IAB207 Assessment 3 - Fully Developed Solution: Details

Overview - Web Application Solution: Event Management System

Due: Wednesday 30th October @ 11:59:00 PM (Week 14)

Task description:

In this assignment, you will complete the implementation of your Event Management System web application in a team of 3 or 4 students. We recommend that you use the previous Assignment 2 front-end prototypes developed by your team as inspiration for this final solution. In fact, you should feel free to use one (or a combination) of them as a start point for this assignment.

NB: Generative AI tools are not allowed in this assessment task.

Unit Learning Outcomes assessed:

1. Analyse client requirements and design a web application in preparation for the software development process
2. Use platform tools to specify and validate web forms, create APIs, and persist information in databases
3. Develop, test and deploy program code of web applications using both a programming language and platform tools
4. Critically reflect upon personal performance, and the performance of others
5. Collaborate and work effectively in a team environment to produce quality deliverables for a client within agreed timelines.

Estimated time for completion	Weighting	Group or Individual	How I will be assessed
30 hours	50% of final grade	40% Group 10% Individual	Using a 7-Point grading scale

What you need to do

1. Read the Criterion-Referenced Assessment Rubric
2. Read the assignment objectives, assignment description, late submission penalties, and background
3. Follow the step-by-step guide of the assignment.

What to submit

A Web Application Solution as detailed below with a succinct submission report.

Overview

In a team of 3 or 4 students you will develop an event management web application that uses the following languages, frameworks and libraries:

1. HTML, CSS and Bootstrap
2. Python
3. Flask (plus Flask-WTF, Flask-SQLAlchemy, Bootstrap-Flask, Flask-Login and an authentication library e.g., werkzeug or bcrypt)
4. WTForms
5. SQLAlchemy

Groups with a team size of 3 will need to develop the same functionality as teams with a size of 4. Your group will need to select one group member's assignment 2 submission as a base to start working on this assignment.

Project Deliverables

Please read the below deliverables (requirements) carefully as every team will be expected to address each of the requirements in their final submission.

You are free to reuse the code from the individual submissions of Assignment 2 of one of the team members to develop all pages again. As part of the project, the following needs to be developed and demonstrated:

Part 1: Project Requirements

1. A landing page that clearly conveys what kind of events the website promotes, as well as some upcoming events. Your home page must allow users to browse the events by category (you are free to support this functionality using a drop-down menu or some other intuitive way). In addition, you might include text-based search functionality. However, this is an advanced feature and is optional to those who would like to include it (although it isn't specified in the CRA, but the more intuitive your search is, the better).
2. The events listed on the landing page should show a useful overview of the events with information in accordance with the context of the design (including the event's status at a minimum).
3. Anybody should be able to choose an event and view the details of it. The event details page should have an image of the event, a description of the event, the date and any other specific

information that is helpful to the user and is in accordance with the context of the design and requirements.

4. To buy tickets, the user should be logged in. The user should provide details such as the quantity of the tickets to be booked. An order is generated by the application, and the order detail (order ID) is provided to the user for reference. Consistent with the booking history view developed in assignment 2, that view should now reflect the user's actual, previously booked events.
5. A logged-in user can post a comment on an event. The name of the commenter, the comment, and the date the comment was posted can be seen by anybody.
6. A logged-in user can create an event by entering relevant details. In addition, an event must have one of the following states: Open, Inactive (event date in the past), Sold Out, or Cancelled. The owner of the event should be able to update the details of the event (**excluding** the event status). They can also cancel their own events.
7. A logged-in user can log out of the application.
8. Anybody can register for the site by providing details of their first name, surname, email, password, contact number, and street address.
9. Error Handling - Ensure that the two errors 404 and 500 are handled appropriately.
10. All your pages should include a navigation menu and utilise data that is dynamically drawn from and stored in the database.

Project Management

Teams are required to maintain a list of tasks and ensure that each task is assigned to individual members within the group in an equitable way. You are free to allocate a task to one or more members of the team. In the final submission report, teams will be required to clearly demonstrate the contribution of individual members to the tasks.

You should have a GitHub repository and add the 'IAB207-QUT' account as a contributor. Please visit the Week 10 '**Deploying your Web Application (https://canvas.qut.edu.au/courses/17367/pages/deploying-your-web-application)** ' page for more information on how to get started with Git.

Please note: Each team member needs to submit a peer review feedback form. Each member will be reviewed by all other members of the team. The plan and the peer review feedback will be considered when allocating marks to team members. All team members may **not** get the same mark/grade for the assignment. Please consider this when reviewing your team members.

An example of a task definition, allocation, and progress is shown below:

Task	**Week 1**	**Week 2**	**Week 3**	**Week 4**	**Week 5**

Landing Page	Liam: Worked on the html template		Liam: Worked with the dynamic content		Abdul: Completed verified all menu items
Registration	Sheru: Complete the form	Sheru: Tested the form validations		Sheru: Integrated the Flask-login support	
Create event		Liam: Added create event functionality			
Database model	Abdul: Completed the models.py with all DB elements	Abdul: Tested all the objects and their relationships	Liam: Added Flask-login support	Sheru: updated the database model	Abdul: Checked all the database updates, queries
Error Handling					Liam: Added errorhandler

Development Guidelines

1. Develop code that is well written with reasonable comments.
2. You should use the assessment starter code zip file that is provided to assist you with organising your code. Please change the folder name from ‘projectfile’ to ‘a3_groupX’ (where X is your group number from Canvas) and the module name from ‘website’ to the name of your application.
3. Implement using HTML, CSS, Bootstrap, and Python (with Flask and the other prescribed modules) only.

