**TEAM INFORMATION MANAGEMENT SYSTEM**

**CORPORATE TRUST(CT)**

**A PROJECT REPORT**

***Submitted by***

**PADMAJA.S**

**SUWETHA.S**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**ANNA UNIVERSITY : CHENNAI - 600 025**

**APRIL 2016**

**APPENDIX 2**

**ANNA UNIVERSITY : CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report **“TEAM INFORMATION MANAGEMENT SYSTEM”** is the bonafide work of***“*BHAVANI.M, BHAVEENA.P, PADMAJA.S, VAISHNI.R, DINESH SURYA.R”**  who carried out the project work under my supervision.

**SIGNATURE SIGNATURE**

**HEAD OF THE DEPARTMENT SUPERVISOR**

#### ABSTRACT

Team Information Management System (TIMS) is fast moving, knowledge-intensive and requires a substantive amount of teamwork. In order to develop quality software, teams need to leverage the skills and knowledge of each team member.TIMS teams who engage in learning at a group level can perform more effectively and efficiently. However, relative to other disciplines, knowledge and literature about team learning in TIMS research is new and dispersed. This fact hampers the cumulative progress in research that seeks to answer questions about how TIMS teams learn to work together and improve their performance. We draw on and extend the classification scheme of Edmondson et al. (2007) and conduct a review of TIMS team learning research literature. We synthesize the main findings and highlight the limitations of existing approaches. We emphasize potential directions for future research while focusing on the resulting implications for TIMS management and methodology. We further demonstrate that there are four distinctive streams in TIMS team learning research that differ in the manner that they conceptualize team learning, underlying theories, and research methodologies. Finally, we illustrate how these differing streams can cross-fertilize and thereby present notable aspects of team learning presently addressed by related disciplines for which there is scant or non-existent TIMS research.

**CHAPTER 1**

**INTRODUCTION**

* 1. **Team Information Management System**

Information Management System (IMS) is a general term for software designed to facilitate the storage, organization and retrieval of information. Information Management System provides information for the managerial activities in an organization. The main purpose of this research is to provide decision-making process and enable the organizations planning, control and operational functions to be carried out effectively.TIMS is basically concerned with processing data into information and is communicated to various department’s team members in an organization for appropriate decision-making.

* 1. **Aim:**

This (TIMS-Team Information Management System) focusses on the individual’s professional performance measurement that relates to the overall financial performance and the production output of the company.

* 1. **Project Objective:**

The main objective is to implement a software that performs the information management tasks:

* To provide continuous updations of releases and resources in each domain as well as team.
* To inculcate the knowledge over every project, it’s members, their skill sets, the training to be provided to them for flexibility in the project involved.
  1. **Overview:**

In this software, the information provided by the team managers get stored in a database and it concentrates on key performance indication of each employee in their allocated projects. Also the personal information about employees that helps in team building exercises which could develop the team spirit and communication between each other is available. By making this software automatic, we can thus reduce a lot of time and manual effort which can be used for other significant purposes.

* 1. **Problem Definition:**

TIMS is a tracker of all activities driven in the division at each sub level of resources and tasks. This is an internal web application which helps to find progress, plans, resources etc. by the managers and team members. It is necessary to monitor and assess the company employee’s performance that links to rewards to performance. Manager’s time and effort is ruined in the process of judging particularly in performance areas while doing the management processes manually. It is important to link the personal objectives of the employee with the production output of the company which requires software to make this process simple and effective.

**CHAPTER 2**

**LITERATURE REVIEW**

There have been numerous kinds of systems that have been developed over the past several years. These information systems had helped to fulfill the needs and requirements of decision making not only at managerial but also at the operational level. Every organization develops its own team information management system (TIMS) which is totally dependent on the personal needs of the organizations. Management information system developed for one organization is useless for other organizations with different requirements.

In the team information management system, not only the system itself is important but to get the maximum advantages from the system it is important that the human intelligence, perception and judgment must be powerful and strong enough to get combined with the system information This combination will provide managers with the unique and valuable tool for the information management in any company.

The information system which is composed of formalized procedures that can provide all of the relevant appropriate information at all levels to all of the managers is called Team information management system (TIMS).

This system uses all of the internal and external resources to provide data and information to the end users which enable them to take timely decisions that are effective from the directing, planning and controlling point of view. As a whole this is an efficient system that can provide all of the required information to the management at all levels at appropriate times. Management information system is basically an affective combination of human and technology resources that results in data storage, collection, and communication, data retrieval and usage. In managerial operations and business planning, these systems play a vital role in business success.

In literature, there are several definitions of the team information management systems. The system is capable of converting the collected data from the routine user and machine interactions in to the effective information which later on is used by the decision makers to make efficient decisions. Information systems are of significant importance in any type of organization, generally, it is really impossible for any organization to operate without any sort of information system.

**CHAPTER 3**

**METHODOLOGY**

**3.1 Existing Methodology:**

The prevailing methodology of Team Information Management System is carried out manually. It involves entering data such as the application details, resource details, task assignment and completion details, release details and each status updates manually.

The manager has to collect information from each and every individual in the form of Excel sheets from the employees and he has to update those information in the database so as the details of anyone could be retrieved easier. But this system involves a lot of human work and requires much more time.

