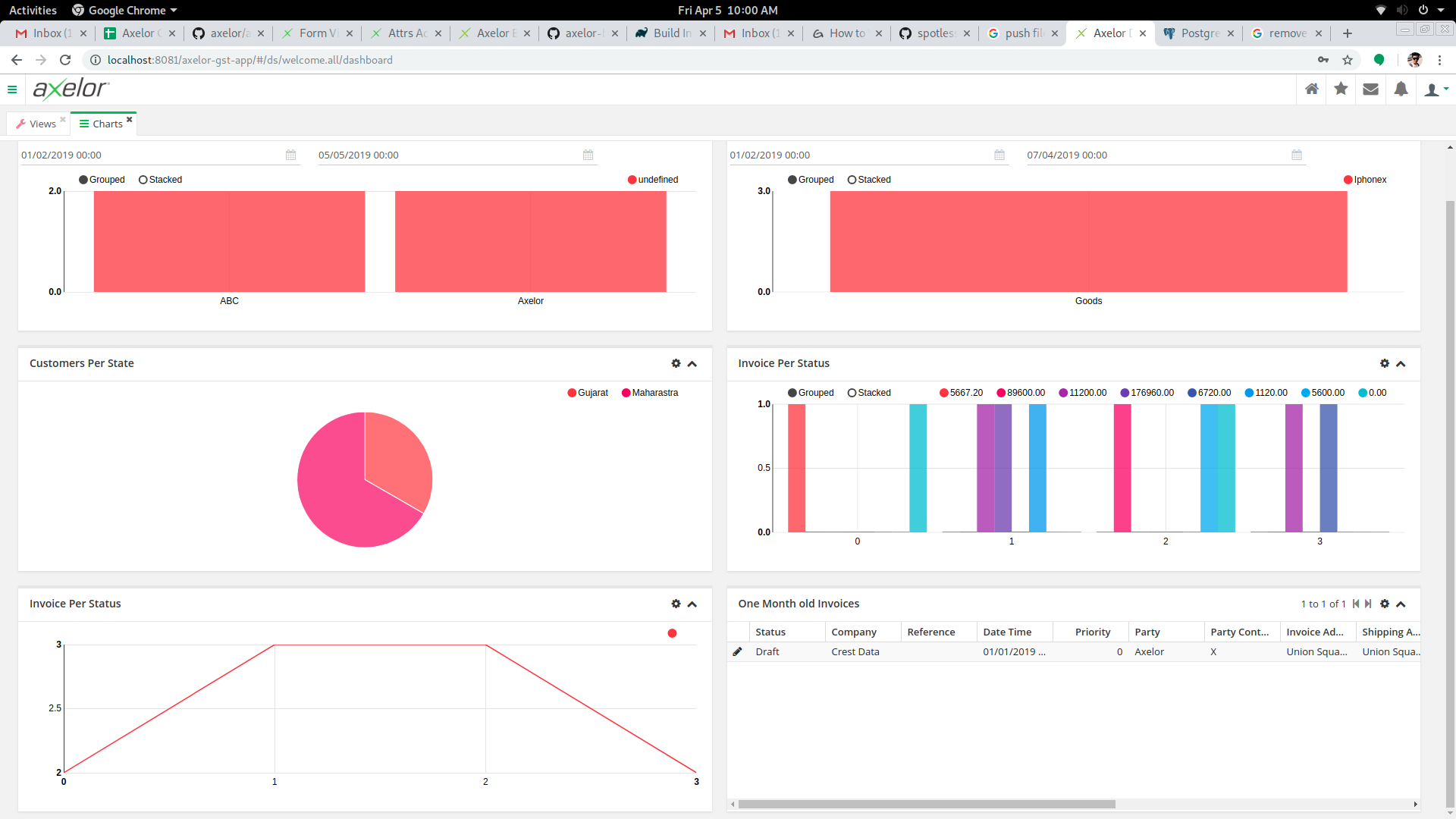
* 1. **INPUT/ OUTPUT INTERFACE DESIGN**

**5.3.1 Samples of Forms, Reports and Interface**



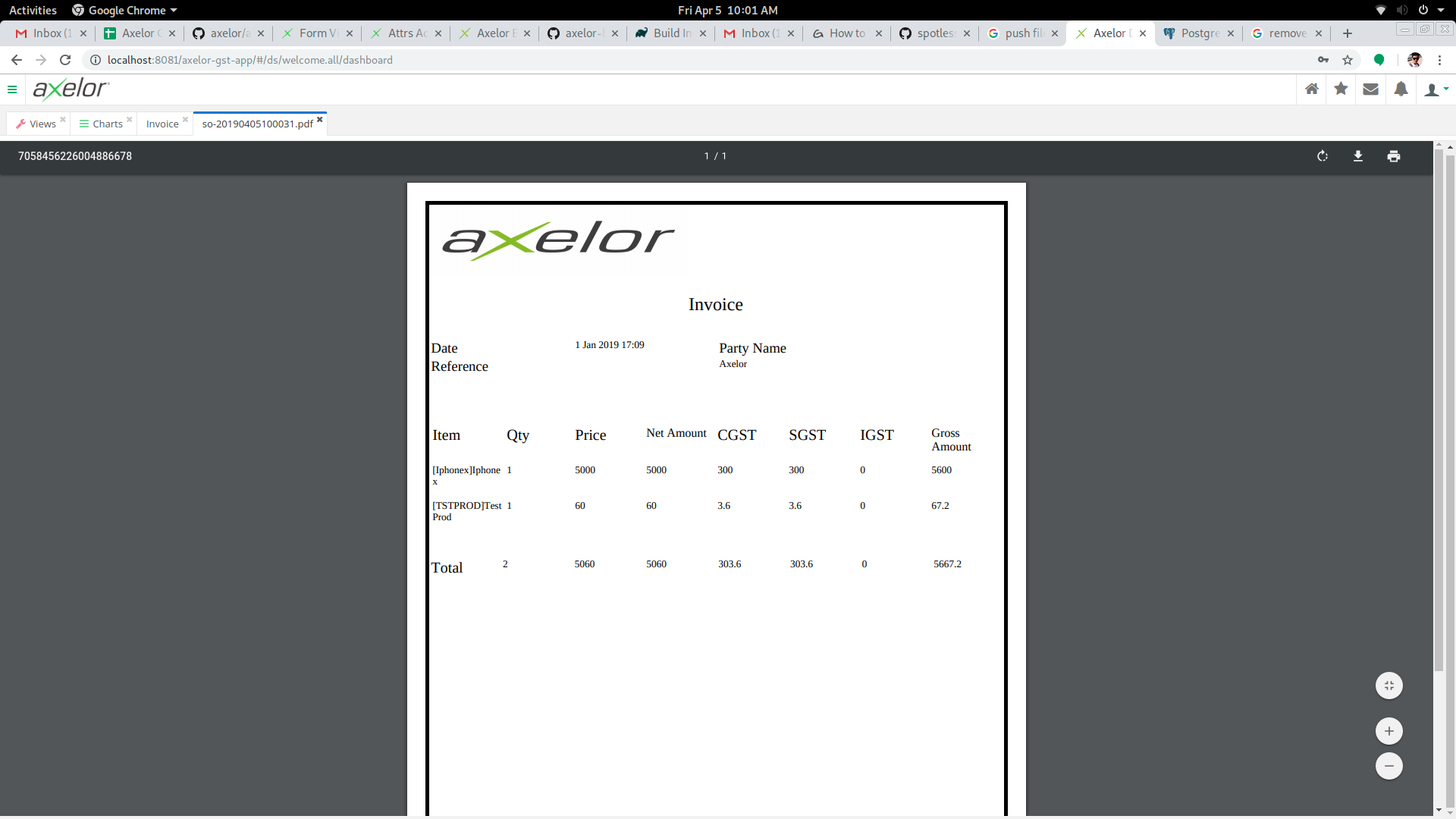
**Fig 5.3.1 Create Invoice**

User