Università Degli Studi di Trento

Employee/Project Tracking Application

Web architecture

Academic year

2013-2014

By
Joseph Brian Musanje Kasozi
166047

Submitted to
Professor Marco Ronchetti

Contents

1.0 Introduction

Web architecture is a course offered to establish an understanding of the way the web and web application are designed and how they work. As a course project to demonstrate this understanding, I developed a web based application to monitor and track employee progress on given projects. This application utilizes the Java Enterprises Edition architecture. Its split into three tiers as the J2EE standards that's, Web tier referred to as etsweb, Business logic referred to as etscore and the persistence tier backed up by the mysql database.

1.2 Project Description

Employee tracing system (ets) is a web based application that can be used to monitor progress and status of employees on given projects. The project is mainly developed to facilitate the work at my own startup.

1.3 Projects Goals

To demonstrate the three tier J2EE architecture

To demonstrate the usefulness of Java persistence API

To develop an application that can be used to track employee progress

2.0 Requirements Analysis

The application is designed to monitor and track employee's progress on the different projects to which they are assigned. This is done by the system administrator who adds and manages projects and the employees. Different Employees are assigned different roles to which they can manage different projects running. The administrator is able to set the status of the project depending on the employee progress. On addition the administrator will manage all the employees in the system.

2.1 Actors

The system is accessed by five actors where each is allocated or performs different tasks and view.

Administrators

The administrator is the main actor of the system. He manages projects and employees; it's the role of the admin to add projects, employees, roles, departments to the system. Then after assign employees to given projects as well as setting the priorities of the projects. After or during the project up period the administrator updates the project status.

Team Leader

The team leader is responsible for a given team working on a given project but the team leader is different from a project manager as the team leader is assigned a a task of only monitoring a group to which he is assigned

Software Engineer

The software engineer is can be assigned to work on a given project under a supervision of a given team leader or project manager or both. The engineer it assigned a project to work on for a given period of time of which it can expire incase it's not finished in time.

Software trainee

This works hand in hand with the software engineer therefore always should be assigned a software engineer as the supervisor.

Project manager

This actor is assigned to manage and supervise projects, employees and teams.

2.2 Things or Entities

The entities and all the detailed attributes of each are in the attached html file "*Database.html"*

