

# Department of Accounting & Information Systems

## ACCT/INFO GROUP PROJECT SUBMISSION

CASE STUDY/PROJECT TITLE: INFO263 Group Project.....

Please complete all sections of this sheet, sign the declaration and attach the sheet to your project.

The next panel must be completed by all team members, **including** the agreed proportion of work done on the project. (For example. if all members of a team of four made equal contributions then enter 25% for each team member.)


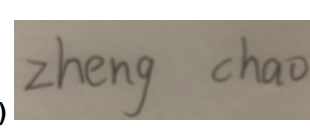
	Student ID No.	User ID e.g. <i>afg21</i>	Student Names: (Surname first & alphabetical order please)	Proportion % (Agreed by group)
1)	..... 98105803..	lca84 .....	Carolán Leo Turlough Oscar .....	47.5..... ...
2)	..... 37969552	wth37 .....	Huang William Teng Hui.....	30.....
3)	..... 21671773	zch65 .....	Zheng Chao .....	12.5..... ...
4)	..... 38053397	Cna45 .....	Natapu Carl.....	10.....
5)	.....	.....	.....	.....

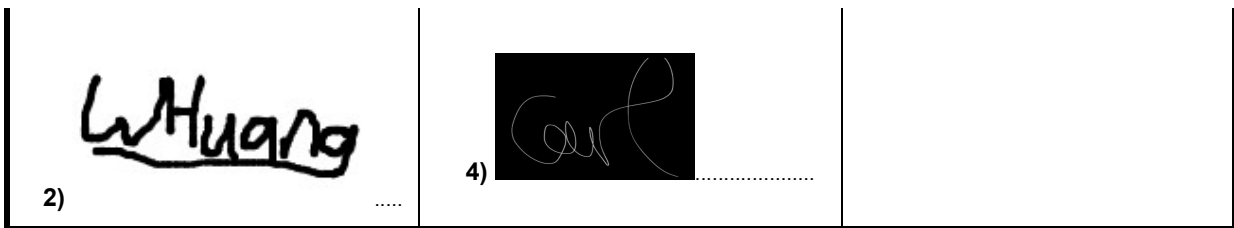
### Honesty Declaration

- I declare that this is an original assignment and is entirely my own work.
- Where I have made use of the ideas, words or work of others, I have acknowledged the source in every instance.
- Where I have used any diagrams (including modifications) prepared by others, I have acknowledged the source in every instance.
- I have read and understood the Dishonest or Improper Practices Statement overleaf.
- I am aware of what constitutes cheating, and the penalties for plagiarism and cheating as described in University publications.
- I am aware that the content of this written work may be checked against an electronic database.
- I have supplied the correct word count and have taken no steps to cause disclosure of an incorrect word count for the assessment.

I have read and fully understand the Honesty Declaration above, and hereby certify that this item of work submitted for assessment is entirely the work of the members of the group, in the proportions stated.

Signed . . .

1) 	3) 	5) .....
--	--	----------



*Under the University Regulations, evidence of any of these or other forms of dishonest practice by any student(s) represents grounds for disciplinary action and may result in penalties ranging from denial of credit for the item or work in question, to exclusion from the University.*

### **Dishonest or Improper Practices**

It is recognised that students will discuss course work and assignments with others, and such discussion is an important part of the learning process. However, any work presented by a student for credit in a course must be that student's own original work. If students are directed to complete work submitted for credit in groups, the work submitted must be the original work of the group. Work submitted in breach of these requirements or which fails to comply with other instructions contravenes the University's Dishonest Practice and Breach of Instruction Regulations. Such work will either not be marked, and all credit for the work in question forfeited, or the matter will be referred to the University's proctor for investigation and possible referral to the University's Disciplinary Committee.

Penalties which may be imposed in the event of a finding of dishonest or improper practice include loss of credit for a course or an item of assessment and, in serious cases, suspension or expulsion from the University. A record is kept of all instances of dishonest conduct.