**3.1.1 Drawbacks of Existing Methodology:**

Manual systems put pressure on people to be correct in all details of their work at all times, the problem being that people aren’t perfect, however much each of us wishes we were.  With manual systems the level of service is dependent on individuals and this puts a requirement on management to run training continuously for staff to keep them motivated and to ensure they are following the correct procedures.  It can be all to easy to accidentally switch details and end up with inconsistency in data entry or in hand written orders.  This has the effect of not only causing problems with customer service but also making information unable be used for reporting or finding trends with data discovery.  Reporting and checking that data is robust can be timely and expensive.  This is often an area where significant money can be saved by automation.

It takes more effort and physical space to keep track of paper documents, to find information and to keep details secure.  When mistakes are made or changes or corrections are needed, often a manual transaction must be completely redone rather than just updated.  With manual or partially automated systems information often has to be written down and copied or entered more than once.  Systemization can reduce the amount of duplication of data entry.

Another impact of manual systems is on Customer service. Customer queries can be difficult to respond to as information is stored in different places and may even require that you find the right person before being able to respond.  This is no good if they are out to lunch or only work part time.

Inconsistency in data entry, room for errors, miskeying information.

* + - Large ongoing staff training cost.
    - System is dependent on good individuals.
    - Reduction in sharing information and customer services.
    - Time consuming and costly to produce reports.
    - Lack of security.
    - Duplication of data entry.

**3.2 Proposed Methodology:**

**3.2.1 Modules Involved:**

* **Master Store**:

This is to create and collect basic info about the team resource, application names, group or division etc. **Master data** represents the business objects which are agreed on and shared across the enterprise. Commonly confused with [Reference Data](https://en.wikipedia.org/wiki/Reference_Data) – Information that is key to the operation of a business – it alsocan cover [transactional](https://en.wikipedia.org/wiki/Dynamic_data), [unstructured](https://en.wikipedia.org/wiki/Unstructured_data), analytical, [hierarchical](https://en.wikipedia.org/wiki/Hierarchical_database_model) and [meta](https://en.wikipedia.org/wiki/Metadata) data. It is the primary focus of the [Information Technology](https://en.wikipedia.org/wiki/Information_Technology) (IT) discipline of [Master Data Management](https://en.wikipedia.org/wiki/Master_Data_Management) (MDM). Master data management (MDM) is a comprehensive method of enabling an enterprise to link all of its critical data to one file, called a master file, that provides a common point of reference. When properly done, master data management streamlines data sharing among personnel and departments. In addition, master data management can facilitate computing in multiple system architectures, platforms and applications.

Processes commonly seen in master data management include source identification, data collection, [data transformation](https://en.wikipedia.org/wiki/Data_transformation), [normalization](https://en.wikipedia.org/wiki/Database_normalization), rule administration, error detection and correction, data consolidation, [data storage](https://en.wikipedia.org/wiki/Data_storage_device), data distribution, data classification, taxonomy services, item master creation, schema mapping, product codification, data enrichment and [data governance](https://en.wikipedia.org/wiki/Data_governance).

The selection of entities considered for master data management depends somewhat on the nature of an organization. In the common case of commercial enterprises, master data management may apply to such entities as customer ([customer data integration](https://en.wikipedia.org/wiki/Customer_data_integration)), product ([product information management](https://en.wikipedia.org/wiki/Product_information_management)), employee, and vendor. Master data management processes identify the sources from which to collect descriptions of these entities. In the course of transformation and normalization, administrators adapt descriptions to conform to standard formats and data domains, making it possible to remove duplicate instances of any entity. Such processes generally result in an organizational master data management repository, from which all requests for a certain entity instance produce the same description, irrespective of the originating sources and the requesting destination.

The tools include [data networks](https://en.wikipedia.org/wiki/Data_networks), [file systems](https://en.wikipedia.org/wiki/File_systems), a [data warehouse](https://en.wikipedia.org/wiki/Data_warehouse), [data marts](https://en.wikipedia.org/wiki/Data_mart), an [operational data store](https://en.wikipedia.org/wiki/Operational_data_store), [data mining](https://en.wikipedia.org/wiki/Data_mining), [data analysis](https://en.wikipedia.org/wiki/Data_analysis), [data visualization](https://en.wikipedia.org/wiki/Data_visualization), [data federation](https://en.wikipedia.org/w/index.php?title=Data_federation&action=edit&redlink=1) and [data virtualization](https://en.wikipedia.org/wiki/Data_virtualization). One of the newest tools, virtual master data management utilizes data virtualization and a persistent metadata server to implement a multi-level automated master data management hierarchy.

There are several ways in which master data may be collated and distributed to other systems. This includes:

* [**Data consolidation**](https://en.wikipedia.org/w/index.php?title=Data_consolidation&action=edit&redlink=1) – The process of capturing master data from multiple sources and integrating into a single hub ([operational data store](https://en.wikipedia.org/wiki/Operational_data_store)) for replication to other destination systems.
* [**Data federation**](https://en.wikipedia.org/wiki/Federated_database_system) – The process of providing a single virtual view of master data from one or more sources to one or more destination systems.
* [**Data propagation**](https://en.wikipedia.org/w/index.php?title=Data_propagation&action=edit&redlink=1)– The process of copying master data from one system to another, typically through point-to-point interfaces in legacy systems.
* **Utility:**

This is to provide data for the other components based on the request which are reusable. In [computer science](https://en.wikipedia.org/wiki/Computer_science) and [software engineering](https://en.wikipedia.org/wiki/Software_engineering), **reusability** is the use of existing assets in some form within the software product development process. Assets are products and by-products of the software development life cycle and include code, software components, test suites, designs and documentation. Leverage is modifying existing assets as needed to meet specific system requirements. Because reuse implies the creation of a separately maintained version of the assets, it is preferred over leverage.

