Online Academic Registration

CS4099 Project

Mid Term Report

Rohit Garlapati (B110316CS)
Debesh Baidya (B111001CS)
Bakki Ebenezer Jayakar (B110822CS)
Setty Prasanna Kumar (B110724CS)

Guided By: Ms. POURNAMI P N

February 29, 2016

Abstract

Identifying the drawbacks of system that exists, compelled us to think, that lead to the designing of computerized system by which we can improve the efficiency of it. We have 14 Departments and 14 hostels altogether in NITC. Procedures during semester registrations are with lot of repetitions that can easily be avoided. For this project we are proposing to design, an online web portal, which computerizes all the manual work.

Contents

1 Introduction

The online academic registration, here forth OAR, is a web application designed and developed to reduce the hassle that students experience during academic registrations as well as the workload off the hostel office staff. The interface of the OAR has been designed keeping in mind, the strength of students and the traffic flow which is usually in peaks during registration. The OAR implements a parallel and synchronized mechanism in issuing tokens to the students during registration. The application also provides a very user friendly interface facilitating convenient ways for the hostel office staff while making changes to the database. Databases are completely secured and a log management system is included. Various aspects of web-designing have been analyzed and implemented.

2 Problem Statement

For the past few years, number of educational institutions have been increasing rapidly, so are the number of students. Many of these institutes are still following the age old registration process and manual ledgers in academic departments and hostel offices. Students and employees of such institutes face difficulties during registrations. Online Academic Registration is a project aimed at reducing the hassle during semester registrations.

3 Literature Survey

We have drawn the fundamentals where we emphasized more on user than browser, from *Adaptive Web Design*. The basic principles underlying the web-designing have been adapted and are this project's first priority. As it is highlighted in following referred website *web-design-basics-guide* the web application follows user friendly arrangements both visually and hierarchically. Web architecture from *AJAX and PHP: Building Responsive Web Applications* has been studied and implemented thoroughly. Database is the backbone of the web application. *Fundamentals of Database Systems 6th edition* has helped a lot in understanding the databases and ways to implement them. Online website *w3schools.org* has various examples and content which we referred in case of small doubts.

4 Project Analysis

4.1 Existing system

The current procedure for the academic registration is as follows,

- Registration date is fixed and announced.
- Students appear for the registration on the announced date.
- Students have to collect no-dues from hostel office and Library.
- Students meet their respective Faculty advisors and get their consent for the academic registration.
- SAC verifies all the documents, and allots a token for each student.
- Students have to wait in queue for their turn for registration procedure.

4.2 Proposed system

In this project report we propose to make the following changes to the academic registration process, by making it online. These steps are to be followed by a student in order to complete his/her registration,

- Student has to login with provided user-id and password.
- If a student wishes to register, for which he/she has to click the appropriate option to start the registration process, step by step.

- In the first step student is shown his hostel dues. He/She is shown a proceed option if and only if he has cleared his dues, or unless he/she updates the payment information.
- If the above step is completed, a checklist is marked online for verification, and that page doesn't appear again.
- In the same way, student is shown Library dues next. If there are no dues he can proceed to the next step.
- After completing the dues verification, student has to upload his Registration slip, DSS fee payment entry, and Fee receipt in the link provided which are accessible by SAC, FA, and Registration Desk.
- After successful SAC verification, and FA verification, a token and time is allotted to the student, and he/she has to appear and get his/her id card renewed.

5 Requirements specification

All the requirements for this project are submitted as SRS (Software Requirements Specification) document along with this report separately. The basic requirements are shown below.

5.1 Backend Software

- Apache Server Ver. 2.4.9
- Mysql Server Ver. 5.6.17
- PHP 5.5.12 as serverside scripting language

5.2 Frontend software

- HTML5
- CSS 3
- Bootstrap Framework Ver. 3.2.0

5.3 System Requirements

- 64-bit Processor
- Min. of 2GB RAM
- 500 GB HDD
- Pentium IV or greater
- Any 64-bit OS
- Works with most of the latest browsers & IE9 or above.

