PROJECT EXPERIENCE REPORT

Raid Static Manager

2020-4-10

THM
Han Gao 200354969

Git-hub Link: https://github.com/Yisumic/Raid-Static-Manger

Abstract

Raid Static Manger (RSM) is a web-based software contains several systems to focus on help in-game leader of group of 8 people, which is called “Static”, to do team management out of the game. It composes several functionality in one software, that can improve the efficiency of team management. The primary functionality is the battle record system that customer can record battle situation such as: attendance, total damage, damage per second and mistakes for each team member, dropped items etc. Additionality, members system can do team member modification and member items` details record for team leader. Moreover, a reward system that called battle points(BP) system that can directly evaluate the members performance in team. RSM also provides auction system for team members to get request their items and comments system for team member to leave comments. Future more, there is a lot problems that happened during the working process, most of them were solved, but some of them cannot be solved. Nonetheless, the entire system can still can be improved by adding extra functionality.

Table of Content

1 Introduction

This majority of this project is to make a website that contains many functionality can help team leader to make team management, such as team members change by members system, record battle situation by record system, evaluate team members performance by battle points(BP) system, distribute items by auction system, and leave comments by comments system.

1.1 Background

In these days, with the development of Massive Multiplayer Online Role-Playing Game (MMORPG), the functions in the game have gradually improved. As the most indispensable functionality in MMORPG, team system also tend to be perfection. In this case, the management between team members becomes more and more complicated. A variety of issues often plague team managers such as reward distribution, team communication. Team leaders need to use many of different software that across different platforms.

1.2 Problem solving

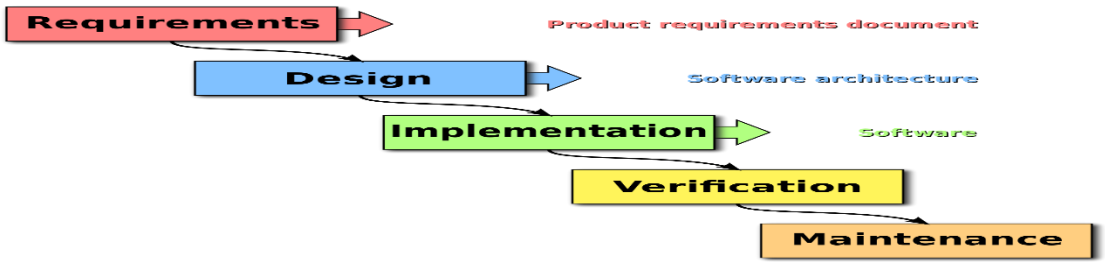
The main purpose of this project is called functionality integration to make it more suitable for a MMORPG team. There is a lot of team management software in the world for different purposes. But there no doubt that team leaders go across several platforms to use different software to manage team will be low efficiency. This project composes most of team management system into one software, and do

some modification to make it more suitable for MMORPG game player to manager their team in higher efficiency.

2 Design Plan

2.1 Methodology

The entire plan of this project is to use waterfall methodology to build the whole system, considering the project is one-person project. By using waterfall methodology can be more straightforward, with more thoughtful in development and cost less time.



2.2 Requirements

- The register system should allow users to register a new account.
- The background color for all windows on the website should be the linen.
- The login system automatically validates the pair of username and password.
- The software system should automatically display the data that relate to the current the user.
- The comments system should allow users to leave comments.
- The battle record system should allow uses to record their battle data in the database

and display on the particular page.

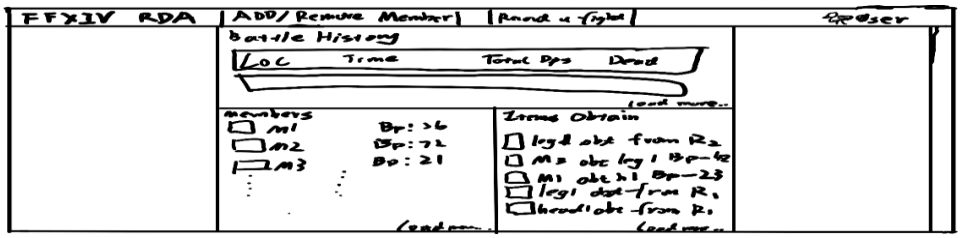
- The members system should allow users to change the members list and coordinate the whole system.
- The auction system should allow users to make an auction for the item and automatically filter the highest bidder and assign item to the bidder.