The ability to reuse relies in an essential way on the ability to build larger things from smaller parts, and being able to identify [commonalities](https://en.wikipedia.org/wiki/Commonality) among those parts. Reusability is often a required characteristic of [platform](https://en.wikipedia.org/wiki/Platform_(computing)) software. Reusability brings several aspects to [software development](https://en.wikipedia.org/wiki/Software_development) that do not need to be considered when reusability is not required.

Reusability implies some explicit management of [build](https://en.wikipedia.org/wiki/Software_build), [packaging](https://en.wikipedia.org/wiki/Packaging), [distribution](https://en.wikipedia.org/wiki/Distribution_(business)) , [installation](https://en.wikipedia.org/wiki/Installation_(computer_programs)) , [configuration](https://en.wikipedia.org/wiki/Computer_configuration), [deployment](https://en.wikipedia.org/wiki/Software_deployment), [maintenance](https://en.wikipedia.org/wiki/Software_maintenance) and [upgrade](https://en.wikipedia.org/wiki/Upgrade) issues. If these issues are not considered, software may appear to be reusable from [design](https://en.wikipedia.org/wiki/Software_design) point of view, but will not be reused in practice.

Software reusability more specifically refers to design features of a software element (or collection of software elements) that enhance its suitability for reuse.

Many reuse design principles were developed at the WISR workshops.

Candidate design features for software reuse include:

* [Adaptable](https://en.wikipedia.org/wiki/Adaptability)
* Brief: small size
* [Consistency](https://en.wikipedia.org/wiki/Consistency)
* [Correctness](https://en.wikipedia.org/wiki/Correctness_(computer_science))
* [Extensibility](https://en.wikipedia.org/wiki/Extensibility)
* [Fast](https://en.wikipedia.org/wiki/Speed)
* Flexible
* [Generic](https://en.wikipedia.org/wiki/Generic_programming)
* Localization of volatile ([changeable](https://en.wikipedia.org/w/index.php?title=Changeable_design&action=edit&redlink=1)) design assumptions ([David Parnas](https://en.wikipedia.org/wiki/David_Parnas))
* [Modularity](https://en.wikipedia.org/wiki/Modularity_(programming))
* [Orthogonality](https://en.wikipedia.org/wiki/Orthogonality)
* [Parameterization](https://en.wikipedia.org/wiki/Parameterization)
* Simple: low [complexity](https://en.wikipedia.org/wiki/Complexity_(disambiguation))
* [Stability](https://en.wikipedia.org/wiki/Stability_Model) under changing [requirements](https://en.wikipedia.org/wiki/Requirements)
* **Transaction Services**:

This is to provide services to handle project’s multiple operations such as task creation, assignment, code release entry etc. These services rely on the master data information captured in the master store module. A transaction is a computation (i.e., program in execution) that accesses and possibly modifies a database:

* Not the source code; not the binaries
* Can be interleaved with other transactions
* But guarantees certain correctness properties
* The purpose of the transaction concept is to avoid the problems (“race conditions”) that may arise from interleaving.

**Functions of Transaction Processing Systems**

A ***transaction*** is an elementary activity conducted during business operations. ***Transaction processing systems*** (TPS) process the company's business transactions and thus support the operations of an enterprise. A TPS records a non-inquiry transaction itself, as well as all of its effects, in the database and produces documents relating to the transaction.

TPS are necessary to conduct business in almost any organization today. TPSs bring data into the organizational databases, these systems are also a foundation on which management oriented information systems rest.

**System Charts**

Systems charts are well-established tools which are used to describe TPSs. These charts show the sources of input into the system, major processing steps, data storage, and systems outputs.

**Transaction Processing Modes**

Transaction processing may be accomplished in one of two modes:

* + - On-line mode
    - Batch mode

**Characteristics of on-line transaction processing**:

* Each transaction is completely processed immediately upon entry.
* OLAP is the most common mode of used today
* More costly than batch processing
* Database is always up to date
* Require the use of fast secondary storage such as magnetic disks

**Characteristics of batch transaction processing:**

* Relies on accumulating transaction data over a period of time and then processing the entire batch at once.
* Batch processing is usually cyclic: daily, weekly, or monthly run cycle is established depending on the nature of the transactions
* Cheaper than on-line processing
* Easier to control than on-line processing
* Database is constantly out of date
* Batch processing is now being captured using disk files
* **Reporting:**

This to provide report for showing the resource allocation, task status, release info etc. This module actually gets data based on the data entered in the transaction services.

MRSs may produce reports either directly from a database collected and maintained by a transaction processing system, or from databases spun off from the central database for the purpose. Separate spin off databases may be created for several reasons, such as:

* + Avoiding interference and delays in transaction processing
  + Maintaining the security of central databases
  + Economizing by using local databases accessible to local managers to counter heavy telecommunications costs of working with a central database.

