CS673F13 Software Engineering
Group Project 4 - Project Life Cycle Management
Software Design Document

Team Member	Role(s)	Signature	Date
Christian	Configuration Leader	Christian Heckendorf	11/10/13
Vipul Agarwal	Environment and Integration Leader		
Vipul Kumar	Requirement Leader		
Rachit,Yuvaraj	Design Leader		
Alan,Manav	Implementation Leader		
Tandhy	QA Leader		

Revision history

Version	Author	Date	Change
1.0	Tandhy	11/04/13	Initial Draft
1.1	Christian	11/05/13	Added Patterns and Algorithms
1.2	Vipul	11/06/13	Class and Algorithm Description
1.3	Tandhy	11/06/13	Added Flowchart in algorithm

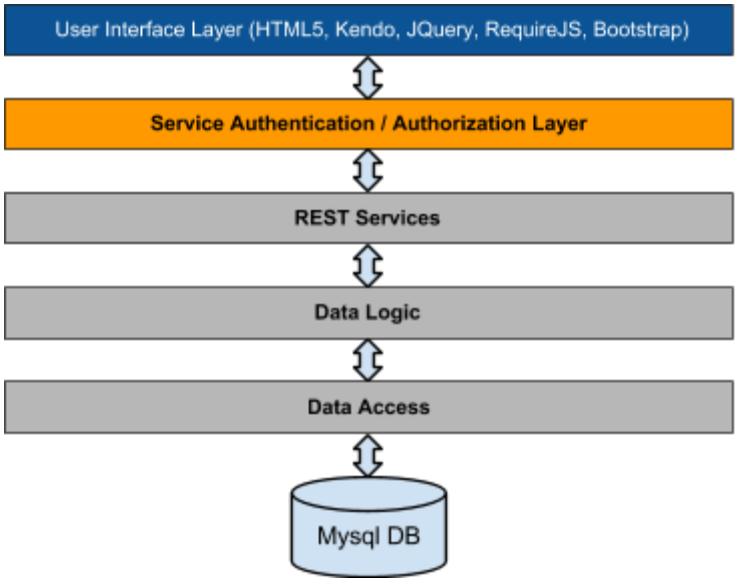
Introduction

In this section, give an overview of this document, and also address the design goals of your software system.

Software Architecture

In this section, you will describe the decomposition of your software system, which include each component (which may be in terms of package or folder) and the relationship between components. You shall have a diagram to show the whole architecture, and class diagram for each component.
The interface of each component and dependency between components should also be described.
If any framework is used, it shall be defined here too.
Database design should also be described if used.
Project Lifecycle Management is build using MVVM (Model - View - View Model) pattern, which similar to MVC (Model - View - Controller) pattern, where View Model provide the same responsibility with Controller.

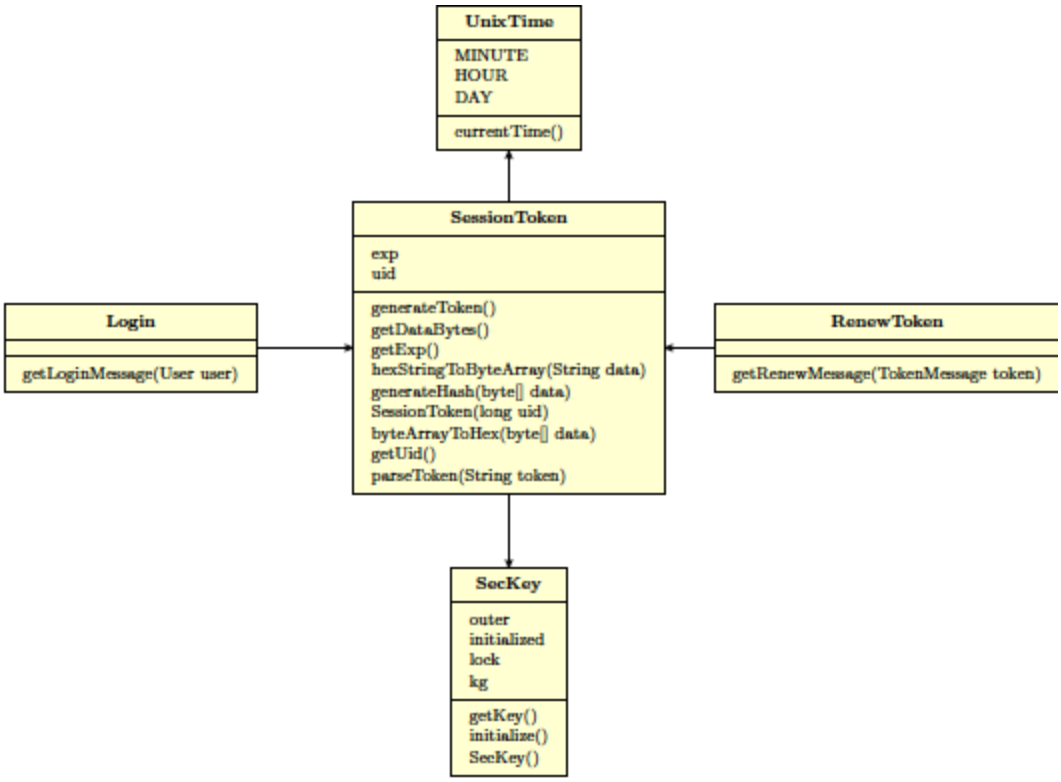
The following is PLM architecture :



[General Description about PLM architecture]
Service Authentication Layer provides service for security purpose, it checks all requests from user interface and in order to pass the layer, each request will embedded with generated key(token). Once authentication is granted, requests will go to Service.