2.3 Lo-fi Prototype

The early design of this project is the website contains 5 webpages and each page indicate different functionality of the project.

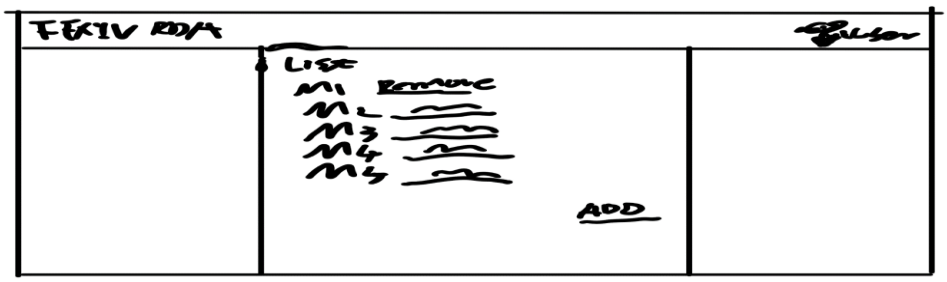
2.3.1 Main page

The Main Page has a quick link to the other pages and will display the short data of battle history, members information, and item distribution with the username.



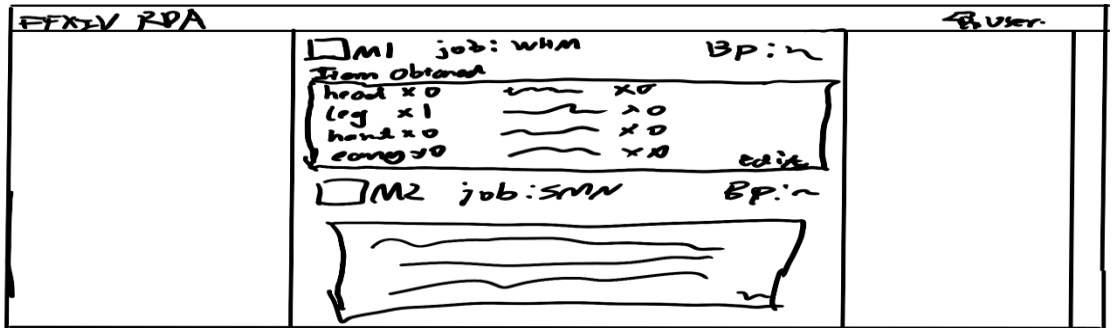
2.3.2 Member List Page

The members list page allowed leader to change the members



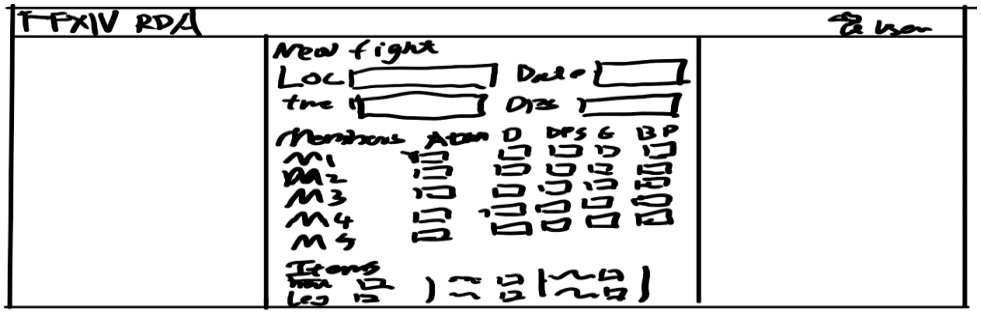
2.3.3 Member Details Page

Member details page shows BP and item obtained situation for each members



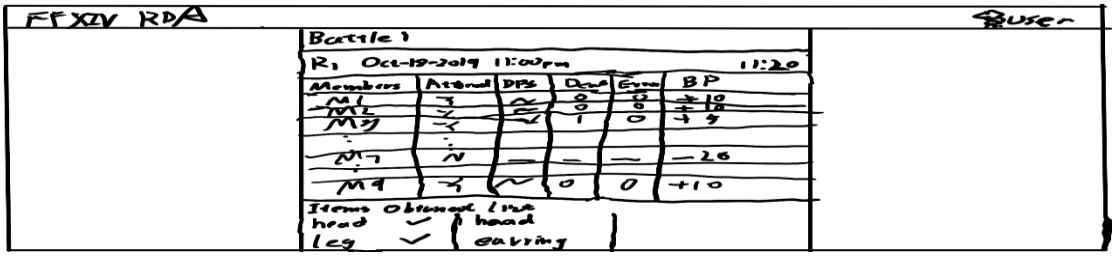
2.3.4 Battle Record page

The battle record page provide a form for user to record their battle situations separate by three parts, the first parts is the battle information which concludes location, date, total time used, and total damage per second(DPS) of the battle.



2.3.5 Battle History Page

The battle history page concludes all the battles information, such as dungen information, battle performance and drooped items



3 Working Experience

3.1 Architecture Changed

During the real implementation of the project, there are some issue that found in the process. First, is there is no page for costumers to create their accounts and login in into the website. So that a propriety signup and login page is necessary. The second issue is the main page, in the lo-fi prototype, there is no space for costumes to leave comments, there an additional area for comments system is needed. Additionally, there is no actually page for customers to make a bid for the drooped item and distribute these items to each particular members.

3.2 Project Procession Analysis

