ICS4U – Project Development Example
Discovery Day
Project Requirements

System Description

The discovery day system is designed to allow students to register themselves for the West Carleton Discovery Day event. Students can visit the site select from available workshops in different time slots during the day. Using this system they can construct a schedule for themselves during the day.

Students will receive a confirmation email that will allow them to sign back in to change their registration at a later date. They will be able to pay for their registrations (if applicable) using PayPal, as well as download any required permission forms.

Admin staff organizing the event should be able to visit the site and read administrative information regarding current registrations, as well as to customize the available sessions, student registrations, and print reports.

Requirements

From Student Perspective:

- Able to visit the site and be presented with a way to pick sessions
- Have the description for each session available to be read
- Students should authenticate using their school credentials
- Receive a confirmation email for each change detailing schedule for the day
- Be able to pay for sessions via paypal during registration, or later by signing in at a later date
- Be sent reminders regarding permission form / payment deadlines
- Have a NOTIFY ME option to have a notification sent when session has an opening
- Provide alternate email to be reached at.

From Admin Perspective:

- Scheduling:
    - Add new sessions
    - Cancel sessions (or remove completely if nobody registered)
    - Edit session info
- Registrations:
    - Provide a way to turn off changes to registration
        - System wide based on date or manual switch (date should be announced)
        - Manually close individual sessions, but allow admin to reg
    - Provide for different schedules (All day activities, All day after keynote, etc)
    - Build schedule for the day
- Display customizable announcement messages to registrants (“Registration closes on….”, etc)
- View logs
- All lists of students should be searchable/sortable (like jquery filter)
- Tools:
    - Register students manually (Optionally send email)
    - Manually change registration (including option for overriding limit sizes) (Optionally send email)
    - De-register students (Optionally send email)
    - Email all registrants
        - System wide
        - In an individual session
        - With template email messages
        - Based on outstanding payment

- Reports
    - Bulk output all students and their registrations (CSV)
    - Bulk output all sessions, their name, slot, instructor, supervisor, room (CSV)
    - Output for printing
        - MAKE ALL REPORTS BASED ON SELECTABLE SESSIONS (checkbox in list?)
        - all session registrants (big wall)
        - Sessions sorted by session (1A, 1B, 1C, 2A, 2B, 2C, etc)
        - Sessions sorted by name
        - Sessions sorted by supervisor
        - Sessions sorted by room
        - Presenter Schedules - simple one-page per presenter, rooms, # of registrants, supervising teacher
        - Attendance tracking
        - Secretary report – all sessions they are responsible for, payments made, tracking sheets, etc

<u>System Perspective:</u>

- Prevent duplicate entries
- Force limited time constraint on registration (can only hold for XX minutes, will kick user from page after time is up)
- Registration will depend on student ID – same as computer password
- Trim student names / IDs when entered (leading / trailing spaces)
- Registration log – adds, changes, deletes, admin actions vs. student
- System log – session adds / edits / deletes, supervisor changes, room changes, etc
- Process paypal payment for students' registrations
- When students are added to a session where they've asked to be notified, they should be removed from the notify list for that section
- Session components
    - Name
    - Description
    - Time slot
    - Capacity
    - Cost
    - Paperwork
    - # of registered students
    - registration buffer (spots to hold open)
    - linked sessions (for double/triple sessions. Ex ids: 1,2,3 or 45,46)
    - Room
    - Presenter
    - Supervisor
    - Secretary (teacher responsible for paperwork)

System Flow

Student Perspective:

Front page welcome, link to PDF of all descriptions, dates, and session writeups (TWO OPTIONS)

(1) Sign-up for sessions

- NEXT:
- Selection page:
    - (hidden) Generate session ID
    - Presented with drop-down boxes for sessions
    - Sessions that require permission forms or money are highlighted
- NEXT:
- Confirmation page
    - starts countdown minute timer
    - asks for first name, last name, homeroom
    - asks for student ID twice, with regex to verify input as: >> S followed by 9 digits
    - details any requirements based on their selections: >> money owed, permission forms to sign, etc
    - final confirmation button
- NEXT:
- Conclusion page
    - Confirms schedule for the day
    - Highlights important details again (cost, etc)
    - Details confirmation code
    - Explains school GMAIL for confirmation
    - Thank you and we're done

(2) Change my registration

- Please enter your registration code (found in your school gmail account)
  >>> Forgot confirmation code?
    - Please enter student ID (EX: S012345678) and we will email you your confirmation code.
    - Please check your school Gmail account.
- NEXT:
    - "Please enter student number and confirmation number"
    - (forget that number? click to email it to your school account)
- NEXT:
    - Present editing page
        - remember to allow re-choosing the same session
        - name & person info not editable