Instances of dishonest or improper practice in coursework and assignments include but are not limited to:

- ❖ Plagiarism. Plagiarism means the dishonest presentation of work that has been produced by someone else as if it is one's own. Please note that the presentation of someone else's work as one's own, even without dishonest intent, may still constitute poor academic practice, and this may be reflected in the mark awarded. There are academic conventions governing appropriate ways to acknowledge the work or part of the work of another person, including the APA and Harvard citation styles. For further information see the UC Library website, under "Citations and Referencing".
- ❖ Submitting for credit in a course without the prior consent of the Course Coordinator for an essay, research paper or any other written work which, although it is the student's own work, is substantially the same as work which has already been (or will be) submitted for credit in another course, whether in the Department of Accounting and Information Systems (ACIS Department) or some other department or academic institution.
- ❖ Copying the work of another student. This includes copying the work submitted by another student for credit for a course in the ACIS Department or some other department or academic institution.
- ❖ Knowingly allowing another student to copy work which that other student then submits for credit for a course in the ACIS Department.
- ❖ Arranging for another person to complete work which is then submitted for credit for a course in the ACIS Department. An example falling in this category is work submitted for credit which has been obtained from a commercial assignment completion service. Care must be taken when using editing services as it is **only** assistance with grammar, punctuation and expression that is permissible and does **not** include the addition or amendment of content.
- ❖ Completing work for another student which is then submitted by that other student for credit for a course in the ACIS Department.
- ❖ Including made up or fabricated material in work submitted for credit for a course in the ACIS Department.
- ❖ Collaborating in the preparation of answers for take home or online tests unless advised otherwise in the take home test instructions.

If you are in doubt about any of the above with respect to a particular course, you should discuss the matter with the lecturer or course co-ordinator concerned.

See also the University Discipline Regulations, Dishonest Practice and Breach of Instructions Regulation, and Academic Integrity Policy – refer to UC Calendar and UC web.

# Info263 assignment documentation

## Setup:

- unzip the folder in ampps/www along with the usual setup.
- connect to the database with username "root" password "mysql"
- Ensure that when creating the schema you name it: tserver
- When confronted with the login page, ensure to use the username: admin, password: 50e04656a6aa90c9c4d70f5a5ab2466c
  - alternatively, you may create your own login by inserting into the user table
- All set!

Our website begins at the login page named index.php. upon entering the login details and clicking submit, an ajax post request will be sent to login.php. This request is then queried to the user table in the database to validate the login. Login.php responds with a success => 0 or 1 depending on whether the request failed or passed. Upon failing, the user will be alerted they have entered invalid credentials, while a succession redirects the user to the home screen dashboard.php.

The dashboard presents a series of actions the user may perform once logged in. They may logout, create new events or view events. Each one of these links contain an href link to their corresponding pages.

The logout link is located at the top right corner and when prompted, will destroy the session, not allowing the user to return to the previous pages. They will then return back to the login page.

## Create events page:

### Create New Event

Assessment Name

Select event cluster

Select Day of Week

Week of the year

Select Year

Start Time

Time offset

Assessment Time

Select Groups

Create Event

[Back to Dashboard](#)

CreateEvent.php consists of a large form the user may fill out to add a new event to the list of events. Some of these fields are in the form of dropdown boxes containing valid parameters. This decision allows the user to select only valid parameters for a better user experience and prevents invalid data when inserting the values into the database. If the user tries to create the event without filling out the required fields, an alert will notify them to do so. Clicking create event with a filled out form sends an ajax post request with the form data to newEvent.php which acts as a backend to insert into the database with the form parameters. A failed request will result in a success => 0 response, alerting the user they have entered invalid input while a successful request responds with success => 1 and will redirect the user to the dashboard. The new event has been created in the database and is visible in view events.

