**SALESFORCE - FORMASSEMBLY INTEGRATION**

**A Project Report Submitted**

**In Partial Fulfilment of the Requirements**

**For the Degree of**

**MASTER OF COMPUTER APPLICATION**

**By**

**GOPAL GOVIND AGARWAL**

**University Roll No. 1900290149046**

**Under the Supervision of**

**Mr. Ankit Verma**

**Assistant Professor**

**KIET Group of Institutions**



**Submitted to**

**Department Of Computer Application**

**Affiliated to**

**DR. A. P. J ABDUL KALAM TECHNICAL UNIVERSITY**

**Lucknow**

**JULY,2021**

# DECLARATION

I hereby declare that the work presented in this report entitled “**Salesforce-Form Assembly Integration**", was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other Institute or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name : Gopal Govind Agarwal

Enroll. No. : 1900290149046

Branch : Master of Computer Application

**(Gopal Govind Agarwal)**

# TRAINING CERTIFICATE

# 

# CERTIFICATE

Certified that **Gopal Govind Agarwal (1900290149046)** has carried out the project work having “**Salesforce-Form Assembly Integration**” for the award of **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other Institute/Institution.

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

**Mr. Ankit Verma External Examiner**

Associate Professor

Dept. of Computer Applications

KIET Group of Institutions, Ghaziabad

**Dr. Ajay Kumar Shrivastava**

Professor & Head,

Department of Computer Applications

KIET Group of Institutions, Ghaziabad

Date:

**Salesforce-Form Assembly Integration**

**Gopal Govind Agarwal**

# ABSTRACT

**“Salesforce-Form Assembly Integration ”** is a Salesforce application which is based on cloud computing.

These days we are using databases in which coding is required to enter the data. But this time we have an application which can do our work simpler and easier i.e. Salesforce. This application is used to store a huge amount of data properly and consistently.

The objective of this application is to show that how a normal person who doesn’t even know programming can use this application easily, it is flexible like data can be deleted, entered or updated easily. A software project means a lot of experience. I learned a lot through this project. This project has sharpened our concept of cloud computing.

This concept of cloud computing has now become a great role to play in today's technical world. These technologies will definitely take database systems far away.

Through this project I learnt so many things that can be managed through this application like sales, marketing, commercials and many more things. The only drawback of Salesforce is that it is expensive but to do great work we have to use good technology as today data security is the best and the most essential thing and Salesforce contains that all. things which I also learnt make Data Flow Diagram, State Event Diagram and Activity Diagram with help of our project Mentors.

# ACKNOWLEDGEMENTS

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Mr. Ankit Verma** for her guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications**, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot in many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kinds of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Gopal Govind Agarwal**

**(1900290149046)**

**Table of Contents**

Declaration ii

Training Certificate iii

Certificate iv

Abstract v

Acknowledgement vi

Table of Content vii

List of Figures viii

**CHAPTER 1: Introduction : 9-15**

1.1 Project details

1.2 Purpose of the Project

1.4 Identification of Need

1.5 Problem Statement

1.6 Hardware / Software used in Project

1.7 Scope

1.8 Project Schedule

**CHAPTER 2: Cloud Computing : 16-19**

**CHAPTER 3: Feasibility Study : 20-25**

3.1 Introduction

3.2 Main Aspects

3.3 Benefits

3.4 System Requirement Specification

**CHAPTER 4: Design : 26-30**

4.1 Introduction

4.2 System Design

4.3 SDLC

4.4 DFD

4.5 UML use case diagram

4.6 ER Diagram

**CHAPTER 5: Salesforce : 31-36**

* 1. Standard Objects
  2. Standard Fields
  3. Important Standard Objects

**CHAPTER 6: Creating Objects, Validations, Triggers : 37-40**

* 1. Creating Salesforce Custom Objects

6.2 Creating a Validation Rule

6.3 Apex Triggers

**CHAPTER 7: Output Screens : 41-44**

**CHAPTER 8: Testing : 45-50**

8.1 Introduction

8.2Types of testing

8.3 Some Important Observations

**CHAPTER 9: Conclusion and Future Scope : 51**

9.1 Conclusion

9.2 Future Scope

**References : 52-53**

## LIST OF FIGURES

**Fig No. Title Page No**

Fig 1.1 Pert Chart : 14

Fig 1.2 Gantt chart for project : 15

Fig 4.1 The planning step : 29

Fig 4.2 Apex Working : 31

Fig 4.3 Context level : 32

Fig 4.4 DFD 1 Level : 32

Fig 4.5 Uml use case : 33

Fig 4.6 ER diagram of system : 34

Fig 6.1 Validation Rule : 34

Fig 6.2 Course Validation Rule : 35

Fig 7.1 Showing home page : 46

Fig 7.2 Showing Course Object : 47

Fig 7.3 Validations : 48

Fig 7.4 Validations on Course Fees Object : 49

Fig 7.5 Validations on Course Fees Object : 50

Fig 7.6 Showing all the Fields of Enrollment Object : 51

Fig 7.7 Showing Validations and Master Detail Relationships : 52

Fig 7.8 Showing all the Fields of Client/User Object : 53

Fig 7.9 Showing Validations on Client/User Object : 54

Fig 7.10 Showing Evaluation Object : 55

Fig 7.11 Showing Mater-Detailed Relationships between Objects : 56

Fig 8.1 Testing Pyramid : 57

## CHAPTER 1

**INTRODUCTION**

## Project Details:

“**Salesforce-Form Assembly Integration**” is taking a concept from “**Cloud Computing**” and Salesforce was founded by **“Mark Russell Benioff”**.

As the Salesforce cloud computing Application consists of a large, waist-level cabinet with a number of clouds in it. Each cloud contains different objects through which we can access the data. Once the application starts, we can enter the particular amount of data which we need to put in the application for sure, the individual can directly add the data to the particular objects and fields thereby adding the student’s data. This does not require much time, only we have to fill and enter all the data related to a particular client.

This application will be developed for PC’s, we can access it from anywhere and even from the phone easily using the salesforce application.

## Purpose:

Salesforce provides you with the fastest path from Idea to build an App. You can concentrate on building your app using Salesforce tools, rather than building the infrastructure and tools yourself. This can save you years of time and millions of dollars.

Salesforce customers generally say that it’s unique for three major reasons:

**Fast** – Traditional CRM software can take more than a year to deploy, compare that to months or even weeks with Salesforce.

**Easy** – Salesforce wins in the easy to use category hands down. You can spend more time putting it to use and less time figuring it out

**Effective** – Because it is easy to use and can be customized to meet business needs, customers find Salesforce very effective.

Salesforce is in the cloud, so your team can use it from anywhere with access to the internet. If you are a business that is rapidly changing or you are a seasoned company that’s been around for years, your business is probably changing too. Salesforce is completely scalable to your growth.

Salesforce seamlessly integrates with 3rd party apps. If you want to integrate Salesforce with Gmail you can do it, if you want to integrate it with your accounting software you can do that too. On the other hand, integration is tough with other CRMs. Salesforce is affordable, especially if you consider its vast variety of capabilities. Even startups and small businesses can use Salesforce.

* 1. **Identification of Need :**

Users need identification and analysis are concerned with what the user needs rather than what he/she wants. Not until the problem has been identified, defined, and evaluated should the analyst think about solutions and whether the problem is worth working. This step intended to help the user and analyst understand the real problem rather than its symptoms. The user or the analyst may identify the need for a candidate system or for enhancement in the existing system.

An analyst is responsible for performing following tasks:

* Studied strength and weakness of the current system.
* Determining “what” must be done to solve the problem.
* Prepared a functional specifications document.

These modules are developed with the aim of reducing time, reducing manpower so that everything can be easily maintained and. The volume of work and complexity are increasing year by year. This system reduces complexity and time. Also provide availability 24\*7.

**1.4 Problem Statement**

In the existing system all the work is done manually. This is a chance of committing errors and it will take more time to perform or checkout any information. There are so many limitations in the existing system. So the existing system should be atomized. If the system is carried over manually, for everything it takes more time. So it is difficult to make immediate decisions.

* In the traditional system, if you wish to analyze any record you have to turn pages many times.
* Existing systems are time consuming as it requires too much planning and so much human involvement.
* As it involves much human involvement, the cost of the system automatically gets increased.
* Existing systems require paper use, which isn’t good for the environment.
* With too much human involvement, there are high chances of risk as well.
* There is too much paperwork too, which makes the tasks in the existing system very tedious.
  1. **Hardware / Software used in Project**
     1. **Hardware Requirement**

|  |  |
| --- | --- |
| Hardware | Configuration |
| Processor | Intel(R)core(TM)i3-7200U CPU  @2.50GHz |
| Ram | 4GB |
| Monitor | Normal |

Table 1.1

**1.6.2 Software Requirement**

|  |  |
| --- | --- |
| Software | Configuration |
| Operating System | Windows 8 |
| Language | Apex,JavaScript |

Table 1.2

## 1.7 Scope:

Client/User can use their device (laptop, mobile, desktop) or you can provide your lab.

We set up a branded solution for you (Software as a Service) and maintain every upgrade of the platform, right from hardware to software.

Our flagship product is built from scratch for a financial goal with one objective – make Client/User successful. We enable Client/User and employees to be successful by capturing, measuring, and analyzing the right data.

Small clients/Users are at the core of the platform. We not only assess but also are part of the journey to accomplish the mission. Whether it is about a small payment job or preparing for higher education, we enable you for success.

Client/User can learn from Financial experts who are mentors on our platform (many of them are your college alumni). Take timely suggestions, join webinars and learn from their experiences.