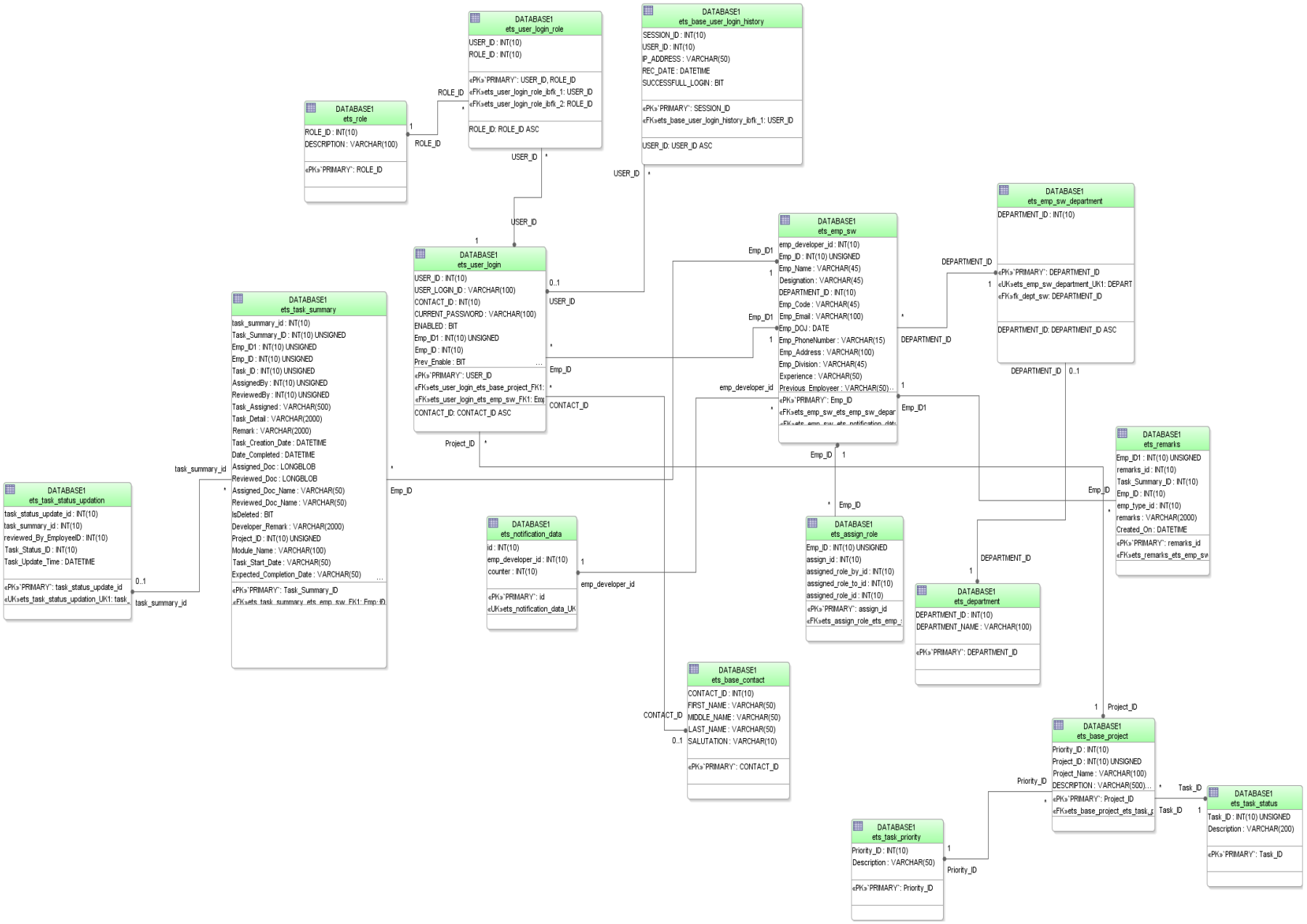
Design

The system was designed using mainly the Object Oriented