**CS673S16 Software Engineering**

**Team X - Project Name**

**Software Design Document**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Xiang Chen | Configuration Leader;  Integration Environment Leader | *Xiang Chen* | 10/24/2017 |
| Yansen Liu | Design leader | *Yansen Liu* | 10/26/2017 |
| Weicheng Yu | Backup Team Leader; Security Leader | *Weicheng Yu* | 10/26/2017 |
| Lu Min | QA Leader | *Lu Min* | 11/08/2017 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
| **alpha** | **Team 3** | **10/30/2017** | **Complete main features** |
|  |  |  |  |

[Introduction](#_87t9hln2vjz0)

[Software Architecture](#_buttcq9i221r)

[Design Patterns](#_x18fj36s1121)

[Key Algorithms](#_mtfbusfb0eq3)

[Classes and Methods](#_7ucksmkf6rzx)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

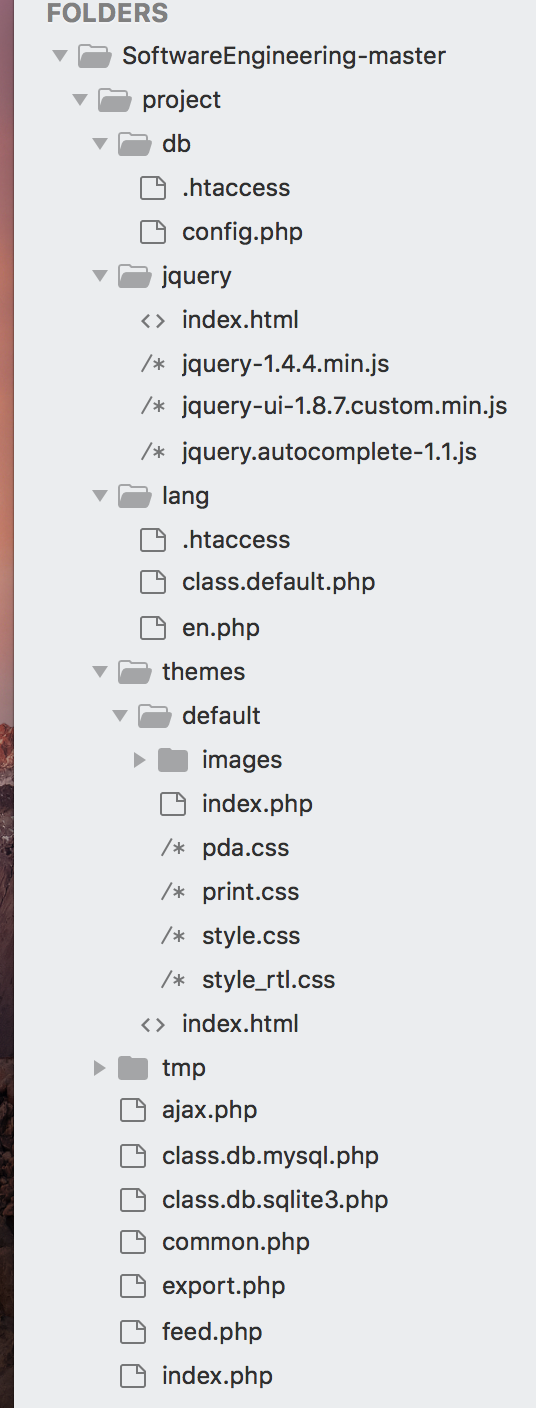
# Introduction (Weicheng)

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

# Software Architecture (Yansen,Dawei,Xiang)

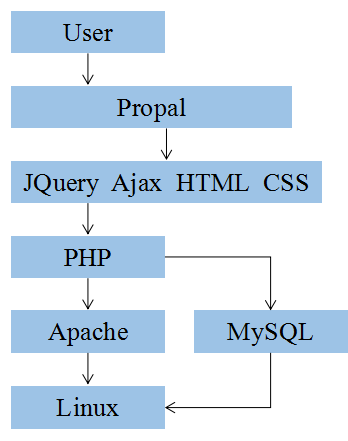
Example: <http://wiki.processmaker.com/index.php/ProcessMaker_Architecture_Diagrams>

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. (Yansen)