can generate sale invoice here.



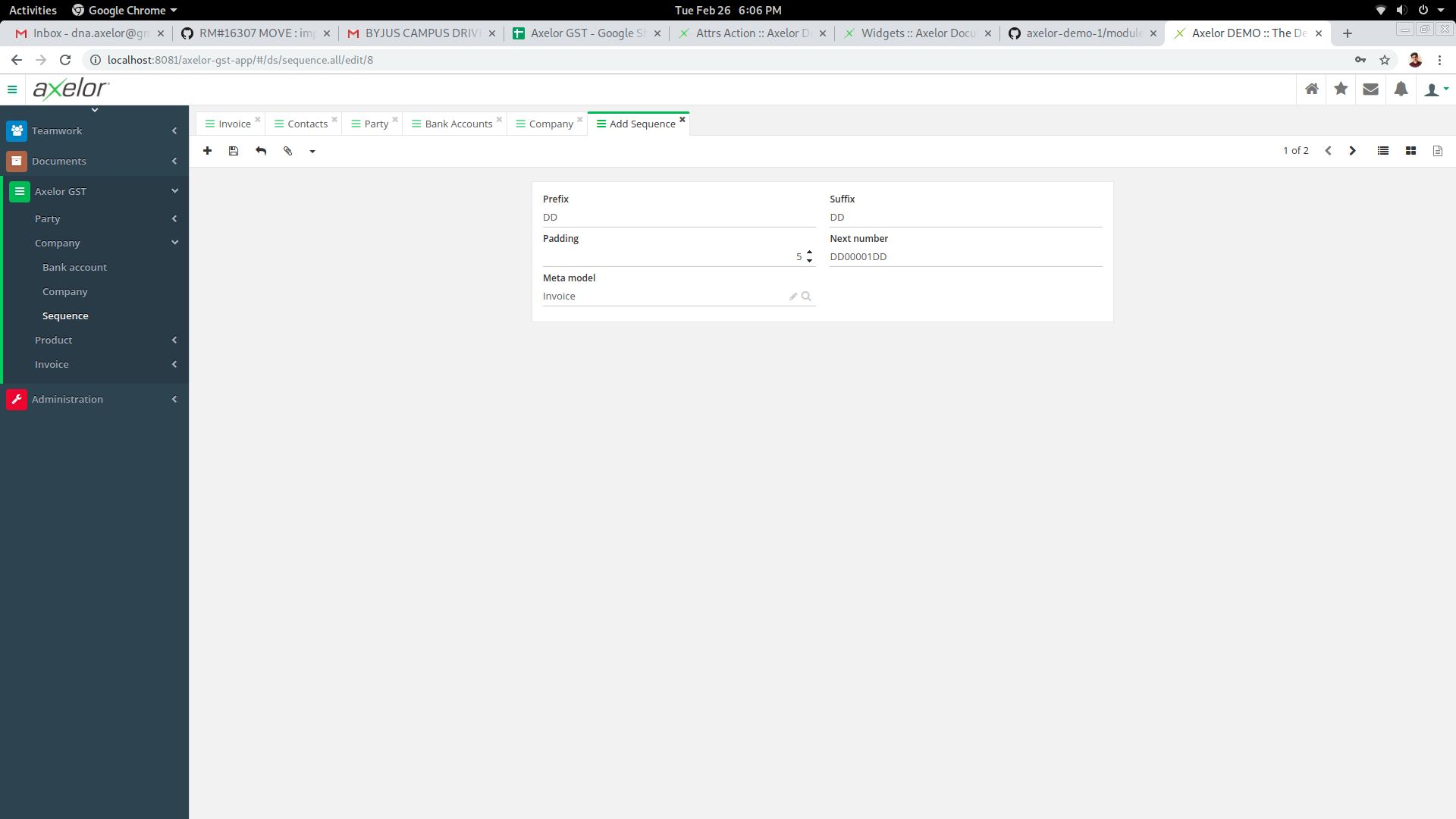
**Fig 5.3.2 Dashboard**

User can see the data in the form of graph generated through JPQL, which shows the invoice, user data.