[Description about our framework]

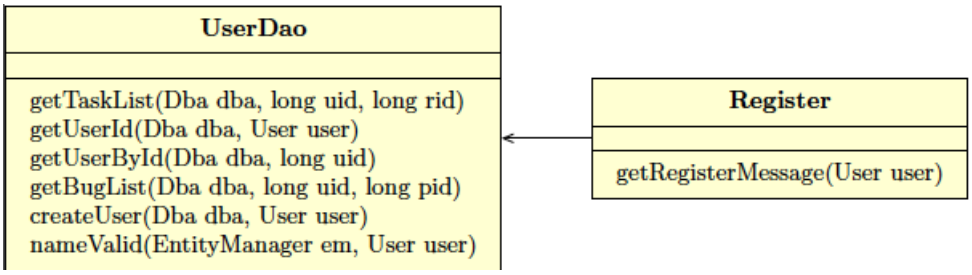
Class Diagram
The following diagrams are classes for each Interface and it's dependency to other objects :
LOGIN



The login system of our PLM application :

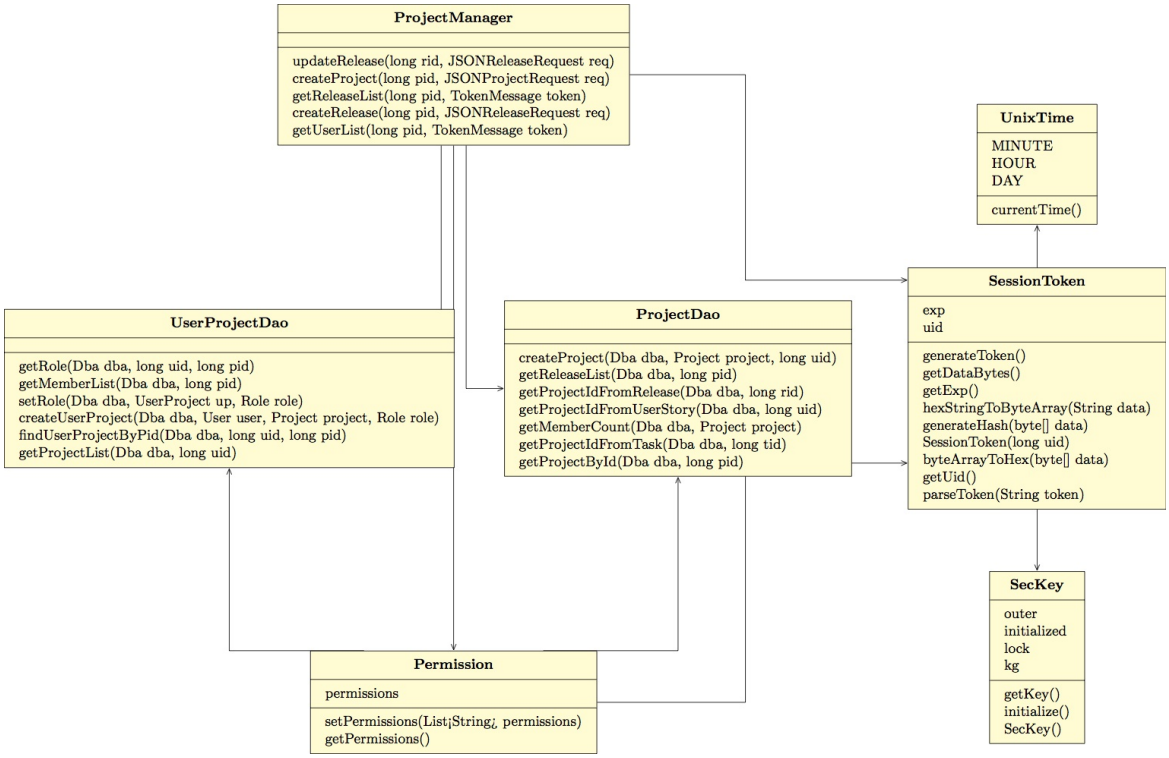
- Classes : Login , Session Token, Time , Renew Token and SecKey.
- All the classes have Directed Relationship. For example Login and RenewToken have a directional relationship with SessionToken. SessionToken has direct relationship with Seckey And UnixTime.
- The bottom partition shows the operations that are associated with each class. Login has the getLoginMessage function and similarly for the others.

REGISTER



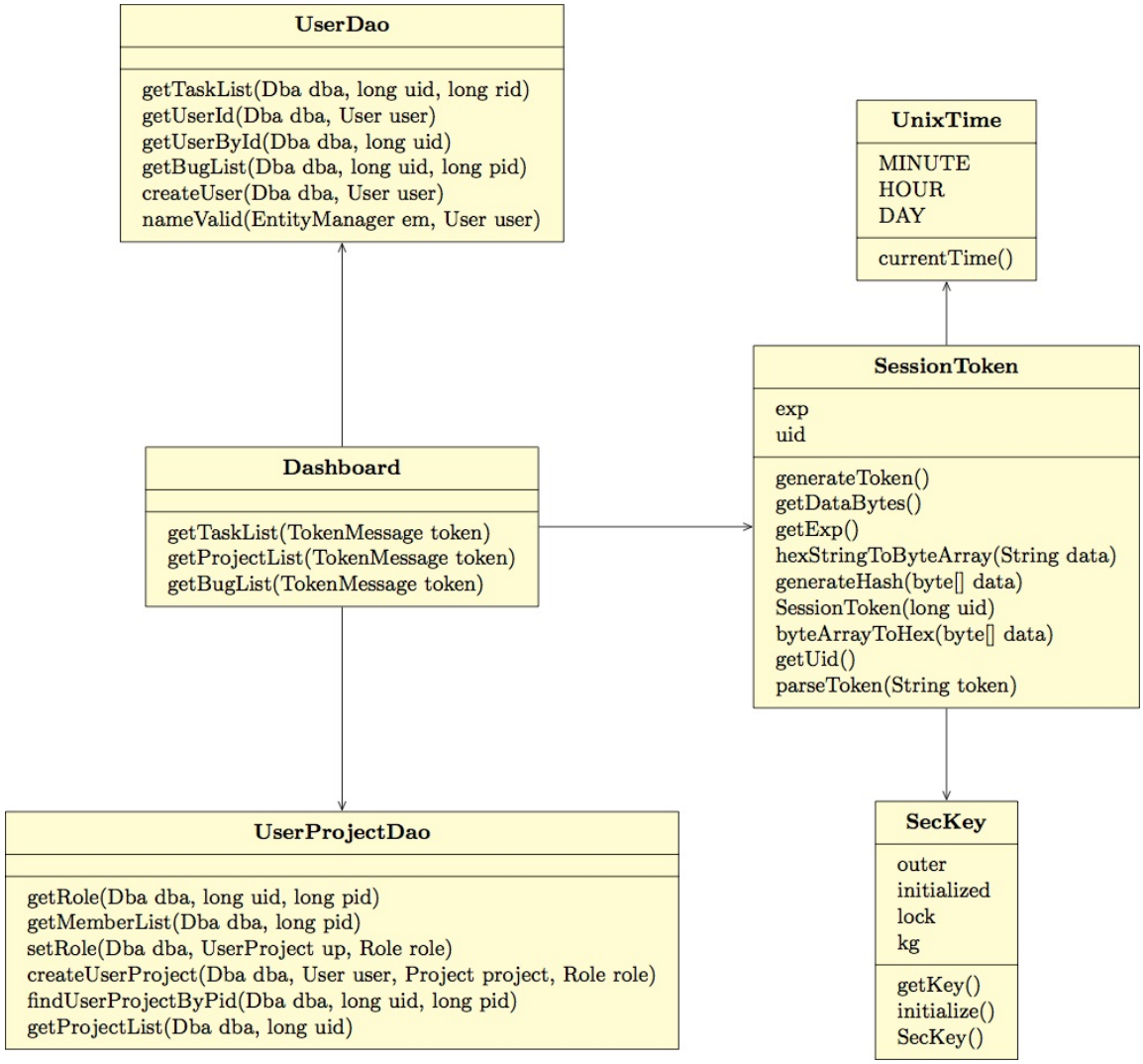
- Two Classes : UserDao and Register .
- Directed relationship between the two classes.
- Register has getRegisterMessage function.