**MRSs provide the following types of reports:**

* **Scheduled (Periodic) Reports**
* Are furnished on a daily, weekly, biweekly, or other regular basis depending on the decision-making need.
* The format and the informational content of scheduled reports are fixed in advance. However, it is crucial to identify the essential informational needs of various managers to facilitate each manager's decision making and to prevent information overload.
* The concept of responsibility reporting is generally applied - managers receive reports within their specific areas of responsibility.
* A hierarchy of performance reports arises, with each report including only the items that the manager can control.
* **Exception Reports**
* Produced only when pre-established about of bounds@ conditions occur and containing only the information regarding these conditions. Exception reporting helps managers avoid perusal of incident figures and concentrate on deviations from the norm and on unusual events.
* **Demand (Ad Hoc) Reports**
* The ability of a manager to request a demand report or screen output as needed enhances the flexibility of MRS use and gives the end user the capability to request the information and format that best suit his or her needs. Query languages provided by DBMSs make data accessible for demand reporting.
* Tracking systems - management reporting systems that continuously track the status of a project or a product under development.
* Asset management systems - TPS (Transaction Processing Systems)and MRS(Management Reporting Systems) that maintain and report on-line the status of financial inventory, and human resources assets.

**CHAPTER 4:**

**SOFTWARE SPECIFICATIONS**

**4.1 Java 7/J2EE:**

Developers today increasingly recognize the need for distributed, transactional, and portable applications that leverage the speed, security, and reliability of server-side technology. **Enterprise applications** provide the business logic for an enterprise. They are centrally managed and often interact with other enterprise software. In the world of information technology, enterprise applications must be designed, built, and produced for less money, with greater speed, and with fewer resources.

With the Java Platform, Enterprise Edition (Java EE), development of Java enterprise applications has never been easier or faster. The aim of the Java EE platform is to provide developers with a powerful set of APIs while shortening development time, reducing application complexity, and improving application performance.

The Java EE platform is developed through the Java Community Process (JCP), which is responsible for all Java technologies. Expert groups composed of interested parties have created Java Specification Requests (JSRs) to define the various Java EE technologies. The work of the Java Community under the JCP program helps to ensure Java technology's standards of stability and cross-platform compatibility.

The Java EE platform uses a simplified programming model. XML deployment descriptors are optional. Instead, a developer can simply enter the information as an **annotation** directly into a Java source file, and the Java EE server will configure the component at deployment and runtime. These annotations are generally used to embed in a program data that would otherwise be furnished in a deployment descriptor. With annotations, you put the specification information in your code next to the program element affected.

In the Java EE platform, dependency injection can be applied to all resources a component needs, effectively hiding the creation and lookup of resources from application code. Dependency injection can be used in Enterprise JavaBeans (EJB) containers, web containers, and application clients. Dependency injection allows the Java EE container to automatically insert references to other required components or resources, using annotations.

**4.2 User Interface:**

**4.2.1: JQuery:**

1. **JQuery scrollex & JQuery scrolly:**

**Nifty scroll events for jQuery**

Adds slick new scroll events to jQuery (like enter and leave) so you can drop scrolling effects.

**Usage**

Load up jquery.scrollex.min.js (after jQuery):

Then call scrollex() on a selector with a **configuration object**, which is where you'll associate handlers with the events you want to use and set various Scrollex options (see **Configuration Reference** below). For example, this uses the enter and leave events to change the background color of #foobar to green when we scroll within its boundaries (its *contact area*), then back again when we scroll out of it:

$(function() {

$('#foobar').scrollex({

enter: function() {

// Set #foobar's background color to green when we scroll into it.

$(this).css('background-color', 'green');

},

leave: function() {

// Reset #foobar's background color when we scroll out of it.

$(this).css('background-color', '');

}

});

});

**Events**

Scrollex supports the following events:

**enter**

Triggered when the viewport enters an element's contact area. Behavior can be tweaked using the mode, top, and bottom options (see next section).

**leave**

Triggered when the viewport leaves an element's contact area. Behavior can be tweaked using the mode, top, and bottom options (see next section).

**initialize**

Triggered as soon as scrollex() is called on an element.

**terminate**

Triggered as soon as unscrollex() is called on an element, which is used to gracefully undo a previous scrollex() call.

**scroll**

Triggered as the viewport scrolls through an element. The handler associated with this event is called with a normalized value representing how far the viewport has scrolled through the element (between 0 and 1, although values outside this range are possible if the viewport is above or below the element). For example:

$(function() {

$('#foobar').scrollex({

scroll: function(progress) {

// Progressively increase #foobar's opacity as we scroll through it.

$(this).css('opacity', Math.max(0, Math.min(1, progress)));

}

});

});

**mode, top, and bottom**

Events that depend on the viewport's position relative to an element's contact area (currently just enter and leave) can be further tweaked using the mode, top, and bottom options.

**mode**

This determines the rules Scrollex uses to figure out when the viewport is considered "inside" or "outside" an element's contact area. Can be any of the following:

| **Value** | **Behavior** |
| --- | --- |
| default | Element's contact area must fall within the viewport. |
| top | Top viewport edge must fall within the element's contact area. |
| bottom | Bottom viewport edge must fall within the element's contact area. |
| middle | Midpoint between top/bottom viewport edges must fall within the element's contact area. |

**top and bottom**

These let you "pad" the edges of an element's contact area using either a pixel value (150) a percentage of that element's height (25%), or a percentage of the viewport's height (20vh). Positive values work inward and shrink the contact area, while negative values work outward and expand the contact area. For example, this expands the contact area of #foobar by 20% of its height in both directions, resulting in enter triggering a bit earlier and leave a bit later:

$(function() {

$('#foobar').scrollex({

top: '-20%',

bottom: '-20%',

enter: function() {

$(this).css('background-color', 'green');

},

leave: function() {

$(this).css('background-color', '');

}

});

});

**Configuration Reference**

| **Name** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- |
| enter | function | null | **Enter** event. |
| leave | function | null | **Leave** event. |
| initialize | function | null | **Initialize** event. |
| terminate | function | null | **Terminate** event. |
| scroll | function | null | **Scroll** event. |
| mode | string | default | Mode (default, top, bottom, or middle). |
| top | integer, string | 0 | Top padding (in pixels, %, vh). |
| bottom | integer, string | 0 | Bottom padding (in pixels, %, or vh). |
| delay | integer | 0 | Delay (in ms) between position checks. |

**License**

Scrollex is released under the MIT license. Copyright © n33

1. **Skel.min(JS):**

Skel is a lightweight framework for building responsive sites and web apps.

