Database Applications
ISYS1101/1102 | Semester 2 2023
Assignment 4: Web Database Applications

Assessment Type	Individual Assessment
Due Date	23:59 Sunday 29 October 2023
Demonstrations	No demos for this assignment
Silence Period	Starts at 5:00PM Friday 27 October 2023
Weight	35%
Submission	Online, via Canvas. Submission instructions are provided on Canvas.

1 Overview

1.1 Assessment Criteria

This assessment will determine your ability to:

1. compare and analyse relational and non-relational database systems;
2. write technical reports suitable for a non-technical audience;
3. write SQL statements required for CRUD (create, read, update and delete) operations on the database you built;
4. by embedding above SQL as appropriate, write the complete web application using HTML, PHP, JavaScript and any other required tools;
5. Demonstrate your complete web database application.

1.2 Learning Outcomes

This assessment will assess how you attained the following course learning outcomes:

CLO 1: apply advanced data analysis and modelling concepts, physical design, integrity, security and transaction management.
CLO 4: build an efficient database application with an emphasis on storage management, indexing and query optimisation;
CLO 6: develop a simple web-based interface for a database.

2 Assessment Details

2.1 Preparation Work

Mandatory:

You are required to be able to write code in HTML, PHP, JavaScript and any other programming/ scripting languages to build a fully-fledged web database application. More importantly, you should be able to use Oracle API (oci8) within a PHP program. In order to acquire this pre-requisite knowledge, you must complete Week 2 lab session.

Optional:

If you haven't done any web programming before, you are highly recommended to complete the LinkedIn Learning tutorials listed in Week 2 Pre-lecture activities.

The assignment solution (web application) must be hosted on a separate folder on school's web server (titan.csit.rmit.edu.au). The recommended folder is

/[Your Home Directory]/public_html/dba/asg4/

Then, your URL of your application homepage will be

https://titan.csit.rmit.edu.au/~s<student_number>/dba/asg4/index.php

In order to protect this website from unauthorised access AND only allowing staff members access it, include a .htaccess file in the /[Your Home Directory]/public_html/dba/asg4/ folder. A pre-configured .htaccess file is available on Canvas in the Assignment 4 folder (along with this specification).

We build the application based on the design you submitted for Assignment 1. If it was partially completed or discovered any shortcoming in the design, a partial schema that you can use to build tables required for this assignment is also available on Canvas in the Assignment 4 folder (along with this specification).

2.2 Assignment Task Description

Task 1: Build a simple web database application

The Australian Electoral Commission (AEC – https://aec.gov.au) is responsible for conducting federal elections and referendums. Australia's manual system of federal elections has one of the most complex and time-consuming counting operations in the world. While it can at times require patience, the federal election counting process delivers (1) integrity to the results, concentrating on (2) accuracy in a (3) highly transparent manner.

While manual process ensures these three key priorities, there are two areas of concern to may stake holders, namely:

1. The time it takes to count votes and the human resources required to complete the process within an acceptable time frame
2. The volume of papers it requires and the environmental impact of running a manual election.

Let's suppose you are employed by a software development company that just received a contract from AEC to build a computerised voting system for federal elections. As in the case with manual elections, the most important aspect of this system is to ensure the integrity of the voting system, accuracy, and transparency.

System requirements

The system is developed in several phases. The first phase, which you are responsible for, is limited to **federal general elections for House of Representatives**. The following voting processes are not in the scope of this phase:

1. Federal general elections for senate
2. Federal by-elections
3. State and territory elections
4. City council and shire council elections
5. Referendums
6. Any other election services provided by AEC

In the first assignment, you had analysed the database requirements, designed the database backend for the voting system, identified various database optimisations, and implemented the backend infrastructure for the electronic voting application.

In this assignment, you are required build a front-end web application for **one** of the main tasks of the application. The task within the scope of the assignment is the interface for voting.

Note: On the election date, registered voters are required to attend a polling station and cast their vote on a ballot paper similar to above. The actual voting process is much more flexible with pre-poll voting, postal voting, absentee voting, and declaration votes. However, for the scope of this assignment, we only consider regular voting process that happen on the election day.

The computerised voting process will mimic the current manual voting process.

Steps to replicate:

When a voter visits a polling station, the polling official will ask the voter the following questions:

Question 1: 'What is your full name?'