The original design of the programming language used for this project were use PHP to build functionality, use HTML5 and CSS to navigate the layout, use MySQL to create the database and use Hercules as the server. The signup page and login page were the first step to implement. There was a issue that happened during the implementation, which is how to automatically use sever to check the duplicated user name in order to not messed up the database. This issue has been solved by one single SQL statement, which is check if the user name can be found in the database, if the number of user name that found in database is not false, then shows the error check with duplicated username:

```
$query = "select * from User where email = $email";
$check = $con->query($query);
```

```
if (count($check)) {
    $validate = false;
    $error0 = "";
    $error1 = "Email has alredy exist";
```

Next step was building a main page. Most of the part of main page was going well, expect of how to only display the information that created by current login user. The solution was to add one column on user table in database with user id and select that to print results on the main page.

```
mysql> desc User;
+--------+--------------+
| Field  | Type         |
+--------+--------------+
| uid    | int(11)      |
| email  | varchar(255) |
| uname  | varchar(255) |
| pwd    | varchar(255) |
| date   | datetime     |
+--------+--------------+
```

```
$comm = "SELECT * FROM Comments WHERE uid ='$user_id' ORD
$com = $con->query($comm);

$history = "SELECT * FROM History WHERE uid = '$user_id'
$his = $con->query($history);

$bp = "SELECT * FROM Members WHERE uid='$user_id' LIMIT 8
$p = $con->query($bp);

$obtained = "SELECT * FROM Obtained WHERE uid='$user_id'O
$obt = $con->query($obtained);
```

Another problem with main page was happened on comments system, that how to remove comments from comments list. In early test, whenever user clicked remove button, the newest data would be removed instead of the clicked data. That was caused by wrong form range. The problem has been resolved after extended form range to the entire comments table.