6 Database design

The online web application for OAR, allows students, faculty advisors, hostel staff, sac members and other necessary people, to login separately. The main database has to have the following tables as understood by our team mates.
a) Student
b) Faculty
c) Department
d) SAC member
e) Hostel staff
f) Hostel dues
g) Unverified payments
h) Library staff
i) Library dues
j) Registration staff
k) Student Notifications
l) Faculty Notifications
m) SAC notifications
n) Tokens

6.1 Student Entity

The student entity has self-explanatory attributes like s_id, password, fname, lname, email, phone, sem, probation, d.no, fa_id, checklist.

6.2 Faculty Entity

The faculty table consists of the following attributes, f_id, password, fname, lname, email, phone, fa_stat, hod_stat.

6.3 Department Entity

Every department in the institute has its own id, and the d_name in the table as its attributes.

Student

s.id	pwd	fname	lname	email	phn	sem	prob	dept	fa_id	chk

Faculty

f.id	pwd	fname	lname	email	phn	fa	hod

Department

d.id	dname

SAC member

s.id	pwd

Hostel staff

h.id	pwd

Hostel dues

s.id	month	due

Unverified

s.id	month	due	amt paid	mode	ref no.	date

Library

l.id	pwd

Library dues

s.id	due

Registration staff

r.id	pwd

Student Notification

n.id	s.id	msg	read

SAC Notification

s.id	read

Faculty Notification

n.id	s.id	read

Tokens

t.no	s.id	registered

Figure 1

A relational schema for main database.

6.4 SAC member Entity

SAC members have separate passwords that of his student password, but can use same id. The attributes are id and password.

6.5 Hostel Staff Entity

All the hostel staff will have their respective id and passwords.

6.6 Hostel Dues Entity

This entity stores the record of every student's hostel dues, as entered by hostel staff, during registration time. The attributes of this entity are s_id, due and month.

6.7 Unverified Payments Entity

If a student has due, he has to update the payment info, which is stored in this table. The data can later be verified with bank records. The attributes are s_id, due, month, amt_paid, mode, ref.no, date.

6.8 Library Staff Entity

Employees working in library will be provided with a user name and password, to login to our website and modify library dues at the time of registration.

6.9 Library Dues Entity

All the changes made by the library staff will reflect on this table.

6.10 Registration Staff Entity

The employees in academic section, who conduct registration procedure, will have their respective id and passwords stored in this table.

6.11 Notifications Entity

The students, faculty and sac notifications entities have id, s_id, read_status, message and from attributes respectively.

6.12 Tokens Entity

Tokens allotted to students are stored in this table. The tkn_id attribute is set to auto increment, for every entry with s_id, time_of_approval and reg.stat as attributes.

7 Working principle

The main database has all the above mentioned tables few of which include the records of users and their passwords. Database has been designed keeping in mind, the individual interface for every kind of user.

7.1 Steps explained

When a particular student logs into his account, he'll be taken to his specific homepage. When he chooses to do the registration for the next semester by clicking on the appropriate link provided, he'll be asked to complete a series of tasks one after the other, in order to finish his registration. After each task is completed, the checklist attribute which is initially 0 by default for every student, is increased by 1. The registration desk only sees that the student has completed all the tasks or not.
Note : $ here refers to a php variable
Step 1: Student is shown his mess dues from the hostel dues entity, which he has to clear if more than relaxation amount.

```
SELECT due
FROM hostel_dues
WHERE s_id = $userid
```

If the user pays the due after the dues have been updated, he has to update his payment info which will be saved in unverified payment info table.

```
INSERT INTO
u_payments (s_id, due, month,....)
VALUES ($userid,$x,$y,$z...)
```

here the php $userid variable is set when the user logs in successfully, and is used throughout the registration process.
After successful verification of dues, the checklist attribute in student entity is increased by 1.

```
UPDATE students
SET chk = $new_chk
WHERE s_id = $userid
```

Step 2: Library dues are verified in the similar manner.

```
SELECT due
FROM library_dues
WHERE s_id = $userid
```

Student can only SELECT the hostel_dues and library_dues table. Thus the data is secure. If a student has dues he has to submit the books, and a library official will login to his account and delete the students row from the dues table

```
DELETE FROM library_dues
WHERE s_id = $userid
```

Step 3: Request approval of FA. If FA approves checklist is increased, by whcih this page doesn't appear again.
When a student requests FA's approval, FA can see the Pre-registration slip of the student, which he uploads after step 3, and either recommends him for the registration of the next semester or ask the student to meet.

```
INSERT INTO
s_notifications (s_id,msg,read)
VALUES ($s_id,"MEET ME",0)
```