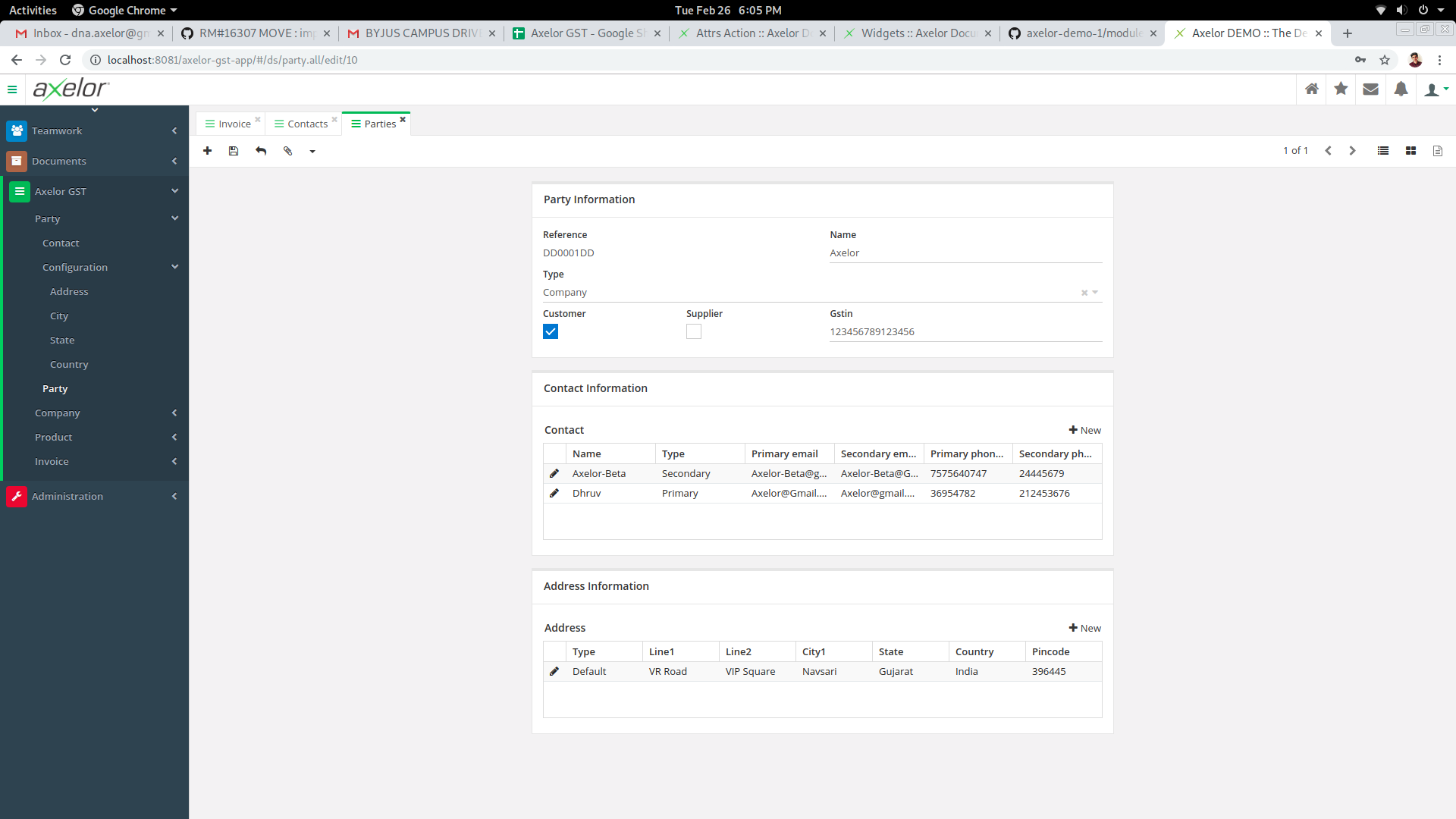
**Fig 5.3.3 Invoice Generation**

This is the bill generated of the sale invoice, which is created with BIRT report generator.