This enables the polling official to look up your name on the electoral role for that division. If your name is on the electoral role, you will be asked the next question.
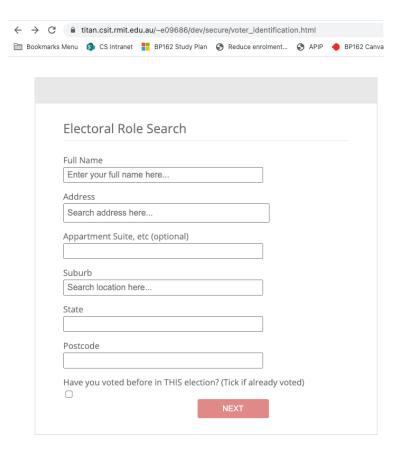
Question 2: 'Where do you live?'

If the address given is the same as that shown on the electoral role you will be asked the next question.

Question 3: 'Have you voted before in THIS election?'
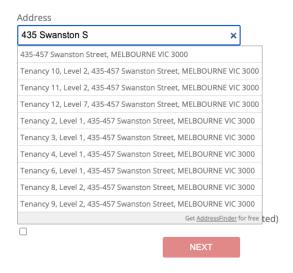
If you have not voted before in this election, you will be issued with a ballot paper.

In your application, a web form will capture the answers to these questions.

Under the Address field, when the user starts to enter their address, the form should use an (externally available) Australian Address Lookup facility and pre-fill the other subsequent fields.
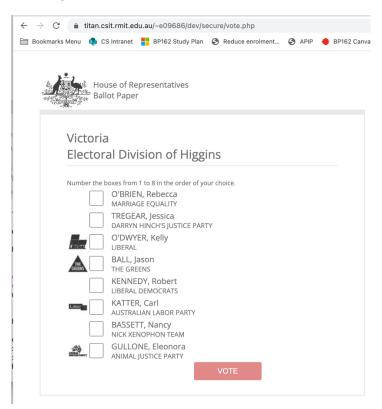
e.g.:

There are a number of free JavaScript-based address lookup facilities available. One of the very customizable facilities is addressfinder.com.au Their free-tier allows you to do 300 address lookup a month, which will be sufficient for the assignment.

In the manual system, you will be issued a ballot paper similar to the following:

In your application, a dynamically generated form will be presented to the voter. It will be similar to the following:

You must use html, JavaScript, CSS, and PHP web technologies to generate these pages.

Once the form is submitted, the data must be stored in appropriate tables in your Oracle database.

[Very important] To ensure integrity and confidentiality of the voting process, once a voter is issued a computerised ballot paper, there should not have any identification records to positively identify who cast that vote. As such, only the following data are stored with each computerised ballot paper.

- Election Event ID
- Electorate Name
- Candidate IDs and the Preferences cast for each candidate (i.e which candidate got the first preference, who got the second preference, etc)

However, in order to avoid double-voting, there must be some mechanism in place to note down when a voter votes in the current election. You may use either the function developed in assignment 1 or a new mechanism for this purpose.

Your application must have appropriate validations at each of the steps:

1. When the voter enters their name and address, it must be cross-checked against the electoral register. If the combination doesn't exist, with an appropriate explanation the voter should present a blank form to re-fill their name and address again.
2. If the voter is valid, however, if they have previously voted, the system should give the voter a formal warning (i.e. voter fraud is a criminal offence) and revert to the home page of the application.

Note: You may not validate the preference votes cast on the computerised ballot paper. While your computerised system is, in fact, capable of validating the preferences entered and make sure that all ballots are formal, the real system does not do any such validations. Voters, in current system, may inadvertently or intentionally enter informal preferences (such as missing numbers, doubled up numbers, writing anything that identifies a voter, etc). Since your task is to mimic the current system, you are not required to validate the preferences entered.

All in all, your application must have at least three pages:

1. An html form for the electoral role check-up;
2. A dynamically generated web form (using PHP) for the ballot paper;
3. Another PHP script to process the ballot. It will do the following tasks:
   a. Store preference data in appropriate tables in your Oracle database. Depending on the schema used, it may require inserting data on to more than one table. For example, if you use the sample schema provided ballot data are stored in Ballot table and BallotPreference table;
   b. Update the ballot issuance record, so, if the same voter attempts to vote again, we know that that's an illegal attempt. If you use the sample schema provided, this information is stored in Voted table.