The read attribute is 0 default by default. When a student sees the message the read is changed to 1. The messages are displayed to the student based on his unique s_id (student id).
Once the fa approves the checklist is increased.
Step 4: Request SAC verification. Token allotment.
After a student gets approval from fa, he can request for SAC verification.

```
INSERT INTO
sac_notifications (s_id,read)
VALUES ($userid,0)
```

The read status here ensures that no student is missed out for the final step. Once a sac member gets a notification from a student he sets the read attribute for that particular student based on his id to "1" and proceeds him to the final registration step. The checklist is marked and increased. A token is generated against student id.

```
INSERT INTO tokens (s_id)
VALUES ($userid)
```

The t_no attribute in this entity is set to auto increment. Thus when a student is approved by SAC, his id is added into the tokens entity.
Step 5: By the time student reaches this step he has to have his checklist at 4. That already implies he has finished all the required formalities. Now the registration desk registers the student and increases the checklist to the maximum 5. Thus the student finishes his registration. The token status can be viewed by every student.

```
SELECT t_id,s_id
FROM tokens
```

After a student is registered the checklist is at 5, which implies that he is registered and it is displayed in his profile. FA can find out who all students under him have registered too.

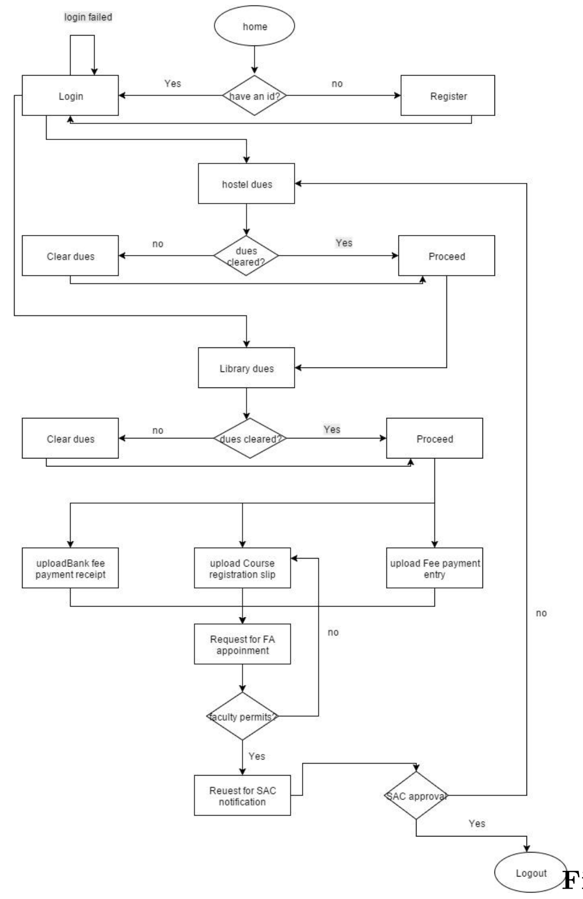

Figure 2 Flow chart for student interface

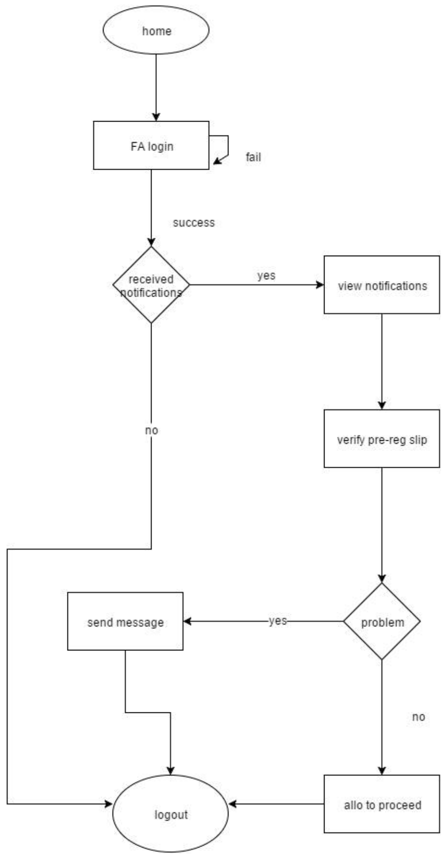

Figure 3 Flow chart for Faculty Advisor interface

```sql
SELECT s_id
FROM student
WHERE f_id = $userid AND chk = 5
```

8 Interfaces Involved

The web application provides the following user interfaces.