### View events page:

## View events

[Back](#)

Event Name	Event ID		
EMTH210-20S1 1-hour	172	<a href="#">Delete</a>	<a href="#">More Details</a>
STAT101-20S1 Thursday	171	<a href="#">Delete</a>	<a href="#">More Details</a>
STAT101-20S1 Wednesday	170	<a href="#">Delete</a>	<a href="#">More Details</a>
EMTH210-20S1 2+hour	169	<a href="#">Delete</a>	<a href="#">More Details</a>
EMTH210-20S1 4-hour	168	<a href="#">Delete</a>	<a href="#">More Details</a>
EMTH210-20S1 3-hour	167	<a href="#">Delete</a>	<a href="#">More Details</a>
MATH110-19S1 Test Wednesday E035	166	<a href="#">Delete</a>	<a href="#">More Details</a>
MATH110-19S1 Test Wednesday E033	165	<a href="#">Delete</a>	<a href="#">More Details</a>
MATH110-19S1 Test Friday	164	<a href="#">Delete</a>	<a href="#">More Details</a>
MATH110-19S2 Test Thursday	163	<a href="#">Delete</a>	<a href="#">More Details</a>

The view events page consists of the 3 files view\_events.js .html and .php which interact with each other to display the page. The html file declares the layout of the page, declaring the header titles as well as the search bar and button. The search bar should take an input which matches an event name exactly. Optionally, if given a substring, it will match with the first result in the order of the query. The php file then queries the database with a select \* from vw\_front\_event. The js file sends an ajax get request to the php file which responds with the vw\_front\_event query. The js file is run as a script tag in the html file. This allows each row of the database query to be displayed. The js file also defines a delete and more details button for each event which may be clicked to perform their corresponding actions. Each button is bound to their own event id.

Deleting performs an ajax post request to delete\_event.php which queries the database to delete from all tables where that event id matches each table. This is shown in more detail from the screenshots below.

The more details button takes the event id bound to the button and sets the event id as a local variable. This way, the user is navigated to more\_details.html and the event\_id is stored to know which event id information to display. The more\_details.html page contains a

series of headers and runs the script more\_details.js. This allows an ajax get request to be made to view\_events.php where each entry of the event id is shown in detail. An edit button is also bound to each event and when clicked, assigns the event id to local storage in order to navigate to edit.php and retain the id of the event being edited. The edit.php page displays some fields that the user may want to change. We thought it was relevant to have a php page rather than an html page so the dropdown boxes could be populated, making interaction easier and pre-validating the data. Upon submission, a script is running to post an ajax request to editEvent.php, sending the form data through. The editEvent.php page is responsible for querying the database with the form information to update the event details. Upon failure, the user is prompted with an invalid input alert and upon success, the user is prompted with a success alert and is returned to the more details screen.

#### View\_events.php-Delete button:

```
var newButton1 = document.createElement( tagName: "BUTTON");
newButton1.id = res[i][0].event_id;
newButton1.innerHTML = "Delete";
boton.appendChild(newButton1);

var newButton2 = document.createElement( tagName: "BUTTON");
newButton2.id = res[i][0].event_id;
newButton2.innerHTML = "More Details";
moreDetails.appendChild(newButton2);

newButton1.addEventListener( type: "click", listener: function(e :MouseEvent ) {
    $.ajax({
        url: "delete_event.php",
        type: 'POST',
        data: "event_id=" + this.id,
        success: function(res) {
            location.reload();
        }
    });
});
```

A delete button is created for each event displayed called newButton1. The button is bound to an event listener which performs an ajax post request to delete\_event.php and sends the event id.

#### delete\_event.php:

```
require_once("config.php");

$event_id = $_POST['event_id'];

$query = "delete from front_action where event_id = '$event_id'";
mysqli_query($conn, $query);
$query = "delete from front_daily where event_id = '$event_id'";
mysqli_query($conn, $query);
$query = "delete from front_weekly where event_id = '$event_id'";
mysqli_query($conn, $query);
$query = "delete from front_event where event_id = '$event_id'";
mysqli_query($conn, $query);
```

The event id which was posted by the event listener is stored and 4 queries are defined and executed to delete the event from all tables.