deigns taking UML as the main language

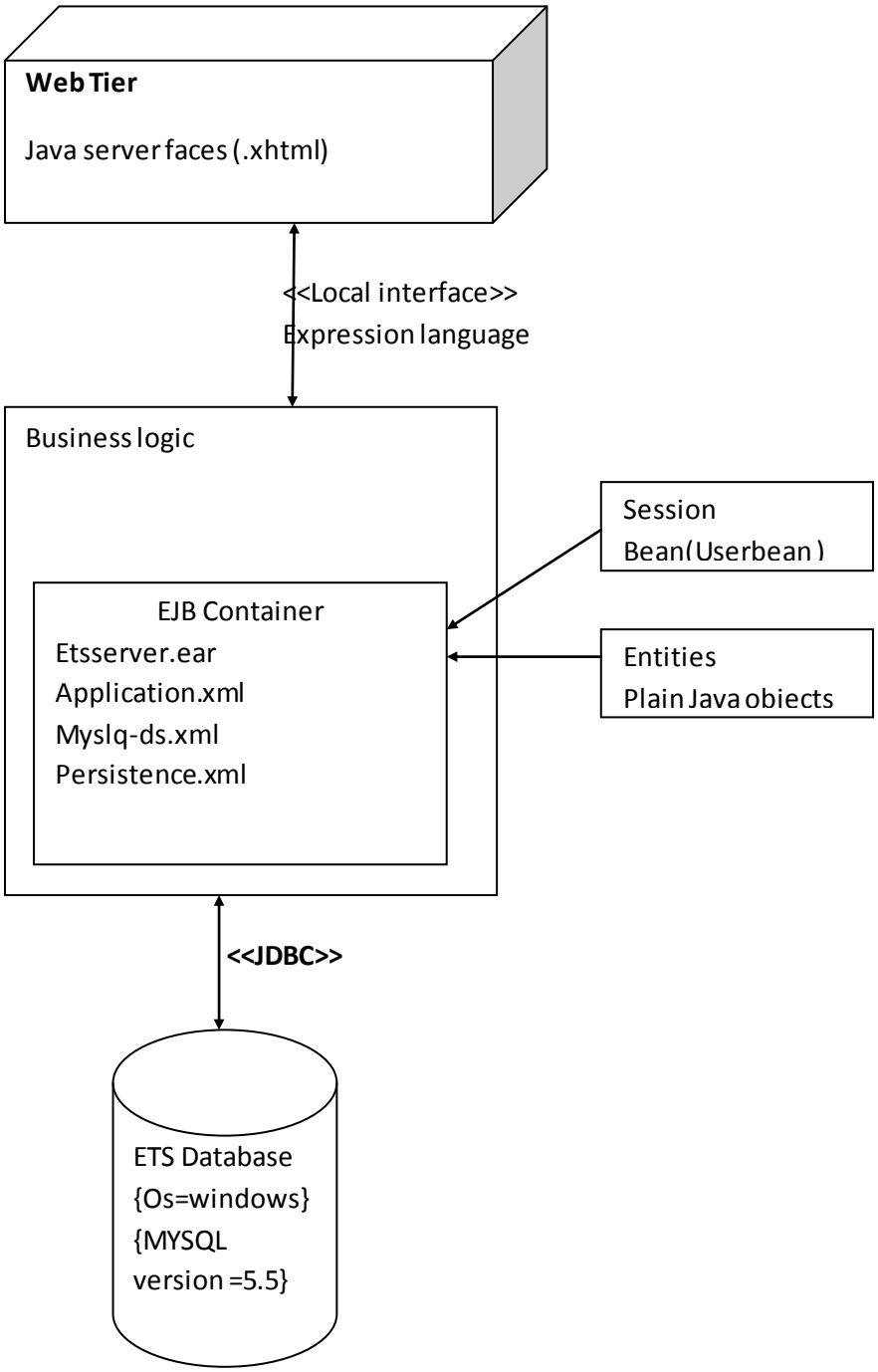
Entity Relation Diagram (ERD)