8.1 Student Interface

The student interface provides the student, an option to register for a semester. The student has his notifications displayed to him based on his unique student id. The student will be able to see his mess dues, update his payment info, library dues, upload required documents, request appointment with FA, request SAC verification, during registration. *See* **Fig 2**

8.2 Faculty Interface

The faculty can see the students under them, students who are registered for the next semester, notify student to meet him, see the pre-registration slip of the student. *See* **Fig 3**

8.3 SAC Interface

The SAC only checks for human errors that can happen. The possible errors that a student can make and a computer can't recognize is, if a student uploads wrong documents. The SAC will have access to the documents uploaded by the student. Once the SAC verifies the documents, the members can approve a student, by which a token is automatically allotted to the student. *See* **Fig 4**

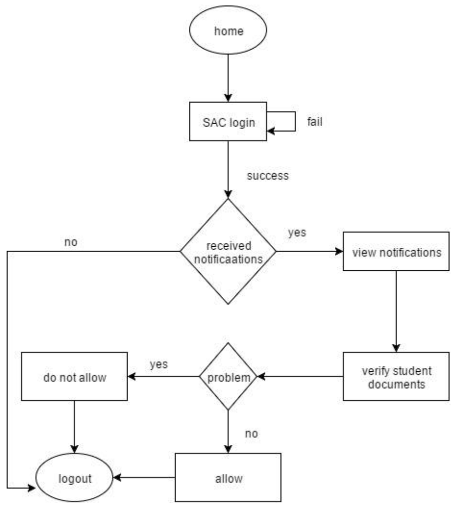

Figure 4 Flow chart for SAC interface

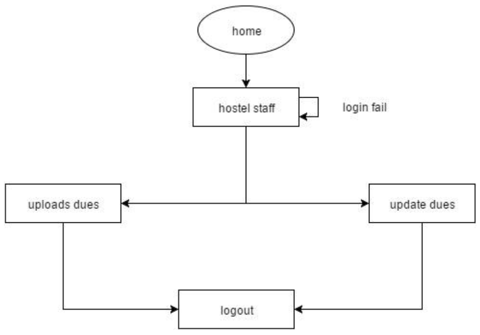

Figure 5 Flow chart for hostel staff interface

8.4 Hostel Staff Interface

The hostel staff will have a user friendly interface once they login. They basically have to redirect the list of dues that they pass during registration to our web app. They have an option to enter the dues into our database, based on the roll number of the student, rather than typing it to excel and printing it. Only a hostel staff employee can make changes to the dues database. *See* **Fig 5**

8.5 Library Staff Interface

In a similar manner library staff can login to the web app, and add the dues of the student. If the student clears his dues, the employee can delete the student id from the table. *See* **Fig 6**

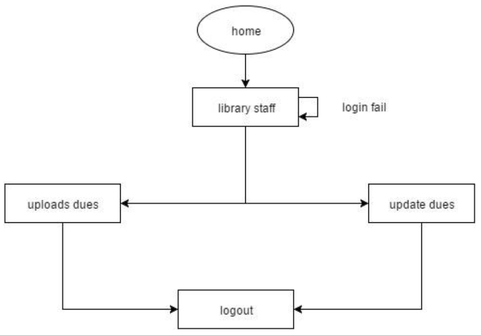

Figure 6 Flow chart for library staff interface

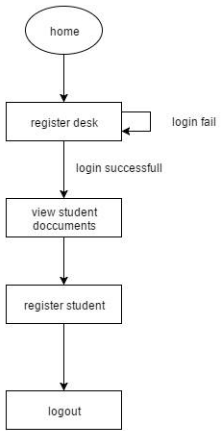

Figure 7 Flow chart for Registration desk interface

8.6 Registration Desk Interface

The registration desk interface shows the student details, if he completed the other required tasks or not. The interface accesses the checklist of a student based on his id, and is displayed. They make the final increment to the checklist marking him registered. *See* **Fig 7**

9 Work done and Description

The design discussed above has been implemented partially. Interfaces for student, faculty, library staff, hostel staff have been created and are being tested.