**Features include:**

* Access to CSS breakpoints via JS (enabling stuff like if (skel.breakpoint("small").active) { /\* do something specific for small displays \*/ }).
* Events, including the commonly used (load, ready) and special ones just for breakpoints (+breakpoint, -breakpoint).
* Vars, for convenient access to information about the client's browser, operating system, and more.
* Extendable with modules (like Layout and Viewport).
* We use version 3.3.0 of skel copyrighted n33

## Modules

Previous versions of Skel were basically monolithic (ie. skel.min.js contained **all** of Skel's functionality, even if you didn't need all of it). As of version 3, Skel uses a modular approach to give you more flexibility in how you use it. Here's how it (currently) breaks down:

* **Skel** (skel.min.js)

(Main framework) Provides JS access to CSS breakpoints, events, and other tools.

* **Layout** (skel-layout.min.js)

(Module) Adds CSS and page layout tools, including a CSS grid system, browser resets and more.

* **Viewport** (skel-viewport.min.js)

(Module) Adds simplified viewport management (including support for multiple viewport <meta> tags).

* **Skel.scss** (\_skel.scss)

(Sass framework) A Sass-based implementation of Skel. Merges certain aspects of Skel and its Layout module (while adding some handy new mixins). Designed to work independently **or** in conjunction with Skel for added effect.

## License

Skel, Layout, Viewport, and Skel.scss are released under the MIT license.

Copyright (c) n33

1. **Backgroundsize.min.htc:**

It uses the elegance of background-size: cover; and background-size: contain;. The first one, for instance, allows an image to completely cover a background, **without** having to send a 1920x1080 background image down the pipes.

1. **Sass:**

**Sass** (**S**yntactically **A**wesome **S**tyle**s**heets) is a [style sheet language](https://en.wikipedia.org/wiki/Style_sheet_language) initially designed by [Hampton Catlin](https://en.wikipedia.org/wiki/Hampton_Catlin) and developed by [Natalie Weizenbaum](https://en.wikipedia.org/w/index.php?title=Natalie_Weizenbaum&action=edit&redlink=1). After its initial versions, Weizenbaum and [Chris Eppstein](https://en.wikipedia.org/w/index.php?title=Chris_Eppstein&action=edit&redlink=1) continued to extend Sass with SassScript, a simple scripting language used in Sass files.

Sass is a [scripting language](https://en.wikipedia.org/wiki/Scripting_language) that is [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) into [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS). SassScript is the scripting language itself. Sass consists of two [syntaxes](https://en.wikipedia.org/wiki/Syntax_(programming_languages)). The original syntax, called "the indented syntax", uses a syntax similar to [Haml](https://en.wikipedia.org/wiki/Haml). It uses [indentation](https://en.wikipedia.org/wiki/Indent_style) to separate [code blocks](https://en.wikipedia.org/wiki/Block_(programming)) and [newline](https://en.wikipedia.org/wiki/Newline) characters to separate rules. The newer syntax, "SCSS", uses block formatting like that of CSS. It uses braces to denote code blocks and semicolons to separate lines within a block. The indented syntax and SCSS files are traditionally given the [extensions](https://en.wikipedia.org/wiki/Filename_extension) .sass and .scss, respectively.

* 1. **Spring 3.0 MVC:**

A model-view-controller framework for Java web application made to simplify the writing and testing of Java web applications Fully integrates with the Spring dependency injection (Inversion of Control) framework Open Source Developed and maintained by Interface21, recently purchased by VMWare.

**4.3.1 Spring MVC Workflow:**

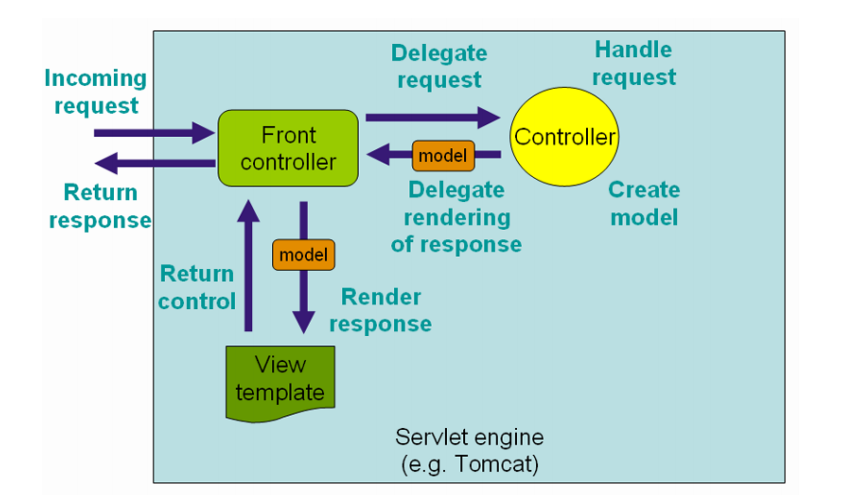
Spring MVC is the web component of Spring’s framework.