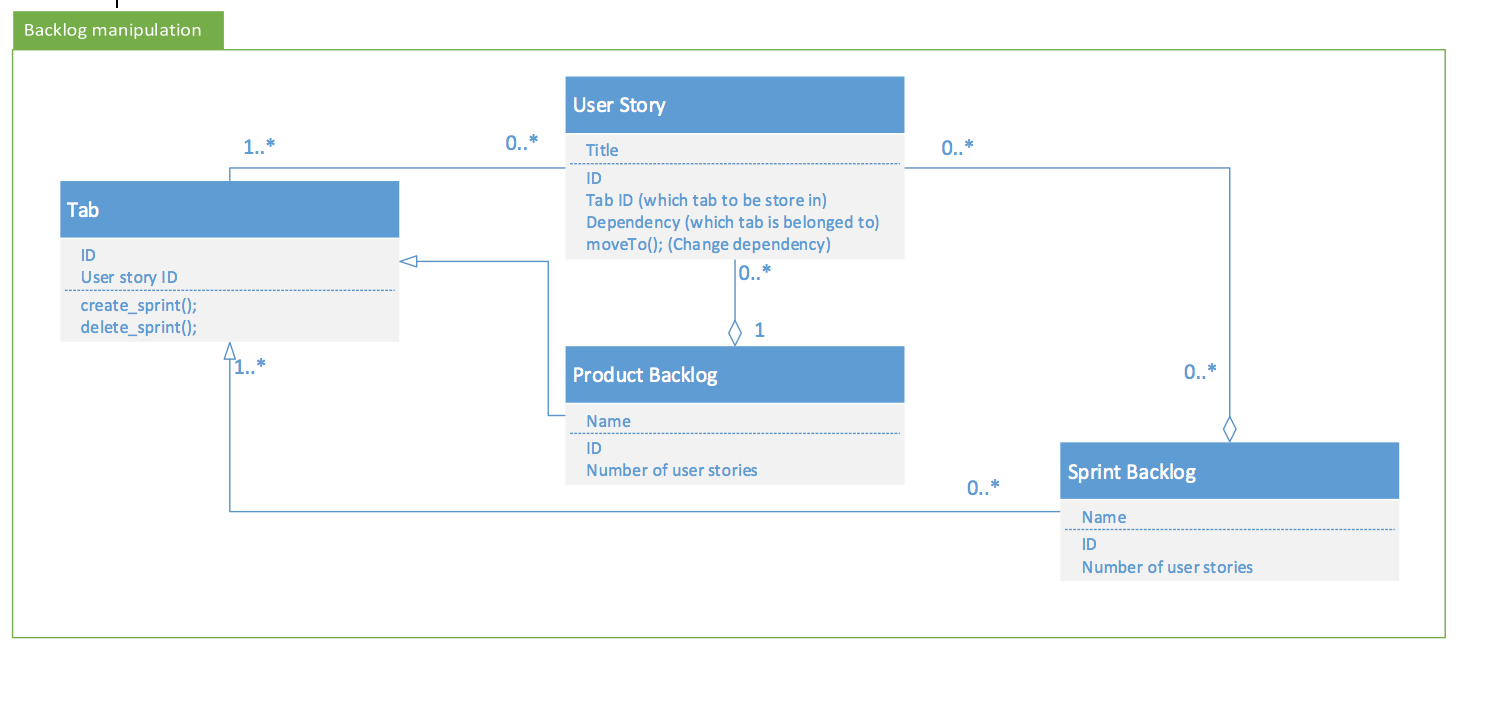


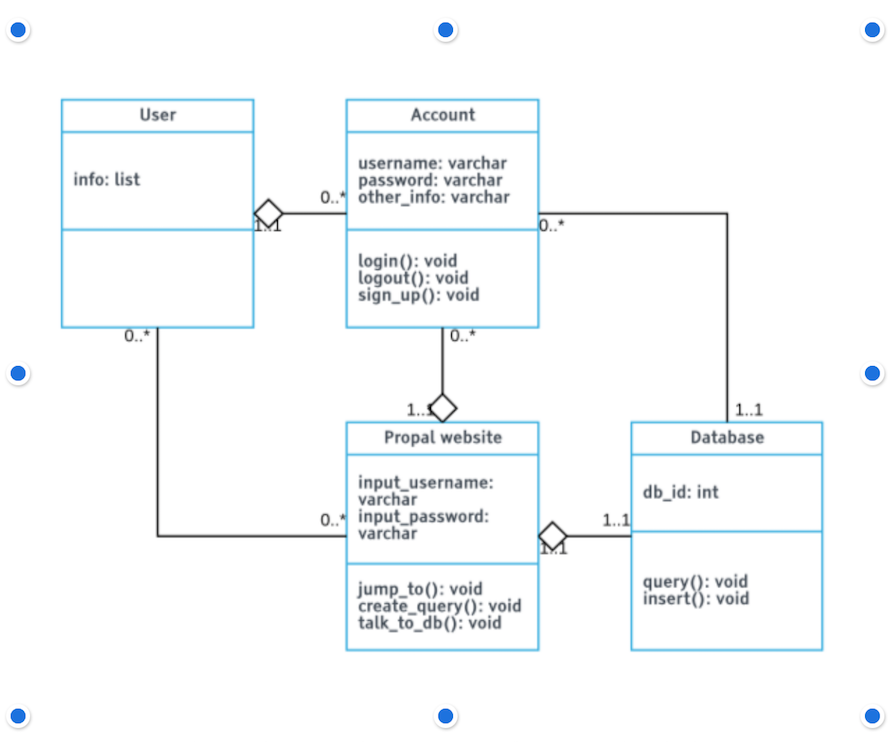
You shall have a **diagram to show the whole architecture**, and **class diagram** for each component. (Xiang )

