CS306 Software Engineering
Compnay Calendar Project
Final Report

Team
Saida HOXA
Ahmed Amine TALEB BAHMED
Aissa HOUDJEDJ

Index

1 Introduction (Purpose of the system, Design Goals)

Purpose of the system:

The system being implemented has the aim to schedule tasks for random employees in a big company with size more than 100. This desktop app will give the opportunity to the users to have more creative space of interactions (eg: amongst workers, workers and their leaders and between teams to be created when needed). The application contains a friendly and smooth user interface in order to make users feel more comfortable. The special feature about this app which makes the difference from other random company calendar apps is the chatting box. One of the design goals of this project is this special feature even though is not accomplished until the given deadline.

Design Goals:

The app will not be used by any random user, actually it will be used by a whole company which will have three different entries corresponding to: admin, manager and user. The relation between them is that the admin can assign tasks to everyone by adding, editing and deleting the tasks and the admin gives grants to the manager to assign this tasks to the workers with the privilege of editing the tasks given by the admin. The workers can only submit the given task. In order to achieve what we want the design goal we tried to achieve is a clear interface to be used without the use of any other external help. The user is now able to navigate throughout the app and to perform the desired operations in a short amount of time. Also the system gives options such as saving the last activity made even without logging out in case of any external interruptions such as shutting down of the system. In this case the user will not be worried of lost data in such special cases.

2 Project Implementation

The project is designed to be a desktop app which is connected to database in order to store the tasks given and other information such as meetings, team info and other stuff to be described below.

Embedding the database with PHP language was a great help for us to proceed faster than expected. Another key point to success in this project was the frequent and the productive meetings we had. Basically the meetings were done online and physically in about twice or more per week whenever needed. In support to this factors we are able to deliver most of the project functioning on time.

Requirement Analysis report along with design report helped in the implementation of the project. We tried to follow the previous given report in each step taken. What we wanted from this app was precise, smooth and easy-to-use interface, so our classes and packets were prepared by following this criterion. For example, the task progress is something included in our project so in order for the user to have an easy access to it we included in the main page class. In addition to this, database was prepared in such order that included 4 tables liked with each other such as: Admin, Meetings, Tasks and Users. This also helped us to have a systematic implementation during the coding part.

Major design changes:

Software development is such that everything can change in any instant because of unexpected events and in our case we had to change the entire language option to implement the project. The unexpected event was the embodiment of Java with the database. Java could not handle the attachment with the database which forced us to change the idea of implementing the code in Java language to PHP, JavaScript languages. There were of course impacts passing from java to JavaScript for the better. There were many helpful websites and sample codes available that made our work faster and easier. Since JavaScript is developing everyday there are lots of forums, questions and answers available that completed all the question marks we had during the implementation. During the design goals report we ensured to accomplish a task such as chat box that normally will help the users to have a faster communication in any situation needed. But because of different

reasons such as time limitation we failed to accomplish this task. Beside this task everything is accomplished as prescribed in the design report.

Some of the design goals prescribed in the design report are as follows: Reliability, robustness, Safety, Good documentation.

One of the important factors that helped us in proceeding quickly throughout the project was **Good documentation**. Our job started to get easier and more effective because we used two parties: one is for the admin and the manager and the other is for workers and normal users to guide them while using our application. The two interfaces have slight changes and each contain its own features.

Another design goal we tried to accomplish is **Robustness**. One example is when a user wants signs and is not registered in database then it won't be accepted then error with be shown by not allowing the user to enter. Another feature of robustness is when the manager and the admin tries to add a user with a name with less than two letters the database won't accept it and an error with appear by warning the admin and the manager to enter a proper name with more than two letters.

Another design goal we presented in the design report was **Safety**. This is an important feature in overall but since our app is not a complicated one it didn't take too much effort to make this feature real. One example is the responsive time which is instant whenever an unwanted action is done and a warning is thrown along with it.

There are lots of things we learned from this project starting from how to decide on design goals to identifying the boundary conditions. The next thing we learned even to practice it after design goals is Subsystem Decomposition where identified the packages. Our project is divided into two subsystems such as: GUI and CalendarComponents.

Also we learned about the hardware and software mapping more as a concept rather than having it practiced since our app is just a simple desktop app needing only a keyboard and a mouse to work on it.

In our system we have some persistent objects which are which values of their attributes have a life time longer than a single execution. And this was realized using a

database and some of the examples are the employees registered along with their attributes such as username, password, name, surname, email and role (admin, manager and workers).

Something to be added to our knowledge was deciding between software control where between event and procedure driven we used event driven programing. Event driven programing executes a specific event and that will call a section of code assigned to a specific event to be triggered by keyboard input or mouse action. For example, when user clicks the command button named 'Sign in' on a form. The mouse click is and event when the click event occurs, php executes the code the Sub procedure named Comment. When the code has finished running, php waits for the next event.

Appendix A.Testing

In the testing part we used an approach where we tested the lowest components first and then we used to facilitate the testing of higher level components. We repeated the process until the components at the top of the hierarchy were tested. First we integrated the bottom or low-level modules and then tested them. This method helped us in determining the levels of software developed already and it was easier to report testing progress among the team members.