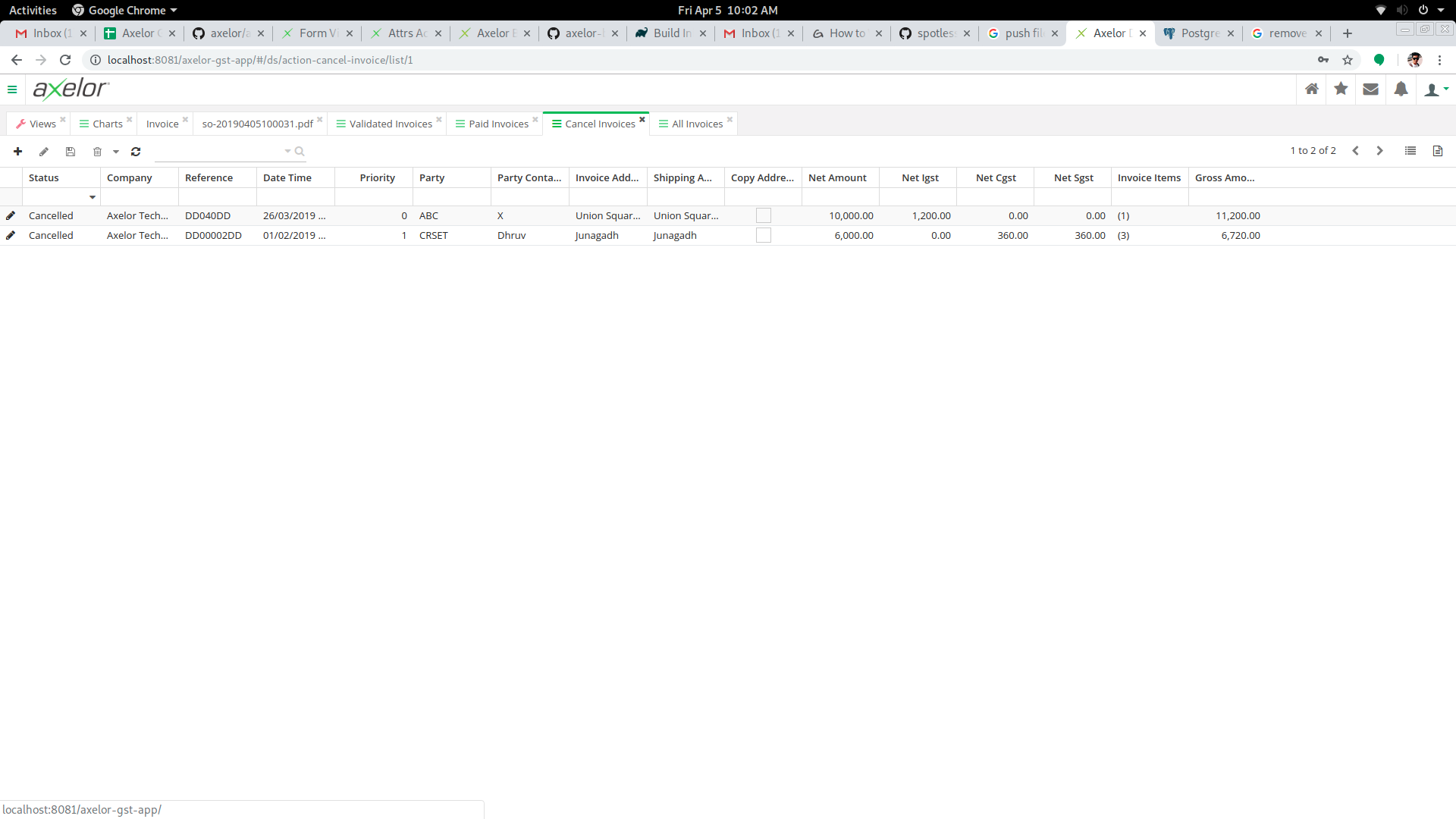
**Fig 5.3.4 Sequence Generation**

Here user can generate Sequence for party and invoice.