The third step of this project was newRecord page. In this step there was a critical problem that happened on data insertion. A group of data that need to insert into database. That means the SQL statement cannot simply submit data separately into database. This caused only one line of data was a valid data, the others were ignored. The solution of this problems was to set input as an array, so that all data would be inserted to database proprietary.

```
        </tr>
        <?php
    while ($row = $members->fetch_assoc()) {
        ?>
<tr>
                <td><?=$row["uname"]?><input type="hidden" name="uname[]" value="<?=$row["uname"]?>" /></td>
                <td><input class="right_msg" type="text" name="atd[]" size="5" /></td>
                <td><input class="right_msg" type="text" name="dps[]" size="5" /></td>
                <td><input class="right_msg" type="text" name="mistake[]" size="5" /></td>
                <td><input class="right_msg" type="text" name="death[]" size="5" /></td>
                <td><input class="right_msg" type="text" name="bp[]" size="5" /></td>
            </tr>

<?php }?>
```

```
foreach ($uname as $_idx => $_uname) {
    $tmp_atd = $atd[$_idx];
    $tmp_dps = $dps[$_idx];
    $tmp_amistake = $mistake[$_idx];
    $tmp_death = $death[$_idx];
    $tmp_bp = $bp[$_idx];
    $mp = "INSERT INTO Performance (batt
    $mbc = "UPDATE Members SET BP = BP +
    $performance = $con->query($mp);
    $bpChange = $con->query($mbc);
}
```

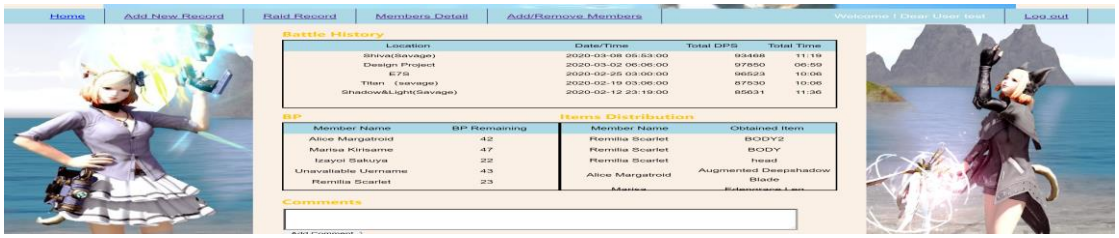
The rest of part were all good in coding process. But for the testing part, the script file which is called “bid.php” cannot not be automatically run in the Hercules by crontab php command due to the security mode in the server, so that maintainer need to manually run the script, unless can get the authorization from the server host to turn off security mode for the file.

Unlike coding part, the UI design part were all went through well by using public library bootstrap scss.

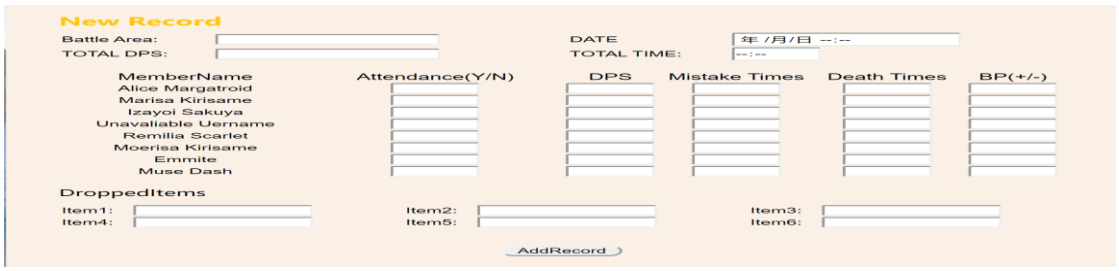
4 Final Result

Login and Signup Page:

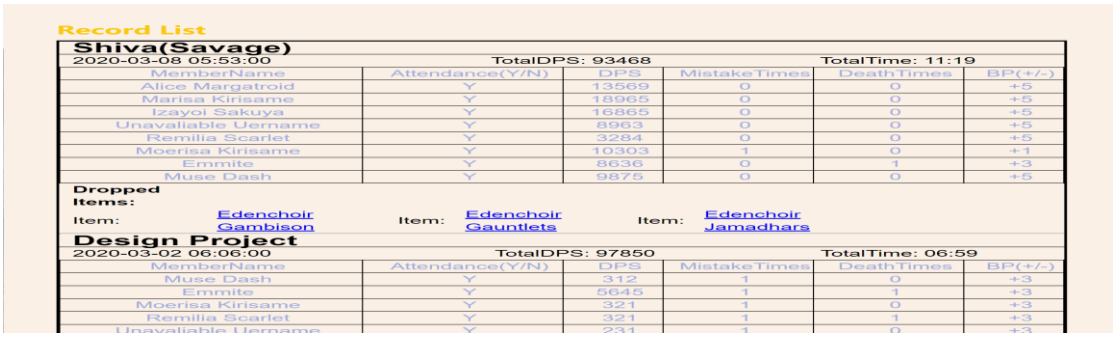
Main Page:



New Record Page:



Battle History Page:



Record List

Shiva(Savage)

2020-03-08 05:53:00 TotalDPS: 93468 TotalTime: 11:19

MemberName	Attendance(Y/N)	DPS	MistakeTimes	DeathTimes	BP(+/-)
Alice Margatroid	Y	13569	0	0	+5
Marisa Kirisame	Y	18965	0	0	+5
Izayoi Sakuya	Y	16865	0	0	+5
Unavaliable Uername	Y	8963	0	0	+5
Remilia Scarlet	Y	3284	0	0	+5
Moerisa Kirisame	Y	10303	1	0	+1
Emmite	Y	8636	0	1	+3
Muse Dash	Y	9875	0	0	+5

Dropped Items:

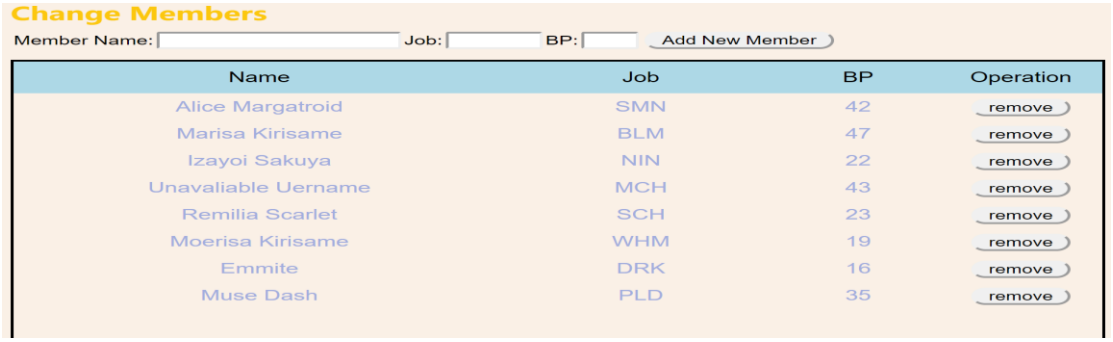
Item: Edenchoir Gambison Item: Edenchoir Gauntlets Item: Edenchoir Jamadhars

Design Project

2020-03-02 06:06:00 TotalDPS: 97850 TotalTime: 06:59

MemberName	Attendance(Y/N)	DPS	MistakeTimes	DeathTimes	BP(+/-)
Muse Dash	Y	312	1	0	+3
Emmite	Y	5645	1	1	+3
Moerisa Kirisame	Y	321	1	0	+3
Remilia Scarlet	Y	321	1	1	+3
Unavaliable Uername	Y	231	1	0	+3

Member List Page:

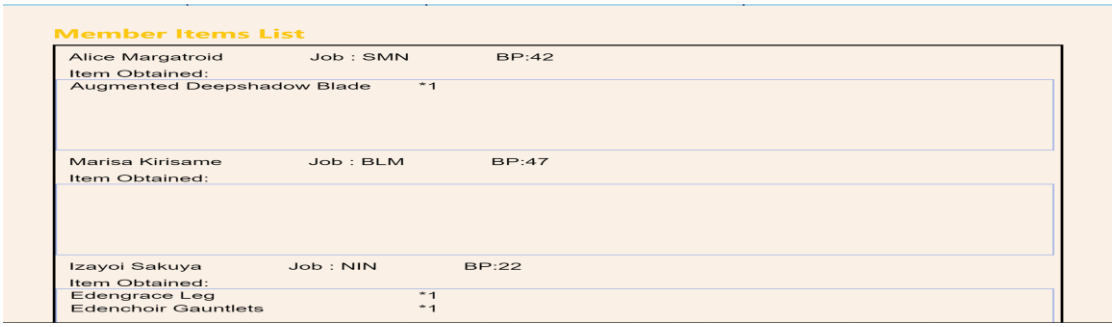


Change Members

Member Name: Job: BP: Add New Member

Name	Job	BP	Operation
Alice Margatroid	SMN	42	remove
Marisa Kirisame	BLM	47	remove
Izayoi Sakuya	NIN	22	remove
Unavaliable Uername	MCH	43	remove
Remilia Scarlet	SCH	23	remove
Moerisa Kirisame	WHM	19	remove
Emmite	DRK	16	remove
Muse Dash	PLD	35	remove

Members Detail Page:



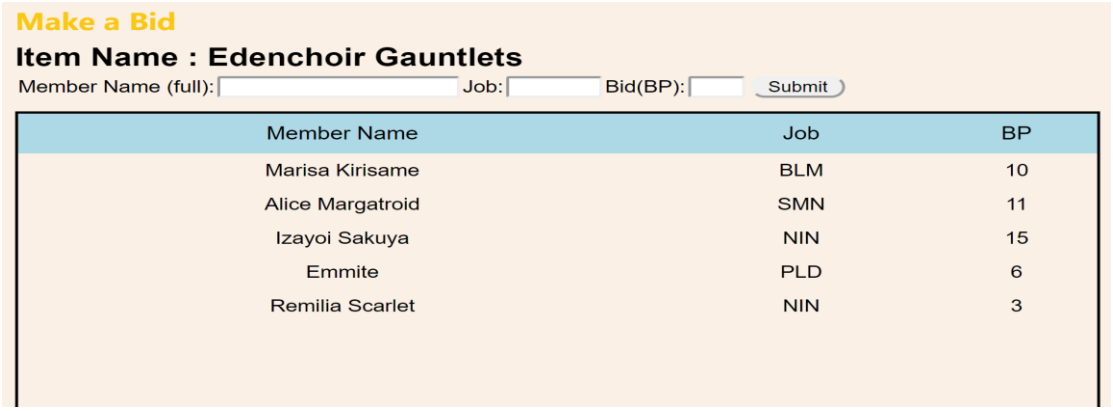
Member Items List

Alice Margatroid Job : SMN BP:42
Item Obtained:
Augmented Deepshadow Blade *1

Marisa Kirisame Job : BLM BP:47
Item Obtained:

Izayoi Sakuya Job : NIN BP:22
Item Obtained:
Edengrace Leg *1
Edenchoir Gauntlets *1

Auction Page:



Make a Bid

Item Name : Edenchoir Gauntlets

Member Name (full): Job: Bid(BP): Submit

Member Name	Job	BP
Marisa Kirisame	BLM	10
Alice Margatroid	SMN	11
Izayoi Sakuya	NIN	15
Emmite	PLD	6
Remilia Scarlet	NIN	3

5 Future Work

Right now, this project can only allow one static use one account to manage their team. There is no other sub accounts allowed for the team members to do their own customization or make request. Another functionality that can be approved is that all data acquiring are need to manually input by the costumes. So that a Web Crawler System can significantly improved the efficiency of data input and reduce the total workload for the team leader. Additionality, the security of the project also can be improved to protect person privacy.

Reference:

Wikimedia Commons, File:Waterfall_model.svg, retrieved form:

https://commons.wikimedia.org/wiki/File:Waterfall_model.svg