Project Manager



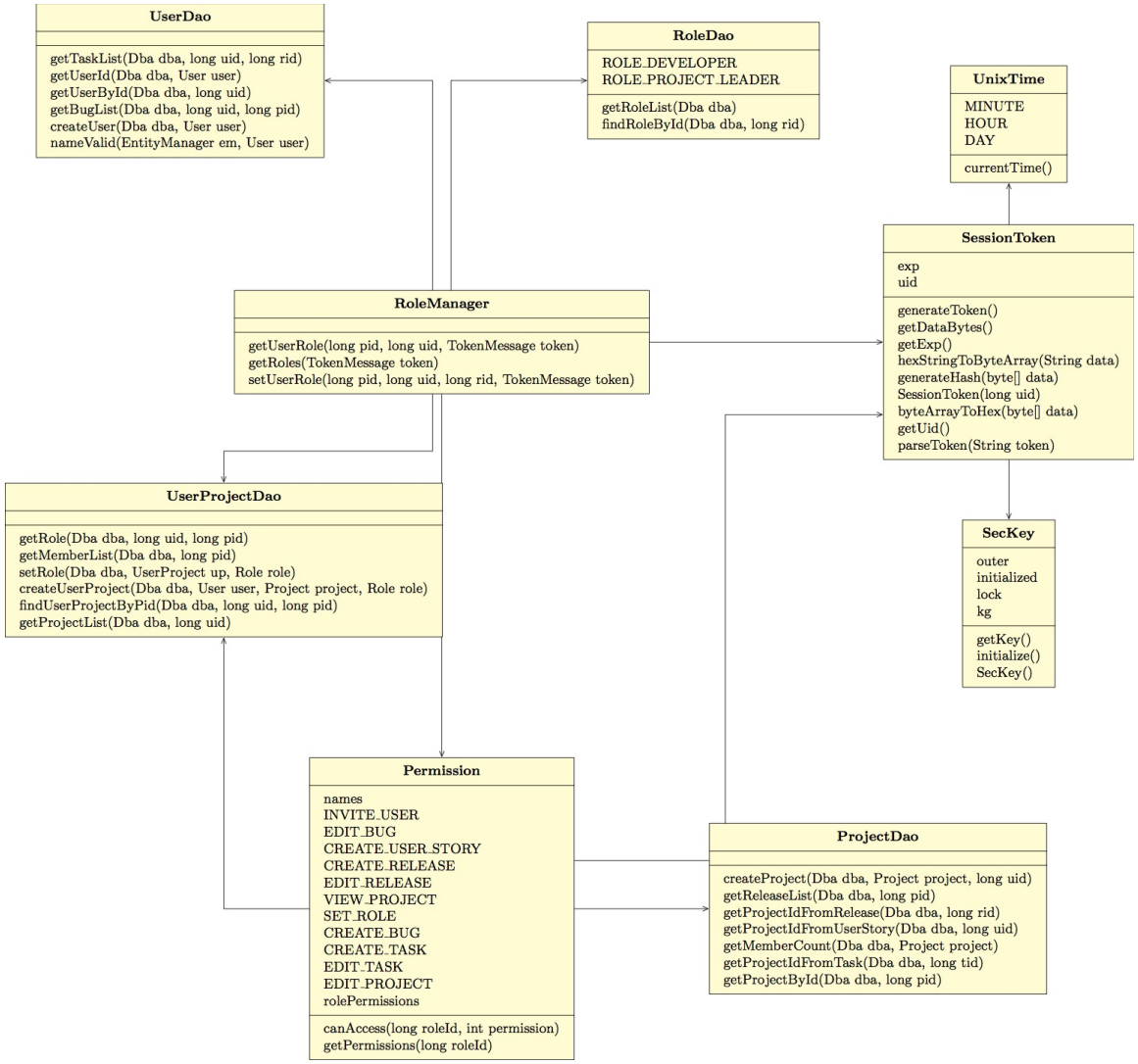
- The various classes are : Project Manager, UnixTime, SessionToken, SecKey , UserProjectDao, ProjectDao and Permission.
- They also directed relationship between them .