**M**odel-The data required for the request.

**V**iew-Displays the page using the model.

**C**ontroller-Handles the request,generates the model.

* Used to handle all incoming requests and route them through Spring
* Uses customizable logic to determine which controllers should handle which requests
* Forwards all responses through view handlers to determine the correct views to route responses
* Exposes all beans defined in Spring to controllers for dependency injection.



* 1. **Apache Maven:**

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

Maven is a project management and comprehension tool. Maven provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

In case of multiple development teams environment, Maven can set-up the way to work as per standards in a very short time. As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

Maven provides developers ways to manage following:

* Builds
* Documentation
* Reporting
* Dependencies
* SCMs
* Releases
* Distribution
* Mailing list

To summarize, Maven simplifies and standardizes the project build process. It handles compilation, distribution, documentation, team collaboration and other tasks seamlessly. Maven increases reusability and takes care of most of build related tasks.

**4.4.1 Maven Phases:**

Although hardly a comprehensive list, these are the most common *default* lifecycle phases executed.

* **validate**: validate the project is correct and all necessary information is available
* **compile**: compile the source code of the project
* **test**: test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
* **package**: take the compiled code and package it in its distributable format, such as a JAR.
* **integration-test**: process and deploy the package if necessary into an environment where integration tests can be run
* **verify**: run any checks to verify the package is valid and meets quality criteria
* **install**: install the package into the local repository, for use as a dependency in other projects locally
* **deploy**: done in an integration or release environment, copies the final package to the remote repository for sharing with other developers and projects.

There are two other Maven lifecycles of note beyond the *default* list above.

They are:

* **clean**: cleans up artifacts created by prior builds
* **site**: generates site documentation for this project.
  1. **Tomcat 7.x:**

**Apache Tomcat**, often referred to as **Tomcat**, is an open-source web server developed by the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation) (ASF). Tomcat implements several [Java EE](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) specifications including [Java Servlet](https://en.wikipedia.org/wiki/Java_Servlet), [JavaServer Pages](https://en.wikipedia.org/wiki/JavaServer_Pages) (JSP), [Java EL](https://en.wikipedia.org/wiki/Unified_Expression_Language), and [Web Socket](https://en.wikipedia.org/wiki/WebSocket), and provides a "pure[Java](https://en.wikipedia.org/wiki/Java_(programming_language))"  [HTTP](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) [web server](https://en.wikipedia.org/wiki/Web_server) environment for [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) code to run in.

Tomcat is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the [Apache License](https://en.wikipedia.org/wiki/Apache_License) 2.0 license, and is [open-source software](https://en.wikipedia.org/wiki/Open-source_software).

**4.5.1 Components:**

Tomcat 4.x was released with Catalina (a servlet container), Coyote (an HTTP connector) and Jasper (a [JSP engine](https://en.wikipedia.org/wiki/JSP_engine)).

**Catalina**

Catalina is Tomcat's [servlet container](https://en.wikipedia.org/wiki/Web_container). Catalina implements [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems)' specifications for [servlet](https://en.wikipedia.org/wiki/Java_servlet) and [JavaServer Pages](https://en.wikipedia.org/wiki/JavaServer_Pages) (JSP). In Tomcat, a Realm element represents a "database" of usernames, passwords, and roles (similar to Unix groups) assigned to those users. Different implementations of Realm allow Catalina to be integrated into environments where such authentication information is already being created and maintained, and then use that information to implement Container Managed Security as described in the Servlet Specification

**Coyote**

Coyote is a Connector component for Tomcat that supports the HTTP 1.1 protocol as a web server. This allows Catalina, nominally a Java Servlet or JSP container, to also act as a plain web server that serves local files as HTTP documents.

Coyote listens for incoming connections to the server on a specific [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol) port and forwards the request to the Tomcat Engine to process the request and send back a response to the requesting client. Another Coyote Connector, Coyote JK, listens similarly but instead forwards its requests to another web server, such as Apache, using the [JK protocol](https://en.wikipedia.org/wiki/Apache_JServ_Protocol).This usually offers better performance.

**Jasper**

Jasper is Tomcat's JSP Engine. Jasper parses [JSP files](https://en.wikipedia.org/wiki/JSP_files) to compile them into Java code as servlets (that can be handled by Catalina). At runtime, Jasper detects changes to JSP files and recompiles them.

As of version 5, Tomcat uses Jasper 2, which is an implementation of the [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems)'s [JSP](https://en.wikipedia.org/wiki/JavaServer_Pages) 2.0 specification. From Jasper to Jasper 2, important features were added:

* JSP Tag library pooling - Each tag markup in JSP file is handled by a tag handler class. Tag handler class objects can be pooled and reused in the whole JSP servlet.
* Background JSP compilation - While recompiling modified JSP Java code, the older version is still available for server requests. The older JSP servlet is deleted once the new JSP servlet has finished being recompiled.
* Recompile JSP when included page changes - Pages can be inserted and included into a JSP at runtime. The JSP will not only be recompiled with JSP file changes but also with included page changes.
* JDT Java compiler - Jasper 2 can use the Eclipse JDT (Java Development Tools) Java compiler instead of [Ant](https://en.wikipedia.org/wiki/Apache_Ant) and [javac](https://en.wikipedia.org/wiki/Javac).

