Assessment:	**Web Programming 37(8)1**
Date	2022-05-20
Assessor	M.Magorimbo
Total:	70 Marks
Duration:	180 Minutes (includes time for uploading etc.)

WEB PROGRAMMING 37(8)1 ST

INSTRUCTIONS

- This assessment does not include any theory. See the attachments for the paper.
- This assessment must be submitted on AssessmentQ. Do not send any solutions via MS Teams (unless authorized to do so)!
- *Unless otherwise stated*, Invigilation is exclusively on (HybeFlex/ Zoom)! Keep your camera on at all times. Any updates to the assessment will be posted on the shared screen (HybeFlex/ Zoom).
- *Unless otherwise stated,* do not use MS Teams for submissions for whatever reason. In case of technical difficulties, you should submit your assessment to your lecturer with a proper subject line that follows the naming convention *WPR37(8)1 ST: StudentSurname StudentName.*
- Any work that is uploaded must be in zip format ONLY. Other file types may give problems!
- At the end of this assessment, you should only upload a *single* zip file. See the saving instructions at the end of this assessment.
- Create a separate project for each of the tasks using NPM!!!! This is very important.
- Should any work need to be uploaded, ensure naming conventions are followed as instructed. Please see the saving instructions at the end of this assessment.
- No instructions or directives of the invigilator shall be disregarded; if any of the instructions are disobeyed, candidates shall expose themselves to disqualification from future assessments.
- Observe the Honor Code[1].
- Each candidate is expected to maintain academic integrity. Any form of plagiarism or cheating that is detected can lead to disciplinary measures being taken.
- The allocated time includes time for uploading/ saving your solution.
- Penalties may be applied for late work or other practices that impact your submission.
- No explanation of the assessment questions shall be given.
- The mark allocation is subject to change.
- Any technical issue related to the assessment platform being used needs to be reported to the invigilator immediately, and evidence of errors must be emailed to support@belgiumcampus.ac.za and venter.f@belgiumcampus.ac.za with a *cc* to your lecturer.
- *It is your responsibility to ensure that you have saved your examination correctly! We will not accept excuses and explanations after the assessment.*
- Marks are reserved for best practices including commenting the important parts of your code etc.
- Delete *node_modules* before uploading your solution, and ensure that you do not delete package.json.

[1] The Honor Code is a statement addressing issues such as cheating, stealing, and misrepresentation, made by a school or other institution in which its participants pledge to adhere to. Honor Codes are self-regulating because under an honor code, students are required to turn in other students in violation of the code. <https://cs.stanford.edu/people/eroberts/cs181/projects/honor-code/honorcodes.html>

Assessment:	**Web Programming 37(8)1**
Date	2022-05-20
Assessor	M.Magorimbo
Total:	70 Marks
Duration:	180 Minutes (includes time for uploading etc.)

OVERVIEW OF THIS SUMMATIVE TEST.

This practical assessment contains two tasks. The first task is about creating a RESTful API, and the second one is on building a GUI-based application that uses a webpage to create interactions with a user. This examination considers your skills in HTML (EJS), CSS, Node, Express, MongoDB and other utilities such as REST client, etc. Read the instructions carefully; and good luck ☺ !

Remember to read the entire question before you begin, including the notes at the end of each task.

THE REQUIREMENT FOR THIS ASSESSMENT

Childcare is the caring for and supervision of a child or children. You have been approached to create a sample web application that can be used to record data for nannies. This examination will require you to use your knowledge and skills to create a proof-of-concept that answers the questions that follow.

Overview

You have been asked to build an application that requires the details as described in the table below to be recorded. Please note that the table also includes the validation rules for the required data, and the table applies to both Task 1 and Task 2.

To store values, please note that Task 1 will use MongoDB and Task 2 will only use an array to store the information. Beyond this, there are no further constraints except those that have been stated in the task itself. Routes will be tested as has been asked in the assessment for both tasks. No marks will be awarded for routes that do not work.