In addition to above, you may require other PHP scripts (required by the above scripts) and/or JavaScript and CSS files.

Task 2: Analyse different database platforms

You have implemented the **Computerised Voting** application and the **AirBnB-lite** application using two very different database backends: (1) Oracle implementation in assignment 1; and (2) MongoDB implementation in assignment 2. In this Task, you are required to write a research report analysing these two implementations.

The system requirements between these two applications are somewhat different. The current computerised voting system caters for about 17 million voters in Australia and manages 3-yearly federal elections. In contrast to that, AirBnb has 2.9 million hosts with 14,000 new hosts each month, and 7 million listings worldwide. A registered voter record (a row in VoterRegistry table) is much smaller than a document in listringsAndReviews document collection.

Based on the findings from your two implementations, write a report identifying the advantages and disadvantages of both backend approaches and a conclusion making your recommendations.

Following metrics can be used to compare two database backend implementations:

- Query performance
- Resource requirements (disk, memory, CPU, network bandwidth, etc)
- Security issues, such as SQL injection
- ACID properties, transaction and concurrency control
- Scalability
- Ability to handle massive volumes of data
- Ability to execute complex queries
- Data integrity
- Differences in (for example media) data types

Note that while the current implementation of the MongoDB database does not deal with security, you should consider how this would be handled in the real application.

Your report may also include case studies (implementations other than your computerised voting and AirBnB applications) for both paradigms and draw conclusions based on their findings.

The report length should be between 1000 - 1500 words. You must be careful about quoting texts extracted from other sources. You can paraphrase them with proper referencing. Before you start your report, please refer RMIT Library Referencing Guide, available at: https://www.rmit.edu.au/library/study/referencing

3 Submission

Follow the instructions on Canvas to complete your submission.

3.1 Task 1 Submission

You will need to submit two things:

1. The URL of the Task 1. Please submit your website URL as submission instruction. It must follow the pattern https://titan.csit.rmit.edu.au/~s<Student_Number>/DBA/asg4/index.php

2. A zipped file containing full codebase for your website. It must include html, JavaScript, CSS, and PHP files and also make sure your Oracle credentials are also included. DO NOT use the RMIT authentication (or any other confidential passwords) for Oracle.

3.2 Task 2 Submission

The report length should be between 1000 - 1500 words. You must be careful about quoting texts extracted from other sources. You can paraphrase them with proper referencing. Before you start your report, please refer RMIT Library Referencing Guide, available at:
https://www.rmit.edu.au/library/study/referencing

You can use Microsoft Word or another word processing application to work on your research report. At the end, convert it into PDF format. Do not submit Word file. if that option is not available on your system there are free pdf converters online you can utilise. e.g. http://convertonlinefree.com/

3.3 Assessment Declaration

When you submit work electronically, you agree to the RMIT assessment declaration.

3.4 Late Submissions & Extensions

A penalty of 10% per day is applied to late submissions up to 5 business days, after which you will receive zero marks.

Short extensions may be granted by the course coordinator up to 1 business day *before* the due date in accordance with RMIT Assessment Adjustment process. However, extensions are not guaranteed and require suitable documentation. The course coordinator may refer requests to Special Considerations.

Special Consideration *may result in an equivalent assessment*, which may take the form of a timed assessment assessing the same knowledge and skills of the assignment and are generally granted on an individual basis. For more information refer to the RMIT Special Consideration process.

3.5 Supported software for assessment and grading

The markers will check the functionality of the hosted web application. Make sure it is up and running for another two weeks after the submission deadline.

4 Marking Guidelines

4.1 Task 1

- 15 marks for the statements required for tasks 1 – 6.

4.2 Task 2

- 20 marks for the report

The detailed breakdown is provided on the marking Rubric available on Canvas.

5 Academic Integrity and Plagiarism (Standard Warning)

Academic integrity is about the honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e., directly copied), summarised, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites. If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviours, including:

- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students

For further information on our policies and procedures, please refer to the RMIT Academic Integrity Website.

The penalty for plagiarised assignments includes zero marks for that assignment, or failure for this course. Please keep in mind that RMIT University uses plagiarism detection software.