Testing plan:

a) Test on sidebar
b) Test on calendar
c) Test on task subsystem
d) Test on user subsystem
e) Test on chart subsystem
f) Test on meeting subsystem
g) Test on event subsystem
h) Test on log in
i) Test on UI subsystem
j) Test on database (this is connected to log in, addUsers, addEvents, addTasks,etc).

Log in testing

We tested by entering different users with different username formats such as using percentage sign and this is supposed to give an error which is shown in the pictures below.

Also we logged in as admin and tried to add a user which contained one character and as stated above this is not allowed and the system naturally should warn the admin to enter a proper username. This is also shown in the picture below. While testing on the log in, in parallel we had to check on database whether the user was added to the data base with the same characters and so on. Beside this in the main page there is a label which says 'Welcome %username%', so we checked logging in as Aissa and the label said 'Welcome Aissa'.

Another testing was done on the calendar. For each task assigned there is a time period given represented by a timeline along with a color in order to make the interface friendlier. The timeline is set after we enter the deadline to a task and then it goes to database and according to this the display must be shown correctly. We checked it by entering several inputs and in each of them the calendar must show the timelines in different colors picked randomly but when the same task is given to 2 different users or more the color is displayed in the same color. So we had to test on that as well by using the admin and checking if the same task added by him to two different users maintained the same color line. Another feature is the color of the current day which is always blue, so we had to check if it was maintained every day.