* 1. **Project Schedule**

The objective of software project planning is to provide a framework that enables the manager to make reasonable estimates of resources, costs and schedule. These estimates are made within a limited time frame at the beginning of a software project and should be updated regularly as the project progresses. In addition, estimates should attempt to define “best case” and “worst case” scenarios so that project outcomes can be bounded.

The first activity in software project planning is the determination of software scope. Function and performance allocated to software during system engineering should be assessed to establish a project scope that is ambiguous and understandable at Presidency and technical levels. Software scope describes function, performance, constraints, interfaces and reliability.

During early stages of project planning, a microscopic schedule is developed. This type of schedule identifies all major software engineering activities and the product functions to which they are applied. As the project gets under way, each entry on the macroscopic

Schedule is refined into a detailed schedule. Here specific software tasks are identified and scheduled.

Scheduling has following principles:

* 1. Compartmentalization: the project must be compartmentalized into a number of manageable activities and tasks.
  2. Interdependency: the interdependencies of each compartmentalized activity or tasks must be determined.
  3. Time allocation: each task to be scheduled must be allocated some number of work units.
  4. Effort validation: every project has a defined number of staff members.
  5. Defined responsibilities: every task that is scheduled should be assigned to a specific team member.
  6. Defined outcomes: every task that is scheduled should have a defined outcome.

**1.8.1 Pert chart**

Program evaluation and review technique (pert) is a project scheduling method that is applied to software development.

Pert provide quantitative tool that allow the software planner to-Determine the critical path-the chain of tasks that determines the duration of the project; Establish “most likely” time estimates for individual tasks by applying statistical models; and

Calculate “boundary times” that define a time “window” for a particular task.

Pert chart (program evolution review technique) for project-



Fig 1.1 Pert Chart

* + 1. **Gantt Chart**

When creating a project schedule, the planner begins with a set of tasks (the work breakdown structure). If automated tools are used, the work breakdown is input as a task network. Effort, duration and start dates are input to each task network. As a consequence of this input, a timeline chart also called a Gantt chart is generated. A timeline chart is developed for the entire project.

Gantt chart for project:



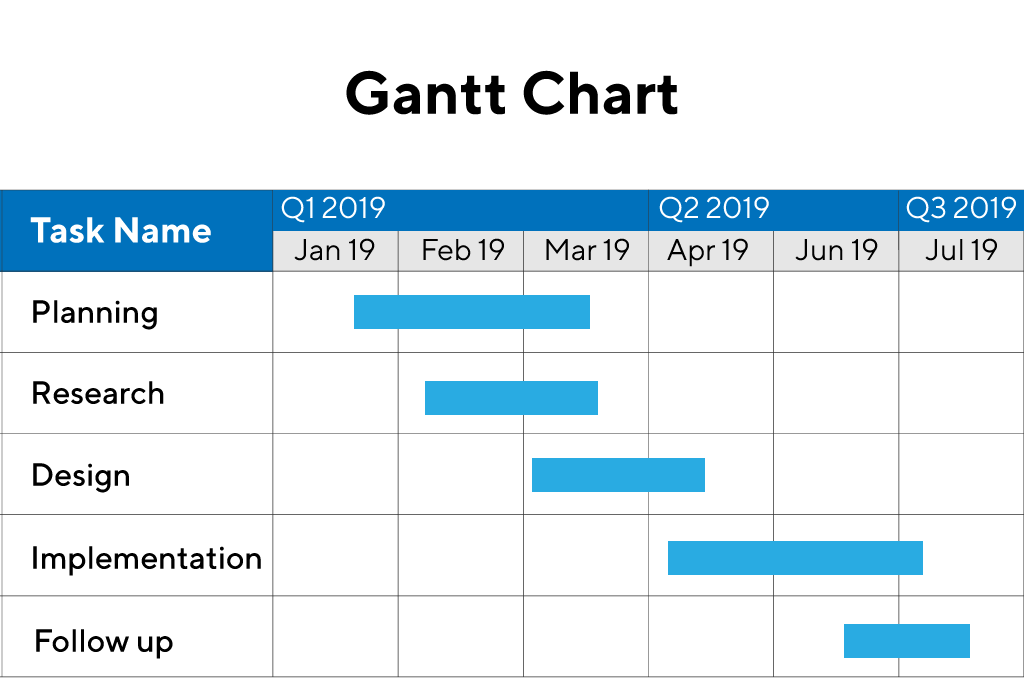


Fig 1.2 Gantt chart for project

Here horizontal bars indicate the duration of each task.

**CHAPTER 2**

**CLOUD COMPUTING**

## About:

Cloud computing relies on software for distributed batch and stream processing, as well as distributed storage. This chapter focuses on an oft-ignored angle of assuredness: performance assuredness.

A significant pain point today is the inability to support reconfiguration operations, such as changing of the shard key in a shared storage/database system, or scaling up (or down) of the number of virtual machines (VMs) being used in a stream or batch processing system.

Cloud computing has become the industry standard for rapid application deployment, scalable server support, mobile and distributed services, and it provides access to (theoretically) infinite resources. Unfortunately, researchers are still trying to converge towards cross-provider cloud computing frameworks to enable compatibility and seamless resource transition between cloud providers. Moreover, users are restricted to using the provider-specific pre-configured options of resources and services, irrespective of their current needs. At the same time, cloud services are provided as a direct service from the providers to the clients. This creates a segregated cloud market clientele, and non-negotiable pricing

Strategies for the cloud services. In this paper, we propose Jugo, a generic architecture for cloud composition and negotiated service delivery for cloud users. Jugo acts as a match-maker for service specifications from the users with the currently available assets from the cloud providers. The engagement of a middle-man as an opaque cloud service provider will create a better opportunity for cloud users to find cheaper deals, price-matching, and flexible resource specifications, with increased revenue and higher resource utilization for the cloud service providers.[1]

We discuss new techniques to support such reconfiguration operations in an online manner, whereby the system does not need to be shut down and the user/client-perceived behavior is indistinguishable regardless of whether a reconfiguration is occurring in the background, that is, the performance continues to be assured in spite of ongoing background reconfiguration. Next, we describe how to scale-out and scale-in (increase or decrease) the number of machines/VMs in cloud computing frameworks like distributed stream processing and distributed graph processing systems, again while offering assured performance to the customer in spite of the reconfigurations occurring in the background. The ultimate performance assuredness is the ability to support SLAs/SLOs (service-level agreements/objectives) such as deadlines. We present a new real-time scheduler that supports priorities and hard deadlines for Hadoop jobs. We implemented our reconfiguration systems as patches to several popular and open-source cloud computing systems, including MongoDB and Cassandra (storage), Storm (stream processing), LFGraph (graph processing), and Hadoop (batch processing).[2]

Many enterprises in industries start using Cloud Computing for their IT infrastructure services. This adoption of Cloud Computing is a part of the enterprise transformation which is the migration from a legacy IT environment to Cloud Computing. On the other hand, one of major targets is an industry solution which provides a critical business service to their end customers. This paper proposes Industry Cloud which is the enhanced design of Cloud Computing for industry solutions. It efficiently supports industry solutions for enterprise business requirements. The paper describes Industry Cloud with a requirement analysis of industry solutions, those adopted functions, and three use case scenarios in the electronics and retail industry. The contribution of the paper is the analysis of industry wide requirements, the definition of Industry Cloud with a common function among industry solutions and the usage with use case scenarios.[3]

The complexity of Cloud infrastructures is increasing every year, requiring new concepts and tools to face off topics such as process configuration and reconfiguration, automatic scaling, elastic computing and healthiness control. This paper presents a Smart Cloud solution based on a Knowledge Base, KB, with the aim of modeling cloud resources, Service Level Agreements and their evolution, while enabling the reasoning on cloud structures and implementing strategies of efficient smart cloud management and intelligence. The solution proposed is composed of Smart Cloud Engine, SCE, the Knowledge Base, KB, and the Supervisor and Monitoring module for data acquisition. It can be easily integrated with any cloud configuration manager, cloud orchestra or, and monitoring tool, since the connections with these tools are performed by using REST calls and XML files. [4]

Current Cloud Computing is primarily based on proprietary data centers, where hundreds of thousands of dedicated servers are set up to host the cloud services.

In addition to the huge number of dedicated servers deployed in data centers, there are billions of underutilized Personal Computers (PCs), usually used only for a few hours per day, owned by individuals and organizations worldwide. The vast untapped compute and storage capacities of the underutilized PCs can be consolidated as alternative cloud fabrics to provision broad cloud services, primarily infrastructure as a service. This approach, thus referred to as the "no data center" approach, complements the data center based cloud provision model. In this paper, we present our opportunistic Cloud Computing system, called cu Cloud, that runs on scavenged resources of underutilized PCs within an organization/community. Our system demonstrates that the "no data center" solution indeed works. Besides providing our concept, model, and philosophy, our experimental results are highly encouraging. [5]

Whether one public cloud, private cloud or a mixed cloud, the users lack effective security quantifiable evaluation methods to grasp the security situation of its own information infrastructure on the whole. This paper provides a quantifiable security evaluation system for different clouds that can be accessed by a consistent API. The evaluation system includes a security scanning engine, security recovery engine, security quantifiable evaluation model, visual display module, etc.

The security evaluation model consists of a set of evaluation elements corresponding to different fields, such as computing, storage, network, maintenance, application security, etc. Each element is assigned a three tuples on vulnerabilities, score and repair method. The system adopts a “One vote vetoed” mechanism for one field to count it's score and adds up the summary as the total score, and to create one security view.