Three new components were added with the release of Tomcat 7:

**Cluster**

This component has been added to manage large applications. It is used for [load balancing](https://en.wikipedia.org/wiki/Load_balancing_(computing)) that can be achieved through many techniques. Clustering support currently requires the JDK version 1.5 or later.

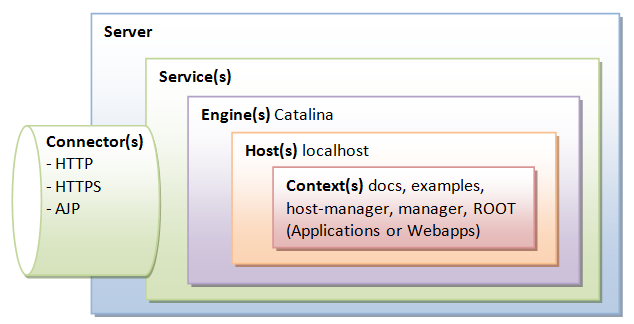
**High availability**

A high-availability feature has been added to facilitate the scheduling of system upgrades (e.g. new releases, change requests) without affecting the live environment. This is done by dispatching live traffic requests to a temporary server on a different port while the main server is upgraded on the main port. It is very useful in handling user requests on high-traffic web applications.

**Web application**

It has also added user- as well as system-based web applications enhancement to add support for deployment across the variety of environments. It also tries to manage sessions as well as applications across the network.

Tomcat is building additional components. A number of additional components may be used with Apache Tomcat. These components may be built by users should they need them or they can be downloaded from one of the mirrors.



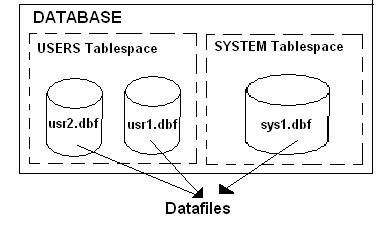
* 1. **ORACLE:**

Oracle is an Object-Relational Database Management System. It is the leading RDBMS vendor worldwide. Nearly half of RDBMS worldwide market is owned by Oracle.

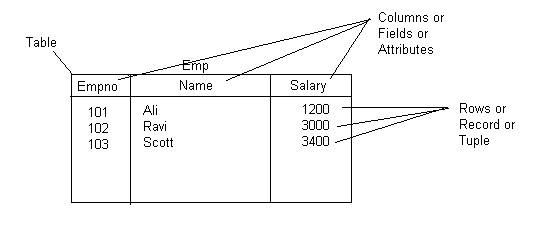
## 4.6.1 ORACLE DATABASE

Every Oracle Database Contains Logical and Physical Structures. Logical Structures are table spaces, Schema objects, extents and segments. Physical Structures are Data files, Redo Log Files, Control File.

A database is divided into logical storage units called table spaces, which group related logical structures together. Each Table space in turn consists of one are more data files.



In relational database system all the information is stored in form of tables.  A table consists of rows and columns



All the tables and other objects in Oracle are stored in table space logically, but physically they are stored in the data files associated with the table space.

Every Oracle database has a set of two or more redo log files. The set of redo log files for a database is collectively known as the database's redo log. A redo log is made up of redo entries (also called redo records).

The primary function of the redo log is to record all changes made to data. If a failure prevents modified data from being permanently written to the data files, the changes can be obtained from the redo log so work is never lost.

Every Oracle database has a control file. A control file contains the database name and locations of all data files and redo log files.

Every Oracle database also has a Parameter File. Parameter file contains the name of the Database, Memory Settings and Location of Control file.

**CHAPTER 6**

**IMPLEMENTATION**

**6.1 Working Principle:**

The working principle of the Corporate Trust Project Management System involves the software components of JSP(Java Server Pages),Spring 3.0 MVC and Oracle Database.

JSP has been used to design the pages which involved in developing the User Interface. It is used to create the web pages dynamically. JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, such as HTML, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java bytecode rather than [machine code](https://en.wikipedia.org/wiki/Machine_code). Like any other Java program, they must be executed within a [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine) (JVM) that interacts with the server's host [operating system](https://en.wikipedia.org/wiki/Operating_system) to provide an abstract, platform-neutral environment.

Spring is a light weight framework for URL mapping at backend and to handle the data. Further it is easy for database insertion, deletion, updation and retrieval of values using the JDBC template of Spring. Further it is pure Java and requires less or no use of external xml files for URL mapping.

Oracle Database Stores the entire details that are obtained from the admin to create the information repository. Whenever there is a need for the stored data, the data will be retrieved from the databases in the cases of requests from both admin as well as the user.

The spring framework follows Model View Controller pattern. The requests are received and handled by the controller. The view decides which output to render based upon the data and finally the model is the format of the final output.

**6.2 Working Modules:**

**Home Page:**

The Corporate Trust Project Management’s home page has been provided with the login and signup buttons. The employees can login by any of the two ways, one as the admin and the other as the user. The admin is provided with the rights of creating new entries in the database as well as updating the existing details in the database. But the user is restricted only with the rights of viewing the contents of the database. The user is not provided with the superior rights of manipulating the database.

The admin or the normal user has to login into the system with a user id and a password. If the id and the password match, the control over the system moves on to the next page. Incase if the id and the password doesn’t match then the control is again redirected to the login page.

**Support Group Form:**

After successful login of admin, the admin has access to several of the available options. The Support Group Form lets the administrator input the Group Name, Application Owner for the project and Group description which contains the details of the members. This information is saved in the database using the JDBC template.