9.1 Login page

The login page provides options for various users to set mode and login with their respective id and passwords. The login page essentially collects the data from the form and sends it to a validation page through ajax call.

```
send_data = {
        'id':id,
        'pwd':pwd,
        'mode':mode};
var validate = $.ajax({
        type: "POST",
        url:  "validate.php",
        data: send_data
});
```

Once the login is successful the ajax call returns a json string with "accept" key.

9.2 Profile page

Based on the mode selected by the user and succesful authentication, the user is redirected to his profile page. The content of the profile page is displayed according to the mode and user id.

```
//individual profile pages
for different users//
if($mode == "library staff")
{
?>
<div class="container"
style="margin-top:60px;
           height:2000px;
           border:2px solid">
<p>
Welcome to library management!
</p>
</div>
<?php
}
```

9.2.1 Student profile

The main functinality for the web app is to provide different features for different users. The student profile will allow a student to view notifications, register for a semester, view dues in hostel and library, request appointment with FA, etc.

Notifications have been tested succesfully. A student can see notifications that are sent to him. The notifications are filtered using the students $user_id.

```
if($mode == "student")
{
$sql1 = "SELECT *
FROM student_notifications
WHERE to_id='$id'";
$result = mysqli_query($con,$sql1);
$notif = array();
$modal = array();
```

The $notif array tags every notification with a number to index them in the notifications dropdown. The $modal array is used to display every notification when clicked in a bootstrap modal.

9.2.2 Faculty profile

In a very similar manner the faculty profile is provided with different features like viewing students under him, send students messages, etc.

9.2.3 Library and Hostel staff profiles

Library and Hostel staff, after they login using their respective user id, password and mode set, are redirected to their profile pages. The only feature these staff get is to update/view dues of students. These users have no use of notifications and hence a "Notifications are not enabled for your account" message is diplayed.

```
$notif[0] = "Notifications are not
   enabled for your account"
```

9.2.4 Features

1. Student who has successfully logged in can view notifications, check dues, and other related content.

2. Faculty can view a list of students registered under him. He/She can also message students individually.
3. Hostel/Library staff can add/delete entries for student dues.
4. User can use the forgot password option to recover password.
5. Admin can add/remove registration dates.

9.3 Database

As directed by our guide, we have implemented the database as per the design above, and 15 sample rows in every table. There have been a few minor changes made as we implemented the design.

9.4 Web Hosting

The project is currently live on Amazon Web Services (AWS) hosting service. The web app can be accessed through http://52.24.74.222 link. The hosting service curretnly is provided on free tier basis, which allows a limited transactions, which made us to restrict public usage.

10 Future work

10.1 Database linking

We'll have to know the structure and design of hostel, library databases which are not readily discussed by the authorities, to facilitate direct display of dues in our web application by connecting their databases to ours. Current database contains separate tables for faculty and other staff members. This can be reduced to a single employee table, once we get access to the structure of current employee database in our college.

10.2 Main functionality

After most of the basic wireframe is done, the main registration function will be implemented, as a part of phase 2. A student should be able to see an option to register for a semester, when admin opens registration. Profile pages for registration desk and sac members will be implemented. Once the implementation completes, we'll proceed to trials.

10.3 Mess Management

By using mess management interface, a student can view his/her dues on day to day basis. Hostel office and mess staff can login to their respective profiles. Web application will be readable and user friendly, for easier entry and retrieval of data. The database required might be very huge, hence proper care will be taken while designing the database.

11 Conclusion

Web app will completely be mobile friendly as the work progresses. The app will constantly be peer-reviewed. Any reported bugs and feedbacks will be considered seriously, and the web app will be improvised accordingly. Our entire project is dedicated to reduce hassle during registration by making it online which indeed saves paper.

References

[1] Aaron Gustafson. *Adaptive Web Design*, Easy Readers, LLC; 1st edition (May 30, 2011).

[2] Alex Bigman's article on 4 key principles of web design *http://99designs.com/designer-blog/2014/11/07/web-design-basics-guide/*, May 21, 2015.

[3] Bogdan Brinzarea, Filip Chereches-Tosa, Mihai Bucica. *AJAX and PHP: Building Responsive Web Applications*, Packt Publishing; 1st edition (March 10, 2006).

[4] *www.w3schools.com*

[5] Ramez Elmasri, Shamkant B. Navathe. *Fundamentals of Database Systems 6th edition.*