We implement the quantifiable evaluation for different cloud users based on our G-Cloud platform. It shows the dynamic security scanning score for one or multiple clouds with visual graphs and guides users to modify configuration, improve operation and repair vulnerabilities, so as to improve the security of their cloud resources. [6]

**CHAPTER 3**

**FEASIBILITY STUDY**

**3.1 Introduction**

Feasibility of the system is an important aspect, which is to be considered. The system needs to satisfy the law of economics, which states that the maximum output should be yielded in minimum available resources.

A feasibility analysis evaluates the project’s potential for success; therefore, perceived objectivity is an essential factor in the credibility of the study for potential investors and lending institutions. There are five types of feasibility study—separate areas that a feasibility study examines, described below.

1. **Technical Feasibility**

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves the evaluation of the hardware, software, and other technical requirements of the proposed system. As an exaggerated example, an organization wouldn’t want to try to put Star Trek’s transporters in their building—currently, this project is not technically feasible.

1. **Economic Feasibility**

This assessment typically involves a cost/ benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility—helping decision-makers determine the positive economic benefits to the organization that the proposed project will provide.

1. **Legal Feasibility**

This assessment investigates whether any aspect of the proposed project conflicts with legal requirements like zoning laws, data protection acts or social media laws. Let’s say an organization wants to construct a new office building in a specific location. A feasibility study might reveal the organization’s ideal location isn’t zoned for that type of business. That organization has just saved considerable time and effort by learning that their project was not feasible right from the beginning.

1. **Operational Feasibility**

This assessment involves undertaking a study to analyze and determine whether—and how well—the organization’s needs can be met by completing the project. Operational feasibility

studies also examine how a project plan satisfies the requirements identified in the requirements analysis phase of system development.

1. **Scheduling Feasibility**

This assessment is the most important for project success; after all, a project will fail if not completed on time. In scheduling feasibility, an organization estimates how much time the project will take to complete.

When these areas have all been examined, the feasibility analysis helps identify any constraints the proposed project may face, including:

* Internal Project Constraints: Technical, Technology, Budget, Resource, etc.
* Internal Corporate Constraints: Financial, Marketing, Export, etc.
* External Constraints: Logistics, Environment, Laws, and Regulations, etc.

**3.2 Main Aspects**

There are three aspects of feasibility to be considered namely.

1. Technical
2. Operational
3. Economical

**TECHNICAL**:

In the technical aspects one may consider the hardware equipment for the installation of the software. The system being centralized will require very little hardware appliances. Hence this helps the system to work smoothly with a limited amount of working capitals.

**OPERATIONAL**:

In the operational aspects may think of the benefits of the workload that many a personal may have to share. This is eased out and the required output may be retrieved in a very short time. Thus there is accuracy in the work on time and also there will be very little work that needs to be performed.

**ECONOMICAL**:

The Economic system is definitely feasible because the hardware requirement is less and the operational working for the system requires less number of recruits. This helps introduce over-staffing and wastage of funds.

We studied the position to evaluate solutions. Most important factors in this study were tending to overlook the confusion inherent in system Development, the constraints and the assumed studies. It can be started that if the feasibility study is to serve as a decision document it must answer three key questions.

1. Is there a new and better way to do the job that will benefit the user?
2. What are the costs and savings of the alternatives?
3. What is recommended?

On these questions it can be explained that feasibility study of the system includes following different angles.

**3.2.1 Technical feasibility:**

This centers on the existing computer system (hardware, software etc.) and to what extent it can support the proposed additional equipment .in this stage of

study, we have collected information about technical tools available by which I could decide my system design as the technical requirements.

**3.2.2 Operational Feasibility:**

In this stage of study we have checked the staff availability. I concentrate on knowledge of end users that are going to use the system. This is also called behavioral feasibility in which I have studied the following aspects; people are inherently resistant to change, and computers have been known to facilitate change .An estimate has been made to how strong a reaction the user staff is having toward the development of a computerized system. It is common knowledge that computer installations have something to do with turnover. I had explained that there is a need to educate and train the staff on new ways of conducting business.

**3.2.3 Economical feasibility**:

Economical analysis is the most frequently used method for evaluating the effectiveness of candidate systems. More commonly known as cost\benefit analysis, the procedure is to determine the benefits and savings that benefits outweigh costs. The decision was to design and implement the system because it is for having a chance to be approved. This is an ongoing effort that improves the accuracy at each phase of the system life cycle.

In developing cost estimates for a system I need to consider several cost elements. Among these is a hardware personal facility. Operating and supply costs.

**3.3 Benefits**

Benefits of conducting a feasibility study:

* Improves project teams’ focus
* Identifies new opportunities
* Provides valuable information for a “go/no-go” decision
* Narrows the business alternatives
* Identifies a valid reason to undertake the project
* Enhances the success rate by evaluating multiple parameters
* Aids decision-making on the project
* Identifies reasons not to proceed

**3.4 System Requirement Specification**

Any system can be designed after specifying the requirement of the user about that system. For this first of all gathered information from users by the preliminary investigation which is starting an investigation about user requirements.

The data that the analysts collect during preliminary investigation are gathered through the various preliminary methods.

Documents Reviewing Organization

The analysts conducting the investigation first learn the organization involved in, or affected by the project. Analysts can get some details by examining organization charts and studying written operating procedures.

**Collected data is usually of the current operating procedure:**

* The information relating to clients, projects and Client/User and the relationship between them was held manually.
* Managing follow-ups was through manual forms.
* Complaints require another tedious work to maintain and solve.
* Payment details had to be maintained differently.

**Gathering Information by Asking Questions**

Interviewing is the most commonly used technique in analysis. It is always necessary first to approach someone and ask them what their problems are, and later to discuss with them the result of your analysis.

**Questionnaires**

Questionnaires provide an alternative to interviews for finding out information about a system. Questionnaires are made up of questions about information sought by analysts. The questionnaire is then sent to the user, and the analyst analyzes the replies.

**Electronic Data Gathering**

Electronic communication systems are increasingly being used to gather information. Thus it is possible to use electronic mail to broadcast a question to a number of users in an organization to obtain their viewpoint on a particular issue.

In my project, with the help of Marg software solutions, I have sent questionnaires through electronic mail to twenty employees of the company and retrieved the information regarding the problem faced by the existing system.

Interviews

Interview allows the analysts to learn more about the nature of the project request and reason for submitting it. Interviews should provide details that further explain the project and show whether assistance is merited economically, operationally or technically.

One of the most important points about interviewing is what question you need to ask.

It is often convenient to make a distinction between three kinds of question that is

* Open questions
* Closed question
* Probes

Open questions are general questions that establish a person’s viewpoint on a particular subject.

Closed questions are specific and usually require a specific answer.

Probes are questions that follow up an earlier answer.

**CHAPTER 4**

**DESIGNs**

**4.1 Introduction**

System is created to solve problems. One can think of the systems approach as an organized way of dealing with a problem. In this dynamic world, the subject system analysis and design, mainly deals with the software development activities.

Since a new system is to be developed, the one most important phase of the software development life cycle is system requirement gathering and analysis. Analysis is a detailed study of various operations performed by a system and their relationship within and outside the system. Using the following steps it becomes easy to draw the exact boundary of the new system under consideration.

All procedures, requirements must be analyzed and documented in the form of detailed DFDs, logical data structure and miniature specifications.

System analyses also include sub-dividing of complex processes involving the entire system, identification of data store and manual processes.

**4.2 System Design**

System design is the process of planning a new system or to replace the existing system. Simply, system design is like the blueprint for building, it specifies all the features that are to be in the finished product.

System design phase follows system analysis phase. Design is concerned with identifying functions, data streams among those functions, maintaining a record of the design decisions and providing a blueprint for the implementation phase.

Design is the bridge between system analysis and system implementation. Some of the essential fundamental concepts involved in the design of application software are:

* Abstraction
* Modularity
* Verification

**Abstraction** is used to construct solutions to problems without having to take account of the intricate details of the various component sub problems. Abstraction allows system designers to make stepwise refinement, which at each stage of the design may hide unnecessary details associated with representation or implementation from the surrounding environment.

**Modularity** is concerned with decomposing the main module into well-defined manageable units with well-defined interfaces among the units. This enhances design clarity, which in turn eases implementation, Debugging, Testing, Documenting and Maintenance of the software product. Modularity viewed in this sense is a vital tool in the construction of large software projects.

**Verification** is a fundamental concept in software design. A design is verifiable if it can be demonstrated that the design will result in implementation that satisfies the customer’s requirements. Verification is of two types namely.

* Verification that the software requirements analysis satisfies the customer’s needs.
* Verification that the design satisfies the requirement analysis.

Some of the important factors of quality that are to be considered in the design of application software are:

**Reliability:**

The software should behave strictly according to the original specification and should function smoothly under normal conditions.

**Extensibility:**

The software should be capable of adapting easily to changes in the specification.

**Reusability:**

The software should be developed using a modular approach, which permits modules to be reused by other applications, if possible.

The System Design briefly describes the concept of system design and it contains four sections. The first section briefly describes the features that the system is going to provide to the user and the outputs that the proposed system is going to offer.

The second section namely Logical Design describes the Data Flow Diagrams, which show clearly the data movements, the processes and the data sources, and sinks, E-R diagrams which represent the overall logical design of the database, and high-level process structure of the system.

**Preliminary Design:**

Preliminary design is basically concerned with deriving an overall picture of the system. Deriving the entire system into modules and sub-modules while keeping Cohesion and Coupling factors in mind. Tools, which assist in the preliminary design process, are Data Flow Diagrams.