**Resource Entry Form:**

Then at the resource entry, the details pertaining to the users entry to the group such as the Date of join to company and group and leaving date from group and then the technology he has expertise in, role of the user, his area of interests and the experience summary is input by the admin for storage in the database and there is an option for reset in case of wrong entry.

**Application Entry Form:**

Next the admin goes to the application entry form where he assigns roles to the users who were previously added. Some of the roles include the Application Owner again, primary support and secondary support. The roles are selectable based on the entries pulled from the database which was previously got as input at the resource entry form using JDBC template of spring-mvc.

**Task Form:**

In the task form, we are going to input the nature of the task such as Task Type, Application Name, Task Summary, Priority, Status, e-mail and Task Number. Ask for an update provides a user with the latest progress of the task. Send reminder is used to put a mail to all the members of the group reminding them regarding the progress they had to achieve.

**Assignment Form:**

Assignment Form allows the admin to allocate a resource for a particular application by specifying the application name, resource name, task description, task number and start date of the application.

**Access Form:**

The access form allows the admin to allow or deny access to a resource person pertaining to a particular application.

**Task Summary Form:**

Now in the task summary form, both the users and admin, can search for the available task within a particular duration specified by pulling already supplied data from the database.

**Release Form:**

In the release form, the release details such as release name, application name, release date, responsibility, status of the application and the summary are specifiable by the admin.

**Resource Summary Form:**

The users and the admin can search for the details of the resource person involved in a particular application within a period of time. This searching process proceeds with the resource name, starting and ending date of the application.

**Release Summary Form:**

This is similar to resource summary form except that here we search for the different releases from various groups. This is obtained by release name, start date, end date and the e-mail id of the resource group.

**Reminder Set Up Form:**

The Reminder Form allows the employee to frame a particular format for a reminder to be sent to their team. The fields include reminder type, email group, repeat time, end date, e-mail subject and email content. This reminder is sent as an e-mail to the team.

**CHAPTER 7**

**TESTING**

**VERIFICATION:**

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words ,to make sure the product behaves the way we want it to .

**VALIDATION:**

Validation is the process to make sure that the product satisfies the specified requirements at the end of the development phase. In other words ,to make sure the product is built as per customer requirements.

**7.1 TYPES OF TESTING:**

There are many types of testing like :

* Unit Testing
* Integration Testing
* Functional Testing
* System Testing
* Stress Testing
* Performance Testing
* Usability Testing
* Acceptance Testing
* Regression Testing
* Beta Testing
* Black Box Testing
* White Box Testing

**UNIT TESTING:**

Unit testing is the testing of an individual unit of group of related units. It falls under the class of white box testing .It is often done by the programmer to test that the unit he/she has implementation is producing the expected output against given input.

**INTEGRATION TESTING:**

Integration testing is the testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box and black box testing.

**FUNCTIONAL TESTING:**

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of blackbox testing.

**SYSTEM TESTING:**

System testing is the testing to ensure that by putting the software in different environment it still works. System testing is done with full system implementation and environment**.** It falls under the class of black box testing.

**STRESS TESTING:**

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

**PERFORMANCE TESTING:**

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating result within a specified time as in the performance requirements. It falls under the class of black box testing .

**USABILITY TESTING:**

Usability testing is performed to the perspective of the client ,to evaluate how GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing

**ACCEPTANCE TESTING:**

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirement and works as the customer expected. It falls under the class of black box testing.

**REGRESSION TESTING:**

Regression testing is the testing after the modification of a system, component, or group of related units to ensure that the modification is working correctly and not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

**BETA TESTING:**

Beta testing is the testing which is done by end-users, a team outside development or publicly releasing full pre-versions of the product which is known as beta version. The aim of beta testing is to cover unexpected errors. It falls under the class of black box testing.

**BLACKBOX TESTING:**

Black box testing is a testing technique that ignores the internal mechanisms of the system and focuses on the input and the output generated against any input and execution of the system. It is also called as functional testing.

**WHITEBOX TESTING:**

White box testing is a testing technique that takes into account the internal mechanisms of a system. It is also called as structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

**CHAPTER 8**

**CONCLUSION AND FUTURE ENHANCEMENT**

Programs must continuously improve the team based performance assessment process to make it more efficient and valuable to current and future clients. The key is not only “lesson learnt” but more importantly “lessons to be used”. When metrics cease to be useful, when comparison data is no long applicable, or when alignment is broken due to changes and policy, adjustments must be made in timely manner.

A data driven decision making nonprofit uses its evaluation efforts to guide the organization, with an eye toward what works and the effective and efficient use of funds. Implementing a team based performance management process is one way to create an environment of data driven decision making within an organization.

Benefits of this approach include an engaged staff and an innovative and responsive organization.

Creating an application that recognizes individuals is a very powerful tool to motivate and improve the performance of staff members. There are a number of ways to recognize employees that are just effective other than monetary rewards.

The goal of recognition should be achieved by this application through different ideas and approaches possible.

**REFERENCES**

1. <https://en.wikipedia.org/wiki/Management_information_system>
2. A performance Management Model for Team-Based Settings CEO Publication G94-20 (263); Allan M.Mohram, Jr and Susan A.Mohram, 1994.
3. <http://www.firstinspires.org/resource-library/ftc/information-management-systems-and-user-support>
4. <http://www.firstroboticscanada.org/main/frc-team/>
5. Strategic Performance Appraisal in team-based organization: one size does not fit all; Academy of Management Executive, 2001 Vol. 15 No.2