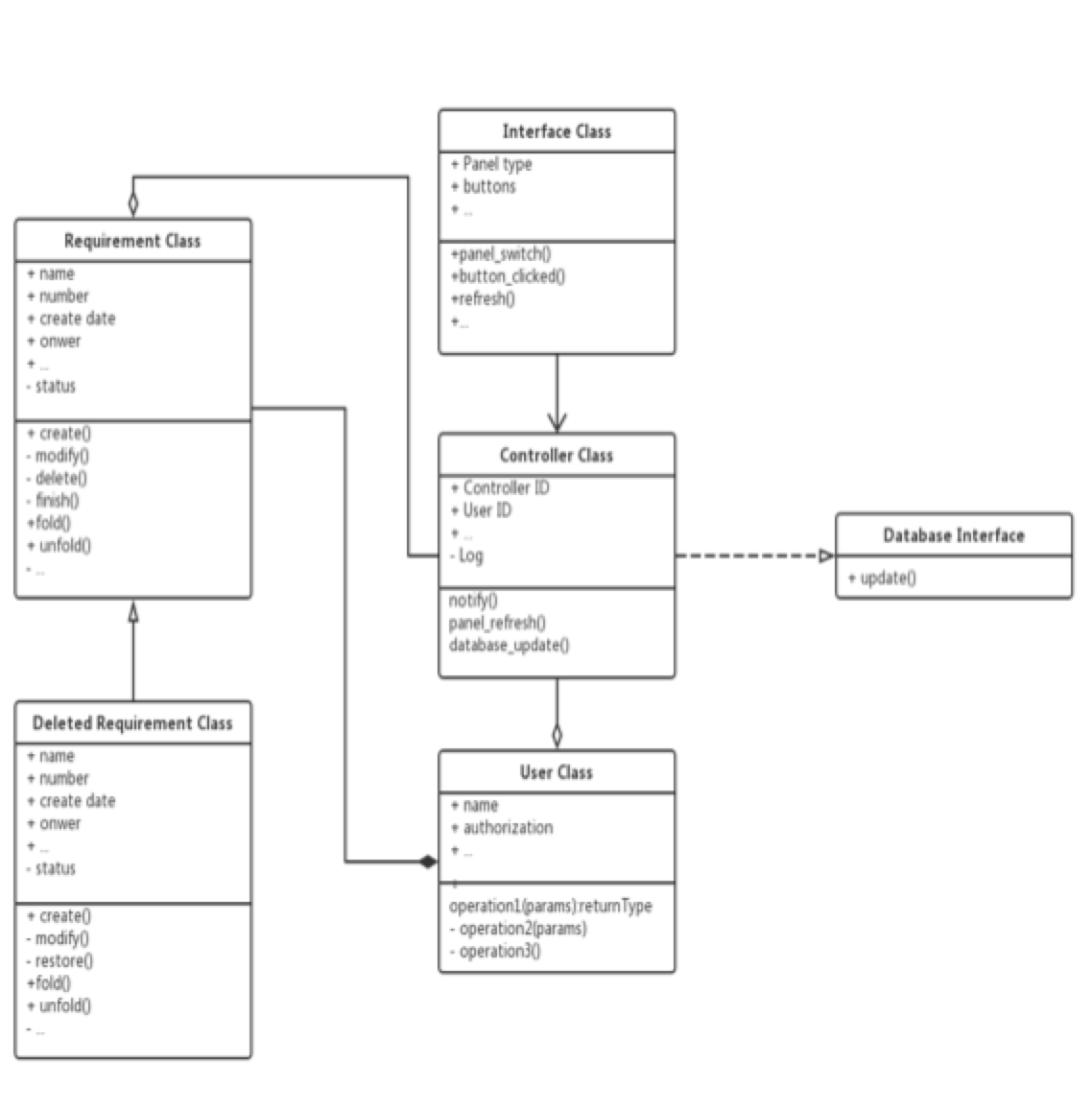
**Whole software architecture**



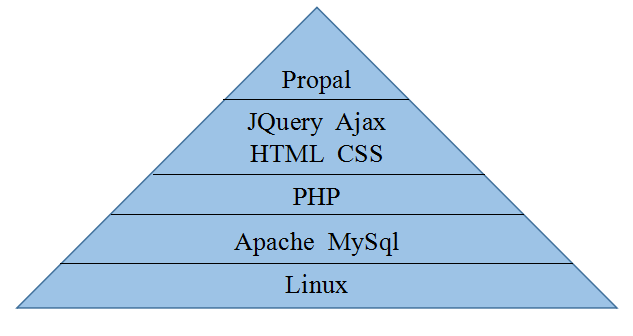
**Class diagram**







**The interface of each component and dependency between components**



Dependency:

The basic in our software is the bottom one which is Linux system, everything were built on it. We use Mysql to build our database and use Apache to set our server. After this, we use PHP to start creating our software. We use Jquery, Ajax, HTML, and CSS based on PHP to design and create the application. And the top component is our web application: “Propal” which create by everything under it.