**Code design:**

The purpose of code is to facilitate the identification and retrieval for items of information. A code is an ordered collection of symbols designed to provide unique identification of an entity or attribute. To achieve unique identification there must be only one place where the identified entity or the attribute can be entered in the code; conversely there must be a place in the code for everything that is to be identified. This mutually exclusive feature must be built into any coding system.

The codes for this system are designed with two features in mind. Optimum human oriented use and machine efficiency are also operable i.e., they are adequate for present and anticipate data processing both for machine and human use.

**Input /Output design**

Is a part of overall system design, which requires very careful attention. The main objectives of input design are:

* To produce a cost-effective method of input.
* To achieve the highest possible level of accuracy.
* To ensure that the input is acceptable to and understood by the user staff.

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also to provide a permanent hard copy of these results for later consultation.

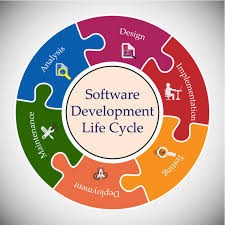
The various types of outputs are required by this system are given below:

* External outputs, whose destination is outside the concern and which require special attention because they project the image of the concern.
* Internal outputs, whose destination is within the concern and which require careful design because they are the user’s main interface within the computer.
* Operation outputs, whose use is purely within the computer department, E.g., program listings, usage statistics etc,

**4.3 SDLC**

Software Development Life Cycle (SDLC) is a framework that defines the steps involved in the development of software at each phase. It covers the detailed plan for building, deploying and maintaining the software.

SDLC defines the complete cycle of development i.e. all the tasks involved in planning, creating, testing, and deploying a Software Product.



ig 4.1 The planning step

**4.3.1 SDLC Phases**

**Given below are the various phases:**

* Requirement gathering and analysis
* Design
* Implementation or coding
* Testing
* Deployment
* Maintenance

**Requirement Gathering and Analysis**

During this phase, all the relevant information is collected from the customer to develop a product as per their expectation. Any ambiguities must be resolved in this phase only.

Business analyst and Project Manager set up a meeting with the customer to gather all the information like what the customer wants to build, who will be the end-user, what is the purpose of the product. Before building a product a core understanding or knowledge of the product is very important.

Once the requirement gathering is done, an analysis is done to check the feasibility of the development of a product. In case of any ambiguity, a call is set up for further discussion.

Once the requirement is clearly understood, the SRS (Software Requirement Specification) document is created. This document should be thoroughly understood by the developers and also should be reviewed by the customer for future reference.

**Design**

In this phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived.

**Implementation or Coding**

Implementation/Coding starts once the developer gets the Design document. The Software design is translated into source code. All the components of the software are implemented in this phase.

**Testing**

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

**Deployment**

Once the product is tested, it is deployed in the production environment or first UAT (User Acceptance testing) is done depending on the customer expectation.

**Maintenance**

After the deployment of a product in the production environment, maintenance of the product i.e. if any issue comes up and needs to be fixed or any enhancement is to be done is taken care of by the developers.

**4.4 DFD**

**DFD** is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. Data Flow Diagrams can be represented in several ways.

Diagram

Description automatically generated

.Fig 4.2 Apex Working

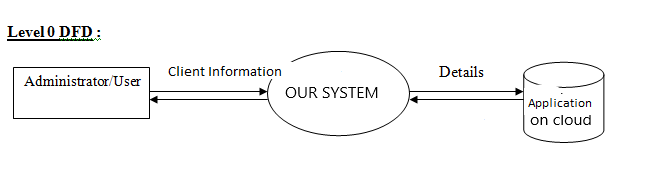


Fig 4.3 Context level

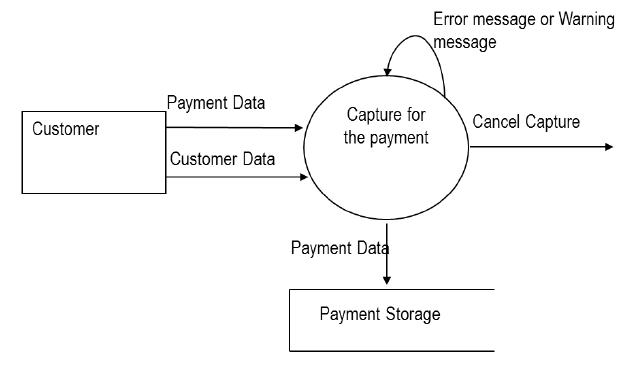


Fig 4.4 DFD 1 Level

**4.5 UML use case diagram**

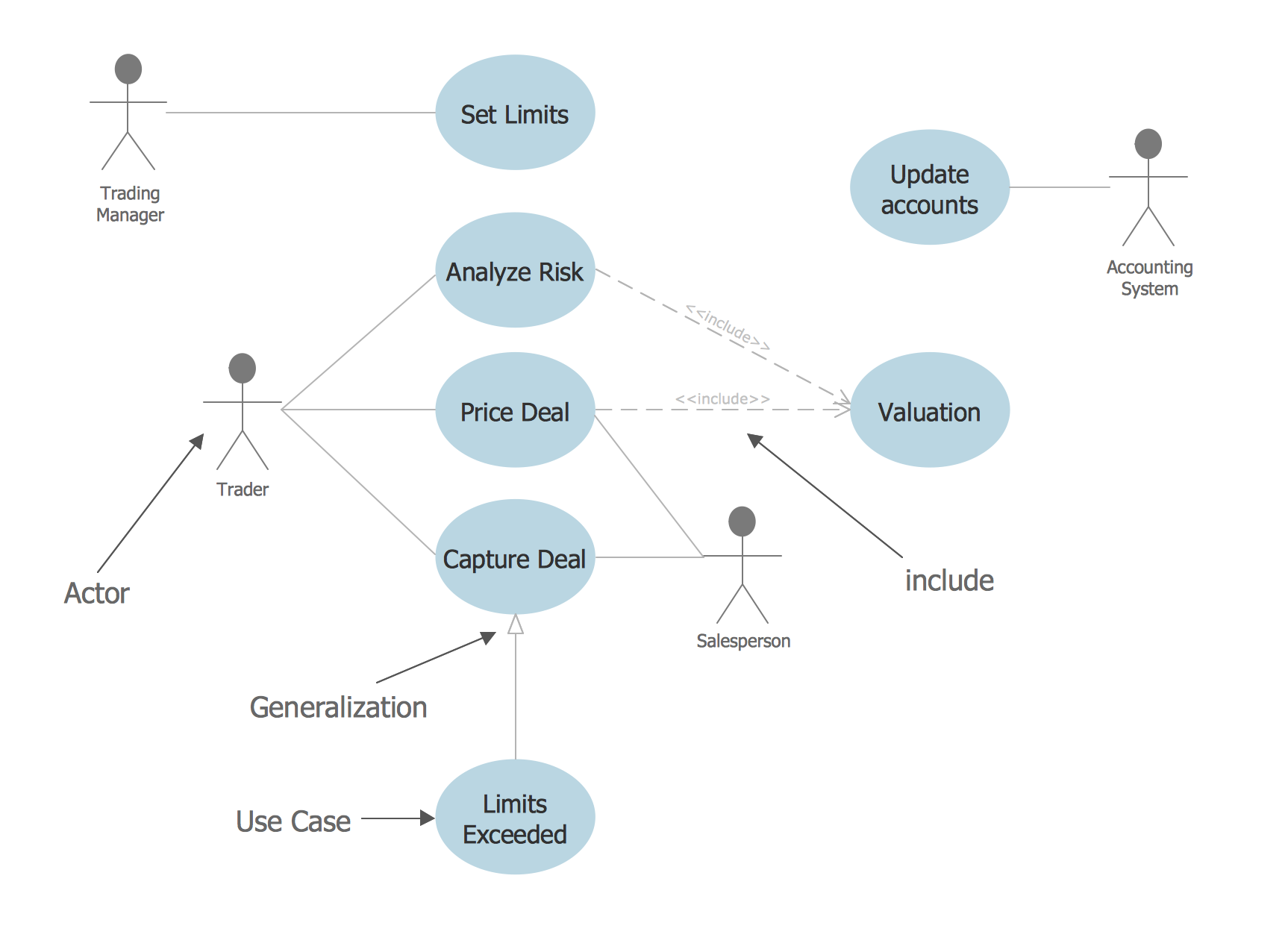
****

Fig 4.5 Uml use case

**4.6 ER Diagram**

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram has three main components:

1. Entity

2. Attribute

3. Relationship

Fig 4.6 ER diagram of system

**4.7.1 ER- Diagram Notations**

ER- Diagram is a visual representation of data that describe how data is related to each other.

* **Rectangles:** This symbol represent entity types
* **Ellipses :** Symbol represent attributes
* **Diamonds:** This symbol represents relationship types
* **Lines:** It links attributes to entity types and entity types with other relationship types
* **Primary key:** attributes are underlined
* **Double Ellipses:** Represent multi-valued attributes

**CHAPTER 5**

**SALESFORCE**

The Salesforce Platform stores data in relational tables. The records in these tables contain data for the structure of the platform itself as well as user created data. For example, the data about the configuration and settings of an account are already in-built as a relational table. But you can also create your own tables to store data specific to your business like the 'dispatch schedule' for a week assuming you are a courier company.