Here is the table of constraints for Task 1 and Task 2:

The required data and the validation is described as: -

Property	Data Type	Constraints
name	String	Required, Any Valid String
experience	String	Required; Can only be either *Basic, Intermediate* or *Advanced*
certified	Boolean	Required; Defaults to false
age	Number	Required; Can only be between 21 and 45

Table 1: Description of values

TASK 1: REST API, EXPRESS, MONGO DB [35 MARKS]

Write a Node application that allows the user to:

1. Add a new record to your database collection by sending a request to the endpoint */add.* The request must be sent using POSTMAN or REST Client, and in JSON format.
2. View all records in the MongoDB collection by sending a request to the endpoint /view.
3. View details for a single record by sending a request to the endpoint /view/<id> where the <id> is a value generated by MongoDB when the record was created.
4. Update details for a single record by sending a request to the endpoint /update/<id> where the <id> is a value generated by MongoDB when the record was created.

5. Delete a record by sending a request to the endpoint /delete/<id> where the <id> is a value generated by MongoDB when the record was created.

Note

- Use the routes stated in the question. The API will be tested only using the stated routes!
- REST Client is preferred but you will not be penalised for using Postman.
- The name of the database should be *nannies*.
- You will use JSON for data interchange for all requests and responses.
- Your schema must be well-defined
- All validation must be handled by the schema
- Separate all your concerns to improve code maintainability
- Use comments where necessary
- Keep your configuration in a separate file. For example, your .js files must not specify details to do with the database or port
- Suppose that your application was supposed to be committed to GitHub, ensure that you only upload what is necessary and ignore the rest

TASK 2: EJS, ARRAY METHODS, EXPRESS [30 MARKS]

Task 2 will no longer use a database, but will use an array of objects to store values instead. The application must receive the values from a form. Validation must be applied using JavaScript (not via a schema, this question does not use MongoDB). Note that you should also pick the most appropriate form controls for data input.

By creating a separate project, develop a form-driven application that allows a user to:

1. Add a new record to your array by sending a request to the endpoint '/add'. This route must load a page that contains a form. You may call this page index.ejs. Initialise the array with a single record.
2. View all the records in the array by navigating to the home route ('/view'). If there are no items in the array, the application must inform the user by writing some text on the web page. Otherwise, the application prints out the requested values. You can use an HTML table.
3. Filter the collection for records with a particular *experience (Basic, Intermediate* or *Advanced)*. For maximum marks, you should use the most appropriate *array* method for this purpose and also ensure that you are as dynamic with the URL as possible. You will use the '/view/*<experience>*' endpoint and if there are records with the requested filter, just display a message saying *Nothing found.*

   Example: If the requested URL is '*/view/basic* and there no records that are in the array with the status uninsured the application should display *Nothing found.*

Note

- Create an HTML form for the data entry. If you struggle with the form formatting, do not bother with CSS! However, you should know that marks are reserved for the choice of form elements, variety of the form elements, and CSS application. This may count for a maximum of 8 marks.
- Use a partial to display the form on the index.ejs.
- The endpoints above will be visited by typing an address in the URL bar of your browser. You are not going to use POSTMAN or any REST client for this task.

Assessment:	**Web Programming 37(8)1**
Date	2022-05-20
Assessor	M.Magorimbo
Total:	70 Marks
Duration:	180 Minutes (includes time for uploading etc.)

Assessment:	**Web Programming 37(8)1**
Date	2022-05-20
Assessor	M.Magorimbo
Total:	70 Marks
Duration:	180 Minutes (includes time for uploading etc.)

HOW TO SUBMIT: ZIP ONLY! ZIP ONLY!

NAMING CONVENTION

You must submit a ZIP file that uses the Campus standard naming convention. Any examination that a student creates for electronic submission must conform to the following naming convention:

WPR3781 Examination_StudentSurname_StudentName.zip

The reason for this is to ensure that your examination is received correctly by your lecturer; this prevents any inconvenience for you as the student and the lecturer.

PACKAGING YOUR SOLUTION

1. Create a folder that follows the naming convention explained above.
2. Move the Tasks folder into the named folder you created in Step 1 (remember to delete node_modules).
3. Finally, create a zip archive of the folder you created in Step 1. This folder should now contain the screenshots and the project files.
4. Upload your zip archive.