If any framework is used, it shall be defined here too. **Database design** should also be described if used. (Dawei Li)

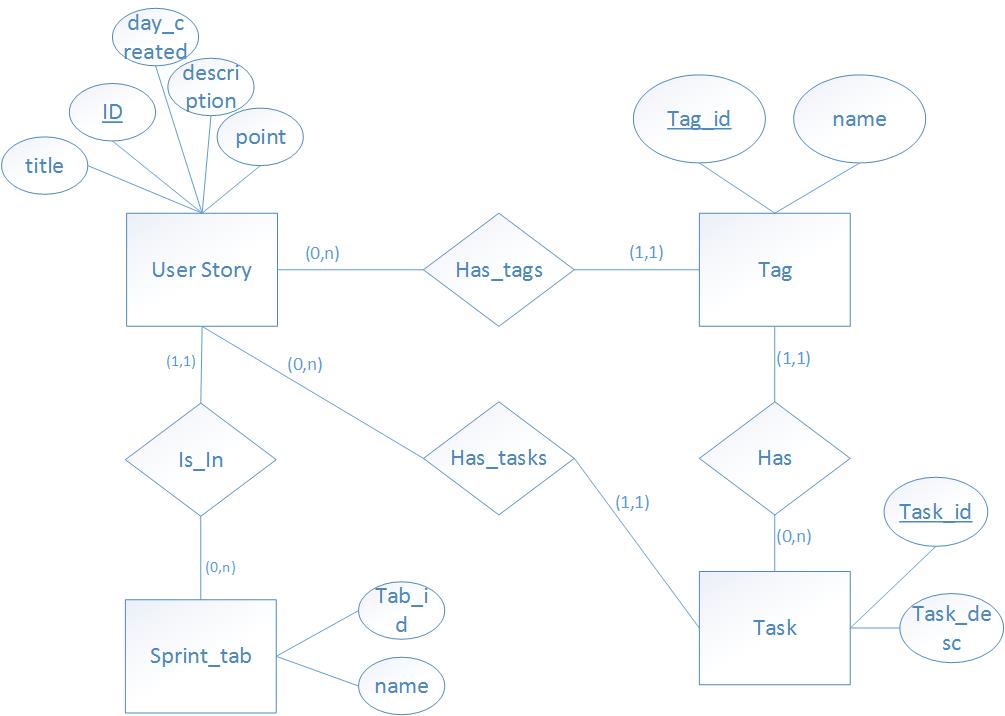
**Database ER diagram (Dawei Li)**

Here is the database design ER diagram.

To user story: the Primary Key is ID. The day\_created will be used to sort the user story by recording the day it is created. The point will be calculated and decided combining men hour.

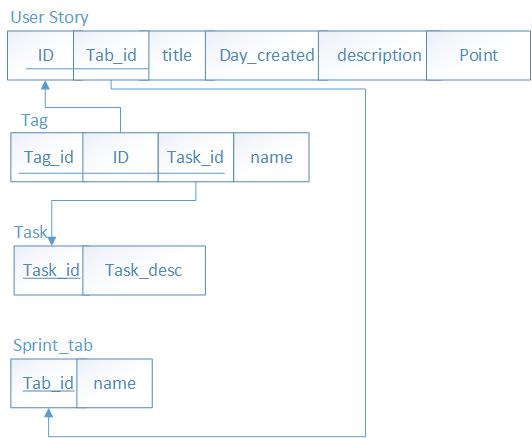
To Tag: Primary key is Tag\_id. Each user story could have a tag; also each task could have a tag.