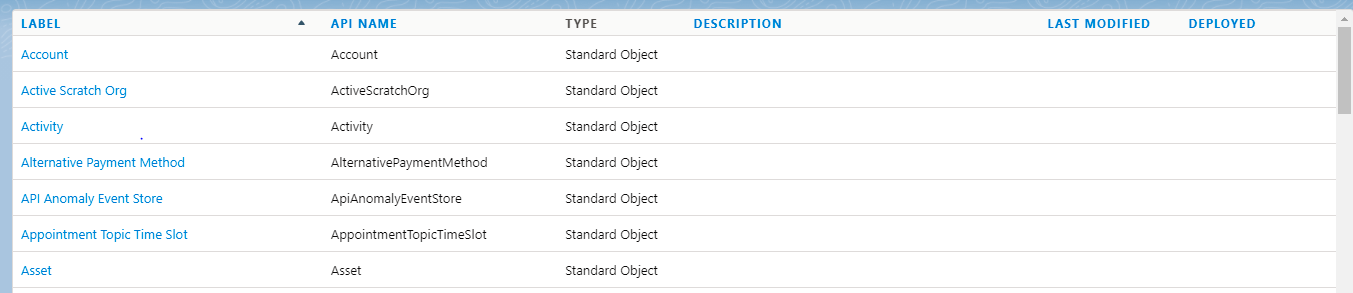
These relational tables are roughly referred to as API Objects or only objects in Salesforce.

There are three kinds of Salesforce objects.

* Standard Objects − The objects already created for you by the Salesforce platform.
* Custom Objects − These are the objects created by you based on your business processes.
* External Objects − The objects which you create map to the data stored outside your organization.

**5.1 Standard Objects**

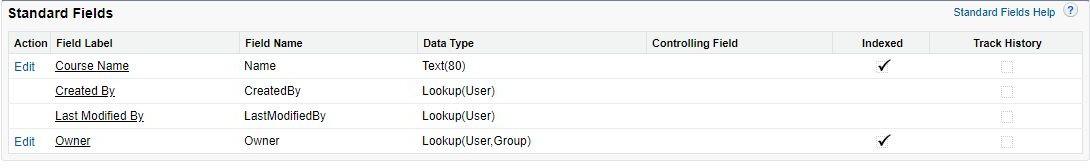
These are the objects which already exist in the Salesforce platform to manage the configurations and settings of the environment. Once you log in to the salesforce platform, you can see the available objects.



**Table 5.1**

**5.2 Standard Fields**

These are the some fields which already exist in the standard objects. When you click the standard object then you can see the standard fields.



**Table 5.2**

**3.3 Important Standard Objects**

In this section, we will discuss the important standard objects in Salesforce. The following table lists down the objects −

|  |  |  |
| --- | --- | --- |
| **Object Name** | **Meaning** | **Usage** |
| Account | Represents an individual account, which is an organization or person involved in the business like customers, competitors, partners, etc. | Use this object to query and manage accounts in your organization. |
| Account History | Represents the history of changes to the values in the fields of an account. | Use this object to identify changes to an account. |
| Case | Represents a case, which is a customer issue or problem. | Use the case object to manage cases for your organization. |
| Contact | Represents a contact, which is an individual associated with an account. | This object is used to manage individuals who are associated with an Account in the organization. |

|  |  |  |
| --- | --- | --- |
| ser | Represents a user in the organization. | This object is used to query information about users and also helps to provide and modify the information concerning the users. |
| Asset | Represents an item of commercial value, such as a product sold by the company or a competitor that a customer has purchased and installed. | This object is used to track assets previously sold into customer accounts. With asset tracking, a client application can quickly determine which products were previously sold or are currently installed at a specific account. |
| Domain | Read-only object that represents a custom Web address assigned to a site in your organization. | This read-only object is used to object to query the domains that are associated with each website in your organization. |

**Table 5.3**

As IT technology advanced, a new style of innovation emerged, in which a leading innovation company invites end-users to its open software service platform. With respect to this type of innovation, a lot of innovation studies were performed to understand the structure of the interaction among users and the platform provider from the perspective of network science. By concentrating only on the internal mechanisms among agents, the previous studies miss the opportunity to consider innovation through collective intelligence. A platform provider plays an important role in the innovation. In this research, we investigate the structure of a service network with empirical data gathered from Salesforce.com App Exchange and discuss the role of a platform provider in innovation through collective intelligence. Our results suggest that the platform provider led the innovation in the initial period and, then, third party developers gradually became innovation leaders. Our findings are expected to re-orient the research focus from internal mechanisms to the role of platform providers. [8]

With the appearance of distributed computing, associations are hoping to move their Customer Relationship Management (CRM) applications from an On-Premise environment or we can say local servers to an On-Demand environment that is on cloud server. On-Premise environment is when an association has the framework and programming inside their system. In an On-Demand environment, an outsider has the base and programming and charges the relationship in light of its participation. Salesforce is the principle On-Demand CRM thing. [9]

The advantages of cloud while supporting real-time service systems using the Salesforce platform. We build here a service management platform for the Polish Billiards and Snooker Association (PBSA), based on a real-time system located in a cloud. It allows PBSA managers to accomplish tasks in this system on-demand. And, it is deployed as a private cloud to grant access only to the employees from the snooker organization. [10]

In a recent scenario, IT industries are growing with the help of proper Utilization of available resources. The IT giants like Microsoft, Infosys, IBM, Oracle, & TCS are switching from their on premises IT setups to the cloud. Cloud computing is replacing the traditional model in which software applications are installed on on premise hardware, from desktop computers to server rooms, depending on the size of the business. The proposed work is about the cloud platform which is going to change all the traditional views of software, application, and product development Technologies. Salesforce.com is one of the best cloud providers available in the recent scenario. There are a number of reasons why IT industries are switching to the Cloud. And there are a number of reasons why Industries have to think about adopting the Salesforce.com cloud. The proposed work is about to focus on important and common features of salsforce.com. These features are common for any developer to learn and use in software, application and product development in salesforce.com. The goal of this proposed work is to show the available resources in the salesforce.com which are still new for the developers. This is an approach to make people familiar with the salesforce.com cloud provider. [11]

Summary form only given as follows. The strongest predictors of new product success is understanding market requirements early in the new product development (NPD) process. A direct salesforce is one of the best sources of new product ideas and market information, although not fully leveraged in many firms. A recent study of 248 salespeople in nineteen high-tech firms indicates wide variation in NPD involvement across and within a firm's salesforce. The study revealed that at the organization level, the length of the NPD cycle was associated with salesforce-initiated and NPD-headquarters initiated activities. In particular, the longer a firm's NPD cycle for product improvements, the lower the involvement the salesforce has in headquarters initiated involvement activities. A significant relationship exists between a firm's new product cycle time and the level of involvement in salesforce-initiated NPD predevelopment activities. At the salesperson level, several factors affect involvement in NPD activities. Most significant was the relationship between salesperson involvement and the distance between a salesperson's location and the NPD office site. The study shows that organizations can affect the degree of involvement that their salesforce or individual salespeople have in early phases of NPD. [12]

Motivation is important because it determines an individual's effort toward performing a task and it leads to an optimistic and challenging attitude at workplace. Incentive motivation is concerned with the way goals influence behavior. Incentive is the promise or stimulus for greater action. Incentives are something that is given in addition to wages. It means additional remuneration or benefit to an employee in recognition of achievement or better work. Organizations in emerging markets are more likely to give group rewards and generally do not encourage risk taking as much as developed economies. Money motivates employees to some extent but there are other powerful sources of motivation such as interesting and challenging tasks and assignments, praise, good interpersonal relations etc. This paper analyzes incentives plan structure of sales people in Croatia. Advantages and disadvantages of different compensation plans and benefits for salespeople are discussed. [13]

Modern business decision models are often very complicated due to a deluge of information. Evaluation and diagnostics of such decision models is extremely challenging due to many factors, including the complexity and volume of data. In addition, since there is no ideal data sample to construct a control group for comparison studies, performance evaluation and diagnostics of business actions can easily be distorted by selection bias. In this paper, we design a framework to analyze this sample bias issue under a practical business scenario. In particular, we focus on: a) identification of the key factors which drive selection bias during the business decision; b) evaluation of the performance of business actions with consideration of the identified selection bias. We evaluate baseline analytics tools on the worldwide sales-force data of a large global corporation and clearly demonstrate that the selection bias issue makes the usual evaluation very unstable and not trustable. However, by removing such detected sample bias, our framework can generate reasonable diagnostics results across different dimensions. The implemented analysis tool was applied to a worldwide business opportunity dataset of a multinational Fortune 500 corporation; the analytics results clearly show the significance of such a bias detection-based evaluation framework for sales-force optimization. [14]

Estimating generalizable relationships between actions and results from historical samples, especially when there is a level of noise or randomness in that signal, is an important problem to address before making decisions on actions to take. Many business analytics problems require the optimal assignment of limited resources to actions and activities to maximize some result or objective such as profit. We present a novel approach to solving this class of analytics problems by modeling the relationship between resource effort and expected return as a dose-response signal and formulating its causal estimation as one of kernel regression. The estimated expected value and variance of the result are then used to optimize resource allocation so as to maximize expected response while minimizing the risk around response subject to business constraints. We apply this approach to the task of optimally assigning salespeople to enterprise clients using real-world data, and show that profit can be substantially increased with fewer salespeople and reduced risk. [15]

**CHAPTER 6**

**CREATE OBJECTS, VALIDATIONS AND TRIGGERS**

* 1. **Creating a Salesforce Custom Object**
* First, follow this path: Setup > Build > Create > Objects > on the screen, click on the new custom object button > enter label name, plural label, and object name > enter record name as a data type.
* To create a record name, first, consider the two data types:
* Text
* Auto-number

You also have some optional features while creating objects:

* Allow reports: If you tick off this checkbox, then only these objects will be available to make reports.
* Allow activities: If you tick off this checkbox, then you are able to make activities on this object.
* Track field history: When you tick off this checkbox, then you can merely track fields. You can track up to 20 fields for a single object.