Dashboard



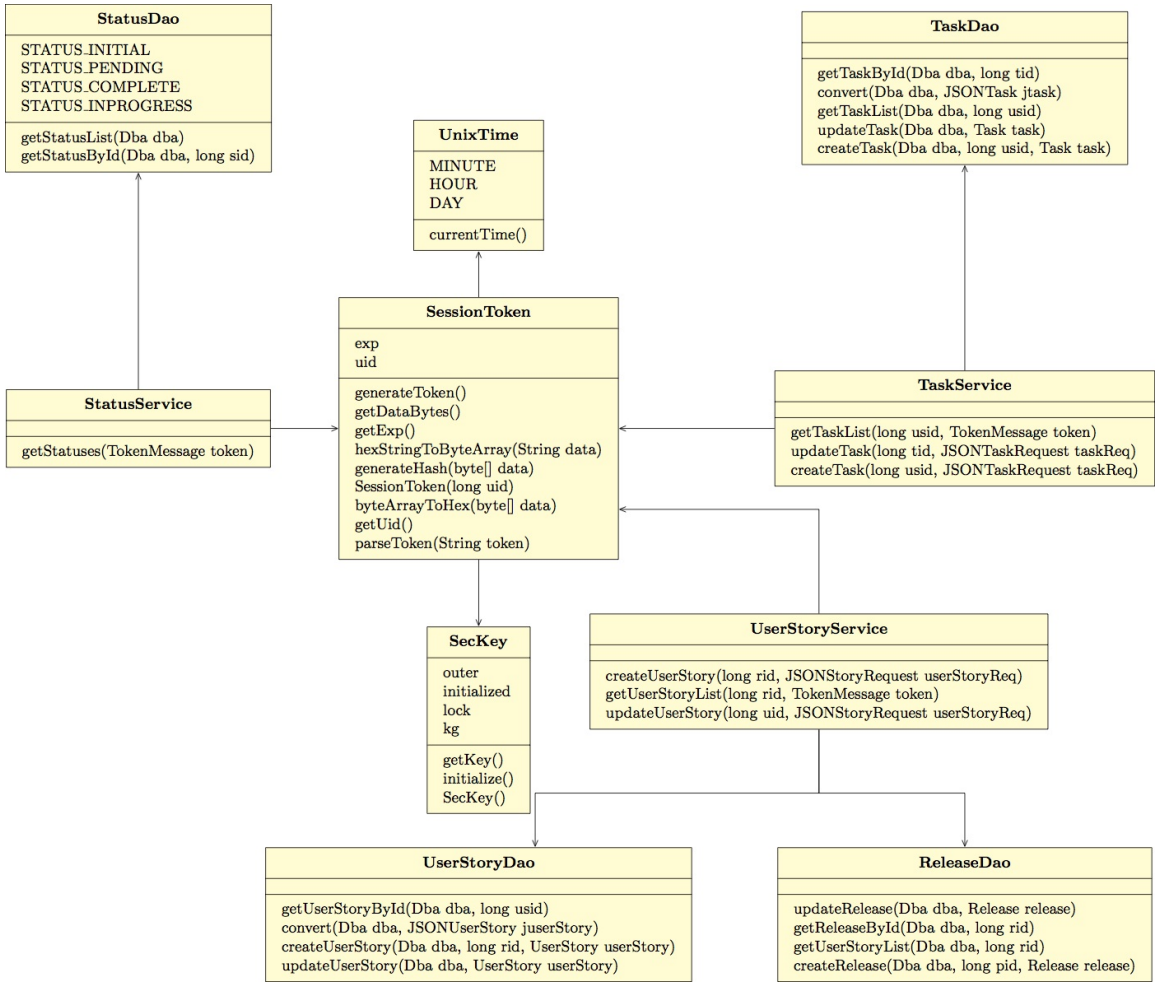
- Classes : UserDao,UnixTime, Dashboard, SessionToken, UserProjectDao and SecKey.
- Relationships : Directed between Dashboard and UserDao.
- Directed relationship between Dashboard and UserProjectDao.
- Directed relationship between Dashboard and SessionToken.
- Directed relationship between SessionToken and Seckey.
- Directed relationship between SessionToken and UnixTime.

Role Manager



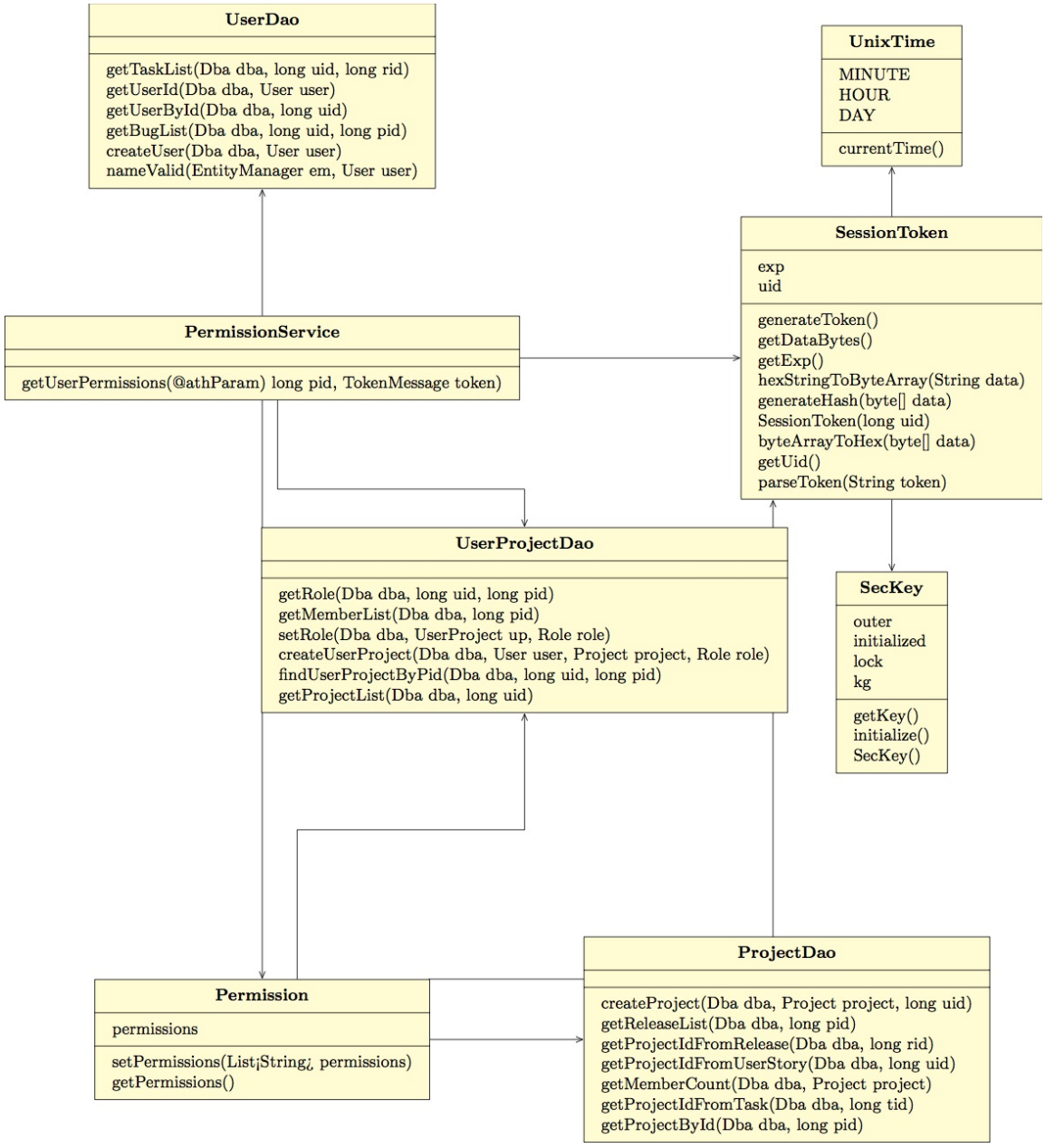
- Classes : UserDao, RoleDao, UnixTime, RoleManager,SessionToken, UserProjectDao,Permission,ProjectDao and Seckey.
- Relatioships : Directed between Role Manager and UserDao.
- Directed between Role Manager and UserProjectDao.
- Directed between Role Manager and RoleDao.
- Directed between Role Manager and Permission.
- Directed between Role Manager and SessionToken.
- Directed between SessionToken and SecKey.
- Directed between SessionToken and UnixTime.
- Directed between SessionToken and ProjectDao.
- Directed between Permission and ProjectDao.
- Directed between Permission and UserProjectDao.