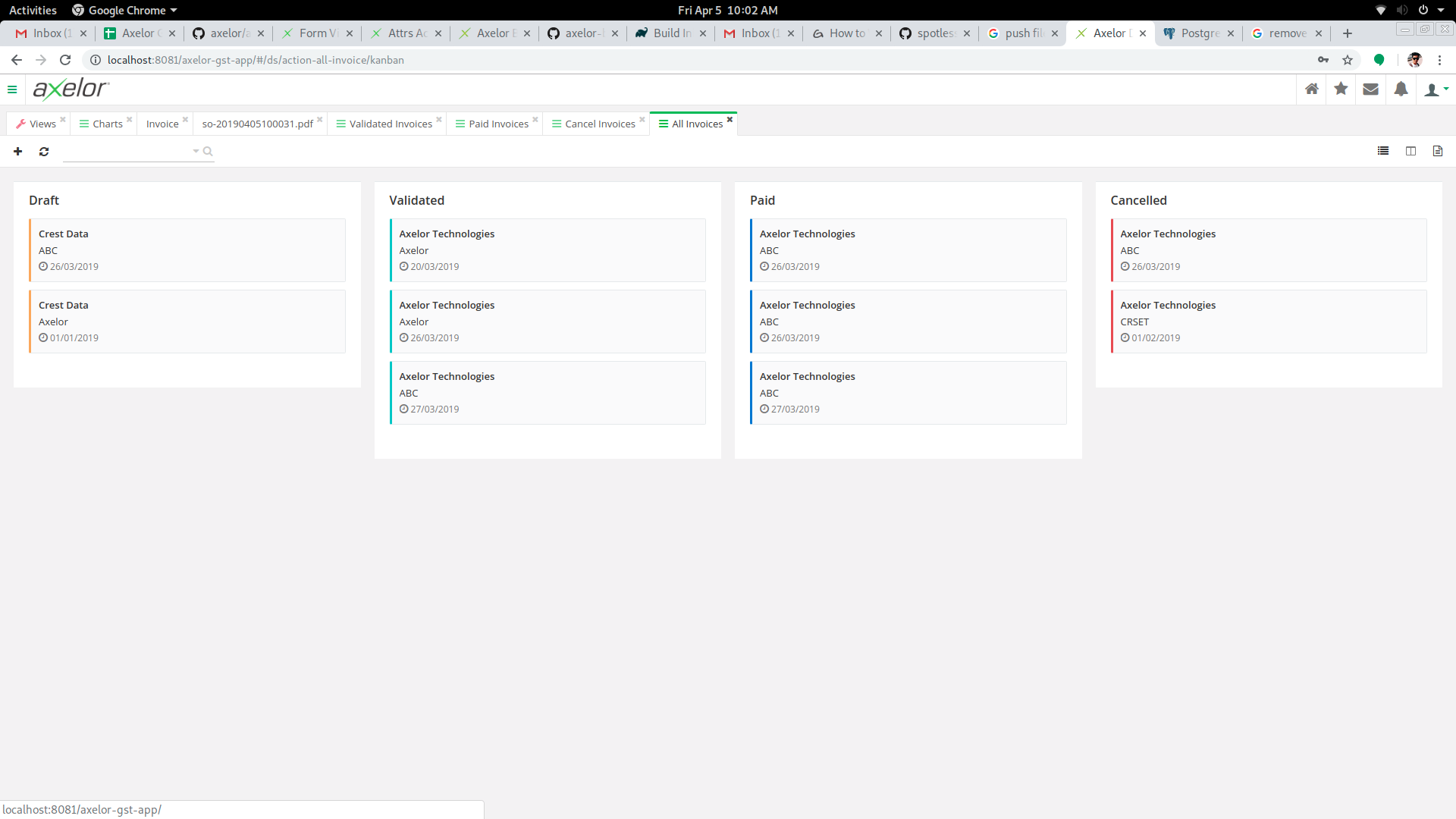
**Fig 5.3.5 Party Generation**

Here Admin can generate Party and contacts of the party

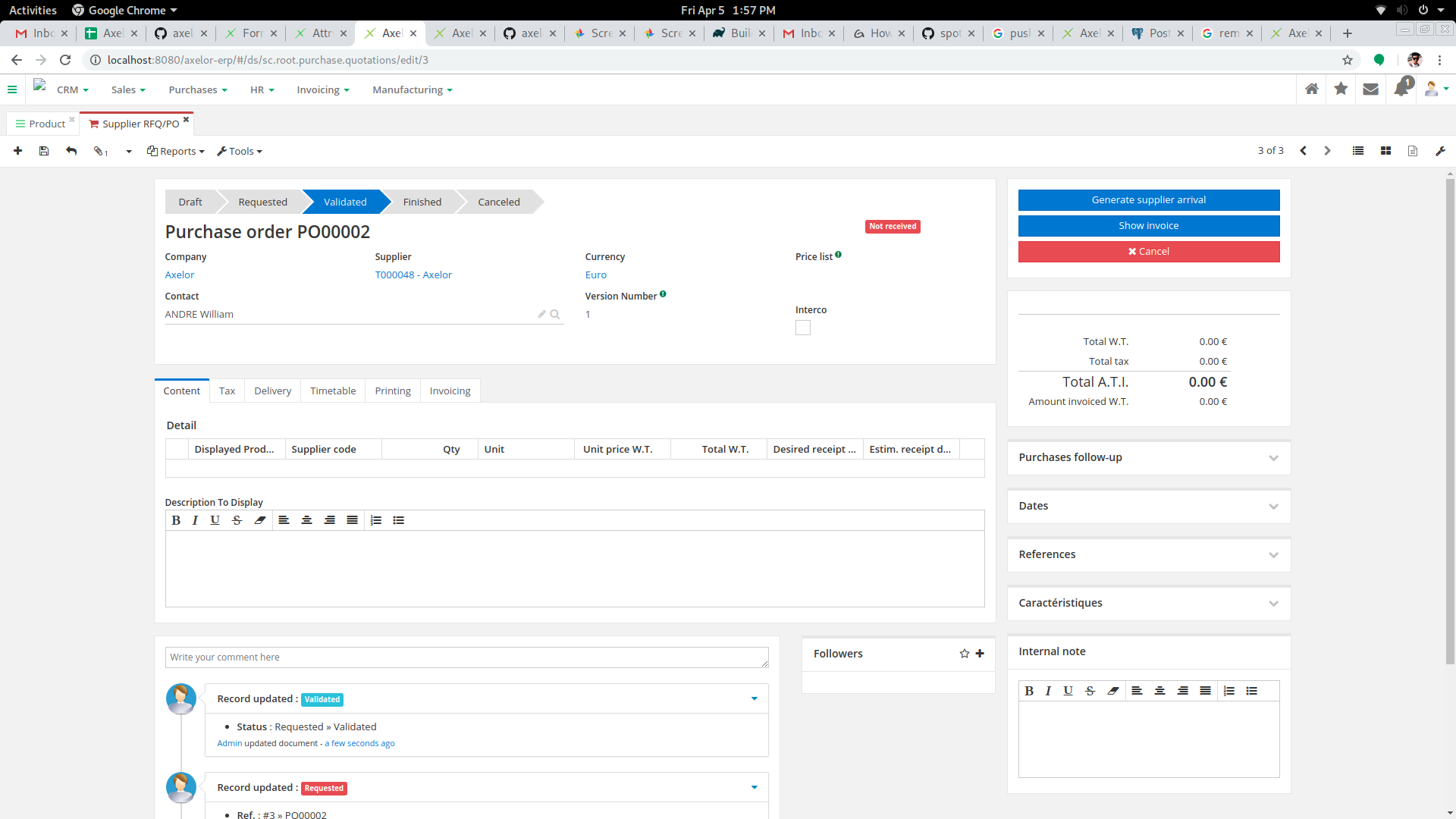


**Fig 5.3.6 Conditional Invoice**

Here user can see the conditional views of the invoice which are based on the status of the invoice.