Now, you have the following deployment modes:

* In development: If you opt for this, then this object will remain in the development mode. It will not be present for deployment.
* Deployed: When you opt for this, the object will be available for deployment.

The following options are available only when you are creating a custom object for the first time:

* Add notes and attachments related list
* Launch a new custom tab wizard after saving this custom object After completing all these settings, press the Save button.

If you do not select ‘Launch new custom tab wizard’ from the object creation page, then the object would be saved without the tab appearance. In such a case, first, you need to make a tab for this object. On the other hand, if you select this option, the object would be saved, along with a tab would be created, which will be visible to you.

**6.2 Creating a Validation Rule**

* From Setup, go to Object Manager and click Account
* In the left sidebar, click Validation Rules.
* Click New.
* Enter the following properties for your validation rule
* Rule Name: Account\_Number\_8\_Characters
* Error Condition Formula:
* LEN( Account Number) != 8
* Error Message: Account number must be 8 characters long.
* To check your formula for errors, click Check Syntax.
* Click Save to finish.



Fig 6.1 Validation Rule

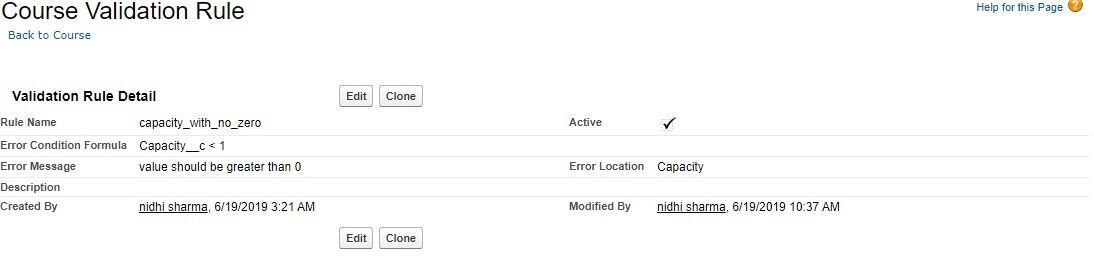


Fig 6.2 Store Validation Rule

* 1. **Apex Triggers**

The following steps show the different types of triggers you can use.

* From Setup, select Customize and then click the object that you want to add the trigger to.
* Click Triggers and then click New.
* To define your trigger, enter Apex code similar to this sample code.
* Replace [ObjectName] with the name of the object that you are adding the trigger to.
* Replace [NameOfCustomSetting] with the integration name you created on the Einstein Discovery Integration information page.
  + 1. **Apex trigger for single object with no data transformation**

trigger SetDealPrediction on [ObjectName] (after insert, after update) { if(System.isFuture()) return; if(ed\_insights.CheckRecursive.runOnce()) {

// custom Settings' name

String CONFIG\_NAME = '[NameOfCustomSetting]'; ed\_insights.TriggerHandler.insertUpdateHandle(CONFIG\_NAME);

}

}

* + 1. **Apex trigger with data transformation:**

trigger SetDealPrediction on [ObjectName] (after insert, after update)

{

if(System.isFuture()) return;

List<Map<String, String>> fieldValues = new List<Map<String, String>>();

// Iterate through all records that are supposed to be processed by this trigger

// Only need to populate the fields that are NOT mapped to the object.

// Fields mapped to the object are automatically queried via the recordID

for ([objectName] o: Trigger.new) {

// the field-value Map to be passed to the setPredition() method Map<String, String> fieldStringMap = new Map<String, String>();

fieldStringMap.put('Id', o.Id); //Must Have this or scoring will fail fieldStringMap.put('<model field 1>',<value 1>); fieldStringMap.put('<model field 2>',<value 2>); fieldValues.add(fieldStringMap);

}

// make sure there are fieldValues populated; otherwise, skip this if(fieldValues.size()>0)

{

ed\_insights.TriggerHandler.insertUpdateHandleForFieldValues(CONFIG\_NAME, JSON.serialize(fieldValues));

}

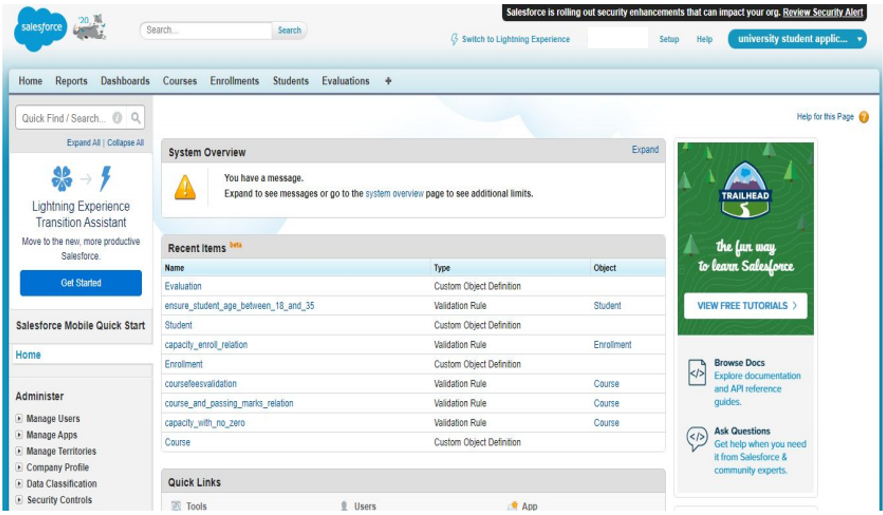
## CHAPTER 7

**OUTPUT SCREEN**

## 7.1 Screenshots of Modules:

## Application main page that shows all the menu related to application in one place.

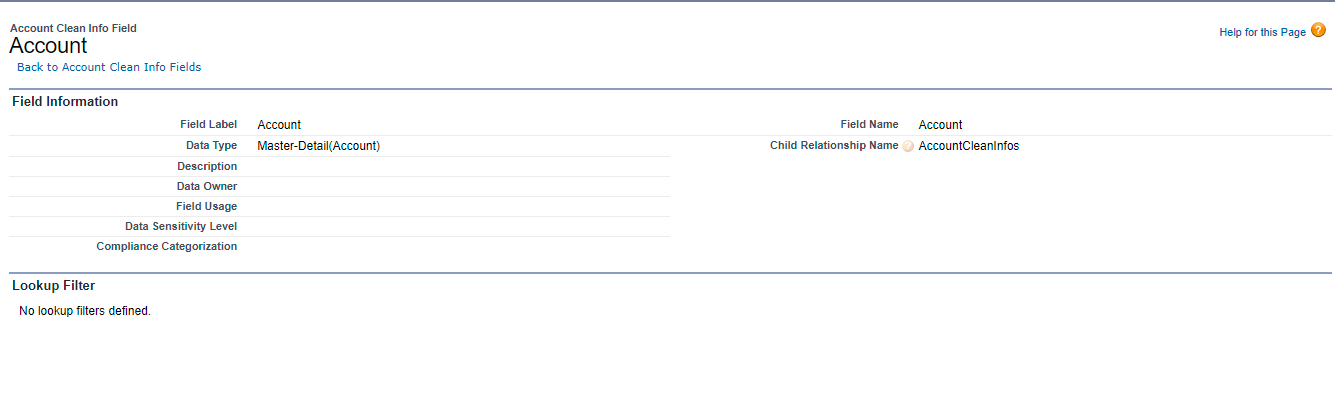
## Its having the menu bar along with dashboard to show the count of the active and deactivate Client/User and also shows the course related things like course wise Client/User no



**Fig 7.1 Showing home page**

## Account Object:

## Users can create the Account and add the course related information to show on the web page and add the several other information like the Available seats and the studies materials, fees, qualification marks and passing marks.



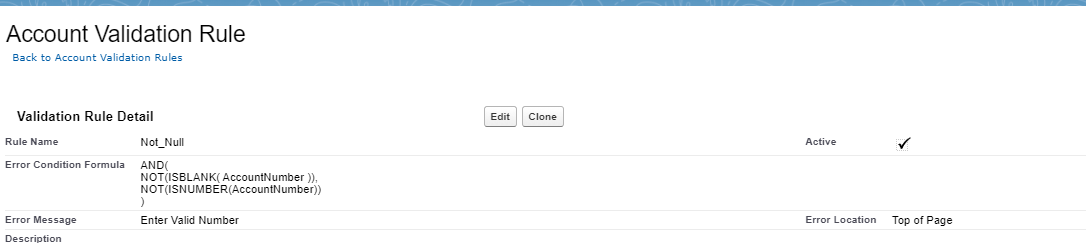
**Fig 7.2 Showing Account Object**

## 7.3 Validations on Account Object:

## On adding the Account to the page we also create some validation related rules for the data verifications.

## Check that Account id ..

## Users can’t click on save button until Account name blank.

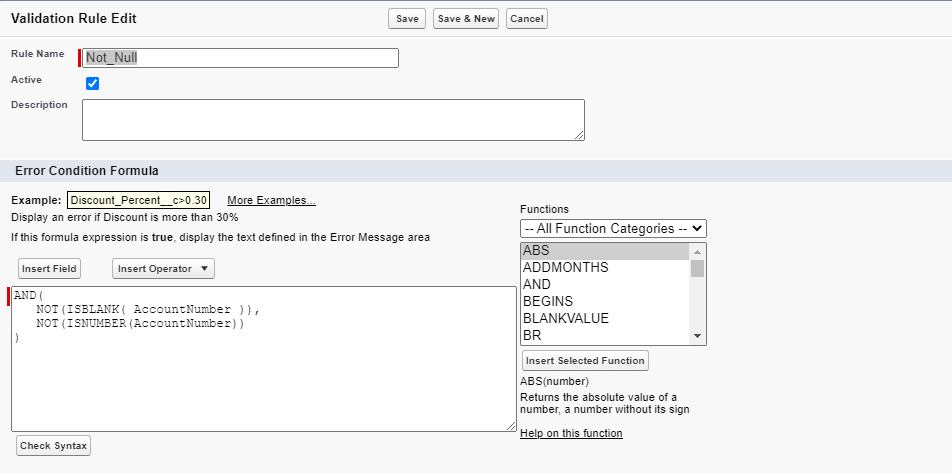


**Fig 7.3 Validations**

We have also put the barriers on the Account structures

1. Account Number Datatype should be Number
2. Account Number Cannot be Blank

.