Status, User Story, and Tasks



- Classes are : UnixTime,StatusDao,TaskDao, StatusService,SessionToken,TaskService, UserStoryService, UserStoryDao and Release Dao.
- Relationship between Classes:
- Directed Relation between StatusService and SessionToken.
- Directed Relation between SessionToken and UnixTime.
- Directed Relation between SessionToken and SecKey.
- Directed Relation between SessionToken and TaskService.
- Directed Relation between SessionToken and UserStoryService.
- Directed Relation between TaskDao and TaskService.
- Directed Relation between UserStoryService and UserStoryDao.
- Directed Relation between UserStoryService and ReleaseDao.

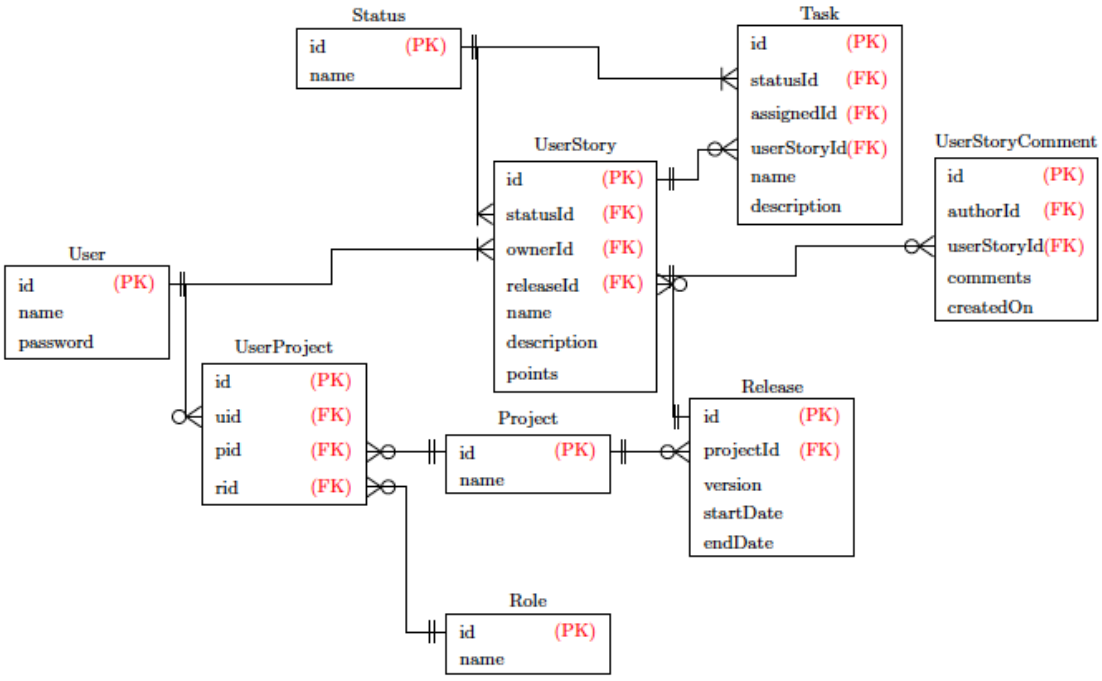
Permission Service



- Classes for this services are : UserDao, UnixTime, SessionToken, PermissionService, UserProjectDao, SecKey , Permission and ProjectDao.
- Directed Relationship between UserDao and PermissionService.
- Directed Relationship between PermissionService and UserProjectDao.
- Directed Relationship between PermissionService and Permission.
- Directed Relationship between PermissionService and SessionToken.
- Directed Relationship between UnixTime and SessionToken.
- Directed Relationship between UserProjectDao and Permission.

- Directed Relationship between UserProjectDao and ProjectDao.
- Directed Relationship between UserProjectDao and SessionToken.
- Directed Relationship between SecKey and SessionToken.
- Directed Relationship between ProjectDao and Permission.

Database Design
The following is our database design of PLM :



Terms :
PK : Primary Key
FK : Foreign Key

Design Patterns

In this section, you shall describe any design patterns used in your software system.
The Dba class uses a singleton pattern to provide database connections to any service that requires it. The SecKey class also uses a singleton pattern but in this case it's to provide the encryption key used in session tokens. In both instances, the singleton pattern allows many different objects to make use of a single resource.