To Sprint\_tab: each user story must be located in a sprint tab.

****

**Database schema(Dawei Li)**

This schema will guide us to create table, decide primary key and decide foreign key.

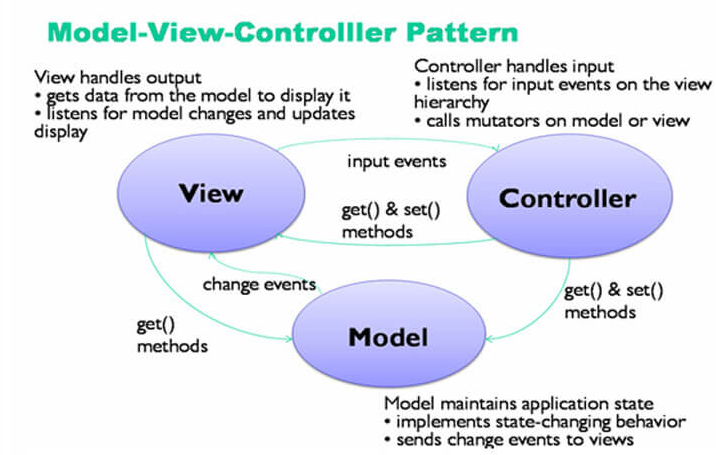
****

# Design Patterns (Weicheng, Yuhao(missing), Dawei)

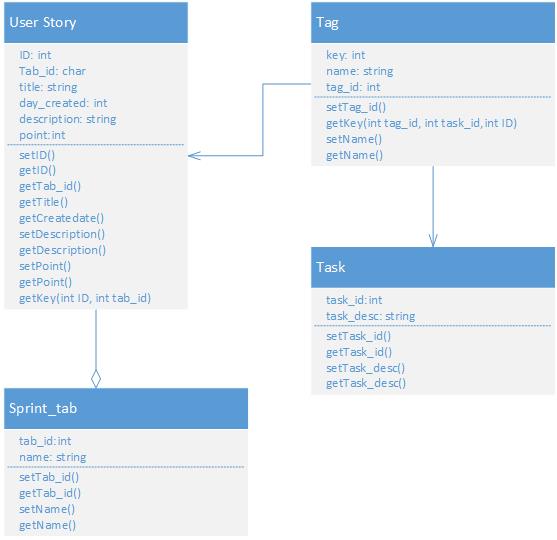
Example : <http://www.siliconinfo.com/open-source-web-application/mvc-development.html>

In this section, you shall describe any design patterns used in your software system.

**MVC design pattern(graphic) (Dawei Li)**

****

|  |  |  |
| --- | --- | --- |
| Model | View | Controller |
| Database | UI, Web Interface | Business Logic |
| MySQL | HTML,CSS,Javascript,JQuery | PHP |
| Store all the data | Present the front-end web page | Exchange data between Model and View |