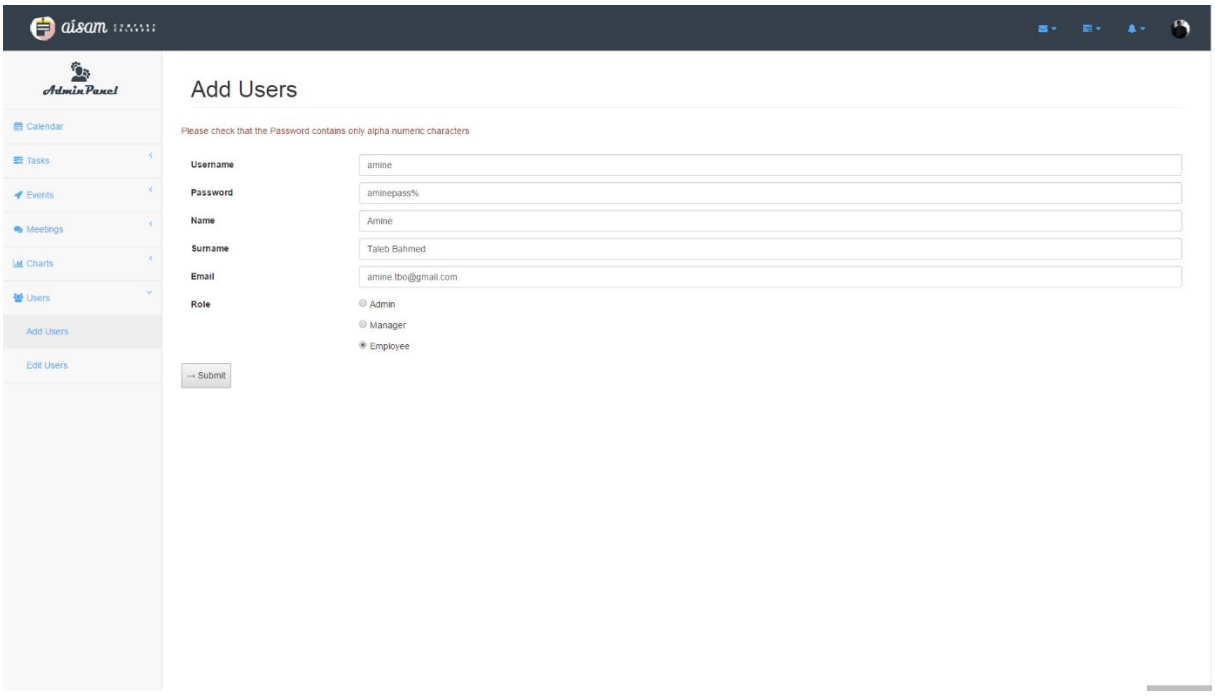

Figure: Error occurs when a password contains non alphanumeric character

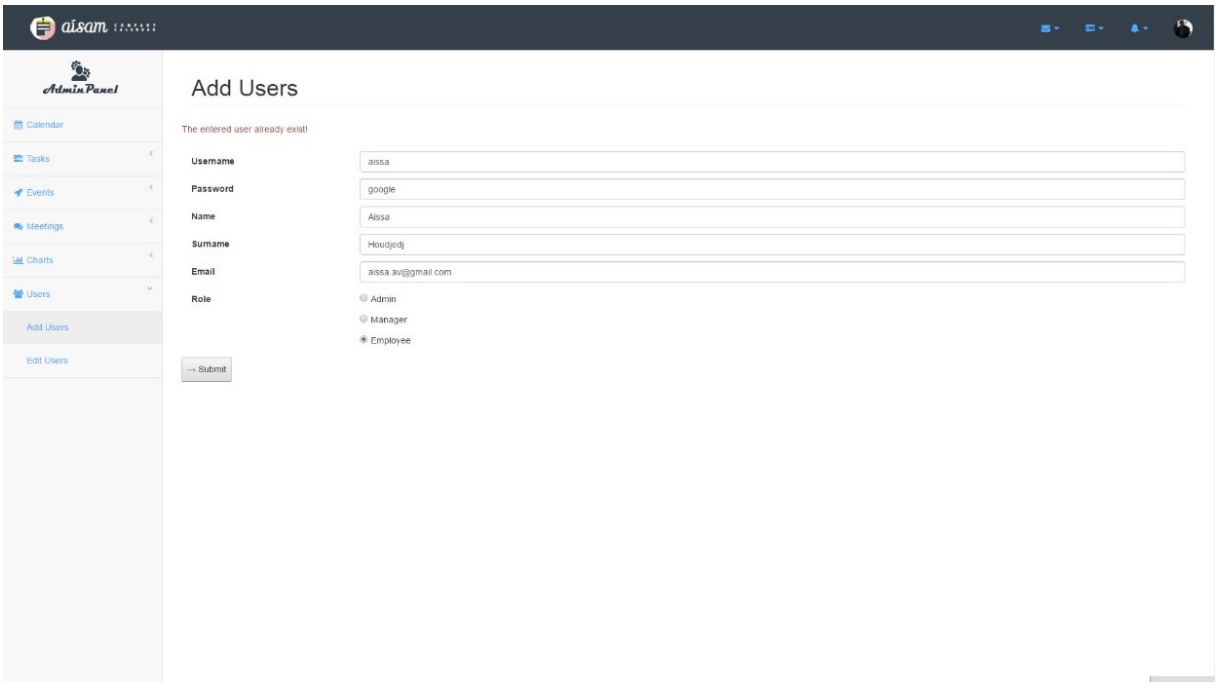

Figure: Error occurs when you entered an existing Username

Figure: Error occurs you submitted an empty input (no password for this case)

Appendix C. Documentation of Class Library.

Package: Login

PhpFiles

- Index.php
- Logout.php
- Database-config.php
- Authenticate.php

CSS Files:

- bootstrap.css
- bootstrap.min.css

JsFiles

- bootstrap.js
- bootstrap.min.js

Package: UInterfaces

- Connect-db.php

CSS Files:

- bootstrap.css
- bootstrap.min.css

JS Files

- sb-admin-2.js

Package: AdminInterface

- index.html

Package: AddTask

- Addtasks.php
- Addtasks_submit.php

Package: EditTasks

- EditTasks.php
- Edit.php
- Delete.php

Package: Login

PhpFiles

- Index.php
- Logout.php
- Database-config.php
- Authenticate.php

CSS Files:

- bootstrap.css
- bootstrap.min.css

JsFiles

- bootstrap.js
- bootstrap.min.js

Package: UInterfaces

- Connect-db.php

CSS Files:

- bootstrap.css
- bootstrap.min.css

JS Files

- sb-admin-2.js

Package: AdminInterface

- index.html

Package: AddTask

- Addtasks.php
- Addtasks_submit.php

Package: EditTasks

- EditTasks.php
- Edit.php
- Delete.php

Package: ViewTasks

- Index.php
- Functions.php
- ViewTasks.php
- Fullcalendar.min.css
- Fullcalendar.print.css

Package: AddEvents

- AddEvents.php
- AddEvents_Submit.php

Package: EditTasks

- ViewEvents.php
- EditEvents.php
- Edit.php
- Delete.php

Package: AddMeetings

- AddMeetings.php
- AddMeetings_Submit.php

Package: EditMeetings

- ViewMeetings.php
- EditMeetings.php
- Edit.php
- Delete.php

Package: Charts

- DonevsProg.php
- TaskProgression.php

Package: AddUsers

- Addusers.php
- Addusers _submit.php

Package: Edit Users

- EditUsers.php
- Edit.php
- Delete.php

Package: ManagerInterface

- index.html

Package: AddTask

- Addtasks.php
- Addtasks_submit.php

Package: EditTasks

- EditTasks.php
- Edit.php
- Delete.php

Package: ViewTasks

- Index.php
- Functions.php
- ViewTasks.php
- Fullcalendar.min.css
- Fullcalendar.print.css

Package: AddEvents

- AddEvents.php
- AddEvents_Submit.php

Package: EditTasks

- ViewEvents.php
- EditEvents.php
- Edit.php
- Delete.php
-

Package: AddMeetings

- AddMeetings.php
- AddMeetings_Submit.php

Package: EditMeetings

- ViewMeetings.php
- EditMeetings.php
- Edit.php
- Delete.php

Package: Charts

- DonevsProg.php
- TaskProgression.php

Package: EmployeeInterface

- index.html

Package: ViewTasks

- Index.php
- Functions.php
- ViewTasks.php
- Fullcalendar.min.css
- Fullcalendar.print.css

Package: ViewTasks

- ViewEvents.php

PackageViewMeetings

- ViewMeetings.php

Package: Charts

- DonevsProg.php
- TaskProgression.php