The ERD shows the things or data being stored in the database. It comprises of tables including ets_emp_sw table which stores all the required information about the employees, ets_Department table which stores all the information about the different departments, ets_base_project table, which stores all the information regarding the projects, ets_task_summary table show and stores all the summary about the projects and the employees working on them.

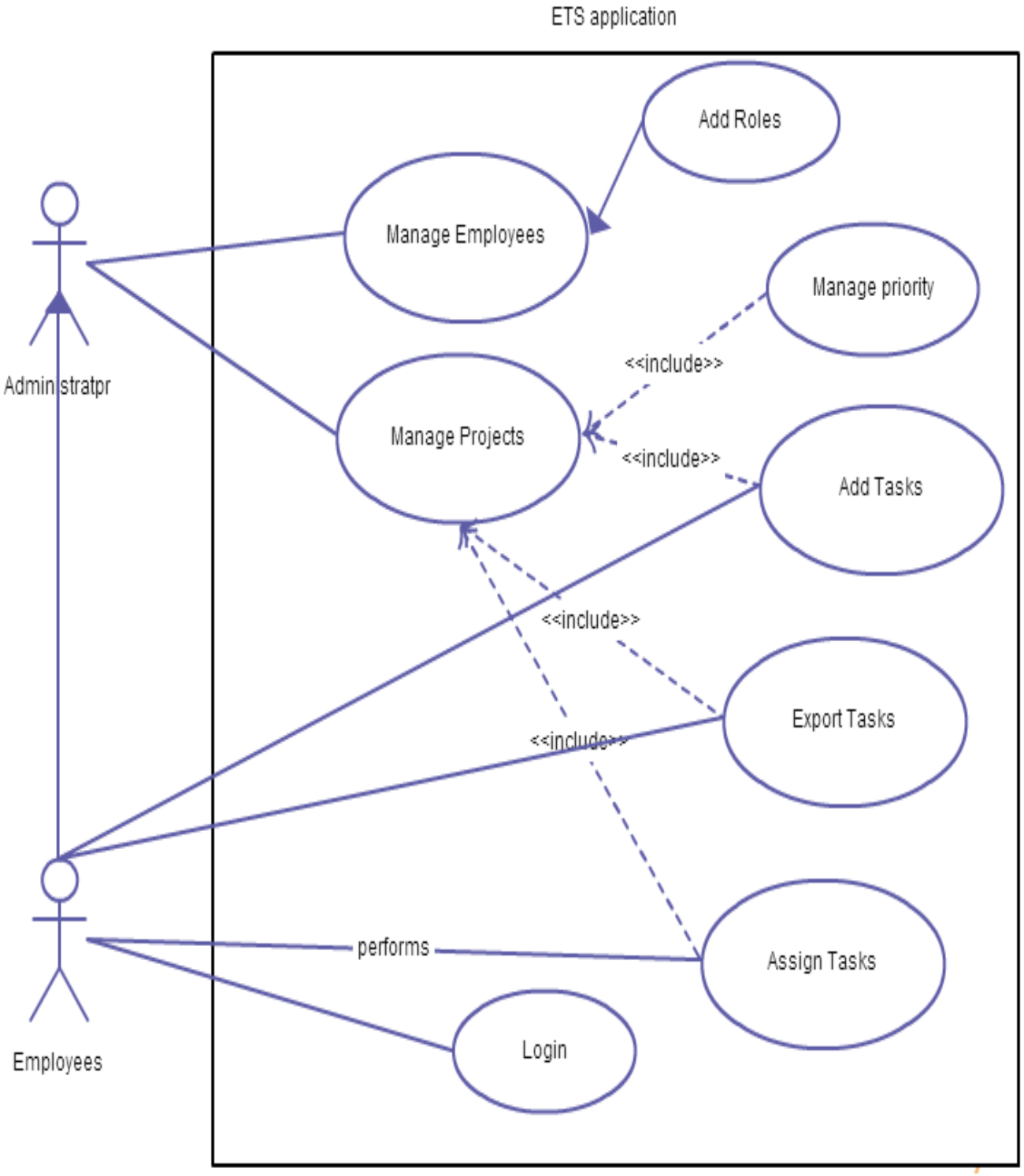
ets_task_status_updation table stores information about who reviewed and updated the project information. ets_assign_role table holds data showing the roles assigned to different users. Ets_user_login table stores information that can enable users to access the application. This includes the user ID, user name and password.