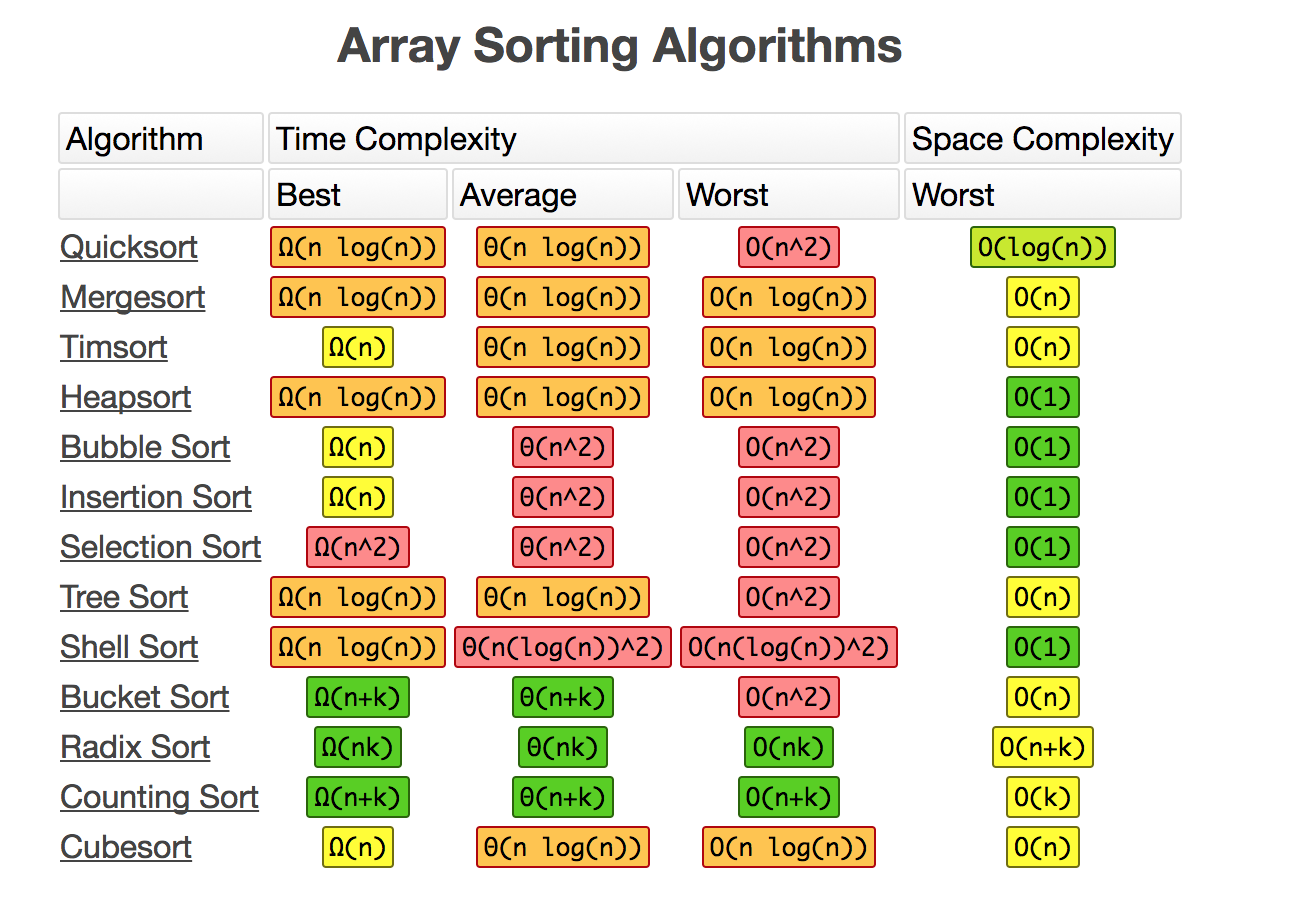
# Key Algorithms (Xiang)

In this section, you shall describe any key algorithms used in your software system, either in terms of pseudocode or flowchart.

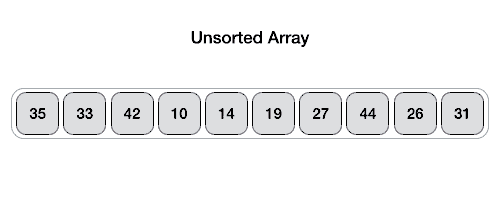
**Sort Algorithm**

We need to sort all the user story by due date, thus we need sort algorithm

According to the picture below and our usage, we chose quicksort algorithm , which is O(nlogn) on average.