Key Algorithms

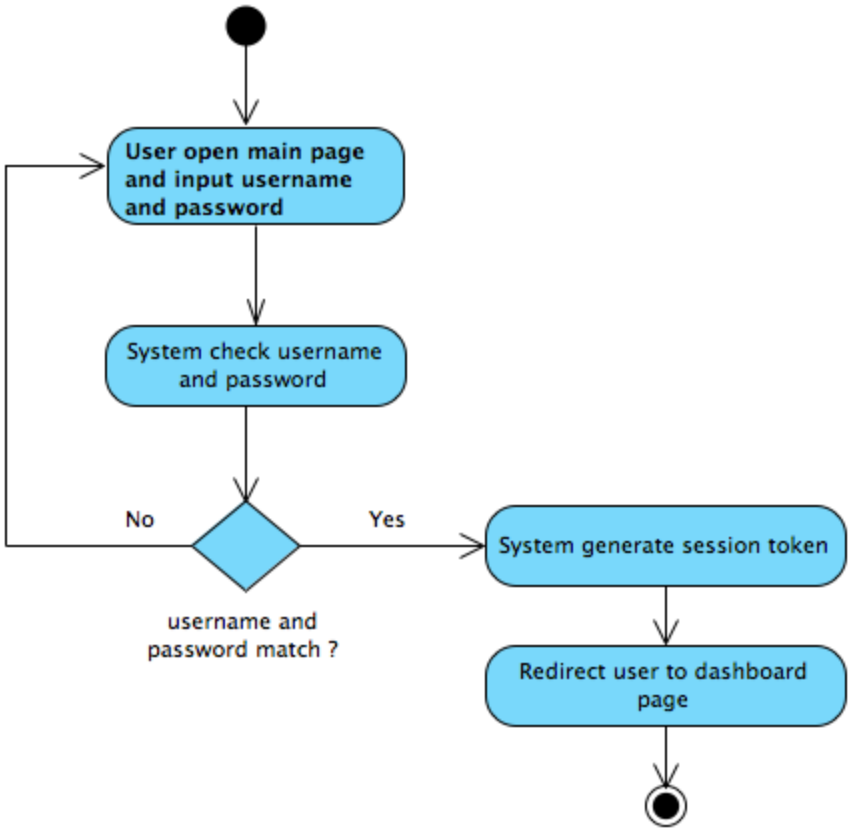
In this section, you shall describe any key algorithms used in your software system,

either in terms of pseudocode or flowchart.

LOGIN

- Username and password are matched against the database
- If the credentials are valid, a session token is generated
    - A string is created containing: user ID and token expire timestamp
    - A SHA-1 hash of that string is concatenated onto the string
    - The entire string is symmetrically encrypted with AES
    - The AES encryption prevents users from modifying the ID or timestamp while the hash prevents users from successfully modifying the encrypted string.
- The token will be decrypted and the hash will be validated by the services in future calls to check a user's identity.

The following is flowchart for Login :

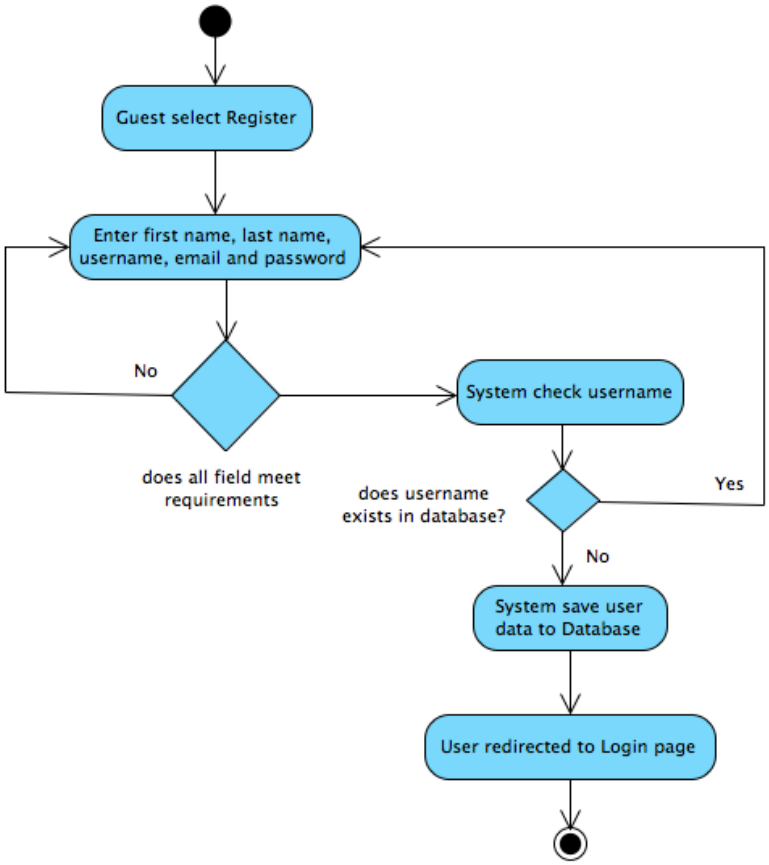


REGISTER

The following is diagram flow for Register :

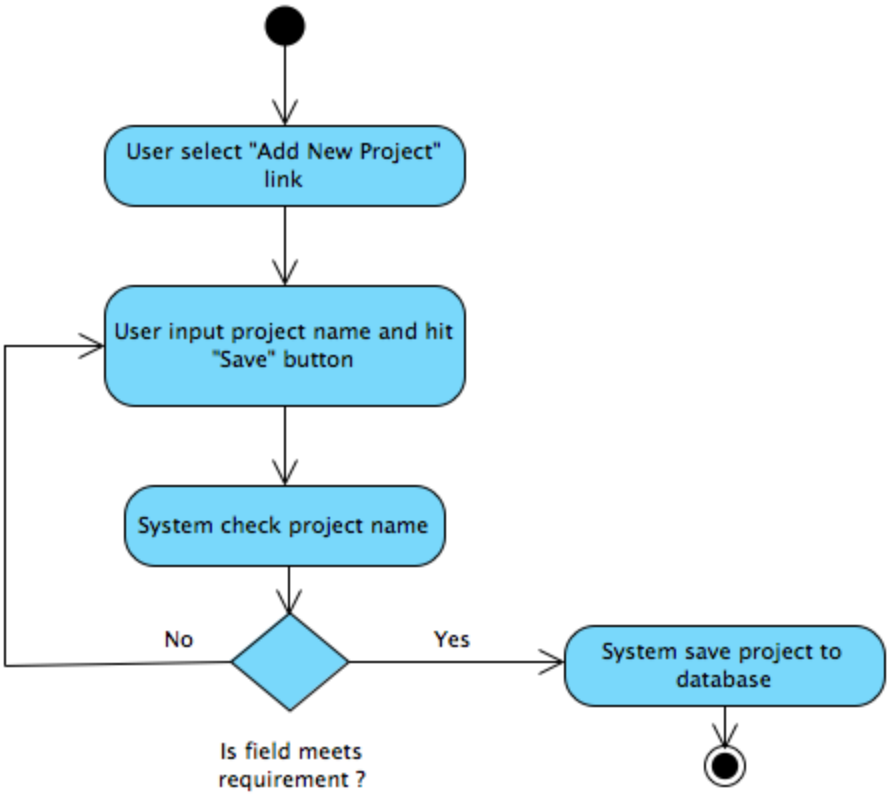
- The guest selects the Register Option which directs him to the registration page.
- Enter first name, last name , username, email and password.