DATABASE1
ets_user_login_role
USER_ID : INT(10)
ROLE_ID : INT(10)
«PK»`PRIMARY`: USER_ID, ROLE_ID
«FK»ets_user_login_role_ibfk_1: USER_ID
«FK»ets_user_login_role_ibfk_2: ROLE_ID
ROLE_ID: ROLE_ID ASC
DATABASE1
ets_base_user_login_history
SESSION_ID : INT(10)
USER_ID : INT(10)
IP_ADDRESS : VARCHAR(50)
REC_DATE : DATETIME
SUCCESSFULL_LOGIN : BIT
«PK»`PRIMARY`: SESSION_ID
«FK»ets_base_user_login_history_ibfk_1: USER_ID
USER_ID: USER_ID ASC
DATABASE1
ets_role
ROLE_ID : INT(10)
DESCRIPTION : VARCHAR(100)
«PK»`PRIMARY`: ROLE_ID
ROLE_ID
ROLE_ID
USER_ID
USER_ID
USER_ID
USER_ID
DATABASE1
ets_user_login
USER_ID : INT(10)
USER_LOGIN_ID : VARCHAR(100)
CONTACT_ID : INT(10)
CURRENT_PASSWORD : VARCHAR(100)
ENABLED : BIT
Emp_ID1 : INT(10) UNSIGNED
Emp_ID : INT(10)
Prev_Enable : BIT
«PK»`PRIMARY`: USER_ID
«FK»ets_user_login_ets_base_project_FK1:
«FK»ets_user_login_ets_emp_sw_FK1: Emp
CONTACT_ID: CONTACT_ID ASC
DATABASE1
ets_emp_sw
emp_developer_id : INT(10)
Emp_ID : INT(10) UNSIGNED
Emp_Name : VARCHAR(45)
Designation : VARCHAR(45)
DEPARTMENT_ID : INT(10)
Emp_Code : VARCHAR(45)
Emp_Email : VARCHAR(100)
Emp_DOJ : DATE
Emp_PhoneNumber : VARCHAR(15)
Emp_Address : VARCHAR(100)
Emp_Division : VARCHAR(45)
Experience : VARCHAR(50)
Previous_Employeer : VARCHAR(50)...
«PK»`PRIMARY`: Emp_ID
«FK»ets_emp_sw_ets_emp_sw_depar
Emp_ID1
Emp_ID1
Emp_ID1
DEPARTMENT_ID
DEPARTMENT_ID
DATABASE1
ets_emp_sw_department
DEPARTMENT_ID : INT(10)
«PK»`PRIMARY`: DEPARTMENT_ID
«UK»ets_emp_sw_department_UK1: DEPART
«FK»fk_dept_sw: DEPARTMENT_ID
DEPARTMENT_ID: DEPARTMENT_ID ASC
DEPARTMENT_ID
DATABASE1
ets_task_summary
task_summary_id : INT(10)
Task_Summary_ID : INT(10) UNSIGNED
Emp_ID1 : INT(10) UNSIGNED
Emp_ID : INT(10) UNSIGNED
Task_ID : INT(10) UNSIGNED
AssignedBy : INT(10) UNSIGNED
ReviewedBy : INT(10) UNSIGNED
Task_Assigned : VARCHAR(500)
Task_Detail : VARCHAR(2000)
Remark : VARCHAR(2000)
Task_Creation_Date : DATETIME
Date_Completed : DATETIME
Assigned_Doc : LONGBLOB
Reviewed_Doc : LONGBLOB
Assigned_Doc_Name : VARCHAR(50)
Reviewed_Doc_Name : VARCHAR(50)
IsDeleted : BIT
Developer_Remark : VARCHAR(2000)
Project_ID : INT(10) UNSIGNED
Module_Name : VARCHAR(100)
Task_Start_Date : VARCHAR(50)
Expected_Completion_Date : VARCHAR(50)
«PK»`PRIMARY`: Task_Summary_ID
«FK»ets_task_summary_ets_emp_sw_FK1: Emp_ID
Project_ID
Emp_ID
CONTACT_ID
emp_developer_id
Emp_ID
Emp_ID
Emp_ID
Emp_ID
DATABASE1
ets_remarks
Emp_ID1 : INT(10) UNSIGNED
remarks_id : INT(10)
Task_Summary_ID : INT(10)
Emp_ID : INT(10)
emp_type_id : INT(10)
remarks : VARCHAR(2000)
Created_On : DATETIME
«PK»`PRIMARY`: remarks_id
«FK»ets_remarks_ets_emp_sw
task_summary_id
task_summary_id
DATABASE1
ets_task_status_updation
task_status_update_id : INT(10)
task_summary_id : INT(10)
reviewed_By_EmployeeID : INT(10)
Task_Status_ID : INT(10)
Task_Update_Time : DATETIME
«PK»`PRIMARY`: task_status_update_id
«UK»ets_task_status_updation_UK1: task_
DATABASE1
ets_notification_data
id : INT(10)
emp_developer_id : INT(10)
counter : INT(10)
«PK»`PRIMARY`: id
«UK»ets_notification_data_UK
emp_developer_id
DATABASE1
ets_assign_role
Emp_ID : INT(10) UNSIGNED
assign_id : INT(10)
assigned_role_by_id : INT(10)
assigned_role_to_id : INT(10)
assigned_role_id : INT(10)
«PK»`PRIMARY`: assign_id
«FK»ets_assign_role_ets_emp_
DEPARTMENT_ID
DATABASE1
ets_department
DEPARTMENT_ID : INT(10)
DEPARTMENT_NAME : VARCHAR(100)
«PK»`PRIMARY`: DEPARTMENT_ID
DATABASE1
ets_base_contact
CONTACT_ID : INT(10)
FIRST_NAME : VARCHAR(50)
MIDDLE_NAME : VARCHAR(50)
LAST_NAME : VARCHAR(50)
SALUTATION : VARCHAR(10)
«PK»`PRIMARY`: CONTACT_ID
CONTACT_ID
Project_ID
DATABASE1
ets_base_project
Priority_ID : INT(10)
Project_ID : INT(10) UNSIGNED
Project_Name : VARCHAR(100)
DESCRIPTION : VARCHAR(500)...
«PK»`PRIMARY`: Project_ID
«FK»ets_base_project_ets_task_
Priority_ID
Task_ID
Task_ID
DATABASE1
ets_task_status
Task_ID : INT(10) UNSIGNED
Description : VARCHAR(200)
«PK»`PRIMARY`: Task_ID
DATABASE1
ets_task_priority
Priority_ID : INT(10)
Description : VARCHAR(50)
«PK»`PRIMARY`: Priority_ID
Priority_ID