Admin Perspective: (/admin)

- Presented with Login box → password is hashed by javascript and sent to the server
- After auth: Landing page presents a jquery-filter style table of all currently registered students. Column headings are:
    - LastName, FirstName, Session1, Session2, Session3, Paid
- Table should have checkable boxes in leftmost column. With students selected, reporting can be selected based on selected students, or in general.
- OPTION TO CHOOSE: Sessions (Menu option somewhere)
    - Page presents a jquery-filter style table of all sessions. Column headings are:
        - Name, Timeslot, Capacity, Slotted, Cost, Supervisor
    - Table should have checkable boxes in leftmost column. With sessions selected, reporting can be selected based on selected sessions, or in general
- For both tables, when row item is clicked on, use ajax to bring up all information in an editable window
- For both tables, should be sortable by clicking on the column heading

System Design

Structural Ideas

- Set Session cookie, to be used for uniqueness on registration
- Store student number once student number is entered.
- Most settings stored in database table: settings and loaded in database.php

Pages

database.php → basic database PDO connection

Index.php → landing page

chooseSessions.php → Select sessions page
→ Where the SQL INSERT will be initiated
→ Also presents option to visit NOTIFY ME page (can only be done once you have registered for other sessions)
→ Make session ID

editStudReg.php → Edit student registration page → Always an SQL UPDATE
editSessionInfo.php → Edit session info page → Always an SQL UPDATE

getSessions.php → returns JSON list of all sessions

forgotCode.php → Lookup code based on stud_id and send email

notifyMe.php → Lists all full sessions and allows students to select from list via checkboxes

admin.php → Admin landing page (defaults to showing sessions Table)

adminSessions.php → Table view and reporting options for sessions (opening one for edit will open editSessionsInfo.php)

adminStuduents.php → Table view and reporting options for student (opening one for edit will open editStudReg.php)

Various reporting files… maybe reports.php which will group them all together and differentiate based on flags?

Databases

sessions

---

id	int	// unique, autoIncremented ID for session
title	varchar(100)	// title of session
timeslot	tinyint	// available time slot (1, 2, 3, etc)
linked	varchar(25)	// JSON encoded array of linked slot and id. Example: {"1":49,"2":50}
capacity	smallint	// # of students allowed in total
buffer	smallint	// integer number of spots to hold open
slotted	smallint	// # of students registered in this session
cost	double	// $$$ cost if applicable, 0 cost otherwise
room	varchar(25)	// full description of room ex: Rm. 216 – ex: P1
desc	varchar(2048)	// long description of session
override_keynote	tinyInt	// if 1, schedule should say no keynote (presuming all-day or 1-2 double session)
presenter	varchar(100)	// name of presenter - ex: Emmell, S
supervisor	varchar(100)	// name of supervisor - ex: Emmell, S
secretary	varchar(100)	// name of secretary - ex: Emmell, S

students

id	int	// unique, autoIncremented ID for student
studID	varchar(50)	// student's studentID
firstName	varchar(100)	// student's first name
lastName	varchar(100)	// student's last name
hr_teacher	varchar(100)	// name of homeroom teacher – ex: Emmell, S
email	varchar(50)	// "s" + studentID + "@scloud.ocdsb.ca"
choice1	smallint	// ID of chosen session for Session 1
choice2	smallint	// ID of chosen session for Session 2
choice3	smallint	// ID of chosen session for Session 3
confirmation	varchar(25)	// Unique confirmation code
paid	tinyint	// 1 or 0 for paid / not paid
reservedUntil	datetime	// deadline to remove registration, -1 for confirmed registrations
sessionID	varchar(100)	// unique id created during registration to prevent duplicate registrations

announcements

id	int	// unique, autoIncremented ID for each announcement
text	varchar(500)	// message text
urgent	tinyint	// 0 or 1 for non-urgent / urgent

notifyList

sessionID	int	// id for corresponding session
studentIDs	varchar(2000)	// comma split list of all student IDs to be notified when session opens up. Ex: 3,457,746

settings

id	int	// unique, autoIncremented ID for each
settingName	varchar(25)	// name. ex: adminSessionColumns
settingValue	varchar(2000)	// value for each setting ex: "name,timeslot,capacity"

Important Functions

Log entries

- updateUserLog($string)
- updateSystemLog($string)

SQL related

- student
  - updateFromSQL // populate a student's info from an SQL query
  - updateDB // send current student info to be stored in DB
- //MORE