- Check if all the fields have been filled and control returns of anyone is missing , if not then the system checks if username already exists.
- Go back to the registration page if username exists otherwise save the userdata to the database and user is redirected to the Login Page .



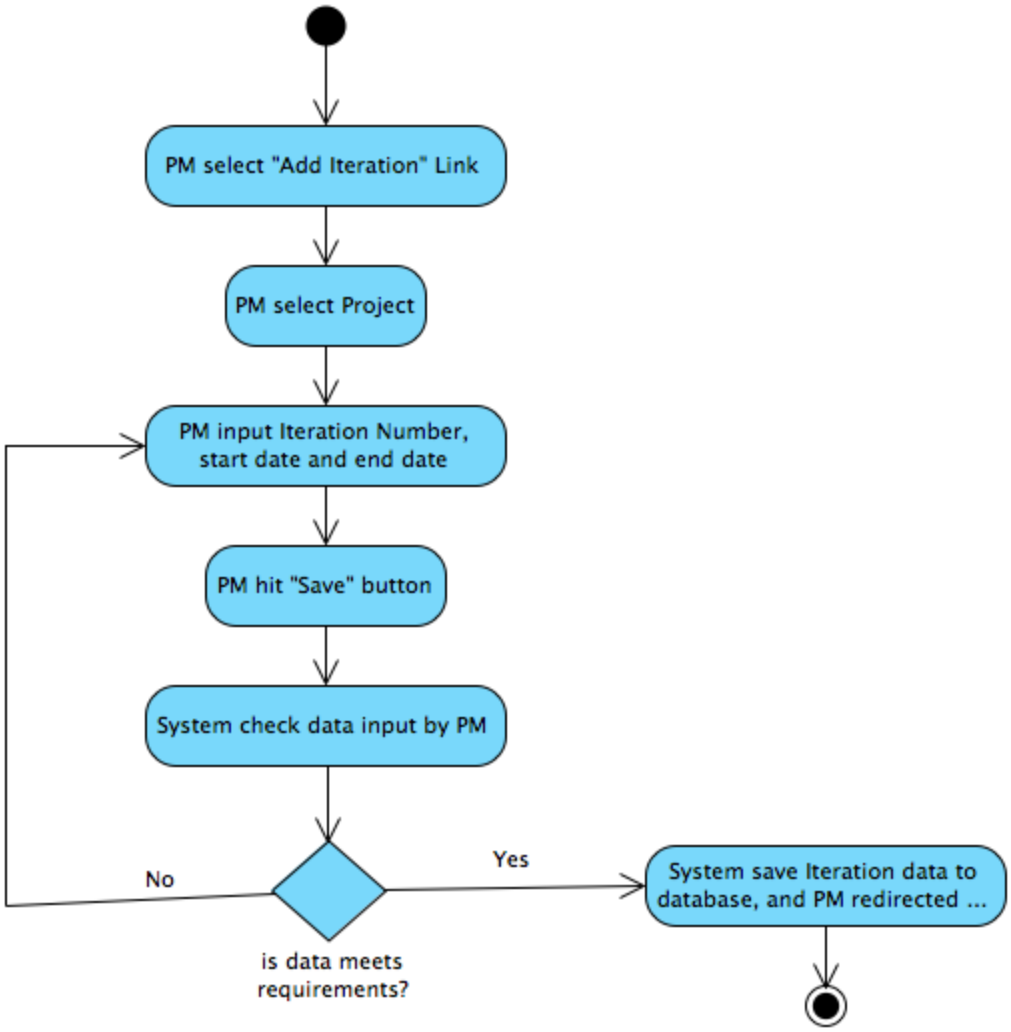
MANAGE PROJECT

- User selects “Add New Project” link.
- Input the Project Name and select save button.
- The name is crosschecked with the database to ensure that duplication does not take place and also if all the field requirements are met. If not go back otherwise save the data to the database.