Database Design

A database solution will be required to achieve desired functionality in your solution. Your database design will consist of at least four tables:

1. An event table containing the event details – you can add more tables to store more information or you can choose to keep it simple if you see fit.

2. A booking table storing information about the price, quantity and date with relationship to a user and an event.
3. A user table storing information about users, including their login credentials.
4. A comments table.
5. Any other tables that you decide to include to normalise data (however, we recommend keeping things as simple as possible).

Things to note

- All code should be written and contributed to by all team members
- JavaScript is not used in this unit and should not be used as it may distract you from completing work that will get you marks. The code you submit will be tested and run, so there should be no unexpected dependencies.

Submission Instructions

Teams must ensure that they are correctly registered on Canvas well before the submission of this assignment. Each team is required to submit a single ZIP file containing a single outer folder named their Canvas team number e.g. “a3_group1”. Within this folder there should be:

1. A Submission Report (**template (https://canvas.qut.edu.au/courses/17367/files/4634962?wrap=1) (https://canvas.qut.edu.au/courses/17367/files/4634962/download?download_frd=1)**) PDF that includes:
   1. GitHub repository link for your team
   2. Team declaration (so we can double-check who is on your team)
   3. A table showing tasks completed by whom (see above)
   4. “`git shortlog > output.txt`” and then copy & paste the contents into the report. This will show us the commits of each user and the total number of commits.
2. Your application code as a package (in a folder). The code should be able to be executed by running a main.py located just above the package folder. You must include a working sqlite db file in the package folder with some synthetic (or dummy) data for events and users (do not expect the tutor to create the database)
3. Please **do not** include any venv directory (Python virtual environment). Please make sure the ZIP folder you intend to upload is a reasonable size (150MB or less - preferably much less).

How you will be marked:

Please see the criteria assessment rubric.

Late Submission Penalties:

Consistent with QUT commitment to real world learning, managing priorities, competing commitments and time are essential skills for effective learning and professional life. Assessment work submitted after the due date will be marked only with an approved extension (MOPP E/6.8.2). Assessment work submitted after the due date without an approved extension or, where

an extension has been granted, after the extended due date, will not be marked and a grade of 1 or 0% will be awarded against the assessment item.

Academic Integrity:

Note that the use of generative AI (e.g., ChatGPT) is not allowed in this unit. Please refer to QUT's policies on academic integrity and plagiarism (MOPP C/5.3).

Feedback:

Under normal circumstances, you will receive marks for each criterion via a Canvas rubric within 10-15 working days of submission. Click on Grades to see your results. Usually the reason for each choice of mark is self-evident, the marker will include some written feedback about your performance. You should use this feedback to strengthen your performance in the next assessment item.

Moderation:

All staff who are assessing your work meet to discuss and compare their judgements before marks or grades are finalised.

IAB207 Rapid Web Development Assessment Task 3 Rubric:

Web application (40 marks - group)

	HD / D	**M / P**	**< P**
Requirements	The site is convincing. Fully functional page without errors. All requested requirements are fully met. Submission of this standard succeeds in the fundamental requirements but also looks convincing to a	The site is not fully convincing. Page is partly incomplete or has some minor errors present. Most of the requirements have been met and the page is functional with a few errors present (i.e. the site still works and the errors only mildly impair the usage of the solution). The solution is good and	The site is unconvincing. Requirements misunderstood or ignored, the page does not meet requirements. Submission of this standard would not function (or partially function) as a minimally viable solution. There would be the presence of serious errors that would impair the solution's basic processes.

	potential customer (e.g., fully responsive). The solution is thorough, comprehensive, robust, and well tested. Consistent with MVC.	relatively convincing to a potential customer.	
Landing page	6	4.5	3
Intuitive search e.g. by categories etc...	4	3	2
Event details page and commenting	8	6	4
Booking an event	7	5.25	3.5
Create / update event	8	6	4
Register a user	3	2.25	1.5
Login / Log-out of user	3	2.25	1.5
Error handling	1	0.75	.5
TOTAL	**40 - 30**	**<30 to 20**	**<20 to 0**

Development and deployment of tutorial example (5 marks - individual)

Marks/Criteria	**5**	**4**	**3**	**2**	**1**
Workshop participation through exercise completion and deployment of website on PythonAnywhere Platform. (5 marks demo)	The travel Web application as developed in all workshop exercises is complete. Successfully deployed the application with Flask and SQLite, works smoothly	The travel Web application as developed in the workshop exercise has most components including the database Successfully deployed the application with Flask and SQLite, works smoothly	The travel Web application as developed in the workshop exercise has some components Partial deployment working - some functionality operational, but certain features break or don't work (e.g. due to missing database file)	The travel Web application as developed in the workshop exercise has used Flask templates Unable to deploy the software on PythonAnywhere. There were some missing steps.	Only the HTML exercises are complete No attempt was made to deploy the app beyond the development environment.

Peer evaluation (5 marks - individual)

Marks/Criteria	**5**	**4**	**3**	**2**	**1**
Peer assessment via survey (5 marks for demo)	(as per student evaluation) Please note that the peer review is an individual component, that is, your individual marks will depend on how your team has rated you.				

NB important note on teamwork contribution: Your overall mark for Assignment 3 may be adjusted if work was not performed equitably within the team. The marker will inspect individual contributions and team performance as measured via various evidence items including GitHub commits and other intelligence and adjust your overall mark to take account of differences in contribution between team members. This is process is in addition to peer survey.