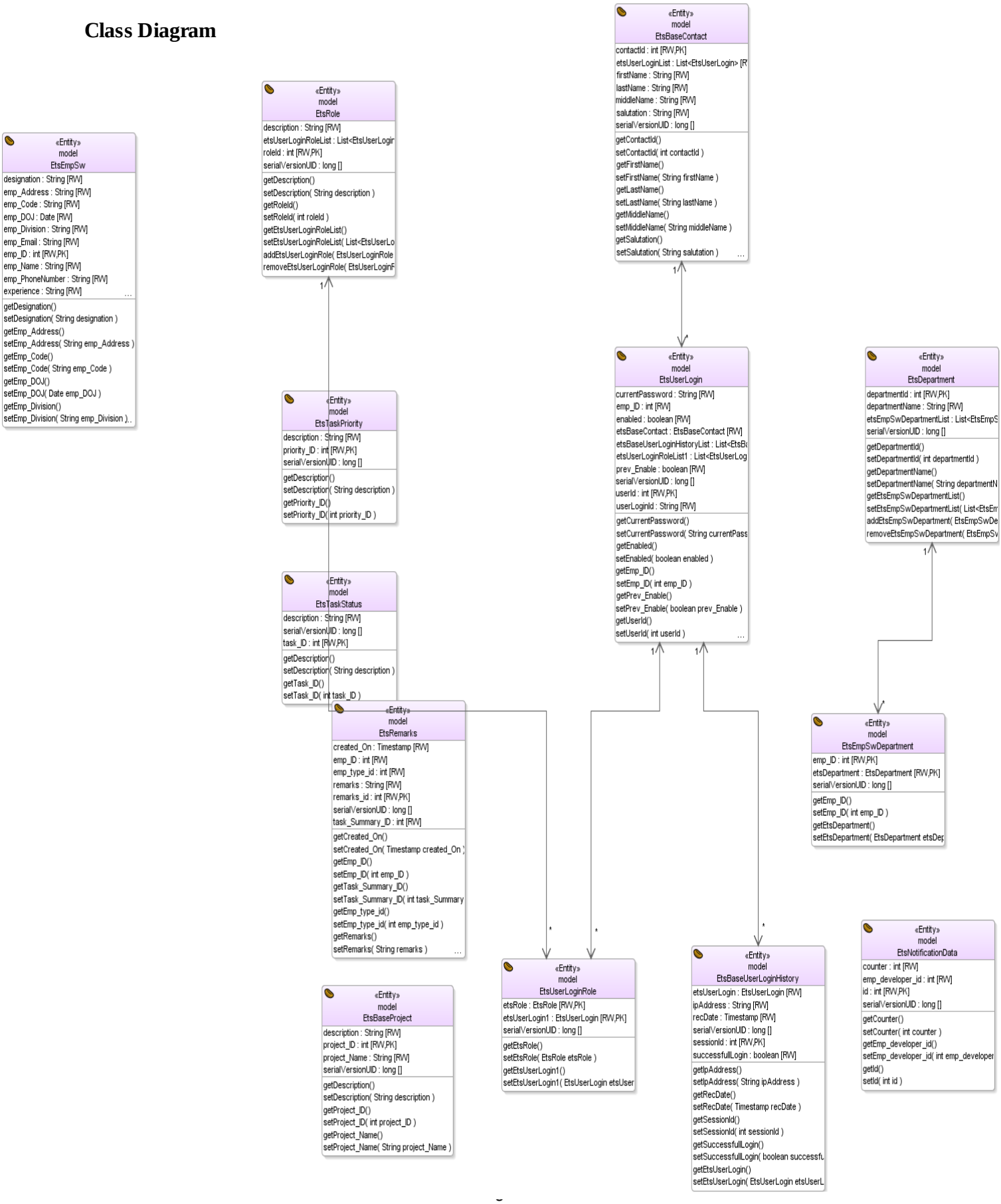
Deployment of Three Tiers



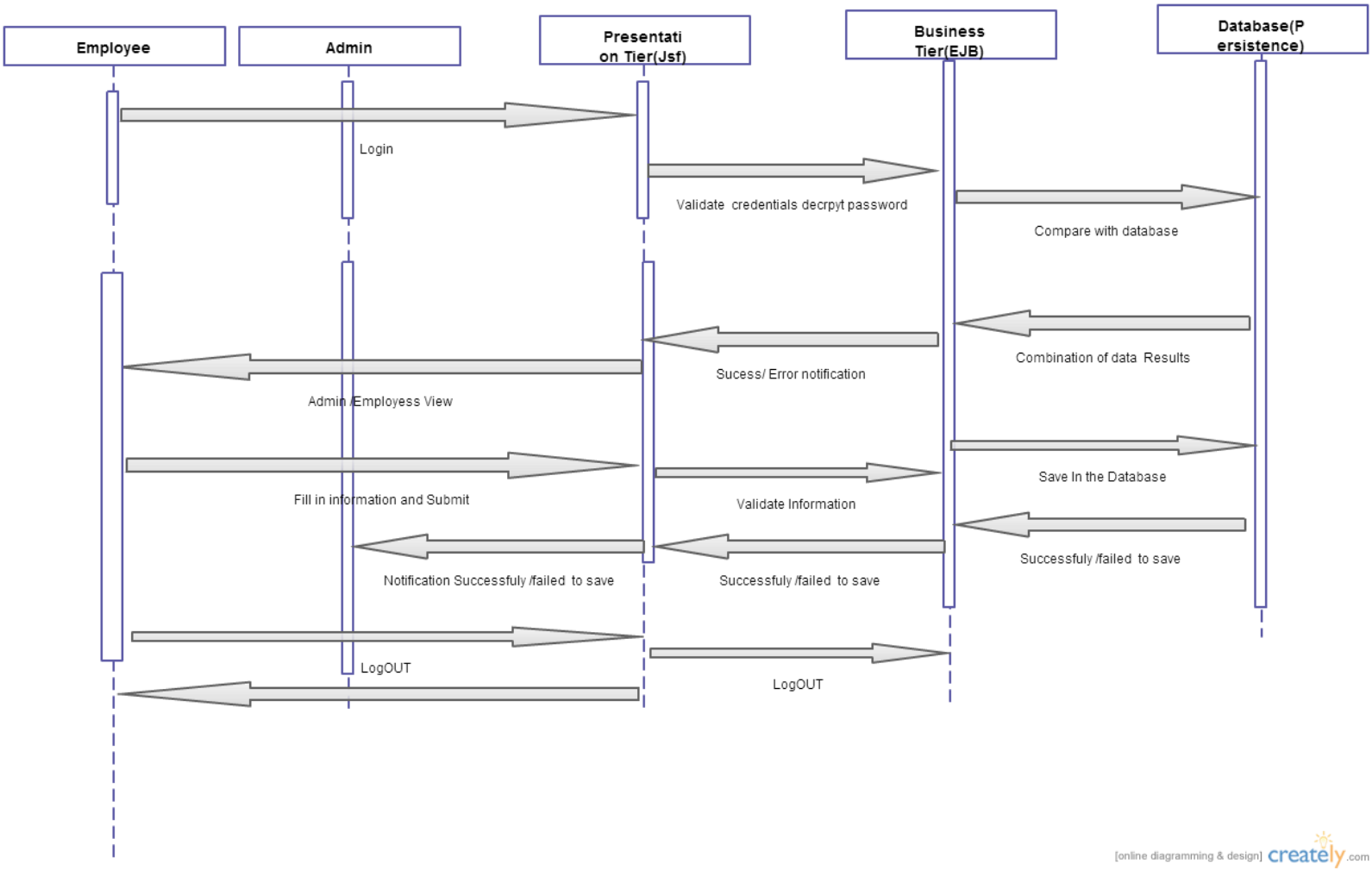
Use Case Diagram



Class Diagram



Sequence Diagram



Deployment of three tiers

The application consists of three tiers i.e. presentation, business logic and the persistence. The persistence tier is composed of a database running on mysql server 5.5 on the localhost .This tier is separate from the business logic tier but they are connected by the Jdbc and persistence API. The business logic is composed of entity classes, session beans and Data access objects which are compressed into the Ear file before deployment and the configurations in the persistence.xml, mysql-ds.xml and application .xml. The business logic is connecting to the presentation through a local interface and with help of expression language which is used by the java server faces to

insert and retrieve data from the business tier.The presentation tier is made up of java server faces (.XHTML) files with the use of facelets (Myfaces Library) .This gives the client or user the view of the system. The user will only interact with the interface provided by presentation tier.

Use case Diagram

The system has mainly two actors that are administrator and employees. Later they are sub-divided into team leader, internees, software developer etc, the administrator has can do all the tasks performed by all other employees that's why in the diagram the actor employee is generalized to administrator. Employee can login to the application, add tasks, export tasks, assign task while the administrator is able to manage employee and manage projects on addition to the tasks performed by the employee.

The manage employees task included add role. This is because one of the main functions while managing employees is to add the roles therefore its included. On addition while managing projects a number of functions are needed to be included and they are all shown in the use case diagram.