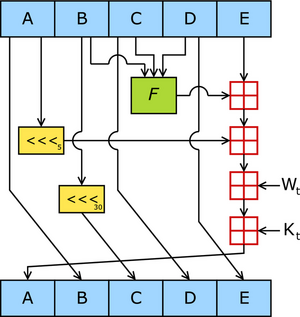


**Animation of Quicksort**

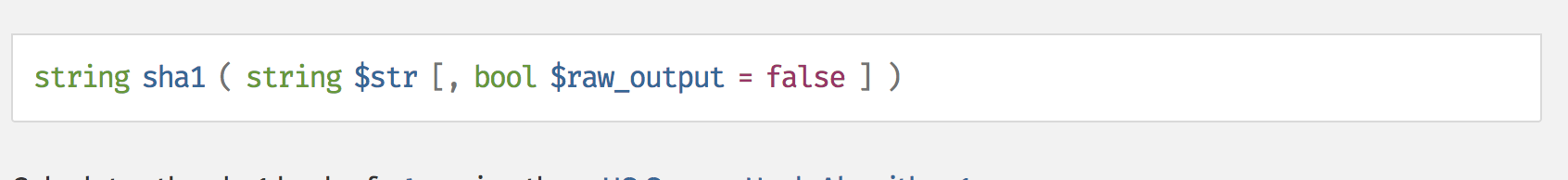


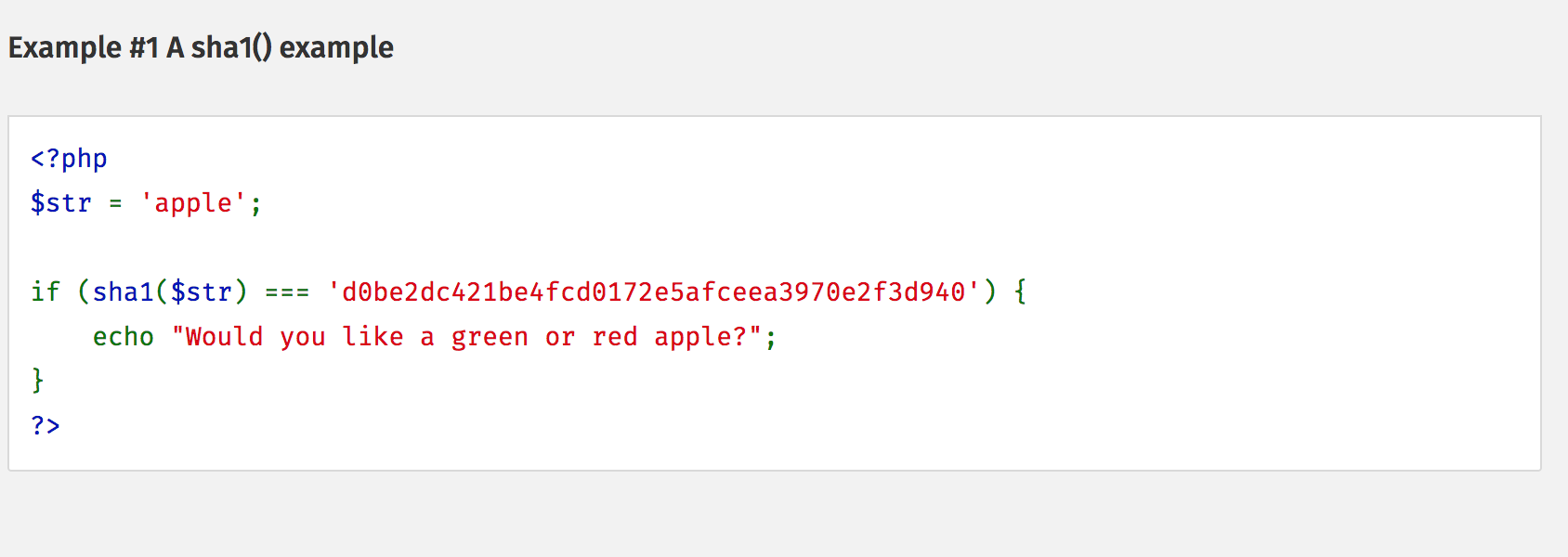
**Hash Algorithm**

We need to protect the privacy of user data, so we need to encrypt the sensitive data stored in the database. We choose to use SHA-1 Algorithm



We leverage the sha function in php

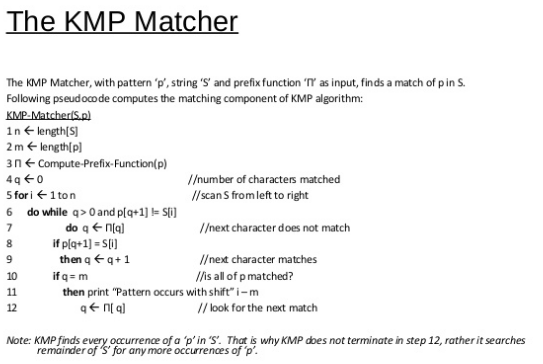




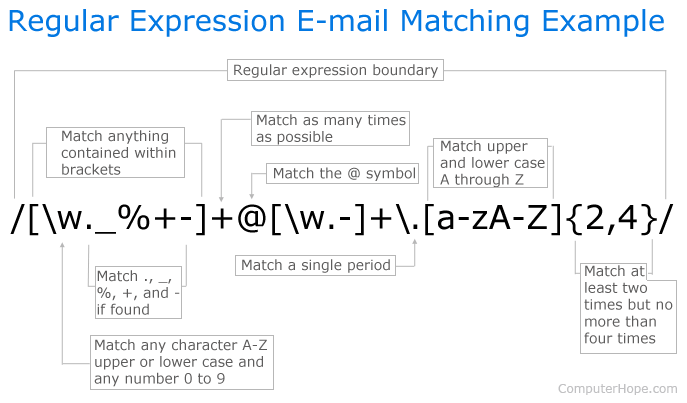
**String Matching and Parsing Algorithm**