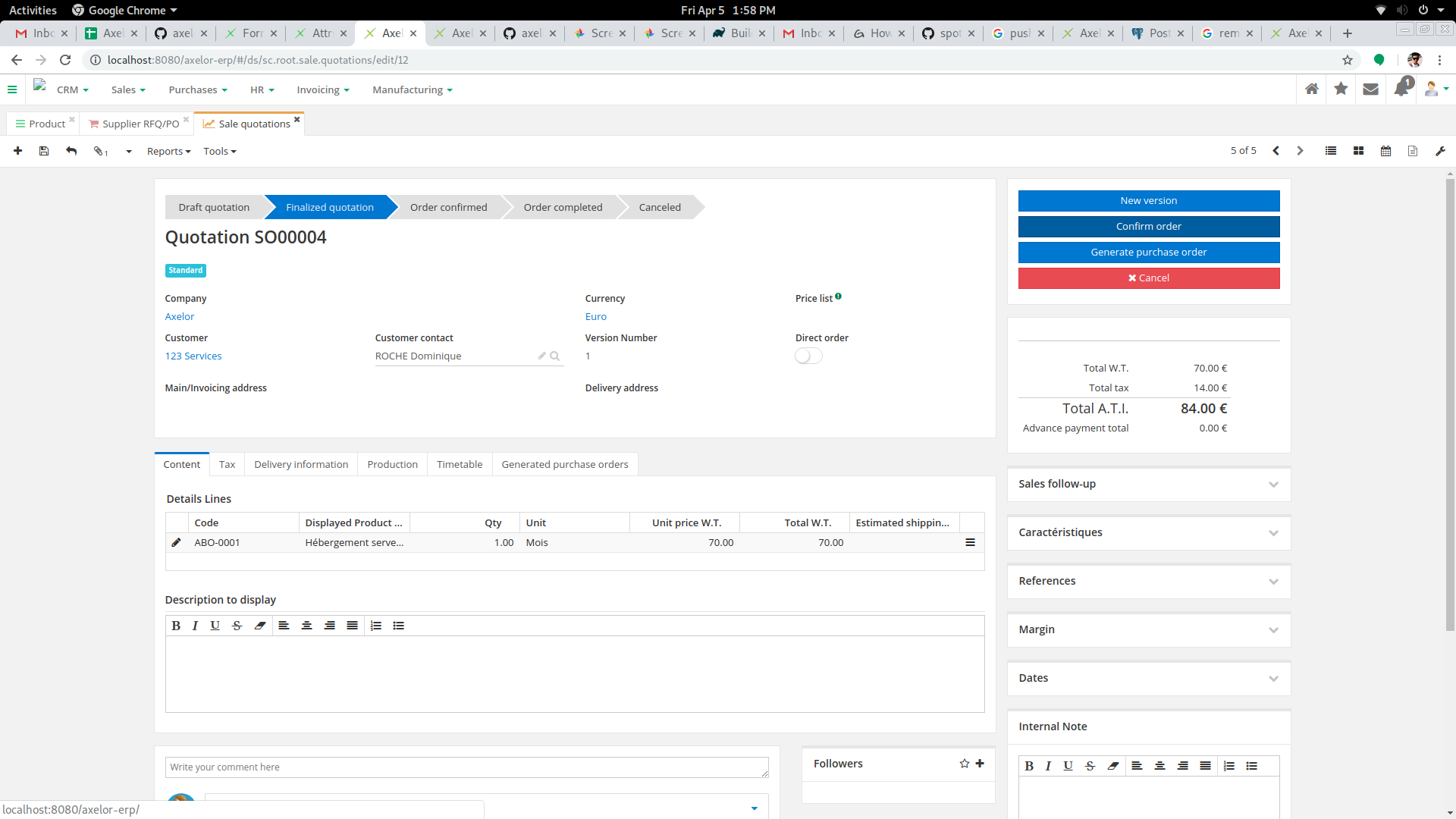


**Fig 5.3.7 Kanban View**



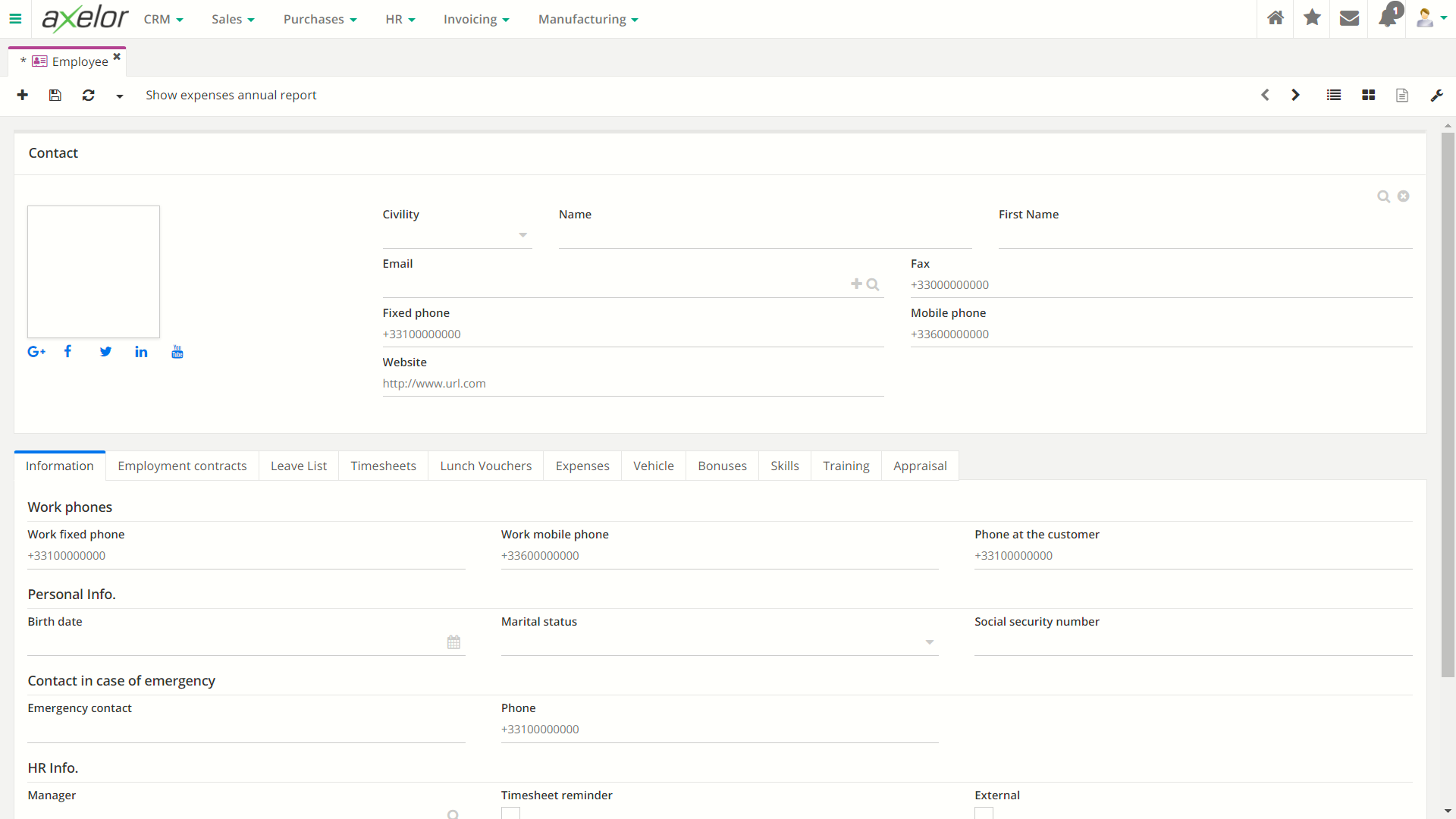
**Fig 5.3.8 Purchase Order**

Here user can place the purchase order of the products.



**Fig 5.3.9 Sale Order**

Here user can place the sale order of the products.



**Fig 5.3.10 Employee Creation**

Here user can enter the employee information

**Implementation Planning**

**6.1 Implementation Environment**

In this project our implementation environment is mainly JAVA and Axelor Development Kit (ADK).

* Axelor development kit

We have used ADK to create the Domain and view of the system, which are the model and view of the MVC structure.

* JAVA

JAVA language is used to create the controller of the MVC Structure, which bridges the gap between the domain and view.

**6.2 Program/Modules Specification**

Main modules of the system are:

**Sales**

This application manages the sales of the company. It allows to create quotations, to print them or to generate them in PDF format and to send them directly from the application. Once the estimate has been validated and finalized, it automatically changes into a sales order.

**Purchase**

This application allows to manage the purchases of the company. It allows you to create quotations that, once validated and finalized, automatically transform into purchase orders. You can generate supplier arrivals from a purchase order to schedule the associated stock movements, and generate the control invoice.

**6.3 Security Features**

Only an authenticated user with valid credentials will be allowed to login in Axelor Business Suite.

A user can only access to the modules and functionality which is assigned them for access except an admin which is the super user of the system.

**6.4 Coding Standards**

**6.4.1 Purpose of coding standards and best practices**

To develop reliable and maintainable applications, one must follow coding standards and best

Practices.

The naming conventions, coding standards and best practices described in this document are those referred by us. There are several standards that exist in the programming industry. None of them are wrong or bad and one may follow any of them. What is more important is, selecting one standard approach and ensuring that the set of standards defined are well adopted.

**6.4.2 Here are several reasons why coding specifications are important:**

Your peer programmers have to understand the code you produce. A coding standard acts as the blueprint for all the team to decipher the code.

Simplicity and clarity achieved by consistent coding save you from common mistakes.

If you revise your code after some time then it becomes easy to understand that code. Its industry standard to follow a particular standard to be more quality in software.