3.0 Architectural considerations

Organization of the deployment on the various (logical) tiers

I used a 3-tier architecture based on open standards that's Java EE, Java Persistence API and Java Server faces.

Persistence

Business

Presentation

The three tiers possibly communicate as shown below

Persistence <-> Business <-> Presentation

The presentation layer never performs persistence operations; it always does it through the business layer. This architecture is meant to fulfill the demands of a high availability web application

Persistence

Performs create, read, update and delete persistence operations. In this case we are using (Java Persistence API) JPA and we currently use Entities with EJB

This layer is divided into multiple classes, where each class deals with a certain type of entities that's entities related to a project or employee might get handled by a single persistence class and is used by one and only one manager in addition this layer also stores JPA entities .

Business

All logic which is tied to the web application functionality is located in this layer. This functionality

Of assigning new projects monitoring projects and employees, adding and users to the system all are implemented in this layer.

This layer has class and this class is annotated with **@stateless** to become a Stateless Session Bean (SLSB).

Presentation

The presentation layer is in charge of presentation. It's responsible for the user interface and shows information to the user by building HTML pages and receiving user input.

The layer calls methods from the business layer to perform operations requested by the user and to receive information to show in the web page. Sometimes the information received from the business layer is less complex types as String's and integers eg. Employee names and numbers, and at other times JPA entities.

With the presentation layer, i used expression language and java server faces (xhtml) to call methods and link the presentation with the business logic tier.

4.0 Implementation and Deployment

4.1 State Management

Stateless Session Beans (Userbean)

A stateless session bean does not maintain a conversational state with the client. When a client invokes the methods of a stateless bean, the bean's instance variables may contain a state specific to that client but only for the duration of the invocation. When the method is finished, the client-specific state should not be retained. This bean was annotated stateless because all instances of a stateless bean are equivalent, allowing the EJB container to assign an instance to any client and thus making it possible to support multiple clients and scalability of the application since it will have multiple clients accessing it at the same time.

Local interface (user local)

The userbean implements the three methods in the userbean. That's

public List getAllObjectFrom(String tableName) throws BaajaException;

public String passwordEncrypt(String password) throws BaajaException;

public ArrayList getResultsetList(String customQuery) throws BaajaException;

The application uses other managed beans as configured in the faces.congif.xml file. On addition there helper classes used which include; BaajaException , BaajaClassNotFoundException, validate etc.

4.2 Transaction Behavior (JTA)

The Java transaction API is used to perform transaction which is managed by the container. An entity manager is class is used to query and manage all the persistence. This is set in the userbean which is also a stateless.

Transaction scoped persistence context

The entity manager is invoked in the active JTA transaction in a managed environment and a new persistence is created. This persistence context will only close if all the entities that were managed are detached and the JTA transaction has completed [*Mastering EJB4 edition].* This was used because the application uses a stateless bean but incase it had used statefull where conversation are extended I would rather use an extended persistence context where entities are not detached even after the JTA closes. This would help not to create persistence context each time a transaction is committed.

4.3 Technologies Used

Java 2 Enterprise Edition (J2EE)

Java 2 platform, Enterprise Edition is a platform that enables solutions for developing, deploying and managing distributed applications. It also provides component-based, server-centric multi-tier application architecture.

Enterprise Java Beans 3.0

Enterprise Java APIs provided programming interfaces for some middleware implementations, like naming, security, transactions, messaging and database. I developed Entity classes, servlets and data access objects

Database

MYSQL database is used for the project. The database ets was created prior the project implementation. This helped in figuring out exactly what entity classes where need and those that were not required.

Rich Faces

With the JSF technology combined with the rich faces library and css. They provide the user interface for the application. These enable the user not to directly interact with the complex server side of the application or database.

Deployment Considerations

Set the mysql–ds file in the Jboss deploy folders and set the user name and password for the database. For the default application the database has user root and no password as shown in the mysql-ds.xml.

Set the workspace for the project (Eclipse) to the workspace provided in the source code (folder eclipse).This helps in setting up the libraries used by application

Set the project runtime environment to Jboss runtime, jsf 2, and jre 1.6

5.0 Comments

The use of transaction is a challenge up to now but I tried to implement them using the container so I had little to work with them in the course of the project. But this being an ongoing work I hope to improve it and do better adjustments.

Although we had not talked about Data access objects and Jdbc I was able to implement them in the project not as additional technologies but they helped in the persistence on addition to the entities