We need to handle several case of string search and matching. In this project, we mainly used two kind of algorithms: KMP Algorithm (String Matching) and Regular Expression (String Parsing)

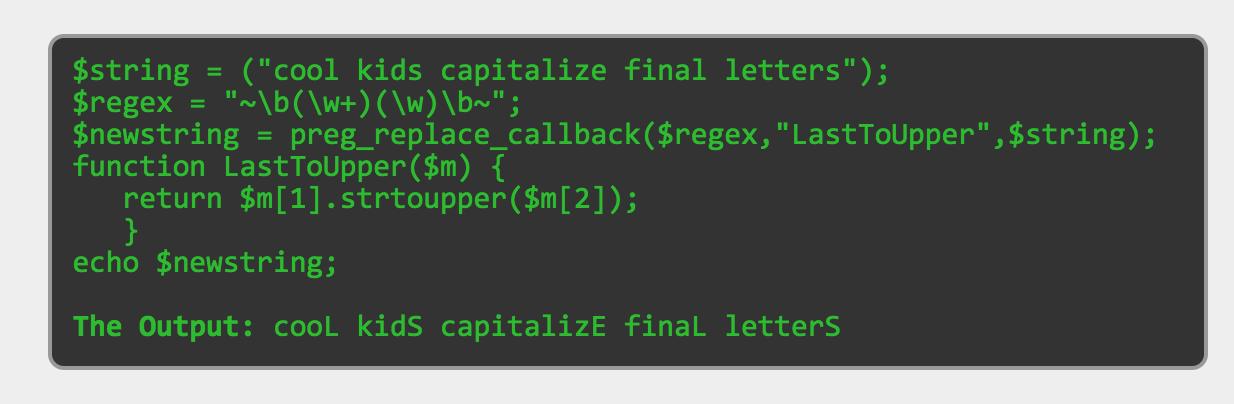
Knuth–Morris–Pratt string searching algorithm (or KMP algorithm) searches for occurrences of a "word" W within a main "text string" S by employing the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters.



Regular expression



Used in PHP, example:



# Classes and Methods (Yansen, Yuhao(missing), Chen Shou)

Example: <http://docs.phpdoc.org/getting-started/your-first-set-of-documentation.html>

Another choice: <https://stackoverflow.com/questions/1182781/how-do-you-document-your-php-functions-and-classes-inline>

1. deleteStory()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: deleting story

1. completeStory()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: complete existing story

1. editStory()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: edit the detail of existing story

1. saveStory()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: save the changing of story

1. searchStory()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: search for story by words

1. submitNewStory()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: add new story to story list, add 1 on the total number of story

1. addList()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: add new sprint list by typing in new id(default: new sprint)

1. renameCurList()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: rename a sprint list by typing in new name

1. deleteCurList()

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: delete a sprint list from database

1. errorDenied():

@author: Yansen Liu

@package: tmp

@version: 1.0

@function: return “Access Denied” when error occur

1. deleteTasks($id):

@author: Chen Shou

@package: back

@version: 1.0

@function: implement a task deletion function and commit onto database

1. moveTask($id, $listId):

@author: Chen Shou

@package: back

@version: 1.0

@function: implement a task moving function and commit updates onto database

1. prepareTags($tagsStr):

@author: Chen Shou

@package: back

@version: 1.0

@function: assignments of tag and return the tags created

1. getOrCreateTag($name):

@author: Chen Shou

@package: back

@version: 1.0

@function: if no existence of tag for certain name create one, otherwise show the existing one

15. getTagId($tag):

@author: Chen Shou

@package: back

@version: 1.0

@function: get tag’s id according to tag given

16. get\_task\_tags($id):

@author: Chen Shou

@package: back

@version: 1.0

@function: get tag’s first row according to specific tag id

17. addTaskTags($taskId, $tagIds, $listId):

@author: Chen Shou

@package: back

@version: 1.0

@function: base on taskId, tagId, and listId, create task

18. prepareList($row):

@author: Chen Shou

@package: back

@version: 1.0

@function: base on taskId, tagId, and listId, create task

19. prepareTaskRow($r):

@author: Chen Shou

@package: back

@version: 1.0

@function: get the most updated information from database schema tag2list

20. check\_write\_access($listId = null):

@author: Chen Shou

@package: back

@version: 1.0

@function: check if the user has write access

# References (All)

# Glossary