**There are few guidelines which can be followed while coding in JAVA:**

**Coding Standards for Components:**It is recommended to write components name by its purpose. This approach improves the readability and maintainability of code.

**Coding Standards for Classes:**Usually class name should be noun starting with uppercase letter. If it contains multiple word than every inner word should start with uppercase.

Eg: String, StringBuffer, and Dog

**Coding Standards for Interface:** Usually interface name should be adjective starting with uppercase letter. If it contains multiple word than every inner word should start with uppercase.

Eg: Runnable, Serializable, And Comparable

**Coding Standards for Methods:**Usually method name should either be verb or verb noun combination starting with lower letter. If it contains multiple word than every inner word should start with uppercase.

Eg: print (), sleep(), setSalary()

**Coding Standards for Variables:** Usually variable name should be noun starting with lowercase letter. If it contains multiple word than every inner word should start with uppercase.

Eg: name, age. mobileNumber

**Coding Standards for Constants:**Usually constant name should be noun. It should contain only uppercase If it contains multiple word than words are separated with ( \_ ) underscore symbol. Usually we declare constants with public static and final modifiers.

**Java Bean Coding Standards:** A Java Bean is a simple java class with private properties and public getter and setter methods

**Getter Methods:**

1. It should be public method
2. Method name should be prefixed with “get”
3. It should not take any argument

**Setter Methods:**

1. It should be public method
2. Return Type should be void
3. Method name should be prefixed with “set”
4. It should take some argument

public class StudentBean{

private String name;

public void setName(String name){

this.name=name;

}public String getName(){return name;

}}

Note: For boolean properties getter method can be prefixed with “get” or “is”

**Coding convention for Listners:**

* To register a Listner method name should prefixed with add

Eg: public void addMyAccountListner( MyActionListner);

* To unregister a Listner method name should prefixed with remove

Eg: public void removeMyAccountListner( MyActionListner);

**TESTING**

**7.1 TESTING PLAN**

The objective of the system testing is to ensure that all individual programs are working as

Expected, that the programs link together to meet the requirements specified and ensure that the computer system and the associated clerical and other procedures work together. Systems are not designed as entire systems but they are tested as a single system. The analyst must perform both unit and system testing.

Different types of testing methods are available. We have tested our system for different

Aspects like Does the application meet the goals for which it has been designed? This was a

Very important question that stood before us as the application was designed to be implemented on such a large network.

To fulfil its goal of being able to run on different systems we went through a series of tests at

Different places where this is supported to be used the most. As we need to make our system

Efficient enough, we need to test it thoroughly.

Finally, we tested the system with the real-time data, for which it is actually designed. We are

Almost successful in satisfying our needs as it was designed according to their requirements. But it is very necessary to maintain this application and so our work is still not over.

**7.2 TESTING STRATEGY**

Once source code has been generated, software must be tested to uncover as many errors as

Possible before delivery to customers. Your goal is to design a series of test cases that have a

High likelihood of finding errors. Software testing techniques provide systematic guidance for Designing tests that

1. Exercise the internal logic of software components
2. Exercise the inputs and outputs domains of the program to uncover errors in program

Function, behaviour and performance.

During the early stages of testing, a software engineer performs all tests. However, as the testing process progresses, testing specialists may become involved. Reviews and other activities can and do uncover errors, but they are not sufficient. Every time the program is executed, the customer tests it! Therefore, you have to execute the program before it gets to the customer with the specific intent of finding and removing all errors. In order to find the highest possible number of errors, tests must be conducted systematically and test cases must be designed using disciplined techniques.

**7.2.1 Testing Objectives**

Testing is a process of executing a program with the intention of finding an error.

* A good test case is one that has a high probability of finding an as-yet undiscovered error.
* A successful test is one that uncovers an as-yet undiscovered error.

**Unit Testing**

Unit testing is a software development process in which the smallest testable part of an

Application, called units, are individually scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to build a product by means of continual testing and revision.

Unit testing involves only those characteristics that are vital to the performance of the unit under test. This encourages developers to modify the source code without immediate concerns about how such changes might affect the functioning of the units or the program as a whole. Once the whole unit of a program has been found to be working in the most efficient and error-free manner, larger components of the program can be evaluated by means of integration testing.

**Subsystem Testing**

After testing each unit, we move on to larger units called subsystem. In subsystem testing, we

Tested the whole ABS as one system. We tested each module and got a successful result. We found no error or bug after the final test.

**System Testing**

Now, it’s time for whole System testing. We have found many cosmetic bugs and minor bugs

.and we have fixed it and again test it. We worked on each error and exception that I got while testing and most of them are removed or made such correction that it will not happen again.

**Recovery Testing**

It is a system test that forces the software to fail in a variety of ways and verifies that recovery is properly performed.

**Security Testing**

It attempts to verify that protection mechanisms built into a system will, in fact, protect it from improper penetration.

**Performance Testing**

It is designed to test the run-time performance of software within the context of an integrated

System performance testing occurs throughout all steps in the testing process.

**Acceptance Testing**

Acceptance testing can be connected by the end user, customer, or client to validate whether or not to accept the product. Acceptance testing may be performed as part of the hand-off process between any two phases of development. The acceptance test suite is run again the supplied input data or using an acceptance test script to direct the tester. Then the results obtained are compared with the expected results. If there is a correct match for every case, the test suite is said to pass.

**Alpha & beta testing**

The alpha test is conducted at the developer’s site by a customer. The software is used in a natural setting with the developer “looking over the shoulder” of the user and recording errors and usage problems. Alpha test is conducted in a controlled environment. The beta testing is conducted at one or more customer site by the end-user of the software. Unlike alpha testing, the developer is generally not present. Therefore, the beta test is a “live” application of the software in an environment that cannot be controlled by the developer.

**Black-box testing**

Also known as functional testing. A software testing technique whereby the internal workings of the item being tested are not known by the tester. For example, in a black box test on software design, the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. The tester does not ever examine the programming code and does not need any further knowledge of the program other than its specifications.

**The advantages of this type of testing include:**

* The test is unbiased as the designer and the tester are independent of each other.
* The tester does not need knowledge of any specific programming languages.
* The test is done from the point of view of the user, not the designer. Test cases can be

Designed as soon as the specifications are complete.