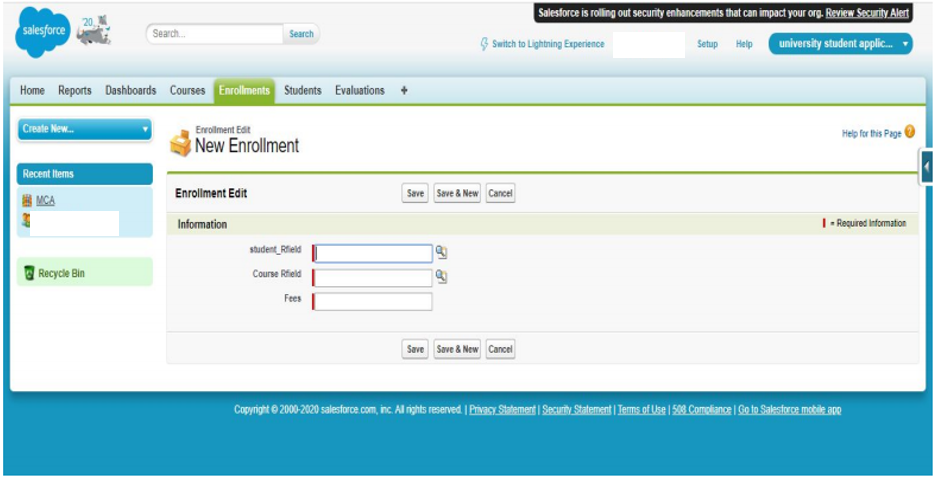


**Fig 7.4 Validations on Account Object**

## 7.4 Enrollment Object:

## Client/User mapping with the Account and Payment.

## This page is used to save or update the Client/User enroll Account and the payment which is charged from the user. This page is useful when a Client is not confused in the Account selection on the enrollment time or any Client/User who is eligible for the special discounts on the Payment fees .



**Fig 7.6 Showing all the Fields of Enrollment Object**

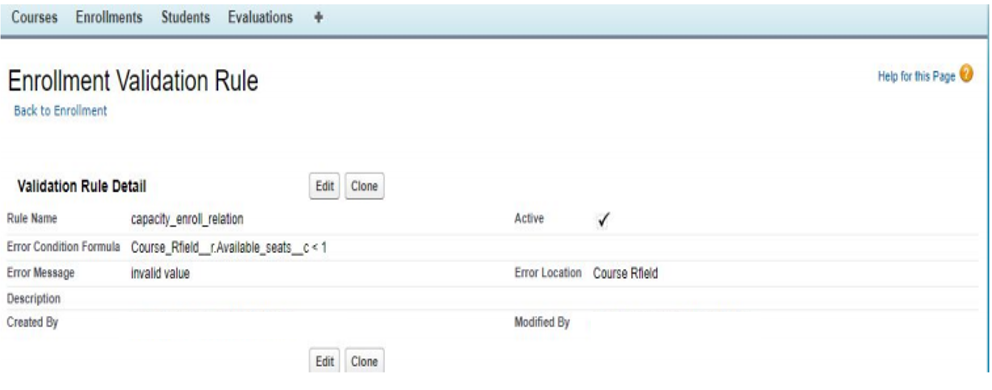
## 7.5 Validation on Enrollment Object

## Validation rules related to the enrollments are \

## For allow Client/User to enroll in Account related available seats should be greater than 0.

## Seats only shows in the integer format.

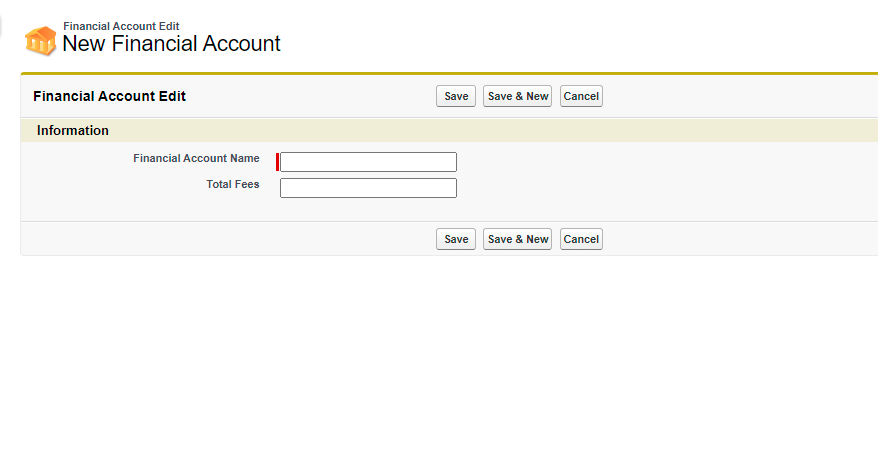
## Once course seats full then Client/User are not able to select that Account.



**Fig 7.7 Showing Validations and Master Detail Relationships**

## 7.6 Financial Account Object

## In this custom object a User can enter his/her financial Account details. This object stores the Client/User Financial data.

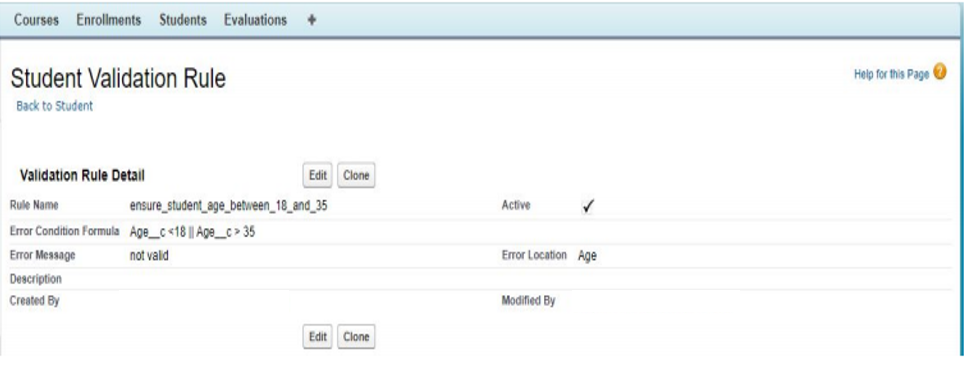


**Fig 7.8 Showing all the Fields of Financial Object**

## 7.7 Validations on Financial Account Object

On the Client/User enrollments we have put the validation like

1. Fess should be greater than 0.
2. Fee should be less than 50000$.
3. Fee should be Currency datatype.

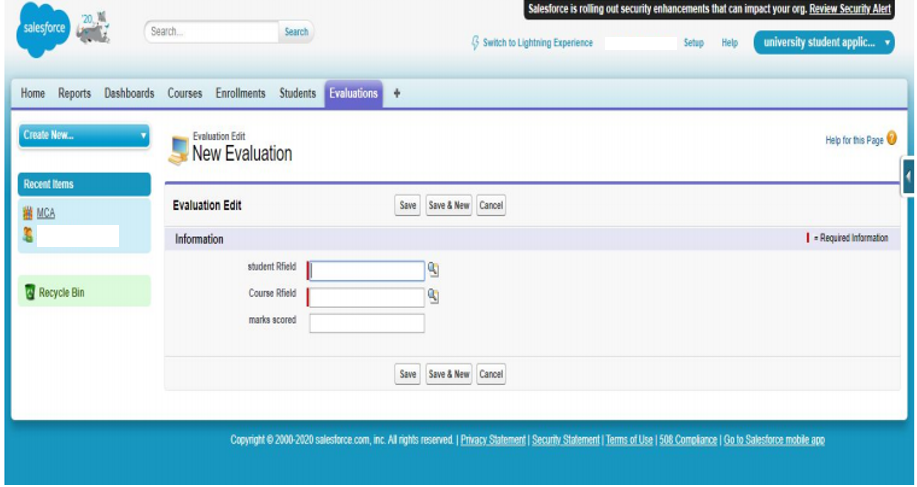


**Fig 7.9 Showing Validations on Financial Account Object**

## 7.8 Evaluation Object

## This page is used as the results page and users can find out the result by the Client Id or client name.

## Once a user give the required field to the page then its shows the marks score by the student’s and also shows the status.