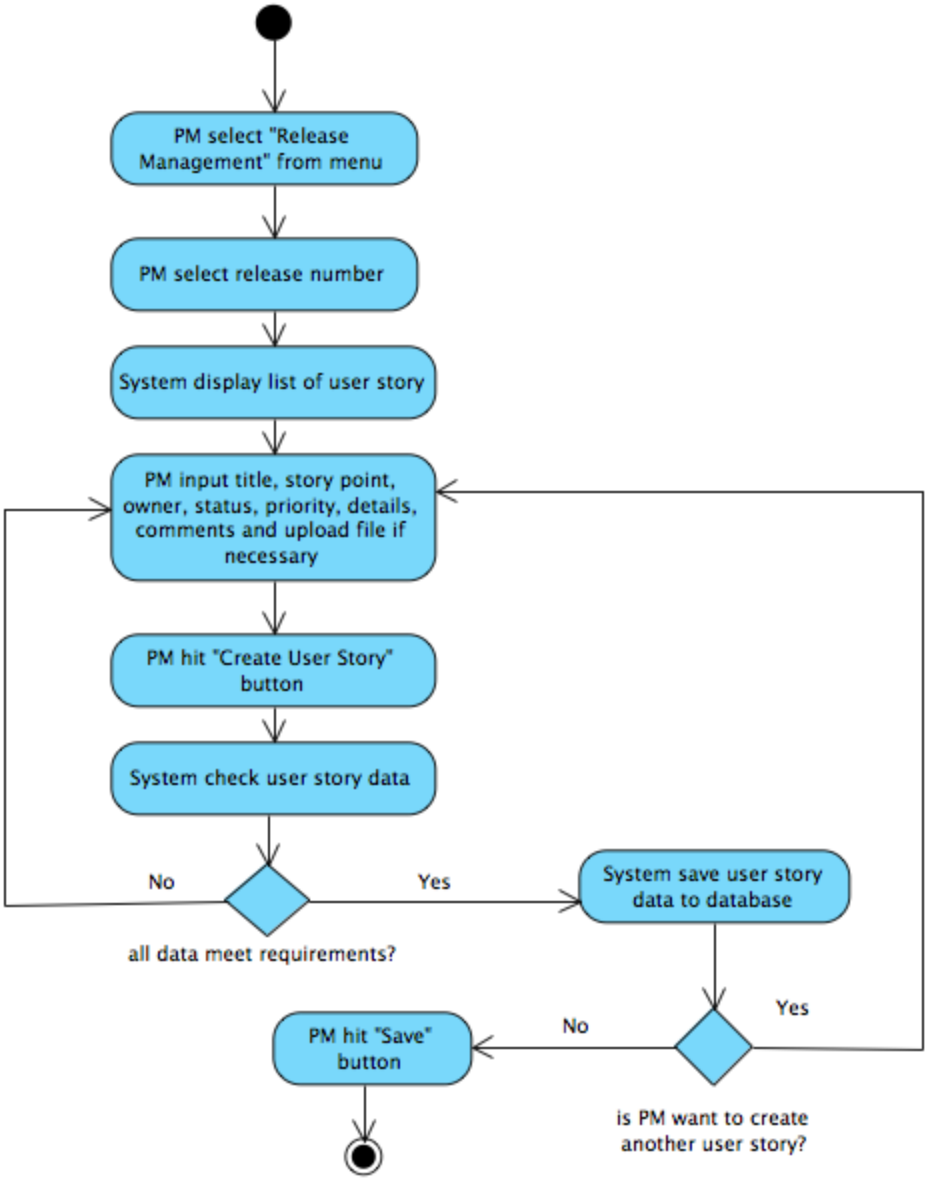
MANAGE ITERATION

- PM selects the “Add Iteration” Link and then selects the desired Project.
- Then inputs the iteration number, start date and end date and click on save.
- System validates the data and saves it to the database if its correct.



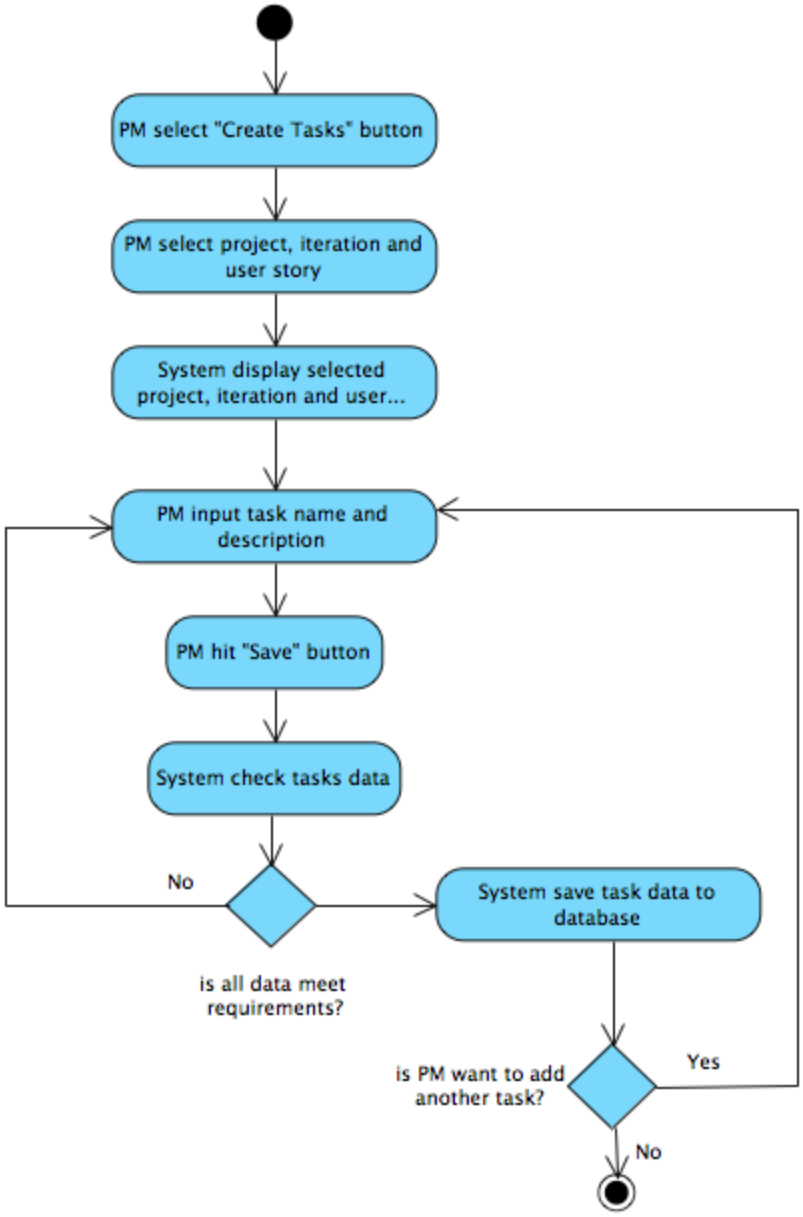
MANAGE USER STORY

- Select Release Management link and then the release number.
- A list of user stories is displayed .
- Input the required data and clink on “Create User Story “.
- The data will be then validated and if it meets the requirements the system then saves the data to the database, if not the data has to be entered again.
- If the PM wishes to add another userstory the entire process is repeated.
- Click on “Save” button to finish.



MANAGE TASK

- PM selects “Create Tasks” button.
- Select Project, Iteration and UserStory and the system will then display them.
- The PM then puts in the task name and description which is validated and if true is saved to the database.
- The PM can repeat this if he wishes to add more tasks otherwise exit.



- Classes and Methods

  This part can be a reference to automatic generated document for all classes and methods.

  All Classes and Methods enclosed in automatic generated documents.

- References

- Glossary