**The disadvantages of this type of testing include:**

* The test can be redundant if the software designer has already run a test case.
* The test cases are difficult to design.
* Testing every possible input stream is unrealistic because it would take an inordinate amount of time.

**White Box Testing**

Also known as glass box, structural, clear box, and open box testing. A software testing

Technique whereby explicit knowledge of the internal workings of the item being tested are

Used to select the test data. Unlike black box testing, white box testing uses specific knowledge of programming code to examine outputs. The test is accurate only if the tester knows what the program is supposed to do. He or she can then see if the program diverges from its intended goal.

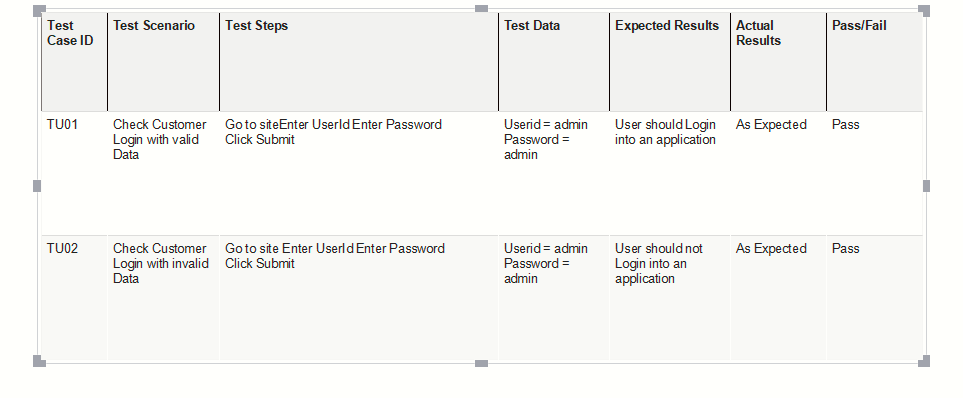
**7.3 TEST SUITE DESIGN**

To minimize the number of errors in software, a rich variety of test design methods has evolved for software. These methods provide the developer with a systematic approach to testing. More important, methods provide a mechanism that can help to ensure the completeness of test and provide the highest likelihood for uncovering errors in software.

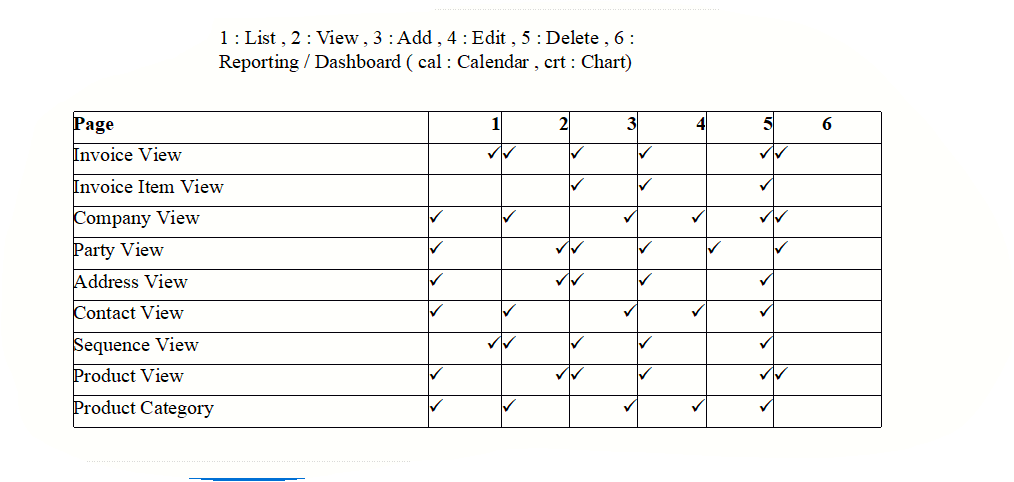
An engineering product can be tested in one of two ways:

* Knowing the specified function that product has been designed to perform, tests can be Conducted that demonstrate each function is fully operational while at the same time Searching for errors in each function:
* Knowing the internal workings of a product, tests can be conducted to ensure that “all gear mesh “, that is, internal oppression is performed according to specifications and all internal components have been adequately exercised.

Here are the testes that have been performed on the views and domains of the module developed.



**Fig. 7.3.1 Test Case**



**Fig. 7.3.2 Test Case**

**CONCLUSION AND DISCUSSION**

**8.1 SELF ANALYSIS OF PROJECT VIABILITIES**

The basic scope of the project included the centralised management of data as well as providing quick and efficient models. One of the major vulnerabilities can be the complexity of the product.

The vulnerabilities can arise from the user requirements, for e.g. if user wants certain models the product need to be configured according to that models only which can sometime lead to unorthodox behaviours.

**8.2 PROBLEMS ENCOUNTERED AND POSSIBLE SOLUTIONS**

One of the main problems while developing was developing module as separate standalone app and then integrating it in the ABS.

As ERP is a complex and integrated application making changes in existing software is a tedious work, but with the help of POJO classes it can reduce the complexity at the Database level.

**8.3 SUMMARY OF PROJECT WORK**

The ultimate aim is that the user can get productivity, flexible and standardized business processes.

It eliminates redundant processes and tasks through automation. Robust, flexible, and configurable modules.

They are not a one-size-fixed modules but can be tailored to the unique needs of a business and can adapt to the ever-changing needs of a growing business.Processes are generally automated, it allows businesses to standardize their own processes and system’s transparency is greatly improved while errors are greatly reduced.

**LIMITATION AND FUTURE ENHANCEMENT**

**9.1 LIMITATIONS**

“No *software is completely failure proof”.* Every software ever built is limited to a specific scope and though a developer should keep in mind every possible scenario while developing a new software, every software is demarcated by its scope and hence has its own limitations

This section will discuss the limitations of the concerned add-on.

**9.1.1 Developed according to specific region**

Currently the ABS is being used in the FRANCE region with there being the common language French it came with the local languages. In future release the company decided to launch the product in the India where addition tax will be added. Currently it’s been hardcoded according to the France region.

# 9.2 FUTURE ENHANCEMENT

Provide mobile application for this web application.

Providing localization to support more language.

**BIBLIOGRAPHY**

* https://docs.axelor.com/
* [*https://github.com/axelor*](https://github.com/axelor)
* [*https://www.youtube.com/*](https://www.youtube.com/)