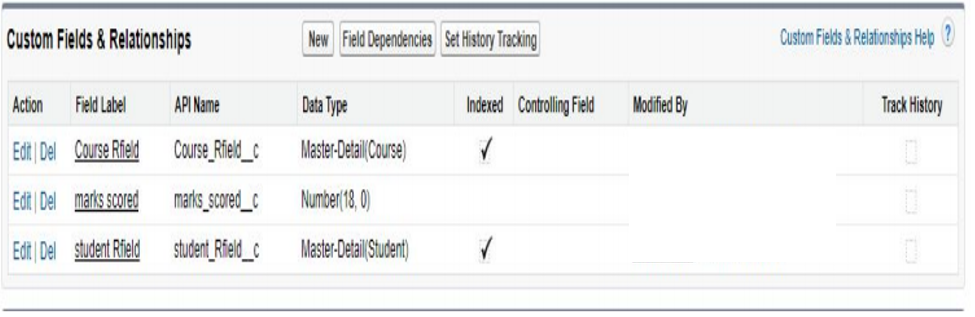
**Fig 7.10 Showing Evaluation Objec**

## 7.9 Relationships between Objects

## Below screen showing the relationship between the pages and the table of the application

## Account id (which is integer in the datatype) has the relation between the student and the marks table.

## Financial Account id (which is integer in the datatype) has the dependency with the Payment id.



**7.11 Showing Mater-Detailed Relationships between Objects**

**CHAPTER 8**

**TESTING**

**8.1 INTRODUCTION**

Testing is the integral part of any System Development Life Cycle. Insufficient and interesting applications tend to crash and result in loss of economic and manpower investment besides user’s dissatisfaction and downfall of reputation.

“Software Testing can be looked upon as one among many processes an organization performs, and that provides the last opportunity to correct any flaws in the developed system. Software Testing includes selecting test data that have more probability of giving errors.” The first step in System testing is to develop the plan for all aspects of system .Software is to be tested for the best quality assurance, an assurance that the system meets the specification and requirement for its intended use and performance.

System Testing is the most useful practical process of executing the program with the implicit intention of finding errors that makes the program fail.

**8.2 Types of Testing**

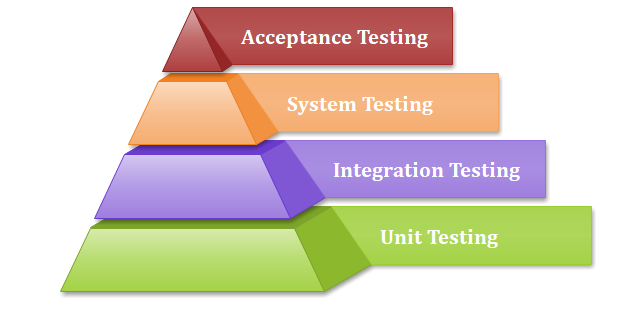


Fig 8.1 Testing Pyramid

**Black Box (Functional) Testing**:

Testing against specification of system or components. Study it by examining its inputs and related outputs. Key is to devise inputs that have a higher likelihood of causing outputs that reveal the presence of defects. Use experience and knowledge of the domain to identify such test cases. Failing this, a systematic approach may be necessary. Equivalence partitioning is where the input to a program falls into a number of classes, e.g. positive numbers vs. negative numbers. This type of test case design method focuses on the functional requirements of the software, ignoring the control structure of the program. Black box testing attempts to find errors in the following categories:

* Incorrect or missing functions.
* Interface errors.
* Errors in data structures or external database access.
* Performance errors.
* Initialization and termination errors.

**White Box (Structural) Testing:**

Testing based on knowledge of structure of components (e.g. by looking at source code). Advantage is that structure of code can be used to find out how many test cases need to be performed. Knowledge of the algorithm (examination of the code) can be used to identify the equivalence partitions. Path testing is where the tester aims to exercise every independent execution path through the component. All conditional statements tested for both true and false cases. If a unit has no control statements, there will be up to 2n possible paths through it. Static tools may be used 34to make this easier in programs that have a complex branching structure. Tools support. Dynamic program analyzers instrument a program with additional code. Typically this will count how many times each statement is executed. At the end print out a report showing which statements have and have not been executed. Problems with flow graph derived testing:

* Data complexity could not be taken into account.
* We cannot test all paths in combination.
* It is really only possible at unit and module testing stages because beyond that complexity is too high.

**Unit Testing:**

Unit testing concentrates on each unit of the software as implemented in the code. This is done to check syntax and logical errors in programs. At this stage, the test focuses on each module individually, assuring that it functions properly as a unit. In our case, we used extensive white-box testing at the unit testing stage. A developer and his team typically do the unit testing; the unit testing is done in parallel with coding; it includes testing each function and procedure.

**Incremental Integration Testing:**

Bottom up approach for testing i.e. continuous testing of an application as new functionality is added; Application functionality and modules should be independent enough to test separately done by programmers or by testers.

**Integration Testing:**

Testing of integration modules to verify combined functionality after integration .Modules are typically code modules, individual applications, client and server and distributed systems.

**Functional Testing:**

This type of testing ignores the internal parts and focuses on whether the output is as per requirement or not .Black box type testing geared to functionality requirements of an application.

**System Testing:**

Entire system is tested as per the requirements. Black box type test that is based on overall requirement specifications covers all combined parts of a system.

* 1. **Some Important Observations**

**8.3.1 System Testing and Validation Results.**

System testing was done after the system was duly coded. Individual modules of the system were checked to ensure they are fully functional units before integrating them. This was done by examining each unit; each script was checked to ensure that it functions as required and that it performs exactly as intended. The success of each individual unit gave us the go ahead to carry out integration testing.

The system was validated using a short questionnaire that was filled by representatives of the users who were let to interact with the system using test data and provided feedback about the system features. This was done to assess if the system met their needs and requirements as regards. It was found out that the system performed in conformance to the then defined user needs and requirements. Results of the validation are shown as percentages of respondents against each requirement.

**8.3.2 Testing Test Scenarios**

1. Check if the page load time is within the acceptable range.  
2. Check the page load on slow connections.  
3. Check the response time for any action under light, normal, moderate, and heavy load conditions.  
4. Check the performance of database stored procedures and triggers.  
5. Check the database execution time.  
6. Check for load testing of the application.  
7. Check for the Stress testing of the application.  
8. Check CPU and memory usage under peak load conditions.

We have checked for scenarios and found that our system performed well in the circumstances.

**CHAPTER 9**

**CONCLUSION AND FUTURE SCOPE**

**9.1 Conclusion**

In this project we can provide the data of Account about Financial to the user so they can easily find the Account according to their needs, requirements and their financial conditions.

They don’t need to go anywhere to get the details of the Account from anyone else from the institutions they can get the data in just a one click.

A software project means a lot of experience. I learned a lot through this project. This project has sharpened our concept of cloud computing.

It provides easy methods to manage the load of work easily for the users.

It is much faster and more efficient as the data once entered can be used and accessed easily.

This project has given me an ample opportunity to design, code, test and implement an application. This has helped in putting into practice various Software Engineering principles concepts like maintaining integrity and consistency of data.

**9.2 Future Scope**

* The Future scope is to make the system more user friendly and enhanced.
* And we will make a Mobile app for our system.
* I will add Helping BOT in the system.
* Online Payment would be introduced to conduct online payment.
* Further, the Client can upload the Screenshot of their review on this site and Client/User also who had missed those payments can provide their feedback.

## References

1. Kovach, J. J., Atasu, A., & Banerjee, S. (2018). Salesforce incentives and remanufacturing. Production and Operations Management, 27(3), 516-530.
2. Anderson, E., & Oliver, R. L. (1987). Perspectives on behavior-based versus outcome-based salesforce control systems. Journal of Marketing, 51(4), 76-88.
3. Samaraweera, M., & Gelb, B. D. (2015). Formal salesforce controls and revenue production: a meta-analysis. Journal of Personal Selling & Sales Management, 35(1), 23-32.
4. Ntayi, J. M., Munene, J. C., & Eyaa, S. (2010). Salesforce behavioural performance of accounts relationship managers (ARMS) in Uganda's commercial banks: A qualitative analysis. Journal of Retail & Leisure Property, 9(1), 5-23
5. . Yaish, H., Goyal, M., & Feuerlicht, G. (2014). Evaluating the performance of multi-tenant elastic extension tables. Procedia Computer Science, 29, 614-626.
6. Peng, J., Zhang, X., Lei, Z., Zhang, B., Zhang, W., & Li, Q. (2009, December). Comparison of several cloud computing platforms. In 2009 Second international symposium on information science and engineering (pp. 23-27). IEEE.
7. Li, M., Kankanhalli, A., & Kim, S. H. (2016). Which ideas are more likely to be implemented in online user innovation communities? An empirical analysis. Decision Support Systems, 84, 28-40
8. Mason, M. (2018). Overcoming the risks of privileged user abuse in Salesforce. Network Security, 2018(8), 6-8
9. Fisher, S. (2007, May). The architecture of the apex platform, salesforce. com's platform for building on-demand applications. In the 29th International Conference on Software Engineering (ICSE 07 Companion) (pp. 3-3). IEEE..
10. Espadas, J., Concha, D., & Molina, A. (2008, October). Application Development over Software-as-a-Service platforms. In 2008 the third international conference on software engineering advances (pp. 97-104). IEEE.
11. Churchill Jr, G. A., Ford, N. M., & Walker Jr, O. C. (1976). Organizational climate and job satisfaction in the salesforce. Journal of Marketing Research, 13(4), 323-332.
12. Dai, Y., & Chao, X. (2016). Price delegation and salesforce contract design with asymmetric risk aversion coefficient of sales agents. International Journal of Production Economics, 172, 31-42.
13. Mietzner, R., & Leymann, F. (2010, December). A self-service portal for service-based applications. In 2010 IEEE International Conference on Service-Oriented Computing and Applications (SOCA) (pp. 1-8). IEEE.
14. Hayes, B. (2008). Cloud computing..
15. Netto, M.; Calheiros, R.; Rodrigues, E.; Cunha, R.; Buyya, R. (2018). "Salesforce performance and satisfaction as a function of individual difference, interpersonal, and situational factors”.